

**Project-End Report**

Initiative for developing eProcurement ontology

Date: 25/01/2021

Doc. Version: 1.1

Template Version: 3.0.1



*This template is based on PM² v3.0*

*For the latest version of this template please visit the PM² Wiki*

**Document Control Information**

|  |  |
| --- | --- |
| **Settings** | **Value** |
| **Document Title:** | Project-End Report |
| **Project Title:** | Initiative for developing eProcurement ontology |
| **Document Author:** | Eugeniu Costetchi |
| **Project Owner:** | Publications Office of the European Union |
| **Project Manager:** | Natalie Muric |
| **Doc. Version:** | 1.1 |
| **Sensitivity:** | Public |
| **Date:** | 2021-01-25 |

**Document Approver(s) and Reviewer(s):**

NOTE: All Approvers are required. Records of each approver must be maintained. All Reviewers in the list are considered required unless explicitly listed as Optional.

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Role** | **Action** | **Date** |
| Natalie Muric | Reviewer | *<Approve / Review>* |  |
| Ioannis Rousochatzakis | Approver | *<Approve / Review>* |  |
|  |  |  |  |

**Document history:**

The Document Author is authorized to make the following types of changes to the document without requiring that the document be re-approved:

* Editorial, formatting, and spelling
* Clarification

To request a change to this document, contact the Document Author or Owner.

Changes to this document are summarized in the following table in reverse chronological order (latest version first).

|  |  |  |  |
| --- | --- | --- | --- |
| **Revision** | **Date** | **Created by** | **Short Description of Changes** |
| 0.1 | 2020-12-01 | Eugeniu Costetchi | First draft |
| 1.0 | 2020-12-22 | Eugeniu Costetchi | Final draft |
| 1.1 | 2021-01-25 | Eugeniu Costetchi | Revision addressing partner comments |

**TABLE OF CONTENTS**

[**Introduction**](#_heading=h.2et92p0) **3**

[**Project Deliverables**](#_heading=h.igt3ydepdbl4) **4**

[Task 1 - Ontology architecture and specifications](#_heading=h.niur3uv1ntti) 4

[Task 2 - Generation of the formal ontology](#_heading=h.2dnw5tshgmf) 5

[Task 3 - Maintenance and final report](#_heading=h.vcbfqlaj6yfl) 6

[**Project Success**](#_heading=h.ds5u6yl76apb) **7**

[Evaluation by Project Manager](#_heading=h.dmzyo8gfoa9l) 7

[Evaluation by Users](#_heading=h.z2ne49eu1bot) 7

[**Lessons Learned and Best Practices**](#_heading=h.z60utsnj40) **7**

[**Post-Project Recommendations - Proposals for Future Work**](#_heading=h.iqupu7hxqxt8) **8**

1. Introduction

The purpose of this report is to summarise and document all the key elements that are found during the life of the project and discussed during the project-end review meeting. The goal is to capture the overall stakeholder satisfaction, perform an overall evaluation of the project experience, and document lessons learned, best practices and offer post project recommendations.

Capturing lessons learned allows the project team and as well the permanent organisation as a whole to benefit from the experience acquired during the project. Capturing ideas and recommendations for post-project work related to the operations of the products and services, is also invaluable for future projects.

Project Deliverables

The Publications Office of the European Union set off to build an eProcurement ontology. The ultimate objective of the project was to put forth a commonly agreed 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. The process and the methodology adopted involve modelling the conceptual model in Unified Modelling Language (UML) and then, by abiding a set of conventions and recommendations, transform that model into a formal ontology expressed in Web Ontology Language (OWL).

The objectives of this project were to establish a reliable way of transforming the conceptual eProcurement model into a formal ontology and other related artefacts. This has been organised as three tasks: *(a) Ontology architecture and specifications* (**Task 1**)*, (b) Generation of the formal ontology* (**Task 2**), and *(c) Maintenance and final report* (**Task 3**).

Task 1 - Ontology architecture and specifications

The goal of this task was to develop an ontology architecture and a set of model conventions that can be automatically assessed. This task has been performed in three steps: develop the overall architecture document, develop a set of UML conventions, implement a convention-checking mechanism.

The first step of this phase aimed to create an Ontology Architecture document (**WP1.1**) that would provide a working definition of what is the architectural stance and the design decisions that shall be adopted for the eProcurement formal ontology along with the specifications on how to generate comprising components. This comprises also performing a state of the art, gathering previous reports on eProcurement, ISA2 and other ontology architecture documents in order to be able to plan the architecture. The document starts with an introduction that contains background considerations, identifying the target audience, context and scope of eProcurement and conformance details and its sections are structured as follows:

* *Requirements* - This section elaborates on the general design criteria along with requirements the ontology should fulfil and it will include functional requirements, non-functional requirements and general design criteria.
* *Process and methodology -* 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 serves as an input for the creation of the ontology, and that this process shall be automatic.
* *Architectural considerations* - This section provides in depth specifications of how these artefacts are structured and how they relate to each other, as the previous section presented the processes used to generate eProcurement ontology components, instance data and how to validate them.
* *Formal considerations* - This section addresses the aspects of logics in the eProcurement ontology. It specifies the semantics, the relationship between the abstract layer of the ontology and the instance data, under what assumptions is the inference done and what level of logical formalism should the ontology components adapt in order to maintain flexibility and reasoning capacity.
* *URI policy* - This section contains guidelines and recommendations to build project URIs.

In the second step, a *technical conventions document for the UML representation of the conceptual model* (**WP1.2**), was created. This document provides a working specification of the guidelines and conventions for the eProcurement conceptual model. The conventions described in WP 1.2 qualify a model as being suitable input for the transformation scripts (developed in Task 2 described in section below). The document was split in two main sections as follows:

* *Conventional constraints* - defining naming and structural conventions for concepts in an ontology and then strictly adhering to these conventions;
* *Technical constraints -* defining the technical, and UML specific constraints and rules in order to aid interpretation and restrict the space of possibilities when the UML is being processed by scripts.

In the third step was developed *a script that checks the conformance to the technical conventions of the conceptual model* (**WP1.3**). This script represents the implementation of conventions from WP 1.2. The output is an interactive HTML report comprising a model nomenclature and UML conformance violations, if any encountered. The script was implemented using XSLT language primarily because the inputs (XMI representation of the UML model) and the outputs (HTML conformance report) are XML based representations.

To ensure its robustness, a *suite of unit tests* (**WP1.4**) accompanies the script, following the Test-Driven Development ([TDD](https://github.com/eprocurementontology/eprocurementontology/tree/v2.0.1)) methodology[[1]](#footnote-2). In general, TDD methodology was adopted for all implementation exercises. The tests ensure a set of expectations and quality assuring criteria are met by the implementation. Also, they guard the script from possible regressions introduced in the future extensions and adaptations. The unit tests for XSLT templates and functions were implemented using the XSpec technology.

In order to easier track the implemented UML conformance checks, *an Excel spreadsheet was developed comprising a systematisation of conformance rules extracted from the UML conventions document* (**WP1.5**). This serves as a concise inventory of all the rules that needed to be respected in the UML model, internally referred to as checkers. In this spreadsheet every checker has an identifier, a scope, a subcategory, a type of constraint, a name and the message to be generated in the HTML conformance report for the users.

In order to improve user experience, an interactive table of contentscomponent was developed, which aids users to navigate easily through the HTML conventions report. Also, a filter component was added, which allows the user to choose the type of constraints to visualise, hiding the rest.

Task 2 - Generation of the formal ontology

This task aimed at implementing a set of scripts that transform the conceptual model into a formal ontology. This task was performed in two steps: first a set of transformation rules was developed and then these rules were implemented as XSLT scripts.

First was created the document that provided a *working definition of the transformation rules from the UML conceptual model into the formal ontology and validation data shapes* (**WP1.6**). These rules were organised in accordance with the eProcurement ontology architecture (**WP1.1**).

This document comprised four sections covering major UML aspects: classes, attributes, main connector types employed in the eProcurement model (associations, dependencies and generalisations), datatypes and enumerations.

Each section comprises a set of transformation rules that are applicable to corresponding UML elements. In the section introduction an inventory table is available offering an overview of the transformation rule set within *core ontology layer, data shape layer and reasoning layer* (see the ontology architecture document for layer definitions). Each transformation rule explanations how the source UML pattern/structure shall be translated into the corresponding RDF representation (OWL or SHACL). In addition, if there were multiple UML structures that lead to the same OWL interpretation, then all have been documented and only one indicated as preferred.

This description constituted a technical specification for implementing the three transformation scripts: (a) a *transformation script for the core ontology* (**WP2.1**), (b) a *transformation script for the SHACL shapes* (**WP2.4**) and (c) *a transformation script for the ontology intensional definitions based on the class and property restrictions* (**WP2.7**). Each of the implemented script corresponds to one of three ontology architecture layers.

Before the implementation, an analysis task was performed on the eProcurement conceptual model ([EPO v2.0.1[[2]](#footnote-3)](https://github.com/eprocurementontology/eprocurementontology/tree/v2.0.1)) in order to assess the suitability to undergo the automatic transformation. This further informed the development team on how the implementation of the transformation rules shall be approached.

Because TDD development methodology was adopted, the implementation of each transformation rule was associated with a *suite of unit tests* (correspondingly **WP2.2, WP2.5, WP2.8**) and *a sample output dataset* (correspondingly **WP2.3, WP2.6, WP2.9**).

Finally, when the development of the transformation scripts was completed, a thorough quality analysis was carried out to ensure the output the RDF artefacts produced by each of the scripts are valid. The core, restrictions and data shape modules have been syntactically and semantically checked using TopBraid Composer and Protege tools accompanied by manually seeking design flaws.

The findings have been documented and fixed in subsequent implementation phase in a timely manner. When the RDF output produced by the transformation scripts were satisfiable, a set of discussions and further adaptations was done both to the transformation scripts and the UML model in order to normalise the management of the namespaces.

The need for normalisation stems from the fact that the standard UML data types need to be mapped to atomic XSD datatypes in order to perform UML to OWL representation. Moreover, custom sets of data types were adopted in the model that require additional mapping rules. This was solved by implementing a generic mapping mechanism controlled from the configuration file.

Task 3 - Maintenance and final report

This task aimed to fix any bugs discovered in the scripts during piloting phase and therefore *minimal maintenance efforts* (**WP3.1**) have been planned until the end of 2020. Additionally, the last work package of the project is *a final report* (**WP3.2**) - this document.

The maintenance phase was foreseen mainly for bug fixing but it also involved changes in the initial requirements as the partner needs of the transformation scripts evolved as well. Additionally, a few easy to make improvements have also been implemented on OP’s request.

A significant effort has been channelled into improving and evolving the UML conventions checking script. Some of the noteworthy change requests and improvements are:

* Building a nomenclature that accounts for the class names, class attributes and where they are used in the model. In addition, accounts of the used data-types and multiplicity values for each class attribute is provided by the nomenclature. Similarly, details about their usage and multiplicity value were generated for all connectors in the model in order to facilitate model editing for the authors.
* A major evolution in the implementation was a consequence of the decision to aggregate connectors into a single definition even if they are used multiple times in the model. This corresponds to the OWL/RDFS semantics and had a major impact on the script.
* Similarly, aggregation of class attributes with multiple usages was implemented.
* Adding the table of contents and a filter for errors, warning and info types of messages improved the process of model cleaning and fixing.
* Added a check for the XMI version restricting it to UML 2.5 and XMI 2.5.1 to prevent unexpected outputs.
* Aggregation of the attributes and connectors led to development of an additional set of checkers at the aggregation level.

The transformation scripts suffered major changes as well during this phase. The most significant efforts were spent on the following improvement requests and change requirements:

* Change the way the OWL datatype property definitions (both for object and datatype properties) are generated, switching from multiple definitions for an attribute name to a single one. This is a result of an aggregation operation of all the usages in the UML model.
* A similar aggregation operation was implemented for transformation of association and dependency connectors. What changed was the way the OWL object properties are generated along with restrictions and data shapes associated with them.
* Implemented property inheritance
* Implemented filters to ignore certain UML constructs for example: connectors between connectors do not have a direct OWL representation and should be ignored.
* Investigating specific cases (classes, attributes and connectors) in the ePO model to diagnose the source of the issue leading to a decision if the model or the script needs to be adjusted.
* Implemented a mechanism that allows customizable configuration for the transformation script supporting multiple configurations to be maintained in parallel.

Project Success

The project was able to provide the Publications Office and its partners a reliable mechanism of transforming the UML model into a formal OWL ontology and derive a SHACL shape file suitable for validation of RDF datasets.

The project offers the editors of the conceptual model a versatile validation mechanism to conform to the UML conventions document. This is an important tool for increasing the quality of the conceptual model.

The deliverables of the project were available in May 2020 on the public [GitHub repository[[3]](#footnote-4)](https://github.com/meaningfy-ws/model2owl) and on the Publications Office internal repository.

Evaluation by Contractor

This project was fairly challenging because of its interrelated nature between various partners and limited timescale to complete. The project required engagement, commitment and a high quality of work both from the external contractors and the Publications Office.

The project ran from March 2020 to June 2020 to complete two main tasks: develop the ontology architecture and specifications (**Task 1**) and develop the formal ontology generation scripts (**Task 2**). The maintenance was provided through the rest of the year until December 2020 when also the project end report (this document) was delivered (**Task 3**).

Despite the limited timescale of the project, all project deliverables were delivered on time.

Evaluation by Users

During the feedback sessions, the users from Publications Office and other external stakeholders actively engaged in using the project outputs acknowledged the high-quality work done. They appreciated the cooperation between the involved partners and responsiveness by which the feedback was taken onboard and implemented which was an important factor for success.

The UML to OWL transformation scripts have been identified as extremely valuable for other similar modelling projects. These scripts could be used to produce formal ontologies in the ISA2 semantic interoperability (SEMIC[[4]](#footnote-5)) projects such as Core Criterion and Core Evidence Vocabulary ([CCCEV](https://joinup.ec.europa.eu/collection/semantic-interoperability-community-semic/solution/core-criterion-and-core-evidence-vocabulary))[[5]](#footnote-6), Common Assessment Method for Standards and Specifications ([CAMSS](https://joinup.ec.europa.eu/collection/common-assessment-method-standards-and-specifications-camss))[[6]](#footnote-7), Core Assessment Vocabulary ([CAV](https://joinup.ec.europa.eu/collection/common-assessment-method-standards-and-specifications-camss/solution/core-assessment-vocabulary-cav))[[7]](#footnote-8) and others.

Some users needed to use a more sophisticated SHACL generation system in order to satisfy their use cases. The identified need requires a mechanism to define the scope for the type of rules that can be conveyed in the UML diagram. Moreover, not all rules can be conveyed by the main model so it must be possible to add more complex rules on a separate basis, such as extensions and refinements.

Some reported difficulties using the project outcomes because they are quite technical. A more high-level explanation in a short introductory document documentation would be necessary in the future. This can be materialised as a user manual for the OWL ontology and the related artefacts, and as a user manual for the transformation scripts.

Lessons Learned and Best Practices

Well defined objectives and scope along with a high level of commitment was decisive for the success of this project. All tasks were conducted in a stepwise manner starting with initial proposal, discussions and clarification of the scope and objectives leading to agreement and a shared understanding of what was required.

The Test-Driven Development (TDD) methodology proved to be highly useful as in countless instances when the scripts were modified, either due to bug fixes or changed specifications, the suite of tests guarded the scripts from functional regressions, and led to identification of potential future bugs well in advance during the implementation step. The application of the methodology proved its return on investment and saved time and efforts spent on bug-fixing.

Post-Project Recommendations - Proposals for Future Work

The generated RDF output has been validated by the users however new requirements and needs were identified.

**Support for application profile models as extensions to ontology model.** There is a need to develop a mechanism for interpreting UML models not as formal models but as application profiles. This has an impact on the way the transformations script generates the output artefacts. This will constitute a mechanism for creating extensions and further adaptations of the eProcurement ontology to the Member State models and possibly nuanced constraints in the existing information systems.

**Modularisation of the generated ontology.** The output artefacts (core ontology, shapes, and restrictions) are monoliths that further need to be broken down into modules based on various criteria, such as extension of externally defined ontologies, management of controlled lists, segregation based on package structure, and other criteria.

**Ontology assessment.** The fitness of the formal ontology needs to be validated in the light of the existent data. In another exercise performed by project partners, XML datasets are transformed into RDF datasets instantiating the eProcurement ontology. An assessment exercise should be conducted to validate, on one hand, the suitability of the ontology for expressing the existent data, and on another hand, to validate the existent data in terms of its completeness and conceptual richness with respect to the ontology. Moreover, deriving new statements by the virtue of reasoning capabilities provided by modern triple stores shall be tested and investigated.

**Data validation service.** A dedicated validation service endpoint can be established to validate RDF datasets instantiating the eProcurement ontology. The validation performed shall conform to the SHACL shape definition files generated from the UML conceptual model. By an extension mechanism these data-shape files can be further configured and adapted to fit the needs of the Member States and constraints stemming from the currently implemented information systems. ITB TestBed could be used as the underlying technology, or other available SHACL validators.

**Refactor transformation scripts for broader use.** In the future, the transformation scripts can be refactored to handle more generic UML models, free of the constraints and the conventions adopted in this project. This will increase the reusability of the transformation scripts for other projects. Possibly separating the transformation rules from the UML conventions in order to provide a possibility of configuring what shall be generated in the output artifacts.

**Automatise testing.** As the development process evolved, the number of test suites evolved. Currently the development IDE is Oxygen which can run one test suite at a time. In future work it would be useful to implement a pipeline for automatic test execution in order to facilitate the future development.

**Transformation service.** The transformation scripts currently take the UML conceptual model in XMI format and produce an output of OWL core, SHACL shape and OWL restrictions files, which need to be further loaded into a triple store to be used. In the future it would be useful to develop a pipeline that runs the transformation and uploads the RDF files into a dedicated triple-store.

**User manual for the generated artefacts.** Develop a user manual for the transformation scripts and one for the OWL ontology and the related artifacts.

1. See <https://en.wikipedia.org/wiki/Test-driven_development> [↑](#footnote-ref-2)
2. See <https://github.com/eprocurementontology/eprocurementontology/tree/v2.0.1> [↑](#footnote-ref-3)
3. model2owl - <https://github.com/meaningfy-ws/model2owl> [↑](#footnote-ref-4)
4. SEMIC - <https://joinup.ec.europa.eu/collection/semantic-interoperability-community-semic/about> [↑](#footnote-ref-5)
5. See <https://joinup.ec.europa.eu/collection/semantic-interoperability-community-semic/solution/core-criterion-and-core-evidence-vocabulary> [↑](#footnote-ref-6)
6. See <https://joinup.ec.europa.eu/collection/common-assessment-method-standards-and-specifications-camss> [↑](#footnote-ref-7)
7. See <https://joinup.ec.europa.eu/collection/common-assessment-method-standards-and-specifications-camss/solution/core-assessment-vocabulary-cav> [↑](#footnote-ref-8)