Initiative for developing eProcurement Ontology

eProcurement ontology architecture and formalisation specifications

Deliverable WP 1.1

Eugeniu Costetchi

17 April 2020

Disclaimer

Project acronym	ePO
Project title	Initiative for developing eProcurement Ontology
Document reference	eProcurement ontology architecture and formalisation specifications
Author(s)	Eugeniu Costetchi
Editor(s)	Eugeniu Costetchi
Contractor	Infeurope S.A.
Framework contract	10688/35368
Actual delivery date	17 April 2020
Delivery nature	Report (R)
Dissemination licence	Creative Commons Attribution 4.0 International
Filename	wp1-1-ontology-architecture
Suggested readers	project partners, future users, legal practitioners, software architects

Abstract

TBD

Contents

1 Introduction

This document provides a working definition of what is the architectural stance and the design decisions that shall be adopted for the eProcurement ontology along with the description of how to generate it.

1.1 Background considerations

Public procurement is undergoing a digital transformation. The EU supports the process of rethinking public procurement process with digital technologies in mind. This goes beyond simply moving to electronic tools; it rethinks various pre-award and post-award phases. The aim is to make them simpler for businesses to participate in and for the public sector to manage. It also allows for the integration of data-based approaches at various stages of the procurement process.

Digital procurement is deeply linked to eGovernment. It is one of the key drivers toward the implementation of the “once-only principle” in public administrations – a cornerstone of the EU’s Digital Single Market strategy. In the age of big data, digital procurement is also crucial in enabling governments to make data-driven decisions about public spending.

With digital tools, public spending should become more transparent, evidence-oriented, optimised, streamlined and integrated with market conditions. This puts eProcurement at the heart of other changes introduced to public procurement in new EU directives.

PSI directive [12] across the EU is calling for open, unobstructed access to public data in order to improve transparency and to boost innovation via the reuse of public data. Procurement data has been identified as data with a high-reuse potential [2]. Therefore, making this data available in machine-readable formats, following the data as a service paradigm, is required in order to maximise its reuse.

Given the increasing importance of data standards for eProcurement, a number of initiatives driven by the public sector, the industry and academia have been kick started in the recent years. Some have grown organically, while others are the result of standardisation work. The vocabularies and the semantics that they are introducing, the phases of public procurement that they are covering, and the technologies that they are using all differ. These differences hamper data interoperability and thus its reuse by them or by the wider public. This creates the need for a common data standard for publishing public procurement data, hence allowing data from different sources to be easily accessed and linked, and consequently reused.

In this context, the Publications Office of the European Union aims to develop an eProcurement ontology.

The objective of the eProcurement ontology is to act as this common standard on the conceptual level, based on consensus of the main stakeholders and designed to encompass the major requirements of the eProcurement process in conformance with the Directives and Regulations [13, 14, 15, 16].

1.2 Target audience

The target audience of the eProcurement ontology, defined in [25], comprises the following groups of stakeholders:

- Contracting authorities and entities, i.e. buyers, such as public administrations in the EU Member States or EU institutions;
- Economic operators, i.e. suppliers of goods and services such as businesses, entrepreneurs and financial institutions;
- Academia and researchers;
- Media and journalists;
- Auditors and regulators;
- Members of parliaments at regional, national and EU level;
- Standardisation organisations;
- NGOs; and
- Citizens [25].

1.3 Context and scope

In the past years much effort was invested into the eProcurement ontology initiative, including definition of requirements, provision of general specifications, identification of the main use cases, and laborious development of a preliminary shared conceptual model expressed using Unified Modelling Language (UML) [4, 6].

The general methodology for developing the eProcurement ontology is described in [11, 3–15]. It describes a process comprising the following steps:

1. Define use cases
2. Define the requirements for the use cases
3. Develop a conceptual data model
4. Consider reusing existing ontologies
5. Define and implement an OWL ontology

The ultimate objective of the eProcurement ontology project is to put forth a commonly agreed OWL ontology that will conceptualise, formally encode and make available in an open, structured and machine-readable format data about public procurement, covering end-to-end procurement, i.e. from notification, through tendering to awarding, ordering, invoicing and payment [25].

Work so far has concentrated on the conceptual modelling of the eNotification phase, taking into consideration the needs of other phases. The UML conceptual model has been created with the forthcoming procurement standard forms (eForms) in mind; the model has not been mapped to the current standard forms.

In the 2020 ISA^2 work programme a new project has been set up to analyse existing procurement data through the lens of the newly developed conceptual model. This means that the conceptual model needs to be transposed into a formal ontology and a subset of the existing eProcurement data (initially only the notification phase is considered) must be transformed into RDF format such that they instantiate the eProcurement ontology and are conform to a set of predefined data shapes.

Working under the assumption that Steps 1–4 have been completed, the current efforts channel on designing, implementing and executing the necessary tasks in order to accomplish Step 5 from the above process.

Once the formal ontology is created and the XML data is transformed into RDF representation, the data can be queried in order to validate the suitability to satisfy the business use cases defined in [11, Sec. 3].

This document comprises of architectural specification and implementation guidelines that shall be taken into consideration when developing the formal ontology. Other related artefacts (i.e. documents, scripts and datasets) are presented in Section 3, where is described, in detail, the process for accomplishing the generation the formal eProcurement ontology, transformation of XML data and the ontology validation.

There is a number of aspects that are excluded from the scope of this project stage:

- Change management and maintenance of the ontology content.
- Content authoring and conceptual design of the domain model.
- Practical implementation of systems that implement the ontology.

Currently in scope are the following items:

- designing an ontology architecture (this document),
- create guidelines and conventions for the UML conceptual model [7],
- develop a set of transformation scripts from the UML model into a formal ontology
- implement a set of scripts to transform the existing XML eProcurement data into RDF format,
- put forward a method to validate the generated formal ontology using the current eProcurement data.

1.4 Key words for Requirement Statements

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [5].

The key words "MUST (BUT WE KNOW YOU WON'T)", "SHOULD CONSIDER", "REALLY SHOULD NOT", "OUGHT TO", "WOULD PROBABLY", "MAY WISH TO", "COULD", "POSSIBLE", and "MIGHT" in this document are to be interpreted as described in RFC 6919 [34].

2 Requirements

2.1 Functional requirements

This section provides the main functionalities and use cases that the ontology should support. These requirements are derived from the use cases identified in the report on policy support for eProcurement [25] and outlined in the in the eProcurement project chapter proposal [10].

1. *Transparency and monitoring*: to enable verification that public procurement is conducted according to the rules set by the Directives and Regulation [13, 14, 15, 16].
   (a) Public understandability
   (b) Data Journalism
   (c) Monitor the money flow
   (d) Detect fraud and compliance with procurement criteria
   (e) Audit procurement process
   (f) Cross-validate data from different parts of the procurement process
2. *Innovation & value added services*: to allow the emergence of new applications and services on the basis of the availability of procurement data.
   (a) Automated matchmaking of procured services and products with
   (b) businesses
   (c) Automated validation of procurement criteria
   (d) Alerting services
   (e) Data analytics on public procurement data
3. *Interconnection of public procurement systems*: to support increased interoperability across procurement systems.
   (a) Increase cross-domain interoperability among Member States
   (b) Introduce automated classification systems in public procurement systems

2.2 Non-functional requirements

This section provides the characteristics, qualities and general aspects that the eProcurement ontology should satisfy.

- The practices, technologies and standards must be aligned with the European Directive on open data and the reuse of public sector information [17] and European Publications Office standards and practices.
- The terminology used in the ontology should be reused from established core vocabularies [33] and domain ontologies as long as their meaning fits into the description of the eProcurement domain.
- The concept and relation labels must be multilingual covering at least the official European Languages [32].
- The formal ontology, and the related artefacts, must be generated from the eProcurement UML conceptual model, serving as the single source of truth, through a set of predefined transformation rules [8].
- The content of the ontology must be consistent with the predefined set of UML conceptual model conventions [7].
- The ontology identifiers must follow a strict URI policy defined in Section 6.
- The ontology design must commit long term URI persistence.
- The ontology, and the related artefacts, must be layered in order to support different degrees of ontological commitment and levels formal specification stacked on each other (see Section 4.2).
- The ontology, and the related artefacts, must be sliced in order to support a modular organisation of the domain in terms of self contained or semi-dependent modules (see Section 4.2).

2.3 General design criteria

For the purpose of knowledge sharing and interoperation between programs based on a shared conceptualisation, Gruber [19] proposes a set of preliminary *design criteria* a formal ontology should follow:

- *Clarity.* An ontology should communicate the purpose and meaning of defined terms. Definitions should be objective and independent of social and computational context even if the underlying motivations arise from them. Formalism is the means to this end, and when possible the logical formulation should be provided.

- *Coherence.* Ontology should permit inferences that are consistent with the definitions. At the least, the defining axioms should be logically consistent. Coherence should also apply to the concepts that are defined informally, such as those described in natural language documentation and examples.
- *Extensibility.* Ontology should be designed to anticipate the uses of the shared vocabulary. It should offer a conceptual foundation for a range of anticipated tasks and the representation should be crafted so that one can extend and specialize the ontology monotonically. This feature supports and encourages reuse and further specialisations of ontologies and creation of the application profiles.
- *Minimal encoding bias.* The conceptualization should be specified at the knowledge level without depending on a particular symbol-level encoding.
- *Minimal ontological commitment.* Ontology should require the minimal ontological commitment sufficient to support the intended knowledge sharing activities. Ontology should make as few claims as possible about the world being modeled, allowing the parties committed to the ontology freedom specialize and instantiate the ontology as needed. An ontological commitment is an agreement to use the shared vocabulary, with which queries and assertions are exchanged between agents, in a coherent and consistent manner. We say that an agent commits to an ontology if its behaviour is consistent with the definitions in the ontology [19].

3 Process and methodology

The main effort of the current stage of the project is to develop a formal ontology. This corresponds to Step 5 of the process described in [11, 3–15] and repeated in Section 1.3.

This section expands and addresses in detail the process of defining and implementing an OWL eProcurement ontology. The underlying assumption is that the conceptual data model developed at Step 3 serves as an input for the creation of the ontology, and that this process shall be automatic.

In addition to producing the ontology as an artefact, we also needs to validates its fitness to represent existing data and test whether the functional and non-functional requirements are respected. Figure 1 depicts the sequence of steps as a BPMN process diagram [37].

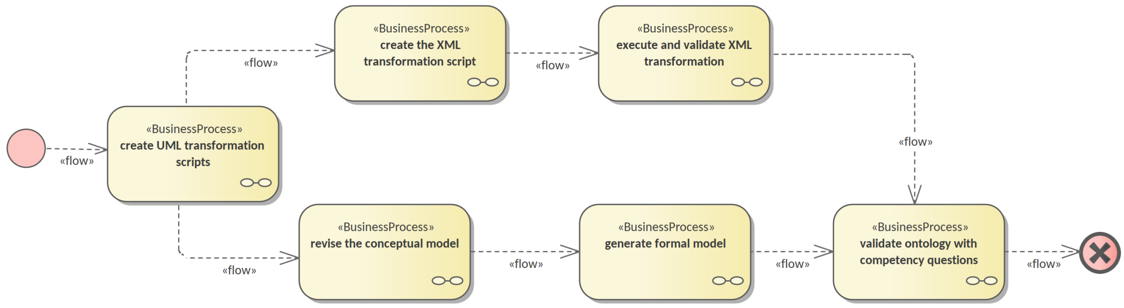


Figure 1: The main steps to implementing and validating the formal eProcurement ontology

The conceptual model serves as the single source of truth, the process starts with development of a series of transformations scripts. The conceptual model needs to be adjusted in order to fit a set of UML modelling conventions [7] making it suitable input for the transformation scripts. Provided that the conceptual model is conform, the transformation can be executed. Finally the validation of the formal ontology can be performed using the existing eProcurement data.

The existing eProcurement data needs to be transformed from XML into RDF format. So, in parallel, after the UML transformations are created and along with them, the ontology architecture and UML conventions, then a set of XML transformation scripts can be developed. Once they are ready, they need to be executed on previously selected datasets, to convert them into RDF data instantiating the formal ontology. Only then, when the datasets are available, the ontology can be validated.

The next subsections describe each of these six steps in more detail in order to provide rationale and introduce each artefact in part.

3.1 UML transformation scripts

The process start with authoring two documents laying the foundations of the entire process: the ontology architecture and the UML modelling conventions [7]. The main purpose of the ontology architecture (this document) specifications is to describe why the ontology is being built, what its intended uses are, who the end-users are, and which requirements the ontology should fulfil. Moreover, it states how the ontology should be structured in order to facilitate maintenance and usage patterns.

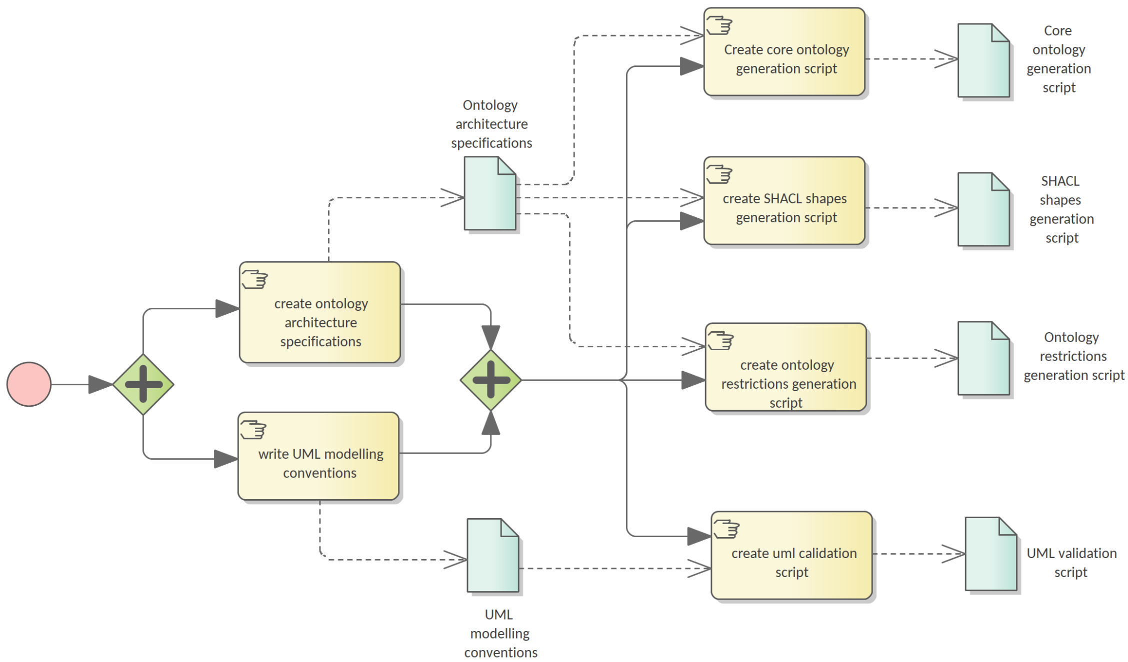


Figure 2: Creation of the specifications documents and the UML transformation scripts

The conceptual model must comply with a set of UML modelling conventions making it suitable input for the transformation scripts, which implement the same conventions. The two parallel actions staring the process are depicted in Figure 2.

The UML conventions document serves, at large, as requirements specification for the XSLT script that checks the UML conceptual model whether it conforms to the conventions.

The ontology architecture specification (this document) serves, at large, as requirement specifications for the development of three XSLT scripts to generate the formal ontology. These scripts can be developed independent of each other as they refer to different aspects of the formal ontology as described in Section 4.1.

The input for these scripts is the UML conceptual model serialised in XMI 2.5.1 format [1].

3.2 Conceptual model revision

The conceptual model revision is an iterative process. The validation script is execution outputs a report indicating if there are any deviations from the conventions, and detailing which are they and eventually what are the necessary actions to resolve them.

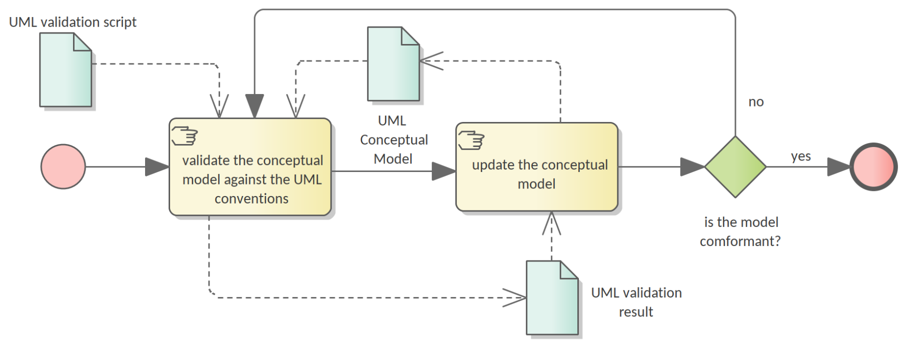


Figure 3: Adjustment of the UML conceptual model guided by the validation script

3.3 Formal ontology generation

3.4 XML to RDF data transformation

4 Architectural considerations

4.1 Separation of concerns

The successful application of an ontology or the development of an ontology-based system depends not just on building a good ontology but also on fitting this into an appropriate development process. All computing information models suffer from a semantic schizophrenia. On the one hand, the model represents the domain; on

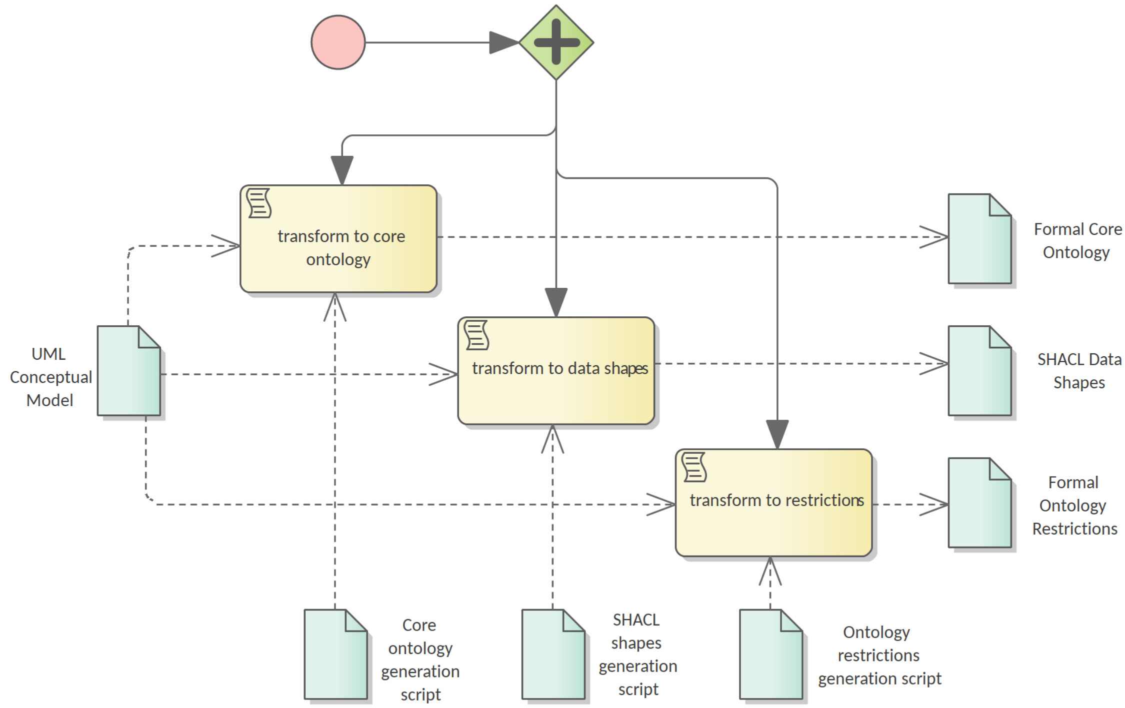


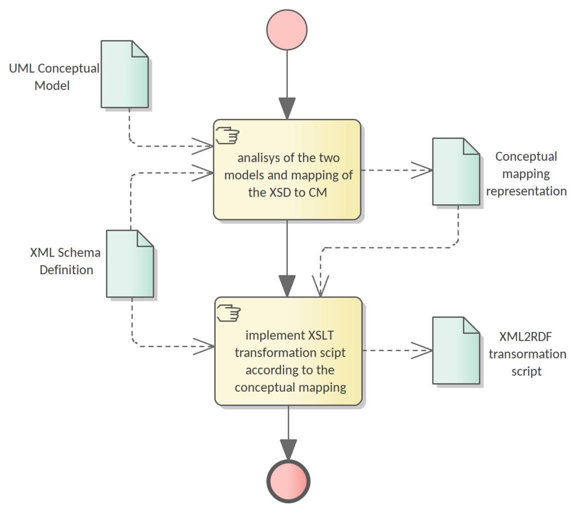
Figure 4: Generation of the formal ontology from the UML conceptual model

the other hand, it represents the implemented system, which then represents the domain. These different representation requirements place different demands upon its structure [29].

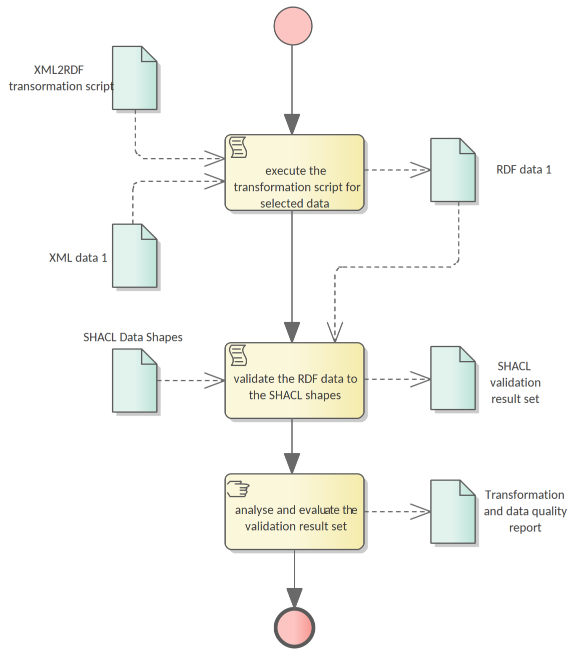
One of the common ways to manage this problem is a separation of concerns. OMG's Model Driven Architecture (MDA) [36] is a well documented structure where a model is built for each concern and this is transformed into a different model for a different concern.

Transformation deals with producing different models, viewpoints, or artefacts from a model based on a transformation pattern. In general, transformation can be used to produce one representation from another, or to cross levels of abstraction or architectural layers [35].

The process described in Section 3 adopted some of these principle and employs model transformation to achieve the project objectives.



(a) Implementation of the XML transformations script based on the mappings from XSD schemas onto the conceptual model



(b) Transformation of the existent eProcurement XML data into RDF representation and validating the end result conformance

Figure 5: XML to RDF data transformation

4.2 Layers and components

This architecture is organised in *horizontal layers* and vertically slicing *components*. The *components* reflect the organisation of the formal ontology based on a logical content division of the UML conceptual model into packages and modules. This division increases the maintainability of entire content. The models and their components are not disconnected from one another. The relations between them are represented in Figure 6 where each component is represented as an UML package. The conceptual model serves as the single source of truth, from which the three components of the formal model are derived. Each of these components can be further divided into modules.

The main ontology packages are: the *core ontology*, *data shapes*, and *formal ontology restrictions*. The these formal modules are derived from the conceptual model through model transformations as described in Section 3 following the rules laid out in Costetchi [8].

The *core ontology* is the foundational, and serves as a backbone for the other components. It establishes the identifiers and the basic definitions of classes and properties.

The *data shapes* represent constraints on how the core ontology can be instantiated and the set of controlled value lists associated to it.

The *formal ontology restrictions* cover intensional class and property definitions used for deriving additional knowledge from factual information. Both, the data shapes and the ontology restrictions are defined as extensions that are flashing out the core ontology which plays, in this case the role of a backbone.

This architecture distinguishes the following layers: the *conceptual layer*, *core definition layer*, *validation layer* and *reasoning layer*. These *layers* can be though of as formal languages with well defined boundaries and extents. The diagram in Figure 7 presents the organisation the ontology layers along two axes: *expressivity* and *detail*.

The *expressivity* of a language is the breadth of ideas that can be represented and communicated in that language. The more expressive a language is, the greater the variety and quantity of ideas it can be used to represent. The design of a language and its formalism involves an inevitable trade-off between the expressive power and its "analyzability", which translates directly into computation difficulty. The more a formalism can express, the harder it becomes to understand, i.e. compute, what do instances of it mean.

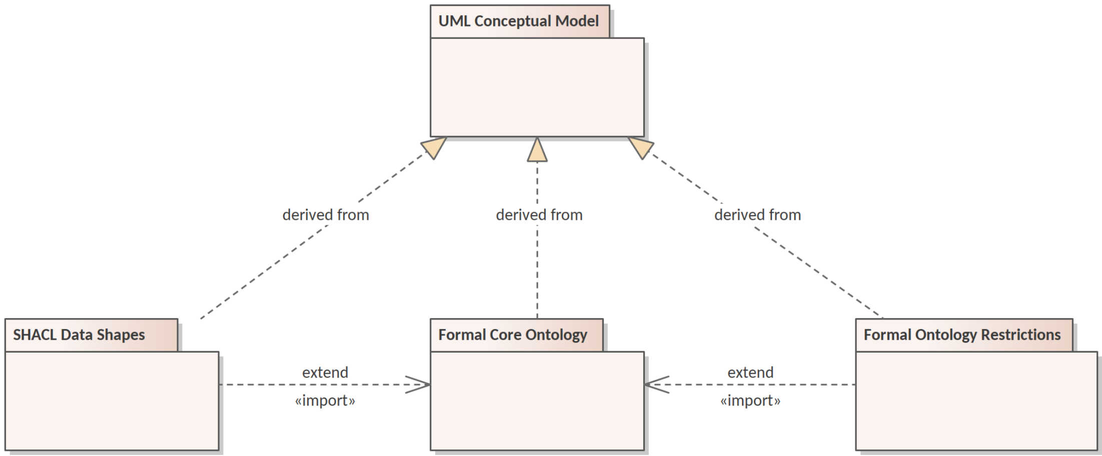


Figure 6: The main components of the eProcurement ontology and their relation to each other and the UML conceptual model

The *detail* refers to how much description and aspects of the domain concepts are considered. This dimension is plays a pragmatic rather than formal role. The rationale is that lower level of detail is useful and re-usable to a wider public, but for a set of relatively simple tasks. As the level of detail increases, the difficulty to operate on it also increases thus the user base shrinks and the task complexity rises.

The detail axis starts, on one side, from establishing the concepts identity, labels, natural language definitions; continues through establishing relations and constraints; and ends, on the other side, with formulating special logical conditions, implications and inference rules.

The *conceptual layer* accommodates informal representation of the business objects, places, things, actors in the "real world" not representations of these things in the information system. It is situated at the base of the diagram with lowest expressivity because it has not formal semantics but with a wide coverage of detail. The dotted line delimits the border between formal and informal layers.

In the conceptual layer is situated the *UML conceptual model*, which is described in Section 4.3. This model is also called *computation independent model* because it is informal and its main purpose is to interface the domain experts with an explicit conceptual representation of the domain.

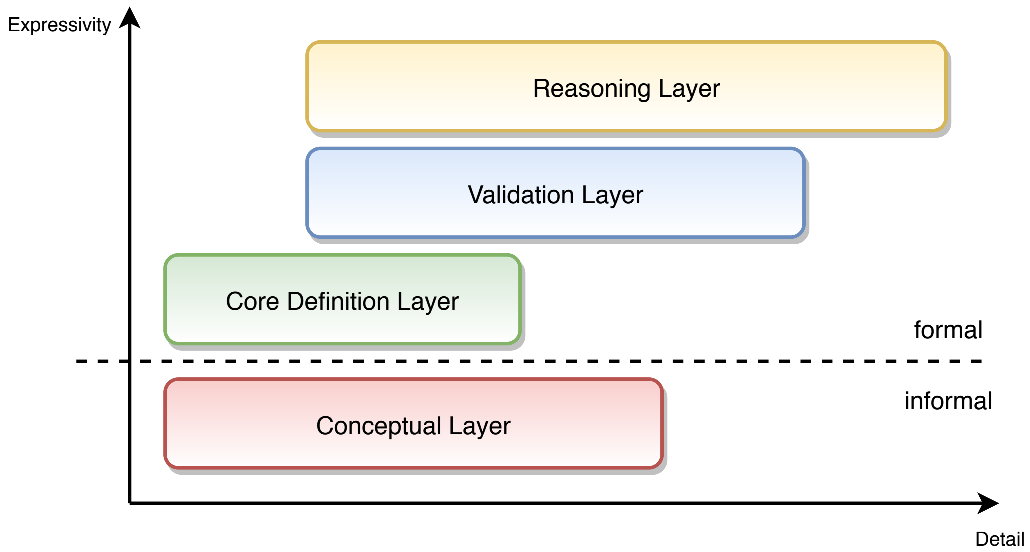


Figure 7: Expressivity and extent of the eProcurement ontology layers

In the formal section of the diagram, the expert knowledge is expressed using descriptive languages with a formal semantics (see Section 5.2), which in this architecture are primarily OWL 2[28] and SHACL[22].

The *core layer*, is situated the core ontology, which has as a primary goal to define the main concepts and relations of the ontology. It is limited to the declarative axioms of the ontology and therefore serves primarily for vocabulary and identity establishment. For this reason it is represented as having the lowest formal expressivity and the lightest level of detail in comparison to other layers. It plays the role of a backbone which is flashed out by other layers.

The *validation layer* accommodates declarations of data shapes which represent constraints on the instance data, i.e. the ABox (see Section 5.1). The assertions in this layer are interpreted done under the closed world assumption, making it possible to support validation functionality. The SHACL shapes are situated in this layer and are addressed in detail in Section 4.4.

The validation layer is situated immediately above the core definitions, being more expressive but also extending it with additional detail. Therefore it is shifted, on the detail axis, to the right towards more detail and leaving out the simple axioms as they are already covered in the core later.

The *reasoning layer* accommodates formal intensional definitions of the classes and properties. It is mostly formed of subclass restrictions and complex class definitions, and as well, domain and range specifications for properties. This layer provides assertions necessary to supports reasoning functionalities for eProcurement ontology. The formal ontology restrictions component is situated in this layer and is described in Section 4.5.

Just like in the case of restrictions layer, the reasoning layer extends the core layer with higher level of detail and comprising assertions with higher expressive power. Therefore the simple details rest in the core layer permitting this one to focus on other ones. For this reason it is depicted as shifted to the towards the right side of detail axis.

4.3 UML conceptual model

The conceptual model is represented in UML [4] serves as the single source of truth. Thus the scope of this architecture is limited by what can be expressed in UML and how that information is utilised to generate formal statements. Each of the above functions will lead to different interpretations of the same UML model.

The primary application of UML [18] for ontology design is in the specification of class diagrams for object-oriented software. However, UML does not have a clearly specified declarative semantics, so that it is not possible to determine whether an ontology is consistent, or to determine the correctness of an implementation of the ontology. Semantic integration in such cases becomes a subjective exercise, validated only by the opinions of the human designers involved in the integration effort [20].

On the other hand, UML is closer than more logic-oriented approaches to the programming languages in which enterprise applications are implemented. For this reason in the current project we have decided to develop agreements on the informal semantics of the UML-based conceptual model. It consists of a set of explicit conventions for naming UML elements [7] and for transforming UML to OWL [8].

4.4 SHACL shapes

Application profiles represent a set of constraints on the logical model tying it to a particular system implementation. The application profiles, in this project, must be expressed using SHACL language [ref shacl spec]. The application profiles are extending the (core) formal model and can be derived from the conceptual model through transformation, or can be designed directly in the formal representation.

4.5 Formal ontology restrictions

5 Formal considerations

5.1 Model and data relationship

DL ontologies are structured into two sets: *ABox* and *TBox*. The *ABox* consists of all (class or property) instance assertions. The *TBox* consists of all terminological axioms, i.e., of all subclass inclusion axioms. The ABox provides information about concrete individuals while the TBox describes general rules that hold for all individuals. In consequence, ABoxes tend to be much larger than TBoxes [23]. In Figure 8 is depicted the delimitation and relations between the TBox and box ABox components of the eProcurement ontology.

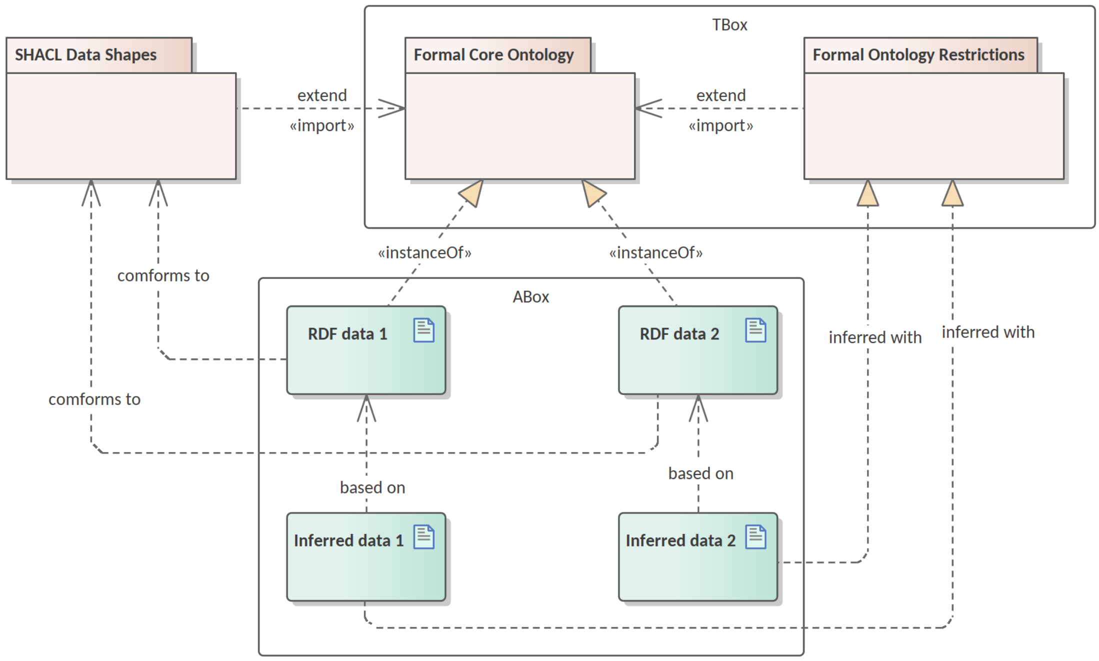


Figure 8: The relationship between the data and each of the ontology components

To explain the relevance of each component, in Figure 8, two arbitrary datasets are brought as examples. They instantiate the core ontology, which means that they comprise of factual statements of concrete entities of eProcurement classes. In order to ensure that the instance data follow minimally the intended ontology design they need to comply with a set of data shapes. Once this condition is satisfied, then, new knowledge can be inferred from the provided facts, given the domain inference rules. The new knowledge is mainly oriented to solve the classification task and does not cover other types of inference.

It is important to note that the data shapes fall out of the TBox as they serve the validation function and are based on a different set of assumptions. First they are interpreted under the closed world assumption, like in XML or RDBM contexts (see Section 5.3). Second they follow a RDF graph based semantics 5.2.

5.2 Semantics

Users of OWL [28] can actually select between two slightly different semantics: *direct semantics* that corresponds to the Description Logics (DL) [3], and *RDF-based semantics* that is based on translation of the OWL axioms into directed graphs. In this document we assume by default the direct semantics. In particular cases (i.e. SPARQL entailments and SHACL data shapes) RDF-based semantics is adopted and is explicitly mentioned in the document.

Description logics provide a concise language for OWL axioms and expressions. DLs are characterised by their expressive features. The description logic that supports all class expressions with $>, \bot, \sqcap, \sqcup, \neg, \exists$ and $\forall$ is known as $\mathcal{ALC}$ (which originally used to be an abbreviation for Attribute Language with Complement). For a formal introduction into DL please consult Baader et al. [3].

Inverse properties are not supported by $\mathcal{ALC}$, and the DL we have introduced above is actually called $\mathcal{ALCI}$ (for $\mathcal{ALC}$ with inverses) [23]. Many description logics can be defined by simply listing their supported features. We will use this notation when discussing degrees of expressivity for the ontology layers in Section 5.5.

Computing all interesting logical conclusions of an OWL ontology can be a challenging problem, and reasoning is typically multi-exponential or even undecidable. To address this problem, the recent update OWL 2 of the W3C standard [31, 28] introduced three profiles: *OWL EL*, *OWL RL*, and *OWL QL*. These lightweight sublanguages of OWL restrict the available modelling features in order to simplify reasoning. This has led to large improvements in performance and scalability, which has made the OWL 2 profiles very attractive for practitioners [23].

On the other hand, the validation data shapes are expressed using Shapes Constraint Language (SHACL) [22]. Its semantics is based on RDF graphs but full RDFS inferencing is not required. SHACL processors may operate on RDF graphs that include RDF entailments [30] and SPARQL specific entailments [27]. The entailment regime specifies conditions that address the fourth condition on extensions of basic graph pattern matching [21, 30].

This architecture delimits different concerns in Section 4.2 in a stack of layers and assigns levels of expressivity to each of the layers in Section 5.5.

5.3 Open/Closed world assumptions

5.4 Conformance

The eProcurement ontology should be expressed in OWL 2 language and in conformance with the conditions listed in Krötzsch et al. [24].

SHACL conformance

RDF conformance for the data

UML conformance for CM

5.5 Expressivity levels

In the layered approach described in [ref architecture section], different expressivity levels are necessary for each layer. In this section we briefly describe these levels of expressivity and relate them to OWL sublanguages [9] and profiles [26].

6 URI policy

7 Final word

Model/Component	Language
Conceptual model	UML (informal)
Core ontology	OWL Lite
Data shapes	SHACL (SPARQL 1.1 entailment)
Ontology restrictions - simplified	OWL QL, OWL EL, OWL RL
Ontology restrictions - complete	OWL 2
Special inference rules (out of scope)	SWRL

Table 1: The components and the corresponding language dialect

Bibliography

[1] Xml metadata interchange (xmi) specification: Version 2.5.1. Standard formal/2015-06-07, Object Management Group (OMG), 2015. URL `http://www.omg.org/spec/XMI/2.5.1`.

[2] P. Archer, L. Bargiotti, M. D. Keyzer, S. Goedertier, N. Loutas, and F. V. Geel. Report on high-value datasets from eu institutions. Deliverable SC17DI06692, European Commission , 2014.

[3] F. Baader, I. Horrocks, and U. Sattler. Description logics. In *Handbook on ontologies*, pages 3–28. Springer, 2004.

[4] G. Booch, J. Rumbaugh, and I. Jacobson. *Unified Modeling Language User Guide, The (2nd Edition) (Addison-Wesley Object Technology Series)*. Addison-Wesley Professional, 2005. ISBN 0321267974.

[5] S. O. Bradner. Key words for use in RFCs to Indicate Requirement Levels. RFC 2119, Mar. 1997. URL `https://rfc-editor.org/rfc/rfc2119.txt`.

[6] S. Cook, C. Bock, P. Rivett, T. Rutt, E. Seidewitz, B. Selic, and D. Tolbert. Unified modeling language (UML) version 2.5.1. Standard formal/2017-12-05, Object Management Group (OMG), Dec. 2017. URL `https://www.omg.org/spec/UML/2.5.1`.

[7] E. Costetchi. eProcurement uml conceptual model conventions. Recommendation, Publications Office of the European Union, April 2020.

[8] E. Costetchi. eProcurement uml conceptual model to owl ontology transformation. Recommendation, Publications Office of the European Union, April 2020.

[9] M. Dean and G. Schreiber. OWL web ontology language reference. W3C recommendation, W3C, Feb. 2004. http://www.w3.org/TR/2004/REC-owl-ref-20040210/.

[10] M. Dekkers, E. Stani, and F. Barthelemy. D02.02 - project charter proposal. Deliverable SC378DI07171, Publications Office of the European Union, 2017.

[11] M. Dekkers, E. Stani, B. Wyns, and F. Barthelemy. D02.01 - specification of the process and methodology to develop the eprocurement ontology with initial draft of the eprocurement ontology for 3 use cases. Deliverable SC378DI07171, Publications Office of the European Union, 2017.

[12] European Parliament and the Council. Directive 2013/37/eu of the european parliament and of the council of 26 june 2013 amending directive 2003/98/ec on the re-use of public sector information text with eea relevance. *OJ*, L 175: 1–8, 2013. CELEX:32013L0037.

[13] European Parliament and the Council. Directive 2014/23/eu of the european parliament and of the council of 26 february 2014 on the award of concession contracts text with eea relevance. *OJ*, L 94(2014/23/EU):1–64, 2014. CELEX:32014L0024.

[14] European Parliament and the Council. Directive 2014/24/eu of the european parliament and of the council of 26 february 2014 on public procurement and repealing directive 2004/18/ec text with eea relevance. *OJ*, L 94:65–242, 2014. CELEX:32014L0024.

[15] European Parliament and the Council. Directive 2014/25/eu of the european parliament and of the council of 26 february 2014 on procurement by entities operating in the water, energy, transport and postal services sectors and repealing directive 2004/17/ec text with eea relevance. *OJ*, L 94:243–374, 2014. URL `http://data.europa.eu/eli/dir/2014/25/oj`. CELEX:32014L0025.

[16] European Parliament and the Council. Directive 2014/55/eu of the european parliament and of the council of 16 april 2014 on electronic invoicing in public procurement text with eea relevance. *OJ*, L 133:1–11, 2014. CELEX:32014L0055.

[17] European Parliament and the Council. Directive (eu) 2019/1024 of the european parliament and of the council of 20 june 2019 on open data and the re-use of public sector information. *OJ*, L 172:56–83, 2019. CELEX:32019L1024.

[18] M. Fowler. *UML distilled: a brief guide to the standard object modeling language*. Addison-Wesley Professional, 2004.

[19] T. R. Gruber. Toward principles for the design of ontologies used for knowledge sharing? *International journal of human-computer studies*, 43(5-6):907–928, 1995.

[20] M. Grunninger. Enterprise modelling. In *Handbook on enterprise architecture*, pages 515–541. Springer, 2003.

[21] P. Hayes. RDF semantics. W3C recommendation, W3C, Feb. 2004. http://www.w3.org/TR/2004/REC-rdf-mt-20040210/.

[22] H. Knublauch and D. Kontokostas. Shapes constraint language (SHACL). W3C recommendation, W3C, July 2017. https://www.w3.org/TR/2017/REC-shacl-20170720/.

[23] M. Krötzsch. Owl 2 profiles: An introduction to lightweight ontology languages. In *Reasoning Web International Summer School*, pages 112–183. Springer, 2012.

[24] M. Krötzsch, I. Horrocks, M. Smith, and B. Glimm. OWL 2 web ontology language conformance (second edition). W3C recommendation, W3C, Dec. 2012. http://www.w3.org/TR/2012/REC-owl2-conformance-20121211/.

[25] N. Loutas, N. Loutas, S. Kotoglou, and D. Hytiroglou. D04.07 - report on policy support for eprocurement. Deliverable SC245DI07171, ISA programme of the European Comission, 2016.

[26] B. Motik, I. Horrocks, B. C. Grau, A. Fokoue, and Z. Wu. OWL 2 web ontology language profiles (second edition). W3C recommendation, W3C, Dec. 2012. http://www.w3.org/TR/2012/REC-owl2-profiles-20121211/.

[27] C. Ogbuji and B. Glimm. SPARQL 1.1 entailment regimes. W3C recommendation, W3C, Mar. 2013. http://www.w3.org/TR/2013/REC-sparql11-entailment-20130321/.

[28] B. Parsia, P. Patel-Schneider, and B. Motik. OWL 2 web ontology language structural specification and functional-style syntax (second edition). W3C recommendation, W3C, Dec. 2012. http://www.w3.org/TR/2012/REC-owl2-syntax-20121211/.

[29] C. Partridge, A. Mitchell, and S. de Cesare. Guidelines for developing ontological architectures in modelling and simulation. In *Ontology, Epistemology, and Teleology for Modeling and Simulation*, pages 27–57. Springer, 2013.

[30] P. Patel-Schneider and P. Hayes. RDF 1.1 semantics. W3C recommendation, W3C, Feb. 2014. http://www.w3.org/TR/2014/REC-rdf11-mt-20140225/.

[31] P. Patel-Schneider, B. Parsia, and B. Motik. OWL 2 web ontology language structural specification and functional-style syntax. W3C recommendation, W3C, Oct. 2009. http://www.w3.org/TR/2009/REC-owl2-syntax-20091027/.

[32] Publications Office of the European Union. Interinstitutional style guide 2011, 2011. doi: doi:10.2830/36616.

[33] PwC EU Services. e-government core vocabularies handbook. Report, ISA programme of the European Comission, 2015. URL `https://ec.europa.eu/isa2/library/e-government-core-vocabularies-handbook_en`.

[34] E. Rescorla, R. Barnes, and S. Kent. Further Key Words for Use in RFCs to Indicate Requirement Levels. RFC 6919, Apr. 2013. URL `https://rfc-editor.org/rfc/rfc6919.txt`.

[35] J. Siegel. Object management group model driven architecture (mda) mda guide rev. 2.0. 2014. URL `http://www.omg.org/cgi-bin/doc?ormsc/14-06-01`.

[36] R. Soley et al. Model driven architecture. *OMG white paper*, 308(308):5, 2000.

[37] S. A. White. Introduction to bpmn. *Ibm Cooperation*, 2(0):0, 2004.