PEPPOL WP2 Pre-VCD Ontology and Reasoning Specification

Wolfgang Groiss, m2n consulting & development gmbh
groiss@m2n.at

History

1. Original Draft
2. Review, adaptation to changes from ontology modelling workshops, adaptation to new reasoning concepts. January 2010

1 General

The Ontology is split into five logical parts as shown in Illustration 1.

Three of those parts (criterion-schema, tenderer-schema, tenderer-criterion-schema) contain the schemas for specifying criteria, evidences, tenderer structure and the relationships between them.

The fourth part (collector-schema) contains the schema for specifiying input and output for the reasoning steps.

The last part (common, not included in the illustration) contains some common classes and properties used by the other four.

These five parts are provided as separate RDF/XML files in conjunction with this document. There are additional files included as examples.

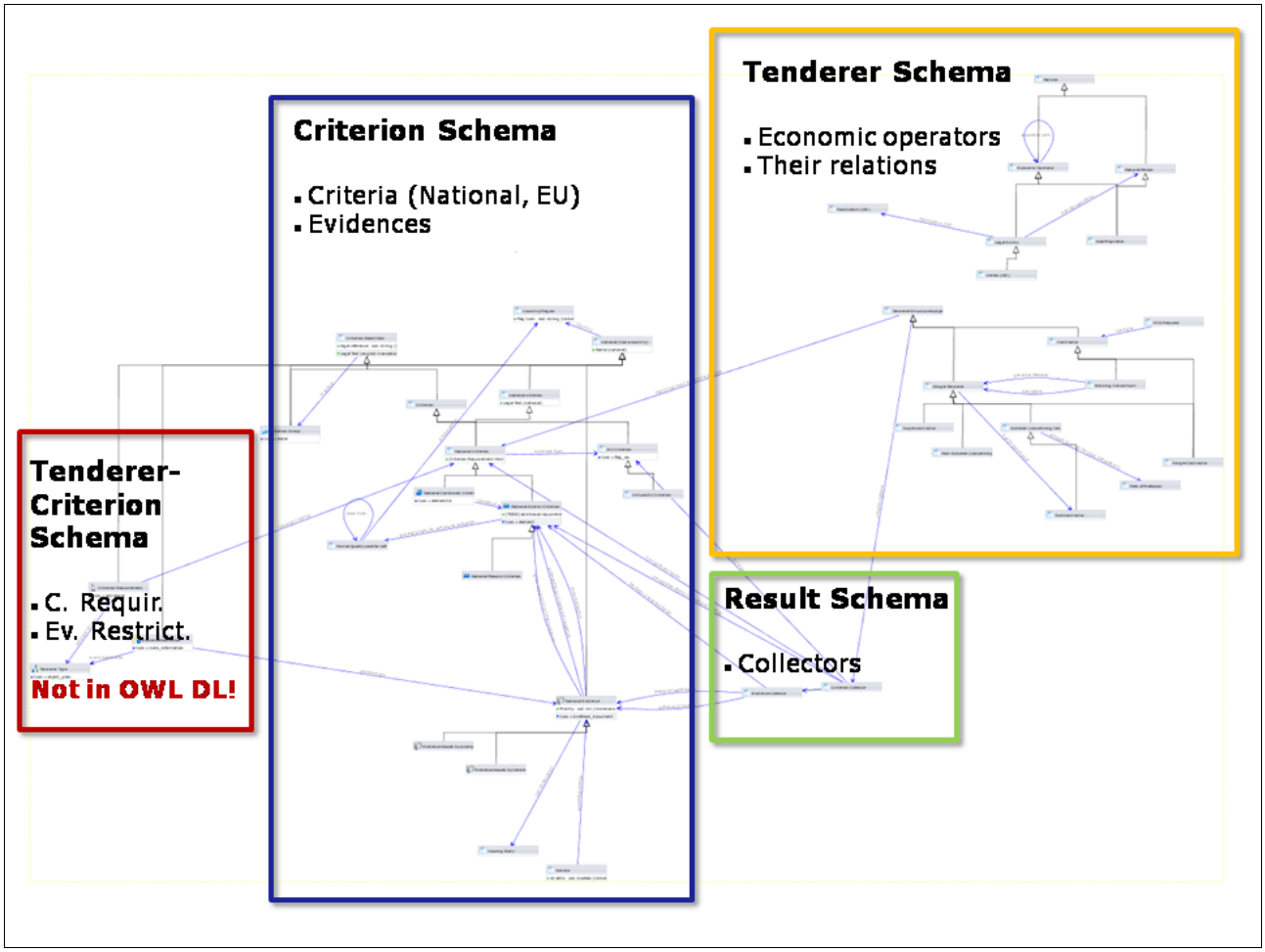

Illustration 1: The different ontology parts

2 Common

This schema contains some classes and properties that will be used throughout the rest of the ontology.

2.1 Abstract vs. Instantiable

The classes owlx:AbstractClass and owlx:InstantiableClass, both being subclasses of owl:Class, are used to distinguish between classes for which we expect instances to exist and classes for which we don't expect any instances (except for inferred type statements).

2.2 Classname and Propertyname

The properties owlx:className and owlx:propertyName are both subproperties of rdfs:label and are used to give human-readable names to classes and properties in the ontology.

2.3 Named

The class peppol:Named is intended to be subclassed by classes whose instances have a simple name. This reduces the amount of re-occuring „Name“-properties and corresponding subproperty relationships to rdfs:label.

2.4 Annotating Open Tasks

The properties peppol:TODO, peppol:assignee, peppol:TODOAnswer and peppol:finished are provided to allow simple and easy annotations of open tasks, the responsible party, their remarks regarding the task and whether the task is finished respectively.

2.5 National Objects

The class peppol:Country is provided to track countries/regions/“ontology domains“. The abstract classes peppol:NationalThing and peppol:NationalCriterionThing allow tieing other entities to a country. NationalThings have a special property for their name in the national language. NationalCriterionThings additionally have a property for the legal text (of the national criterion) in the national language.

3 Criterion Schema

This file defines the schema for criteria and evidences.

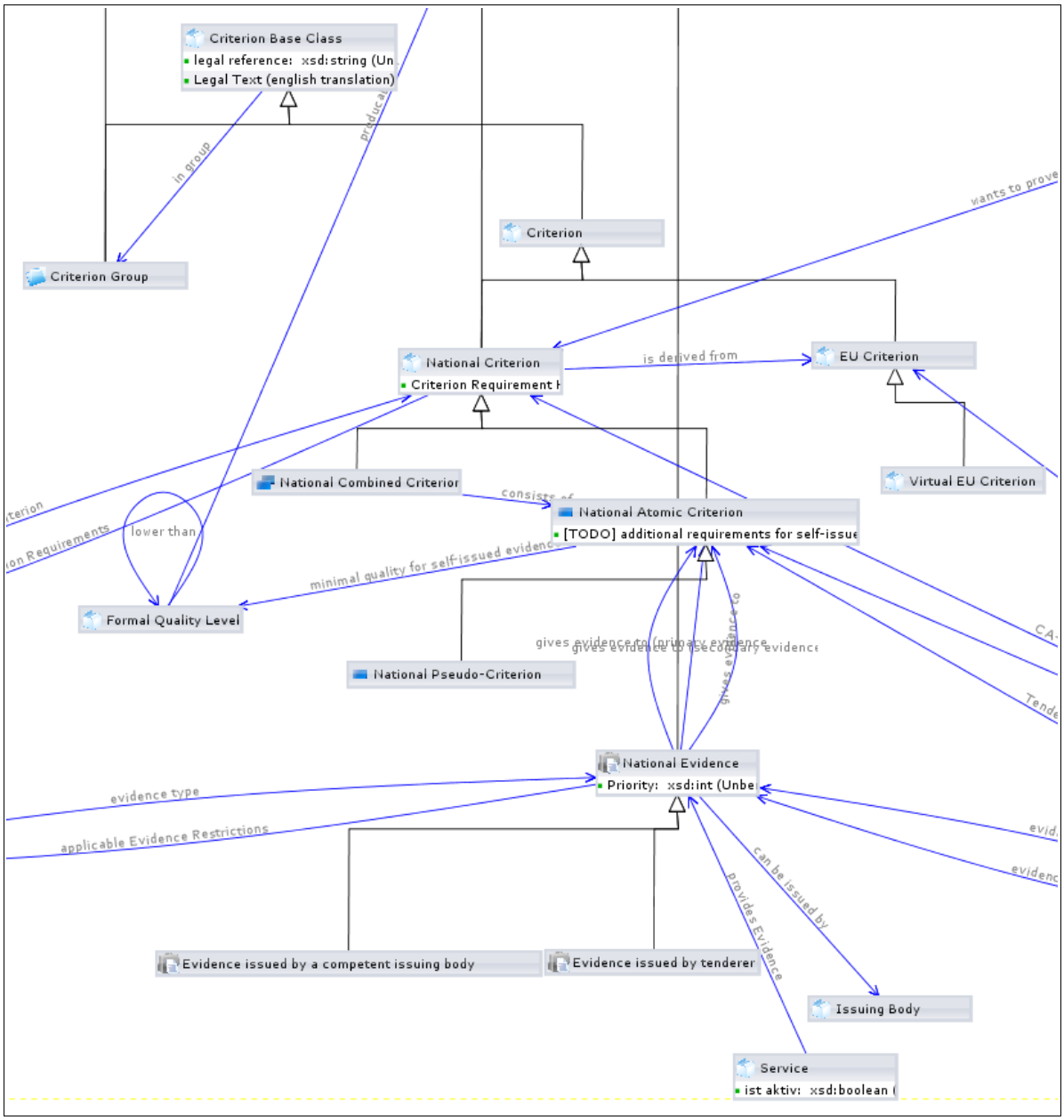

Illustration 2: The Criterion Schema

Criteria can be grouped hierarchically – using the class CriterionGroup – to model the structure of a legal document corpus. References into such corpora can be given at any level of the structure.

There are five instantiable classes for criteria in this schema: EUCriterion, NationalAtomicCriterion and NationalCombinedCriterion as well as VirtualEUCriterion and NationalPseudoCriterion.

Virtual EU Criteria are Criteria that are not reflected in the EU directive and are only included to allow mapping of national criteria that could not be mapped otherwise.

National Pseudo-Criteria are used where there's no correspondence in the national

law, but there is still an EU Criterion that might need to be proven.

Combined Criteria are only possible on a national level. In order to prove a combined criterion, a tenderer always has to prove all of its subcriteria. So in combination with multiple evidences suitable for proving a single criterion, three cases can be modelled:

- Subcriteria of Combined Criteria are always AND (all subcriteria have to be proven)
- Multiple Evidences on one Criterion are always OR (exactly one of them has to be delivered in order to prove the criterion).
- It is possible to combine those cases: Given a Combined Criterion CC, consisting of Atomic Criteria AC1 and AC2, where AC1 can be proven by Evidences E11 or E12, and AC2 by Evidences E21 or E22. Suitable combinations to prove CC would then be (E11, E21), (E11, E22), (E12, E21) and (E12, E22)

Evidence can be distinguished into two types:

- Evidence issued by a competent issuing body
- Evidence issued by the tenderer

Evidences can be assigned a priority. If several evidences would work in a given situation, this means that the highest-priority one should be used, if possible. This works well in conjunction with Evidence Restrictions, for example if there is an Evidence A that can only be provided by Companies satisfying some criterion X, and an Evidence B that can be provided by everyone.

We also distinguish between primary and secondary evidence, where primary evidence is „backed by law“ whereas scondary evidence is NOT backed by law but often still accepted by CAs. The two properties „gives evidence to (primary evidence)“ and „gives evidence to (secondary evidence)“ are subproperties of the more general „gives evidence to“. To reduce visual complexity, this has been omittet from the diagram.

Tenderer-issued evidence can be issued on several different quality levels:

1. Declaration on Oath
2. Solemn Statement
3. Self-Declaration

A solemn statement can also be used in place of a self-declaration, and a declaration on oath could be used in place of any of the other two.

If a tenderer can't produce third-party-issued evidence for a required criterion – for example because there is no such evidence in his country – he will instead substitute self-issued evidence.

(CA) countries can define for each criterion the „Substitution Level“ that is required for tenderer-issued evidence to be acceptable as proof for that category. If a tenderer can't provide third-party-issued evidence for the category, he will instead provide self-issued evidence of that level (or higher, e.g. if the CA country requires a solemn statement, the tenderer could also provide a declaration on oath).

We also provide a property „Criterion Requirement Hint“, in which Hints for the User as to when certain criteria will typically be required can be provided for cases

when the business logic cannot be easily modelled with CriterionRequirements.

3.1 Additional validation rules

There are a few validation rules that cannot be modelled using OWL and are currently not formalized in this schema:

- Each EU criterion has to have – in each country – exactly one national criterion.
- Each national atomic criterion has to have at least one evidence.

4 Tenderer Schema

The Tenderer Schema specifies how a tenderer structure is represented in the ontology.

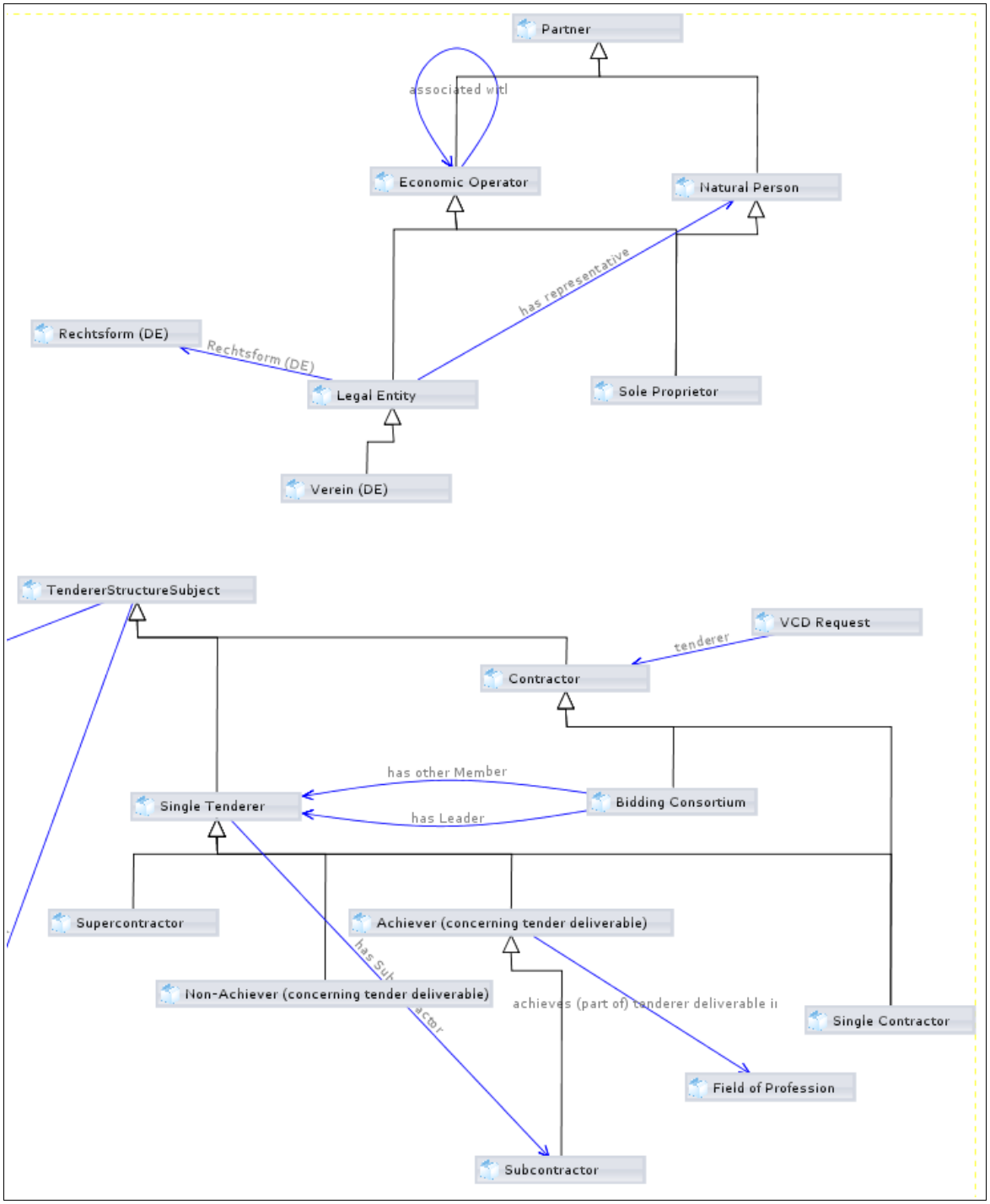

Illustration 3: The Tenderer Schema

Different aspects of a tenderer are modelled in distinct classes here. As denoted, every instance of (a subclass of) TendererStructureSubject has to have a second asserted type which must be a subclass of Partner.

This models the two orthogonal dimensions of companies in the tendering process. One one hand, they have characteristics that describe them as a company (e.g. their

representatives), on the other hand they have characteristics that describe their role in the whole tender (eg. they can be leader of a consortium, or they could be a subcontractor to some other company).

The fact that someone wants to create a VCD is modelled by the class VCD Request.

The structure of a tenderer – Consortia, Subcontractorships, Fields-of-Profession relevant for the tender – are modelled by the class TendererStructureSubject and its subclasses. This part of the schema describes how different companies (plan to) cooperate for the tender at hand.

The structure of companies is modelled by the class Partner and its subclasses. These classes are concerned with different types of companies (mainly SoleProprietor vs. LegalEntity), associations between companies and the natural persons representing those companies.

All the classes in this schema are instances of the class TendererType specified in the tenderer-criterion-schema so they can act as Instances for the tenderer-criterion-schema. Please note that this breaks OWL DL conformance as discussed in section 5. This can be remedied by following the reasoning workflow described in section 7.3.

It is highly probable that this schema will be extended individually for specific Evidence Restrictions. The system should provide OWL reasoning to allow the expressivity that is probably required for this. Based on our experience so far, we assume that Criteria Requirements are more general and can be expressed in terms of the common european tenderer schema.

Components dealing with instances of the tenderer schema (especially user interaction components) will have to deal with these national extensions reasonably.

5 Tenderer-Criterion-Schema

This is the schema for two distinct kinds of rules.

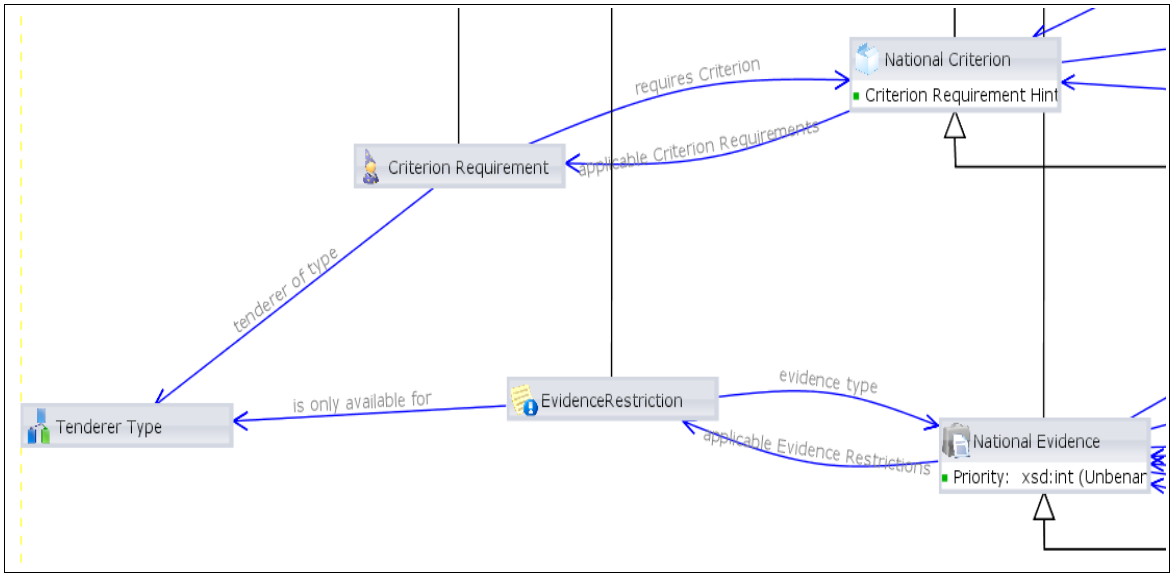

Illustration 4:The Tenderer-Criterion-Schema

First, we have the rules that govern which criteria have to be proven by whom. Those are called the criterion requirements and are represented by the class CriterionRequirement. These are only required if we want to provide a suggestion of probable criteria based on tenderer data; they are not needed if we expect the tenderer to provide the criteria he wants to prove by himself.

The second kind of rule specified by this schema governs limitations of availability of evidences. For example, criminal records might only be available for natural persons, but not for legal entities. These rules are called evidence restrictions and are represented by the class EvidenceRestriction.

The class TendererType used in this schema is the class of which all the classes in the tenderer-schema are instances. This fact (classes as instances of classes other than owl:Class) breaks a OWL DL restriction discussed in http://www.w3.org/TR/2004/REC-owl-guide-20040210/#DesignForUse. Hence, our ontology as described ends up being in OWL Full.

The classes NationalCriterion and NationalEvidence referenced in this schema are specified by the criterion-schema.

Extending the tenderer-schema by providing for example more specific subclasses is probably neccessary to specify exact tenderer types for these rules.

6 Examples

A set of examples for the different schemas described so far is provided in this section. These examples are intended to show how the schema would be applied to instances and how those instances would be related to each other in terms of the schema.

6.1 criteria-and-evidences-example.owl

This example shows how the classes defined in the criterion-schema are used to model both the EU directive and national public procurement acts (NPPAs) down to the level of EU Criteria and National Atomic Criteria, as well as Evidences.

This example also includes criteria requirements and a simple limitation of availability for evidence types so it also demonstrates the usage of the tenderer-criterion-schema.

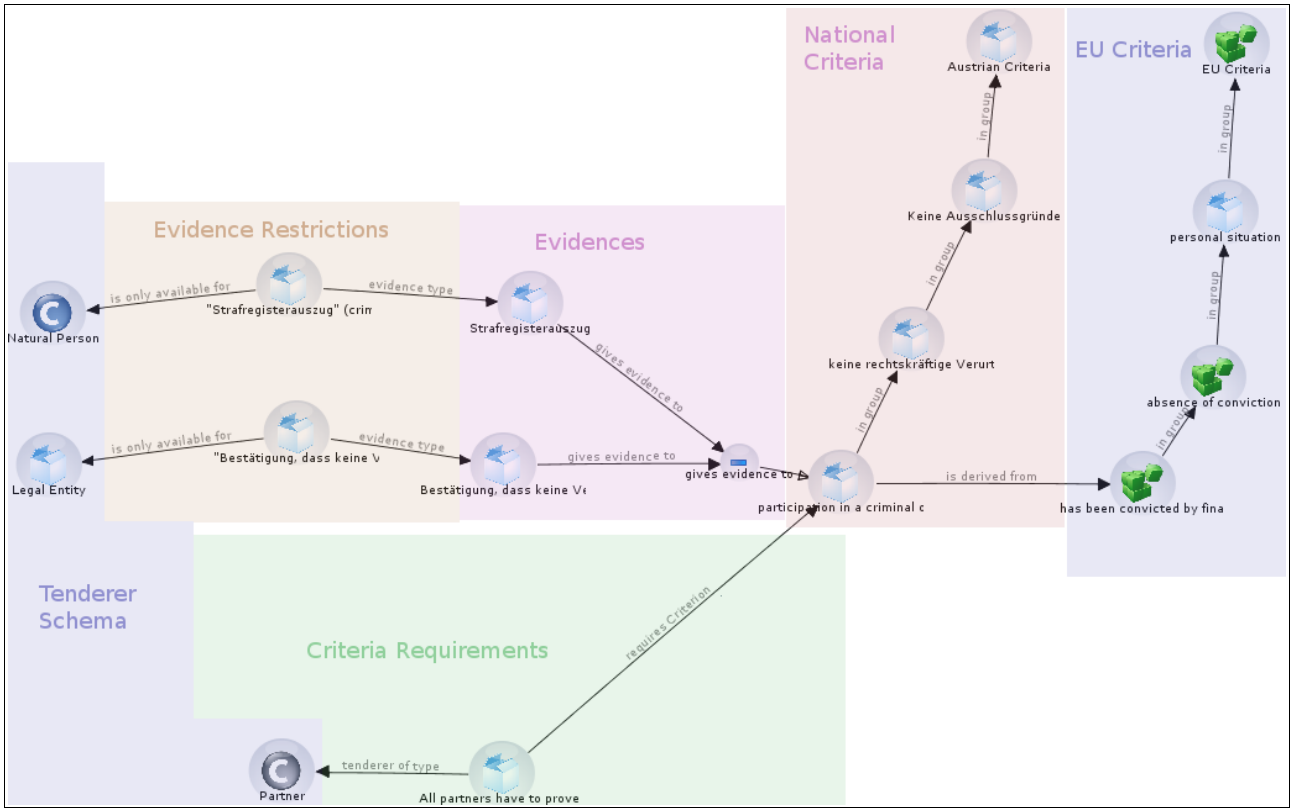

Illustration 5: Example of Instances across the schemas

This diagram shows how the different schemas interact in the example.

6.2 Reasoning Example

Section 7.5 gives a simple example that shows how the entities shown in section 6.1 are used to infer criteria and evidences for a given tenderer structure.

7 Reasoning on Instances

In this chapter we present rules that formalize our intended reasoning for different use cases.

In any case, OWL reasoning is required to allow specification of specialized tenderer-schema subclasses for specific Criteria Requirements and Evidence Restrictions.

While it is probably technically possible to provide all reasoning we want in pure OWL by providing suitable meta schemas, this would lead us even further from OWL DL and would also be overly complex given our goals. Instead, this chapter will specify simple rules that govern reasoning steps for deriving possible Evidences from Criteria of the Contracting Authority's nationality.

The output of one reasoning step can be treated as input to the next one, with possible further modification of the intermediary results by the user. An example for such an interaction would be a selection (and possible addition) of criteria intended to be proven from a set of suggestions.

This can be viewed as the invocation of different reasoning services that extend the supplied input data with inferred triples. The arrangment of calls to those services and of steps of user interaction into a meaningful sequence is then called the „Reasoning Workflow“ and detailed in Section 7.3.

7.1 Use Cases

This chapter will provide a very simplistic view of the use-cases and the involved asserted data as well as the expected inferred data.

7.1.1 Ontology Maintenance

These are presented here in a very simplified way and serve only to show where the ontology data comes from.

7.1.1.1 Maintenance of EU Criteria

Instances of the classes CriterionGroup and EUCriterion are provided/changed.

7.1.1.2 Maintenance of national Criteria

Instances of the classes CriterionGroup, NationalAtomicCriterion and NationalCombinedCriterion are provided/changed.

7.1.1.3 Maintenance of Evidences and Evidence Restrictions

Instances of the classes NationalEvidence and EvidenceRestriction are provided/changed. The tenderer-schema may be extended to allow specification of certain kinds of Tenderers for Evidence Restrictions.

7.1.1.4 Maintenance of Criteria Requirements

Instances of the class CriterionRequirement are provided/changed. The tenderer-schema may be extended to allow specification of certain kinds of Tenderers for Criterion Requirements.

7.1.2 Generation of Required Criteria for a provided Tenderer Structure (probably not in ESP)

Instances for classes from the tenderer-schema are provided.

The national criteria for the countries of the tenderer and the contracting authority, the national evidences and evidence restrictions for the country of the tenderer and the criteria requirements for the country of the contracting authority are loaded as well as the EU criteria and any commonly needed ontology parts.

For each instance provided in the first step, a list of required criteria in the country of the contracting authority as specified by the criteria requirements is inferred.

7.1.3 Deriving Criteria from Contracting-Authority-National Evidences

While this has come up as a potential use case for the reasoning, this will not be handeled by the reasoner. Instead we suggest that the Evidences can be used to let the user search for Criteria in the interaction step.

7.1.4 Generation of possible Evidences for a provided Tenderer Structure with required Criteria

Instances for classes from the tenderer-schema are provided as well as for each of those instances the criteria (in the country of the contracting authority) intended to be proven. This can be the outcome of the use case described in section 7.1.2, possibly after a user interaction where the user adds/removes some of the criteria to be proven.

The national criteria for the countries of the tenderer and the contracting authority and the national evidences and evidence restrictions for the country of the tenderer are loaded as well as the EU criteria and any commonly needed ontology parts.

For each criterion provided in the first step, a list of possible chains of CA-national-criterion → EU-criterion → T-national-criterion → Evidence for each of the evidences suitable for proving the criterion and available to the Tenderer Structure Element is inferred.

7.2 Collector Schema

The Collector Schema provides the classes and properties used as input and output for the reasoning deriving Evidences from required Criteria (as described in section 7.1.4).

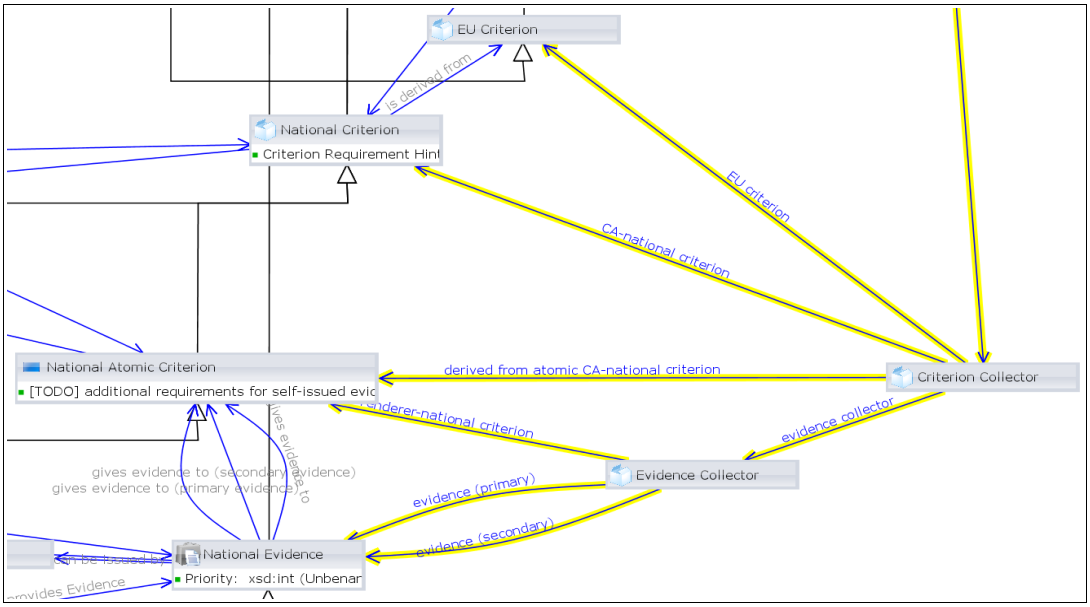

Illustration 6: The Collector Schema

The input for the reasoning is specified by linking the Tenderer Structure Subjects to the criteria we want to prove for them with the wantsToProveCANC property.

The output of the reasoning will be grouped into so-called collectors. The collectors are structured so that they show the actual path from ca-national criteria via eu criteria to t-national criteria and finally evidences.

The reasoning process for deriving evidences from ca-national criteria works like this:

1. A criterion collector for every wantsToProveCANC relationship is created. Atomic criteria that belong to the same combined criterion are grouped into a single collector.
2. The CA-national Criterion to be proven is added to that collector, along with the corresponding EU criterion.
3. For every Tenderer-national Atomic Criterion mapped (directly or via combined criteria) to that EU criterion, an evidence collector is added to the criterion collector, with the Tenderer-national Atomic Criterion added to it.
4. Finally, all evidences suitable to prove that criterion (respecting Evidence Restrictions) are added to the evidence collector.

7.3 Reasoning Workflow

In order to fulfill meet all the requirements, the reasoning will be split up into several steps. Between some of the steps, we expect the user to modify the RDF data, for example to add/remove criteria to be proven. This interaction is *not* part of the reasoner itself.

By carefully choosing the sets of schemas and data that are loaded at each step, the limitations with regard to OWL DL reasoning can be worked around: if nothing that violates OWL DL (specifically, the tenderer-criterion-schema and its instances) is loaded, OWL DL reasoning on the tenderer data can be performed, the output of that reasoning step can be captured and used as input for the other reasoning steps.

This workflow is outlined in Illustration 7: The reasoning workflow. This illustration shows (in the third swimlane, „Process“) the process involving user

interactions and reasoning steps. In that process, the second, third and fifth step are performed by the reasoner while the first and third step will be performed by the user and controlled by some other component.

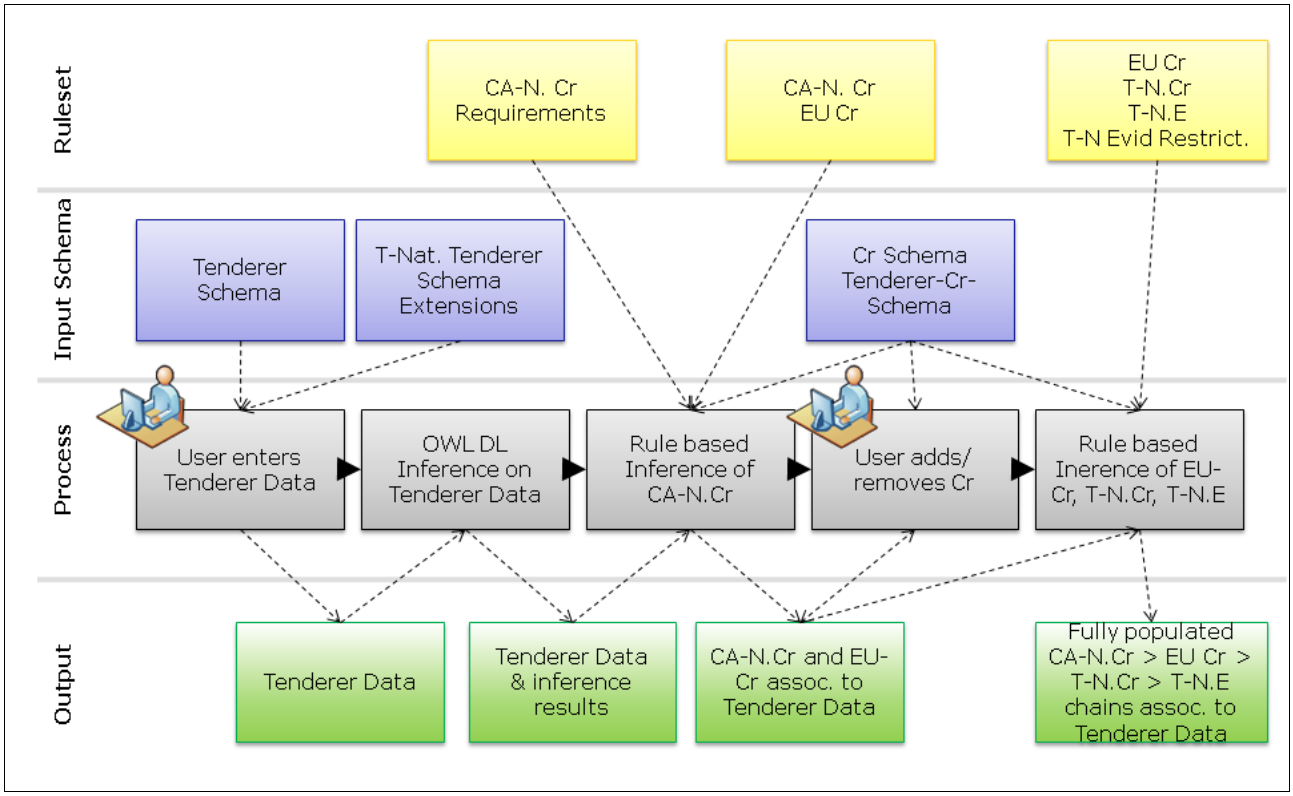

Illustration 7: The reasoning workflow

7.4 Inference Rules

This chapter describes the rules required for the use cases from chapters 7.1.2 and 7.1.4.

We assume OWL DL reasoning has already been performed on the tenderer data and the results of that (especially any additional type statements for instances of the tenderer-schema classes) are available as input to the rules described here.

7.4.1 Rule Syntax

This chapter describes the syntax for the rules. The syntax isn't specified formally yet and this description of the syntax can only serve as a guideline to help the reader understand the rules shown later on, not as a formal specification of the rule semantics.

Rules consist of preconditions and results, with the preconditions on the left, then a `=>` and finally the results. Preconditions and results are sets of triple patterns. A single triple pattern is denoted as `(<s> <p> <o>)`, where `<s>` denotes the subject, `<p>` the predicate and `<o>` the object of the RDF triple. A predicate of `a` is shorthand for `rdf:type`.

Each element of a triple can be either a URI node, a variable node or a creation-variable node.

URI nodes are denoted as simple strings, variable nodes are strings prefixed with a `?` and creation-variable nodes are strings prefixed with a `$`.

URI nodes can appear anywhere in a rule. Variable nodes can only appear in the results of a rule if they have appeared in the preconditions of that rule. Creation-Variable nodes can only appear in the results of a rule.

Rules are applied by trying to bind variable nodes in the preconditions to URI nodes in the base asserted data so that the preconditions are all satisfied. Variable bindings from one triple in the preconditions of a rule carry over to all other triples in that rule. When all preconditions are met, the triple patterns in the results – with the variable nodes replaced according to the variable bindings in the matching step – are emitted. When emitting, each creation-variable node in the results triple patterns is replaced by a single newly-created URI node.

Apart from triple patterns, results can also contain nested rules that become active when the preconditions of the outer rule are met. A nested rule can include an ELSE clause that will be used to produce triples if the preconditions of the nested rule are NOT met (but the preconditions of the outer rule are met). Variable bindings from the outer rule carry over to the nested rule.

Nested rules can be denoted in `[]` or by giving them a name in caps with variables to be carried over as parameters in `()`.

7.4.2 Deriving suggested criteria from criteria requirements (see 7.1.2)

```
(?x a ?c) (?req a CriterionRequirement) (?req tendererType ?c) (?req requires ?crit) => (?x wantsToProveCANC ?crit)
```

7.4.3 Deriving Tenderer-National Criteria from Contracting-Authority-National Criteria

Input for this step would be either the (possibly interactively modified) results from 7.4.2 or data provided by the tenderer himself expressed in the same schema.

7.4.3.1 Building the criterion collectors (reducing atomic to combined CA-national criteria if possible)

```
(?x wantsToProveCANC ?canCrit) => [

  (?canCrit a AtomicCriterion) (?canCombined consistsOf ?canCrit) => CRIT_COLLECTOR(?x, ?canCombined, CriterionCollector, criterionCollector, canCriterion)

  ELSE CRIT_COLLECTOR(?x, ?canCrit, CriterionCollector, criterionCollector, canCriterion)

]

COLLECTOR(?subject, ?criterium, ?type, ?subjectToCollector, ?collectorToCriterium): (?subject ?subjectToCollector ?alreadyExistingCollector) (?alreadyExistingCollector ?collectorToCriterium ?criterium) => () ELSE (?x ?subjectToCollector $collector) ($collector ?collectorToCriterium ?criterium) ($collector a ?type)

(?x wantsToProveCANC ?canCrit) (?canCrit a AtomicCriterion) (?canCombined consistsOf ?canCrit) (?x criterionCollector ?cc) (?cc canCriterion ?canCombined) => (?cc derivedFromAtomicCANCriterion ?canCrit)
```

7.4.3.2 Deriving TN-Criteria via EU-Criteria (reducing combined T-National Criteria to atomic) and building the evidence collectors

```
(?cc canCriterium ?canCrit) (?canCrit isDerivedFrom ?euCrit) (?tnCrit isDerivedFrom ?euCrit) => (?cc
euCriterium ?euCrit) [

  (?tnCrit a CombinedCriterion) (?tnCrit consistsOf ?tnSub) => COLLECTOR(?cc, ?tnSub, EvidenceCollector,
evidenceCollector, tnCriterion)

  ELSE COLLECTOR(?cc, ?tnCrit, EvidenceCollector, evidenceCollector, tnCriterion)


  (?canCrit minimumSubstitutionLevel ?sl) => (?cc minimumSubstitutionLevel ?sl)

]
```

7.4.4 Deriving Evidences from Tenderer-National Criteria

```
(?tse criterionCollector ?cc) (?cc evidenceCollector ?ec) (?ec tnCriterion ?tnc) (?ev givesEvidenceTo ?tnc)
=>[
  (?er a EvidenceRestriction) (?er evidenceOfType ?ev) => RESTRICTED(?tse, ?ec, ?ev, ?er)
  ELSE (?ec evidence ?ev)
]

RESTRICTED(?tse, ?collector, ?ev, ?er): (?er isAvailableFor ?tType) (?tse a ?tType) => (?collector evidence
?ev)

  (?cc minimumSubstitutionLevel ?sl) => (?ec minimumSubstitutionLevel ?sl)
```

7.5 A (simplified) Example of the Reasoning Process

This section gives an example of how the reasoning process works. Illustration 8: An example of the reasoning process shows the process steps on top, and a simplified illustration of the input and output data of the process steps below. Color-coding ties the process steps to their output.

We assume the tenderer is Austrian, and the contracting authority Italian.

The process begins with the user providing the tenderer structure (gray): A single tenderer, m2n, which is also a legal entity and has a business register number 183286p, and a legal representative, Doris, who is a natural person.

On this data, OWL DL reasoning is performed (yellow), yielding additional types: m2n must be a registered business because it has a business register number. Doris must be a representative (rdf:type) because it occurs as the object of the „has representative“ property. To save space, we don't show in the example illustration that and how those rules are encoded in the austrian tenderer-schema extensions using OWL DL constructs.

The result of this step (gray and yellow) is now input into the first rule-based inference step. This results in the creation of criterion collectors (blue). This corresponds to sections 7.4.2 and 7.4.3.1 above. The output of this step is shown in blue.

The next step is a user interaction, where parts of the results from the previous step (shown in blue) can be removed by the user, and similar parts added. (Another, probably better, possibility would be to have this user interaction between the steps described in sections 7.4.2 and 7.4.3.1, simplifying the schema that the user has to deal with a bit, but making our illustration even more complex.)

For simplicity's sake, we assume that the user was perfectly satisifed with the results and didn't change anything.

The last step now performs the reasoning described in 7.4.3.2 and 7.4.4, resulting in evidence collectors tied to the tenderer-national criteria (green) and ultimately the

evidences (light green).

We can't show in the illustration that „Business Register Extract“ is tied via an evidence restriction to the class „Reg. Business (.at)“, and that „Criminal Record Extract“ is tied to „Natural Person“. Those evidence restrictions are satisfied by the tenderer data, and therefore the evidences are included in the evidence collectors. Any potential evidences whose evidence restrictions are not satisfied will be excluded from the evidence collectors and are hence not shown (e.g. a special kind of criminal record extract only available for companies and not for natural persons).

Evidence collector 1 illustrates that this can result in having multiple evidences for a single evidence collector, implying that the user is free to choose one of the evidences (but might be bound by their relative priorities and their primary/secondaryness). That choice is outside of the scope of the reasoning process described here but will probably goverened and controlled by the same component that controls this process.

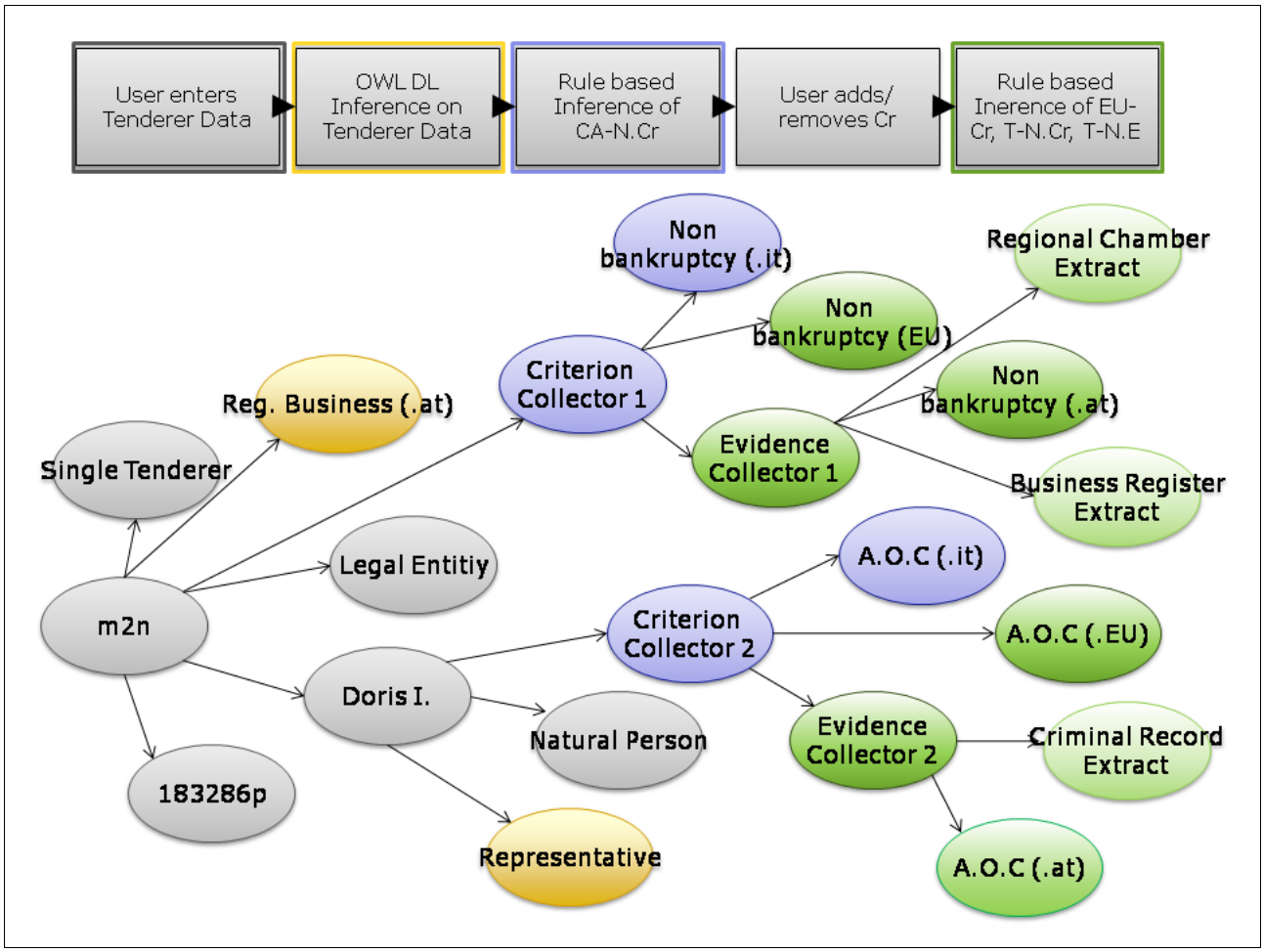

Illustration 8: An example of the reasoning process