SULO – a simplified upper-level ontology

Michel Dumontier[1,*,†], Remzi Çelebi[1], Komal Gilani[1], Isabelle de Zegher[2], Katerina Serafimova[3], Catalina Martínez Costa[4] and Stefan Schulz[5,6,†]

[1]Institute of Data Science, Department of Advanced Computing Sciences, Maastricht University, Netherlands
[2]b!loba, Tervuren, Belgium
[3]Graphwise, Sofia, Bulgaria
[4]Department of Informatics and Systems, University of Murcia, Spain
[5]Institute for Medical Informatics, Statistics and Documentation, Medical University of Graz, Austria
[6]Averbis GmbH, Germany

Abstract

Ontology-based data integration remains a core challenge in healthcare and the life sciences, where the need to analyse heterogeneous datasets demands robust semantic interoperability. Upper level ontologies (ULOs) play a key role in ontology integration, by relating their concepts using common upper-level classes and relations. While ULOs offer a formal foundation for knowledge representation, their complexity often limits adoption by domain experts and standards developers unfamiliar with formal logic or ontology engineering. In this paper, we introduce the Simplified Upper-Level Ontology (SULO) — a lightweight framework comprising a minimal, intuitive set of classes and relations designed to support common modeling needs in biomedical terminologies, ontologies, and schemas. We introduce two ontology design patterns that systematically decompose complex predicates into reusable semantic structures grounded in SULO. We apply the patterns to represent class expressions in the SNOMED biomedical ontology, and to formalise SPHN, an RDF-based biomedical schema. Finally, we compare our approach with other upper level ontologies. The proposed approach facilitates interoperability and promotes reuse, while preserving the expressiveness necessary for rich domain modeling.

Keywords

Upper level ontology, Foundation Ontology, Formal Knowledge Representation, Knowledge Graphs, Schema

1. Introduction

1.1. Background

The need to capture facts and knowledge about a domain in a standardised representation has led to the development of domain-specific terminologies and ontologies. In the past three decades, the evolution of the discipline of Applied Ontology has stimulated the development of upper-level ontologies (ULOs), also known as top-level and foundational ontologies, which attempt to provide a general theory of *what exists*. Examples are BFO [1], UFO [2], GFO [3], DOLCE [4], YAMATO [5], SIO [6] and BioTop [7].

ULOs are grounded in different metaphysical commitments and modeling styles. They are motivated by different scopes of application, e.g., BFO for natural sciences, DOLCE for linguistic and cognitive engineering, and UFO for information science. ULOs claim their usefulness as an overarching axiomatic framework for domain ontologies (DOs), which represent *what exists in a specific domain*, by offering guidance on the conceptualization and formalisation of the domain, enabling automated reasoning to check conformance of the domain ontology to the ULO. However, creating alignments between domain ontologies and ULOs, and between ULOs, remains deeply challenging owing to the complexity of the

Proceedings of the Joint Ontology Workshops (JOWO) - Episode XI: The Sicilian Summer under the Etna, co-located with the 15th International Conference on Formal Ontology in Information Systems (FOIS 2025), September 8–9, 2025, Catania, Italy
*Corresponding author. Michel Dumontier, michel.dumontier@maastrichtuniversity.nl
†These authors contributed equally.
0000-0003-4727-9435 (M. Dumontier); 0000-0001-7769-4272 (R. Çelebi); 0000-0001-7292-1026 (K. Gilani); 0000-0001-7292-1026 (I. d. Zegher); 0009-0004-0797-7313 (K. Serafimova); 0000-0003-1857-1744 (C. M. Costa); 0000-0001-7222-3287 (S. Schulz)

task [8]. In contrast, SIO and BioTop each offer an ULO that is directly extended to their particular domains.

Apart from terminologies and ontologies, many domain schemas have also been developed. Even sometimes called "ontologies" or "schemas" (such as the SPHN [9] up to 2024) or "semantic models" (such as CARE-SM [10]), they do not claim the rigor of formal ontologies. They support domain data structures from a data modelling perspective rather than a domain modelling one. Finally, large terminologies and (meta)thesauri such as SNOMED CT [11], NCIT [12], and the UMLS [13] include their own ontological or conceptual upper levels, as well as systems that are primarily intended to provide support to purpose-driven data collection, e.g. HL7v3 [14]. These upper levels have been developed based on pragmatic considerations, rather than through an ontology-driven analysis of the application domain.

1.2. The practical use of ULOs and upper-level models

The aim to develop interoperable and computable knowledge graphs must navigate complex choices and address various challenges posed by current upper-level distinctions and models, whether from upper-level ontologies, domain-specific upper ontologies, legacy terminology frameworks, or domain-specific schemas. Such decisions have to take into consideration:

- The requirements include understanding nuanced philosophical considerations and familiarity with logic. Even ULO experts struggle to correctly and consistently apply the ontology in classification tasks [15]. ULOs adopting particular philosophical stances such as realism facilitate some domain applications, while struggling with others [16][17]
- ULOs have known missing or underrepresented areas. E.g., GFO explicitly separates universals and particulars. BFO supports ternary relations only in its Common Logic formalisation (with time as the third argument), but refrains from a concrete time model. Not all ULOs support realizables, social entities and information entities. BFO, GFO, and DOLCE do not offer any data properties to capture literal values, e.g. numeric values characterizing measurements.
- The foundational categories of ULOs and the top level of domain ontologies often use labels that carry implicit assumptions, e.g. *Object*, *Process*, *Event*, *Function*, *Quality*, and *Disposition*. These assumptions differ not only between models using the same labels, but also evoke varying associations among domain experts. Not always are these categories defined in precise terms, which can lead to misunderstanding and unintended modeling results [15]. This is particularly true with systems that carry a strong legacy, such as SNOMED CT [11] with upper-level categories labeled such as *Clinical Finding*, *Observable*, *Qualifier Value*. They lack clear definitions, are difficult to translate and therefore difficult to map to ULOs[18].
- Some ULOs deliberately adopt unfamiliar or technical labels derived from philosophical ontology such as *Continuant*, *Endurant*, *Perdurant*, *Specifically dependent continuant*. Although this approach can more precisely define terms, they may be less clear or intuitive ot a general audience. On the other hand, using terms that are broadly acceptable and reusable across fields may be interpretable diffrently in defferent disciplines.
- Domain ontologies often show an excess of different relation types. To support the structuring of data graphs according to specific shapes, informal top levels often include numerous specialized but relatively unformalised relations of the form "**hasRangeType**", such as **hasPatient**, **findingSite**, **hasTemperature** where the range is implicitly constrained by the target class.

It is therefore no wonder that a large majority of ontologies have been created without any reference to a formalised upper level. In BioPortal, a repository of currently 1,204 biomedical ontologies, only for BFO, a significant re-use of upper-level content can be demonstrated with 138 reference to the class BFO_0000002 (bfo:continuant), whereas references to other ULOs are in the low single-digit range. A study published in 2017 had analyzed 355 ontologies hosted in BioPortal and found that 21% of them had imported one or more BFO classes [19].

2. Proposal: A Simplified Upper-Level Ontology (SULO)

Towards addressing the aforementioned issues, we propose SULO (*Simplified Upper-Level Ontology*), an upper level ontology that takes a minimalistic approach to guide the alignment, formalisation, and reusability of upper level and domain ontologies. SULO attempts to balance formal rigour with simplicity and practical usability. This is achieved by reducing the number of upper-level classes and relations to an absolute minimum while maintaining compatibility with existing ULOs, domain ontologies, and hybrids that fuse elements of terminologies, ontologies and schemas. We hypothesize that such a minimized ULO will foster adherence to intended semantics by domain specialists.

SULO aims to reconcile expressivity with usability addressing the following principles:

- **Minimalism**: SULO proposes a small taxonomy of disjoint classes and a minimal set of constrained relations to ensure broad applicability across domains.
- **Compatibility**: SULO maintains compatibility with core components of well-known ULOs while remaining accessible to domain experts. It does not claim to compete with established ULOs but claims to serve as a compatibility layer between established ULOs.
- **Accessibility**: SULO aims to be accessible to users with no or little training in formal ontology through friendly labeling, a simple taxonomy, and a general overview that fits in a single diagram.
- **Composability**: SULO will provide the building blocks to construct complex, machine-readable class expressions.
- **Interoperability**: SULO fosters interoperability by providing a common semantic foundation, including two ontology design patterns, that help domain experts adhere to explicit an implicit semantics.
- **Data validation**: SULO constrains real-world knowledge graphs through automated reasoning and schema validation.

These principles support a flexible framework for aligning and integrating biomedical data representations and standards, by offering a basic ordering principle for knowledge representation assets in a domain. It can be seen as a low-threshold entry into principled domain modeling, without requiring engagement – at least in initial iterations – with ontological and logical intricacies, such as those imposed by more fine-grained and expressive ULOs.

3. Methods

We followed the NEON methodology [20], consisting of i) domain analysis, ii) requirement gathering, iii) development of modular and pattern-based designs, iv) alignment with standards and existing ontologies, iv) iterative development and validation, and v) integration and maintenance. SULO development was initiated at a workshop with six participants, three of whom had strong ontology engineering expertise, three with domain expertise and specific use cases, two being ETL developers, and one being a high-level domain expert. The domain analysis was primarily drawn from the functional requirements in the AIDAVA project [21], which targets the rendering of healthcare processes as knowledge graphs rooted in semantic standards, and which currently uses the SPHN schema[9] to create personal health knowledge graphs from heterogeneous data sources, as well from challenging formalisation use cases drawn from SNOMED-CT. The workshop participants discussed key approaches adopted by existing ULOS, and set out to identify a minimal and accessible set of classes, relations and design patterns to cover key use cases (namely plans and processes around heart transplantation, fractured femurs, and in admission and discharge of patients in a hospital setting. Participants discussed at candidate classes and relations, mapped these to existing ULOs, and proposed possible simplifications that could form the basis of the SULO. The in-person workshop was followed by four additional online workshops for further iterative development, discussion, and validation involving over two dozen biomedical phenomena. The project

is on github[1], where syntax and consistency are automatically checked using Raptor and Robot with Hermit, and HTML documentation is generated using Ontospy [22] and PyLODE [23].

4. Results

4.1. SULO Overview

SULO is implemented as an OWL2 ontology using Protege. It comprises 17 classes, 18 object properties (9 inverse properties), and 1 data property. An overview of the classes and relations can be seen in Figure 1, and are further described in this section.

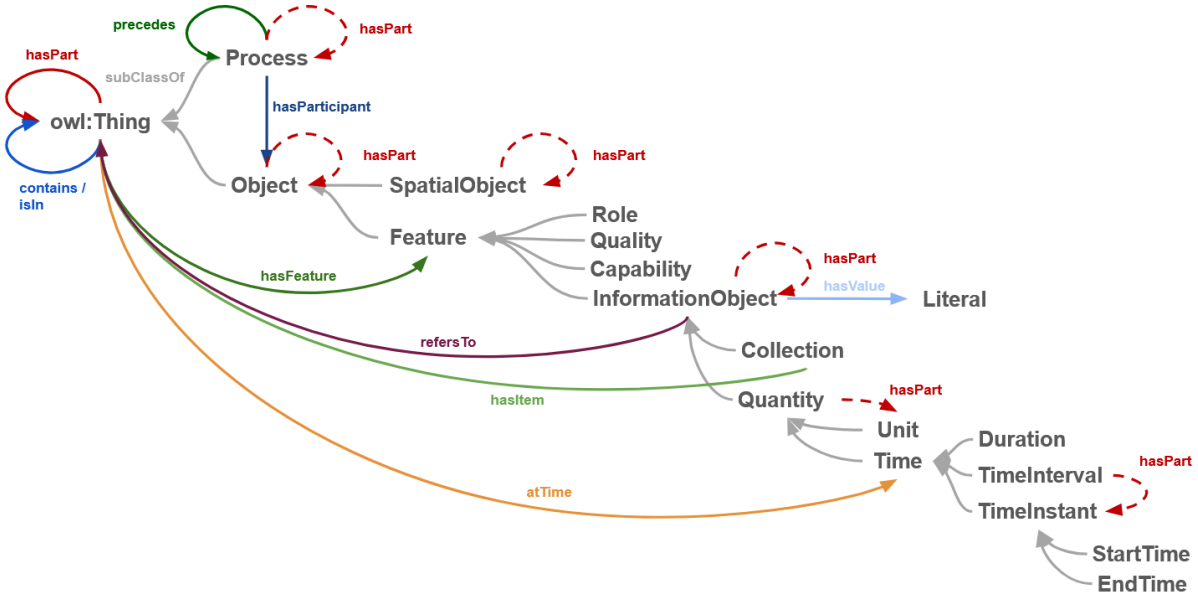

Figure 1: Graph representation of SULO classes and properties. Solid lines indicate domain and range of properties, while dotted lines indicate their use in class axioms.

4.2. SULO classes

SULO comprises seven taxonomic levels, which are mutually disjoint at each level (see Figure 2).

A *Process* has temporal parts, unfolds in time, has a duration and has objects as participants (*e.g. the life of an organism, a heart transplant, the administration of medication*). Processes that have temporal parts are fully determined when the last part of the process is finished.

An *Object* maintains its identity through time, does not have processes as its parts, but participates in processes. Objects are dichotomized into *SpatialObject* (e.g. a person, a particle, a paper document, a hospital) and *Feature*. A *SpatialObject* exists in space and can be characterised in one ore more dimensions of spacetime. They can gain and lose parts and their features can change over time. A *Feature* is an object that existentially depends on other things, whether objects or processes. The class *Feature* includes *Quality* (e.g. the redness of an apple), *Role* (e.g. the role of a care subject, a care provider, a student or a blood donor), *Capability* (e.g. the capability to breathe or to produce offspring), as well as *InformationObject* (e.g. the content of a document, a clinical practice guideline, a mathematical formula).

An *InformationObject* is a *Feature* whose existence depends on and is about something. Two subclasses of *InformationObject* are introduced, *viz. Collection* and *Quantity*. A *Collection* is an *InformationObject* pertaining to classes and mathematical sets, while a *Quantity* is an *InformationObject* that captures a numerical aspect (of an attribute) with a value and optional *Unit*, and is further divided into two

[1]https://github.com/AIDAVA-DEV/sulo; with the namespace https://w3id.org/sulo/ and an OWL/Turtle representation at https://w3id.org/sulo/sulo.ttl

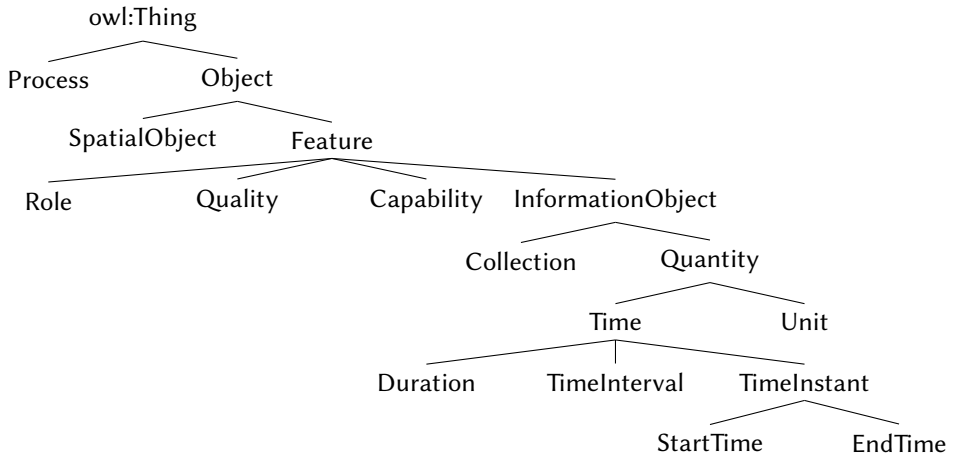

Figure 2: Taxonomy of SULO classes.

subclasses *Time* and *Unit*. In SULO, *Time* is a *Quantity* - it is a measurement of time (it does not offer an ontology of time). SULO's *Time* class is divided into 3 subtypes: a *TimeInstant*, *TimeInterval*, and *Duration* of time. *StartTime* and *EndTime* are role-playing *TimeInstant* that are useful in demarcating the boundaries of a *Process* or *TimeInterval*.

4.3. SULO relations

SULO offers 18 object properties (9 direct + 9 inverses) and one data property.

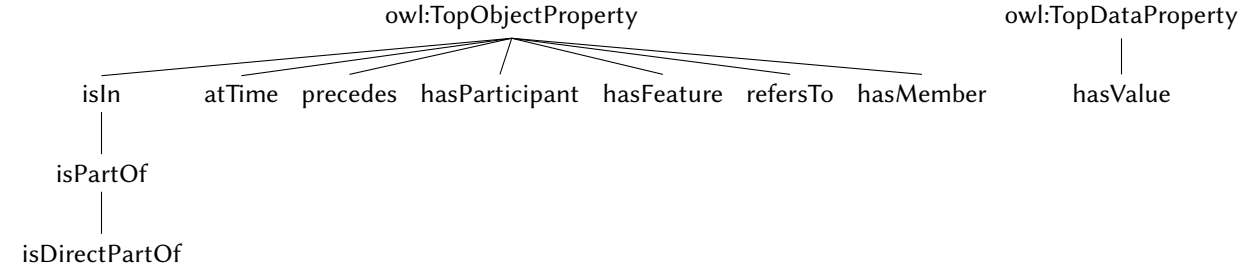

Figure 3: Taxonomy of SULO object and data properties.

4.3.1. Space and Time

SULO does not explicitly admit or refer to spatial or temporal regions, rather it focuses on associating entities with where they are located and at what time they exist. **`isIn`** (inverse: **`contains`**) is a *transitive* binary relation where the object includes the subject within its spatial, temporal, structural, or conceptual extent. The **`atTime`** (inverse: **`isTimeOf`**) relation holds between any entity and a time measurement. It captures the notion of when in time something exists or occurs. The **`precedes`** relation holds between two distinct processes, $p1$ and $p2$, indicating that $p1$ ends before $p2$ begins. It captures strict temporal ordering and sequential relationships between processes.

4.3.2. Parthood

The **`isPartOf`** (inverse: **`hasPart`**) relation is a *transitive* and *reflexive* relation expressing that entity x is incorporated within the entity y, where x contributes to the composition or structure of y in such a way that removing x from y would change the integrity or identity of y. The relation does not have a reference to time, and as such indicates the notion that x **`isPartOf`** y during some *Time t*. Examples

include *myLeftVentricle* `isPartOf` *myHeart*, *this administration of ibuprofen* `isPartOf` *treatment of my headache*, *this premise* `isPartOf` *that argument*. The `isDirectPartOf` (inverse: `hasDirectPart`) relation is a non-transitive subrelation of `isPartOf` to specify cardinality constraints in an OWL class expression. This relation is needed because OWL 2 does not allow cardinality constraints over transitive relations. `hasDirectPart` is the inverse relation of `isDirectPartOf`. An example class expression is *Human Heart* `subClassOf` *SpatialObject that* `hasDirectPart` *exactly 2 Ventricle*.

4.3.3. Descriptors

The `hasFeature` (inverse: `isFeatureOf`) relation holds between any entity and a *Feature*. For instance, a three-dimensional *SpatialObject* is associated with the *Quality* of *Volume*, which may be measured and reported as a *Quantity* `atTime` *Time*.

4.3.4. Participation in Processes

The `hasParticipant` (inverse: `isParticipantIn`) relation holds between a *Process* and an *Object*. We capture the notion that when a *Feature* participates in a *Process*, then so does the *Object* for which it is a *Feature* of through the role chain `hasParticipant` o `isFeatureOf` → `hasParticipant`. A key design pattern is to represent the different roles that objects play in a particular process. For instance, the *Process* of *Administration of Medication* might involve one individual playing a *Care Provider Role*, which administers the medication to another individual playing a *Care Subject Role*.

4.3.5. Relations for information objects

An *InformationObject* always `refersTo` (inverse: `isReferredToIn`) some other thing it mentions, describes, represents or otherwise is information about. The `hasValue` relation is a functional data property to capture a single literal associated with an *InformationObject*. No other data property is admitted. The `hasItem` (inverse: `isItemIn`) relation holds between a *Collection* and the items that constitute this collection. Finally, the physical representation of an *InformationObject* to a *SpatialObject* can be indicated with the `isFeatureOf` relation.

4.4. SULO Design Patterns

To foster semantic interoperability across domain-specific knowledge representations, SULO emphasizes a set of ontology design patterns (ODPs) towards structuring data graphs to a limited set of interoperable shapes. Upper level ontologies, domain ontologies, and schemas often include numerous specialized relations of the form "**hasRangeType**", such as **hasPatient**, **findingSite**, **hasTemperature** where the range is implicitly constrained by the target class. However, there are several problems with these. First is that they lead to a proliferation of relations which complicates the alignment between relations from different ontologies. Moreover, if the modeler wants to constrain the value range to something other than what is defined, they would create yet another relation and place a range constrain. Second, many such relations are just a range specialization of an existing property, and otherwise add no additional semantics. Third, such relations are simply shortcuts that bypass good modeling practice, namely the alignment of the data graph to that constrained by the upper level ontologies, thereby potentially introducing modeling errors that would otherwise be detectable through automated reasoning. We propose two design patterns that help domain modelers align their data to SULO ontology. The first design pattern, SOLID, focuses on data relations, while the second, PRO, focuses on roles that objects play in particular situations.

4.4.1. SOLID (Single Object Literal Information Datum) Design Pattern

The SOLID pattern uses SULO's single functional datatype property, `hasValue`, to assign a literal value to an instance of an *InformationObject*. This pattern transforms the introduction of domain-specific data

properties such as **hasTemperature** by first extracting the implied classes (e.g., *Temperature*) that pertain to a relevant *InformationObject*, and secondly finding the right relation to associate the information object to either a process or an object. Consider the application of the SOLID ODP represent the temperature of an individual (Figure 4). Instead of using arbitrary relations such as **hasTemperature** or **hasTemperatureInCelcius**, the design pattern reuses SULO's **hasValue** data property, **hasFeature**, **refersTo** and **hasPart** object properties in conjunction with two externally defined classes, namely *Temperature* from PATO and *Celcius* from the Unit Ontology. This now becomes a proper SULO compatible description that allows for further statements (e.g. that the temperature measurement was performed using a particular device, in a particular place and time, etc). Thus, SOLID pushes the semantics out of data properties into instances of relevant classes, facilitating more elaborate descriptions now, or in the future. Moreover, the implementation of the pattern ensures a predictable location for where data values are stored, allowing for predictable querying of the **hasValue** data property.

```
@prefix sulo:   <http://w3id.org/sulo/> .
@prefix uo: <http://purl.obolibrary.org/obo/> .
@prefix pato: <http://purl.obolibrary.org/obo/>
@prefix : <http://example.org/> .

:alice a sulo:SpatialObject, :Person;
    sulo:hasFeature :alice_temperature_measurement_1 .

:alice_temperature_measurement_1 a sulo:Quantity;
    sulo:hasValue "37.8"^^xsd:double ;
    sulo:refersTo [ a pato:PATO_0000146] ; # the quality of temperature
    sulo:hasPart [ a uo:UO_0000027 ] . # the celcius unit
```

Figure 4: Using the SOLID pattern to represent Alice's temperature of 37.8C

4.4.2. PRO (Process-Role-Object) Design Pattern

The Process-Role-Object (PRO) ODP provides a modular way to represent how (non-process) entities participate in processes through their specific roles. The pattern, expressed as a Manchester Syntax[24] class expression in Figure 5 requires that particular roles are directly linked to a process instance using **hasParticipant** and to their respective role holders using **isFeatureOf**:

```
EquivalentTo:
  Process and
      hasParticipant some (
          Role and isFeatureOF some Object
      )
```

Figure 5: Process-Role-Object Ontology Design Pattern

We recover the participation of the role holder by inference through a role chain (Figure 6):

```
hasParticipant o isFeatureOf -> hasParticipant
```

Figure 6: PRO Role Chain for Inference

This pattern is key to discriminating the roles or functions of multiple, interacting entities in the same process. Consider a scenario in which a care provider (Dr. Smith) attends to a subject of care (Alice) to discuss Alice's condition (Listing 7). Applying the PRO pattern, the health care encounter (*encounter* is a *Process*) in which the two instances of roles (*alice_subject_of_care_role*, *smith_physician_role*) instantiating *'patient role'* and *'health care provider role'* are added as specific participants. The two roles are related to their role holders, alice and drsmith, respectively. Reasoning over this graph using the role chain, we obtain two additional triples, *encounter* has *alice* and *drsmith* as participants.

```
@prefix sulo: <http://w3id.org/sulo/> .
@prefix obo: <http://purl.obolibrary.org/obo/>
```

```
@prefix : <http://example.org/> .

:encounter a  sulo:Process, obo:OGMS_0000097 . # health care encounter
    sulo:hasParticipant :alice_subject_of_care_role, :smith_care_provider_role .

:alice_subject_of_care_role a obo:OMRSE_00000011; # patient role
    sulo:isFeatureOf :alice .

:smith_care_provider_role a obo:OMRSE_00000012; # health care provider role
    sulo:isFeatureOf :drsmith .

:alice a sulo:SpatialObject, taxon:NCBITaxon_9606 . # human
:drsmith a sulo:SpatialObject, taxon:NCBITaxon_9606 . # human

### inference from PRO role chain
:visit_1 sulo:hasParticipant :alice, :drsmith .
```

Figure 7: Inference on exemplar data graph using PRO design pattern

Thus, the PRO pattern combined with the PRO role chain allows a clear mechanism by which dynamic, context-dependent roles of objects can be described in relevant processes, which retain their association to facilitate downstream query answering or data validation. The pattern also removes the need for role-based relations such as `hasPatient`, `hasCareProvider`.

4.5. Application of SULO

4.5.1. SULO with SNOMED CT

The use of upper-level ontologies with domain terminologies such as SNOMED CT is increasingly recognized as essential for achieving semantic interoperability in healthcare and biomedical research. In [25], an alignment of SNOMED CT with the upper-level ontology BioTopLite2 (BTL2) [26] was proposed and an alignment of SNOMED CT findings and disorders, as well as the SNOMED relations `findingSite` and `RoleGroup` were proposed in [18]. Based on that, SNOMED CT upper-level classes and relations have been aligned with SULO. Table 1 show the mapping of the main SNOMED CT classes and relations. It was done manually, by analysing the meaning of the candidate classes and relations, considering the formal axioms as well as text definitions and hierarchical context.

Table 1
Mappings between SNOMED CT Top Concepts and Relations and corresponding SULO Classes and Properties

SNOMED CT Top Concept	SULO Class	SNOMED CT Relation	SULO Object Property
Clinical finding (finding)	sulo:Process	Occurrence (attribute)	sulo:atTime
Procedure (procedure)		Finding site (attribute)	sulo:isIn
Action (qualifier value)		Procedure site - Direct (attribute)	
Organism (organism)	sulo:SpatialObject	Using device (attribute)	sulo:hasParticipant
Specimen (specimen)		Causative agent (attribute)	
Substance (substance)		Due to (attribute)	
Pharmaceutical / biologic product (product)		Procedure device (attribute)	
Environment or geographical location (environment/location)		Laterality (attribute)	sulo:hasFeature
Physical object (physical object)		Associated morphology (attribute)	
Body structure (body structure)		Has basic dose form (attribute)	
Observable entity (observable entity)	sulo:InformationObject	Access (attribute)	
Record artifact (record artifact)		Component (attribute)	
Situation with explicit context (situation)		Has active ingredient (attribute)	sulo:hasPart
Staging and scales (staging scale)		Specimen substance (attribute)	
Physical force (physical force)	sulo:Feature	Method (attribute)	
Time (property) (qualifier value)	sulo:Time	Role Group (attribute)	
Unit of measure (qualifier value)	sulo:Unit		

SNOMED Ct is expressible in OWL-EL. Classes are defined using an `is-a` hierarchy of primitives and attribute-value axioms, for which equivalence can be stated for new, fully defined classes. Attributes always correspond to OWL object properties and values to existentially qualified expressions. SNOMED CT class expressions may be formalised against SULO using SULO classes, relations, design patterns,

and SULO-SNOMED mappings together. All SNOMED concepts with the label *x* (disorder) have the implicit meaning that part of a patient's life include the disorder *x* as per [18]. Consider *Fracture of femur (disorder)* shown for SNOMED in Figure 8, and transformed to a SULO-based representation in Figure 9. A disorder in SNOMED CT is represented as a *Process* in SULO which is located in a bone structure of femur that has the morphologic abnormality of being fractured.

```
Class: 'Fracture of femur (disorder)'

  EquivalentTo:
    'Disease (disorder)'
    and 'Role group (attribute)' some (
        'Associated morphology (attribute)' some 'Fracture (morphologic abnormality)'
        and
        'Finding site (attribute)' some 'Bone structure of femur (body structure)'
    )
```

Figure 8: SNOMED class definition for Fracture of Femur (disorder)

```
Class: 'Fracture of femur (disorder)'

    EquivalentTo:
    'Disease (disorder)'
        and sulo:hasPart some (
            (sulo:Process and sulo:isIn some
            'Bone structure of femur (body structure)'
                and sulo:hasFeature some 'Fracture (morphologic abnormality)')
        )
```

Figure 9: SULO representation of SNOMED Fracture of femur

4.5.2. SULO with SPHN

The Swiss Personalized Health Network (SPHN) initiative developed an RDF schema to share health-related data between health data providers [9]. While schema are valuable in constraining the shape and content of data, the SPHN schema suffers from several problems including: extensive use of non-reusable specialized predicates that necessitates class-specific lookups and incompatible changes in schema classes and relations across subsequent versions that add burden to implementers. We posit that adoption of SULO as a foundation ontology would increase ease of use.

Let us first consider the schema for the class *Administrative Case* (see Table 2). *AdministrativeCase* aims to capture a set of information relating to the admission, care and discharge of a patient during a contiguous patient encounter. The class includes relations such as **hasAdmission** and **hasIdentifier**, pointing to instances of the class *Admission* and the datatype *xsd:string*, respectively. However, the SPHN schema for *AdministrativeCase* has been revised over time, changing both the names of the relations as well as the set of classes. *AdministrativeCase* in the 2023.2 version of SPHN was defined by a drastically different schema. In the 2023 version, the admission datetime and location, as well as the discharge location and datetime were specified as part of the *AdministrativeCase*. The new version introduces two new classes, namely *Admission* and Discharge (see Table 3), are related by **hasAdmission** and **hasDischarge**, respectively. While both classes reuse a **hasDatetime** relation to capture the time of the event, they use distinct relations (**hasOriginLocation**, **hasTargetLocation** to indicate a location. Confusingly, **hasOriginLocation** specifies the physical location from where the individual came from (prior to admission), while **hasTargetLocation** indicates the location that the individual supposed to go after discharge. While one might want to specify the actual location (e.g. departments in the care facility) of the admission or discharge, the schema lacks the capability to allow this.

The application of SULO and its ODPs can better inform schema development so as to maximize modularity, minimize complexity, and make enhance backwards compatibility of new schemas. Let us fully consider the *AdministrativeCase* using SULO semantics, captured as a parameterized class

Table 2
Comparison of SPHN Administrative Case Schema (v.2023.2 vs v.2025.1)

2023.2	2025.1	Property	Cardinality	Range
☒	☒	hasIdentifier	1..1	xsd:string
☒	☒	hasSubjectPseudoIdentifier	1..1	SubjectPseudoIdentifier
☒	☒	hasCareHandling	0..1	CareHandling
-	☒	hasAdmission	1..1	Admission
☒	-	hasAdmissionDatetime	1..1	xsd:dateTime
☒	-	hasAdmissionLocation	0..1	Location
-	☒	hasDischarge	0..1	Discharge
☒	-	hasDischargeDatetime	0..1	xsd:dateTime
☒	-	hasDischargeLocation	0..1	Location
☒	-	hasDataProviderInstitute	1..1	DataProviderInstitute
-	☒	hasSourceSystem	1..*	SourceSystem

Table 3
Schema definition for SPHN (v.2025.1) Admission and Discharge

2023.2	2025.1	Class	Property	Cardinality	Range
–	☒	Admission	hasOriginLocation	0 .. 1	Location
–	☒		hasDatetime	1 .. 1	xsd:dateTime
–	☒	Discharge	hasTargetLocation	0 .. 1	Location
–	☒		hasDatetime	1 .. 1	xsd:dateTime

expression in Figure 10, where the '?var' indicates the variable to be filled from query on the schema. `hasSubjectPseudoIdentifier` specifies a pseudoanonimized patient identifier. In SULO, a patient pseudoidentifier is a kind of *InformationObject*, which `isFeatureOf` a person (a kind of *SpatialObject*) that plays the *SubjectOfCareRole* in an *AdministrativeCase*. In the earlier formulation of SPHN, the properties of `hasAdmissionDateTime` and `hasDischargeDateTime` indicate the time instants at which the admission and discharge processes were recorded in the information system, rather than being an accurate description of either the start or end of those processes. If we assume that these time instants are contained within the actual admission and discharge processes, the SULO representation is that the administrative case has an admission process and a discharge process as its parts, and each part is associated with distinct time instants and values. However, given the semantics of `hasOriginLocation` and `hasTargetLocation`, we treat these as separate pre-admission and post-discharge processes that are located in the respective sites. While the schema for *AdministrativeCase* has undergone major changes, we can faithfully represent the intended knowledge by reusing SULO relations and ODPs.

4.5.3. SULO vs BFO

To validate SULO's alignment with established ULOs and ensure broader interoperability, we performed an explicit mapping to BFO 2.0. BFO is widely adopted in biomedical and scientific communities [1], offering a foundational framework for rigorous ontological modeling. We manually mapped the classes and relations between SULO and BFO through careful conceptual analysis, matching each SULO class and relation to the most semantically appropriate BFO counterpart (omitted owing to space constraints).

Some mappings revealed noteworthy differences. For instance, `bfo:existsAt` links an *occurrent* to a *temporal region* (which is also an occurrent), whereas `sulo:atTime` relates any entity, including *continuants*, to `sulo:Time`, which is modeled as a *generically dependent continuant*. This demonstrates an difference in the treatment of temporal concepts. Additionally, BFO (v1/v2) does not explicitly support linking information artifacts to universals. In contrast, the Information Artifact Ontology (IAO), which many BFO-based ontologies adopt, permits `iao:isAbout` to refer to both individuals and classes. SULO's `hasFeature` relation, intended as a structural link between an entity and its features (e.g., quality, role), is loosely aligned with `iao:isAbout`. However, `iao:isAbout` implies an epistemic or representational relationship, leading to a semantic mismatch. This distinction underscores

```
Class: 'Administrative Case'

    subClassOf:
    sulo:Process
        and sulo:hasParticipant some (
            'Subject of Care Role'
                and sulo:isFeatureOf some (
                    Person
                    and sulo:hasFeature some (
                        SubjectPseudoIdentifier
                        and sulo:hasValue value xsd:string # subjectpseudoidentifier
                    )
                )
            )
        and sulo:hasPart some (
            PreAdmission
            and sulo:isIn some SpatialObject # sourceLocation
        )
        and sulo:hasPart some (
            Admission
            and sulo:atTime some (
                TimeInstant
                and hasValue value xsd:dateTime # admission datetime
        )
        and sulo:hasPart some CareHandling
        and sulo:hasPart some (
            Discharge
            and sulo:atTime some (
                TimeInstant
                and hasValue value xsd:dateTime # discharge datetime
        )
        and sulo:hasPart some (
            PostDischarge
            and sulo:isIn some SpatialObject # targetLocation
        )
        and sulo:isReferredToIn some InformationObject # sourceSystem
```

Figure 10: Class Expression for a SULO-compliant representation of SPHN class Administrative Case

a broader challenge in harmonizing ontological and epistemological roles of predicates. Finally, SULO features may be features of everything, while BFO dependent continuants exclude processes. Thus, BFO precludes the definition of qualities of processes. BFO:Quality is therefore strictly limited to non-processes, whereas SULO does not impose any restriction. This mapping exercise demonstrates SULO's capability to function as an intermediary layer, preserving compatibility with BFO-based ontologies while simplifying conceptual complexity.

4.5.4. SULO vs SIO

SULO and SIO share a very high degree of similarity in the set of classes and relations as well as the taxonomic organisation of the ontology. The *core* [2] version of SIO comprises 15 classes, 34 object properties, and one data property. SULO and SIO (core) share a large set of common classes including *Process*, *Object*, *Feature* maps to SIO's *attribute*, *Quality*, *Capability*, *Role*, *InformationObject* maps to SIO's *information content entity*, *SpatialObject* maps to SIO's *material entity*. The top level of the ontologies is structured differently. *entity* is at the root of the SIO ontology, which then specialises into three subclasses *process*, *object*, and *attribute*. In contrast, SULO's *Process* and *Object* classes are at the top, and its *Feature* class is a subclass of *Object*. As such, the ontological commitment is that dependent entities (attributes in SIO) are disjoint from objects and processes, and this is incompatible with SULO's representation where dependent entities (features in SULO) are types of objects. Therefore, while the is a correspondance between classes, a strict mapping between these ontologies would yield an inconsistent ontology. SULO and SIO share set of core object properties including **hasParticipant**,

[2] http://semanticscience.org/ontology/sio/v1.59/sio-subset-core.owl

`isPartOf`, `refersTo`, `precedes`, `isIn`, whereas `atTime` maps to SIO's `exists at`, `hasFeature` maps to `has attribute`, and `hasMember` maps to `has data item`. However SIO has a more expansive set of relations that are role specializations that SULO rejects (e.g. `has input`, `has output`, `has agent`) as well as spatial, temporal and information-based relations that provide additional expressivity in the formalisation of domains (e.g. `realizes`, `is variant of`, `is connected to`, `in relation to`. Finally, both ontologies share the `hasValue` data property, and support the modeling advocated by the SOLID design pattern.

5. Discussion

SULO takes a minimalistic approach to guide the alignment, formalisation, and reusability of upper-level and domain ontologies, aiming to balance formal rigour with simplicity and practical usability. SULO's core principles are central to its design and intended impact:

Minimalism: SULO proposes a small taxonomy of 17 disjoint classes, a minimal set of 18 object properties (9 direct + 9 inverse), and 1 data property to ensure broad applicability across domains. This minimal set covers the representational needs in the biomedical use cases examined.

Compatibility: SULO aims to maintain compatibility with core components of well-known ULOs (e.g., BFO, SIO, UFO, GFO, DOLCE, YAMATO, BioTop) while remaining accessible to domain experts. Owing to its minimalist approach, SULO serves as a compatibility layer between ULOs by focusing on key aspects of ontology, for which more specialised theories may be found in other ULOs.

Accessibility: SULO strives to be accessible to users with little or no training in formal ontology through friendly labeling, a simple taxonomy, and a general overview that fits in a single diagram. This approach aims to provide a low-threshold entry into principled domain modeling, avoiding the ontological and logical intricacies of more fine-grained ULOs. The widely adopted Ontology Development 101 guide [27] with the well-known Pizza and Wine ontologies has shaped ontology education over decades based on frame-based modeling, which neglected key concerns central to applied ontology, *viz.* ontological clarity, interoperability, and alignment with upper-level ontologies [28]. Future ontology education must move beyond illustrative examples and tool-centric instruction, towards ontological rigor, modularity, and principled alignment with SULO.

Composability: SULO offers a set of building blocks, namely classes and object and data properties, with which more complex class expressions can be formulated. In particular, we show how select complex class expressions in SNOMED CT can be rewritten as SULO expressions without introducing additional relations. Similarly, SULO is sufficiently expressive to capture the semantics of different version of the SPHN schema. SULO is available as an OWL-DL ontology, which is amenable to automated reasoning in order to check the satisfiability of expert-composed class expressions.

Interoperability: SULO's effectiveness as a interoperability hub is significantly enhanced the adoption of ontology design patterns (ODPs). We introduces two such patterns - SOLID and PRO - which aim to reduce the proliferation of domain-specific role-based relations. The ODPs are simultaneously useful to map relations found in the SNOMED ontology and the SPHN schema to SULO-compatible representations. However, the adoption of such patterns may require the introduction of new classes that currently don't exist. The formalisation of additional ontology design patterns [29] grounded to SULO may improve their uptake in biomedical ontologies [30]. Our analysis of the SPHN schema illustrates how such schemas can be formulated without referring to specific domain relations, and that adopting a set of common design patterns maximizes modularity and enhances backwards compatibility.

Application-Driven Validation: Data validation is a key part of modern information systems. However, many schemas are constructed without considering the semantic foundation that ontologies provide. This yields schemas that closely mirror their data, and are otherwise difficult to reuse or extend for similar dataset. While much if not all of SPHN can be mapped to SULO, it remains future work to show how a SULO-based SPHN schema could bring significant benefits to the construction and maintenance of evolving schemas for data-validation.

6. Conclusion and Future Work

SULO was developed to address the significant challenges of ontology-based data integration in healthcare and life sciences, where the complexity of existing upper-level ontologies (ULOs) often limits the extent and correctness of adoption by domain experts and standards developers. Despite the recognised need for interoperable representations, many domain ontologies and schemas are created without reference to a foundation ontology. SULO takes a minimalistic approach that balances formal rigour with simplicity and practical usability. Future work will explore more complete mappings to well known ULOs and to apply SULO to a broader set of use cases in biomedicine and healthcare, as well as to other disciplines to uncover limitations and potential opportunities for further refinement.

Declaration on Generative AI

During the preparation of this work, the author(s) used GPT-4 to help with LaTeX styling. The authors reviewed and edited the content as needed and take full responsibility for the publication's content.

Funding

This work is supported by the Horizon Europe Framework Program under Grant Agreement Nos. 101057062 (AIDAVA), 101112022 (iCare4CVD), 101095435 (REALM), 101057603 (RES-Q+), 101080875 (CMC), and the Spanish Research Agency Grant RYC2020-030190-I.

References

[1] J. N. Otte, J. Beverley, A. Ruttenberg, BFO: Basic Formal Ontology, Applied Ontology 17 (2022) 17–43.

[2] G. Guizzardi, A. B. Benevides, C. M. Fonseca, D. Porello, J. P. A. Almeida, T. P. Sales, UFO: Unified Foundational Ontology, Applied Ontology 17 (2022) 167–210.

[3] F. Loebe, P. Burek, H. Herre, Gfo: The general formal ontology, Applied Ontology 17 (2022) 71–106.

[4] S. Borgo, R. Ferrario, A. Gangemi, N. Guarino, C. Masolo, D. Porello, E. M. Sanfilippo, L. Vieu, Dolce: A descriptive ontology for linguistic and cognitive engineering1, Applied Ontology 17 (2022) 45–69.

[5] R. Mizoguchi, S. Borgo, YAMATO: Yet-another more advanced top-level ontology, Applied Ontology 17 (2022) 211–232.

[6] M. Dumontier, C. J. Baker, J. Baran, A. Callahan, L. Chepelev, J. Cruz-Toledo, N. R. Del Rio, G. Duck, L. I. Furlong, N. Keath, et al., The Semanticscience Integrated Ontology (SIO) for biomedical research and knowledge discovery, Journal of biomedical semantics 5 (2014) 1–11.

[7] S. Schulz, M. Boeker, C. Martinez-Costa, The BioTop family of upper-level ontological resources for biomedicine, Studies in health technology and informatics 235 (2017) 441–445.

[8] C. Trojahn, R. Vieira, D. Schmidt, A. Pease, G. Guizzardi, Foundational ontologies meet ontology matching: A survey, Semantic Web 13 (2022) 685–704. URL: https://doi.org/10.3233/SW-210447. doi:10.3233/SW-210447.

[9] V. Touré, P. Krauss, K. Gnodtke, J. Buchhorn, D. Unni, P. Horki, J. L. Raisaro, K. Kalt, D. Teixeira, K. Crameri, et al., Fairification of health-related data using semantic web technologies in the swiss personalized health network, Scientific Data 10 (2023) 127.

[10] R. Kaliyaperumal, M. D. Wilkinson, P. A. Moreno, N. Benis, R. Cornet, B. dos Santos Vieira, M. Dumontier, C. H. Bernabé, A. Jacobsen, C. M. A. Le Cornec, M. P. Godoy, N. Queralt-Rosinach, L. J. Schultze Kool, M. A. Swertz, P. van Damme, K. J. van der Velde, N. Lalout, S. Zhang, M. Roos, Semantic modelling of common data elements for rare disease registries, and a prototype workflow for their deployment over registry data, Journal of Biomedical Semantics 13 (2022).

[11] SNOMED International, SNOMED Concept Model, 2025. URL: https://confluence.ihtsdotools.org/display/DOCGLOSS/SNOMED+CT+concept+model.
[12] N. Sioutos, S. de Coronado, M. W. Haber, F. W. Hartel, W.-L. Shaiu, L. W. Wright, NCI Thesaurus: a semantic model integrating cancer-related clinical and molecular information, Journal of biomedical informatics 40 (2007) 30–43.
[13] A. T. McCray, S. J. Nelson, The representation of meaning in the UMLS, Methods of information in medicine 34 (1995) 193–201.
[14] T. Benson, G. Grieve, The HL7 v3 RIM, Springer International Publishing, Cham, 2016, p. 243–264.
[15] R. Stevens, P. Lord, J. Malone, N. Matentzoglu, Measuring expert performance at manually classifying domain entities under upper ontology classes, Journal of Web Semantics 57 (2019) 100469. URL: https://www.sciencedirect.com/science/article/pii/S157082681830043X. doi:https://doi.org/10.1016/j.websem.2018.08.004.
[16] M. Dumontier, R. Hoehndorf, Realism for scientific ontologies, in: Proceedings of the 2010 Conference on Formal Ontology in Information Systems: Proceedings of the Sixth International Conference (FOIS 2010), IOS Press, NLD, 2010, p. 387–399.
[17] P. Lord, R. Stevens, Adding a little reality to building ontologies for biology, PLoS ONE 5 (2010) e12258. URL: http://dx.doi.org/10.1371/journal.pone.0012258. doi:10.1371/journal.pone.0012258.
[18] S. Schulz, J. T. Case, P. Hendler, D. Karlsson, M. Lawley, R. Cornet, R. Hausam, H. Solbrig, K. Nashar, C. Martínez-Costa, Y. Gao, SNOMED CT and Basic Formal Ontology – convergence or contradiction between standards? the case of "clinical finding", Applied ontology 18 (2023) 207–237.
[19] C. Ochs, Z. He, L. Zheng, J. Geller, Y. Perl, G. Hripcsak, M. A. Musen, Utilizing a structural meta-ontology for family-based quality assurance of the bioportal ontologies, Journal of Biomedical Informatics 61 (2016) 63–76.
[20] M. C. Suárez-Figueroa, A. Gómez-Pérez, M. Fernández-López, The NeOn methodology for ontology engineering, in: Ontology engineering in a networked world, Springer, 2011, pp. 9–34.
[21] I. de Zegher, K. Norak, D. Steiger, H. Müller, D. Kalra, B. Scheenstra, I. Cina, S. Schulz, K. Uma, P. Kalendralis, E.-M. Lotman, M. Benedikt, M. Dumontier, R. Celebi, Artificial intelligence based data curation: enabling a patient-centric european health data space, Frontiers in Medicine 11 (2024). URL: http://dx.doi.org/10.3389/fmed.2024.1365501. doi:10.3389/fmed.2024.1365501.
[22] lambdamusic, Ontospy, 2025. URL: https://github.com/lambdamusic/ontosPy.
[23] N. Car, pylode, 2025. URL: https://github.com/RDFLib/pyLODE.
[24] M. Horridge, P. F. Patel-Schneider, OWL 2 web ontology language Manchester syntax (second edition), 2021. URL: https://www.w3.org/TR/owl2-manchester-syntax/.
[25] S. Schulz, C. Martínez-Costa, Harmonizing SNOMED CT with BioTopLite: an exercise in principled ontology alignment, in: MEDINFO 2015: eHealth-enabled Health, IOS Press, 2015, pp. 832–836.
[26] S. Schulz, M. Boeker, BioTopLite: An upper level ontology for the life sciences. evolution, design and application, in: INFORMATIK 2013–Informatik angepasst an Mensch, Organisation und Umwelt, Gesellschaft für Informatik eV, 2013, pp. 1889–1899.
[27] N. Noy, D. McGuinness, et al., Ontology development 101: A guide to creating your first ontology, 2001. URL: http://www.ksl.stanford.edu/people/dlm/papers/ontology-tutorial-noy-mcguinness-abstract.html.
[28] S. Schulz, M. Boeker, J. A. V. Ramos, L. Jansen, Pizza & wine: The need for educational tools for foundational ontologies, in: S. Borgo, O. Kutz, F. Loebe, F. Neuhaus (Eds.), Proceedings of the Joint Ontology Workshops 2017, volume 2050 of *CEUR Workshop Proceedings*, CEUR, 2017. URL: https://ceur-ws.org/Vol-2050/#foust-paper9, ISSN: 1613-0073.
[29] A. Gangemi, V. Presutti, Ontology Design Patterns, Springer Berlin Heidelberg, 2009, p. 221–243. URL: http://dx.doi.org/10.1007/978-3-540-92673-3_10. doi:10.1007/978-3-540-92673-3_10.
[30] C. Ochs, Y. Perl, J. Geller, S. Arabandi, T. Tudorache, M. A. Musen, An empirical analysis of ontology reuse in bioportal, Journal of Biomedical Informatics 71 (2017) 165–177. URL: http://dx.doi.org/10.1016/j.jbi.2017.05.021. doi:10.1016/j.jbi.2017.05.021.