

# Alignment of Core Vocabularies with Schema.org

## 1. Context

This document describes the steps taken to align Core Vocabularies with Schema.org within the SEMIC action.

It is assumed that the reader is familiar with the main concepts of Linked Data and the Core Vocabularies, nevertheless acronyms and abbreviations are described at the end of the document.

For the purposes of an alignment, the following definitions are given<sup>1</sup>:

- **Ontology matching:** the process of finding relationships or correspondences between entities of different ontologies.
- **Ontology alignment:** a set of correspondences between 2 or more ontologies, as outcome of ontology matching process.
- **Ontology mapping:** the oriented, or directed, version of an alignment, i.e., it maps the entities of one ontology to at most one entity of another ontology.

## 2. Process

Within this section, the reader can find the details of the process followed to align Core Vocabularies with Schema.org; such process could be applied to other data models.

The alignment process is divided into 6 steps<sup>2</sup>:

1. Staging: defining the requirements
2. Characterization: defining source and target data and performing data analysis
3. Reuse: discover, evaluate, and reuse existing alignments
4. Matching: execute and evaluate matching
5. Align and map: prepare, create the alignment, and render mappings
6. Application: make the alignment available to be reuse by applications

---

<sup>1</sup> [https://www.researchgate.net/publication/319160792\\_A\\_Multi-strategy\\_Approach\\_for\\_Ontology\\_Reuse\\_Through\\_Matching\\_and\\_Integration\\_Techniques](https://www.researchgate.net/publication/319160792_A_Multi-strategy_Approach_for_Ontology_Reuse_Through_Matching_and_Integration_Techniques)

<sup>2</sup>

[https://www.academia.edu/70179066/Towards\\_a\\_Metadata\\_Model\\_and\\_Lifecycle\\_for\\_Ontology\\_Mapping\\_Governance](https://www.academia.edu/70179066/Towards_a_Metadata_Model_and_Lifecycle_for_Ontology_Mapping_Governance)

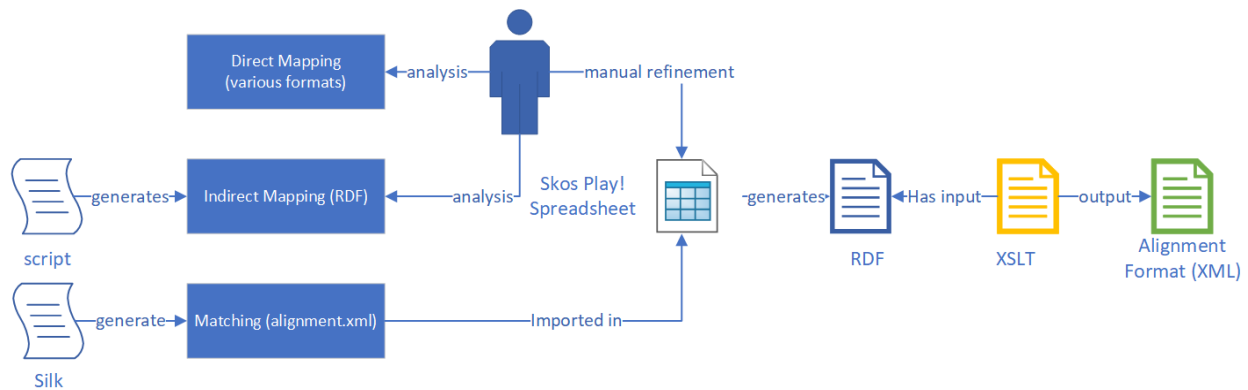


Figure 1: Overall alignment process

The image above describes the overall process which is semi-automatic, and it depends on 3 types of mappings: direct, indirect and mappings coming out of the matching process.

Direct and indirect mappings (see section “2.3 Reuse”) are manually analysed; in particular a dedicated [script](#) has been developed to find indirect mapping including the automatic querying of DBpedia and Wikidata.

Mapping generated out of the matching process (outcome of the Silk Workbench, see section “2.4 Matching”) are imported into a spreadsheet (see section “2.5 Align and map”), where manual refinement is performed considering the analysis done on direct and indirect mapping.

The spreadsheet is then converted into an RDF, which in turn, is transformed back into the Alignment Format thanks to an XSLT file.

## 2.1 Staging

Schema.org has been chosen as part of the alignment because of different requirements raised over time for different Core Vocabularies and Application Profiles:

- Core Business (request for alignment): issue [#38](#)
- Core Public Event (discussion over dependency): issue [#11](#)
- CPOV (proposal for alignment): issue [#10](#)
- CSPV-AP (request for mapping relation): issue [#109](#)

## 2.2 Characterisation

In the below table a comparison between Core Vocabularies and Schema.org is given:

	Core Vocabularies	Schema.org	Comment
<b>Specification</b>	HTML per core vocabulary	HTML per entity type	Given that both specifications are represented in RDF, the HTML is not used for the matching process.
<b>Format</b>	UML, RDF, JSON-LD context, SHACL	<a href="#">RDF and CSV</a> (1 for Types and 1 for Properties)	Being both expressed in RDF, they can be easily compared.
<b>Deprecation</b>	Not yet	Yes, with schema:supersededBy	Deprecated terms are excluded from the mapping.
<b>Inheritance</b>	Single	Multiple	Inheritance mechanisms are important to consider when applying mapping relations as new relations can be created.
<b>Language</b>	English	English	Being in the same language, there is no loss of meaning that could happen when translating.
<b>Labels</b>	rdfs:label, shacl:name (within SHACL shapes)	rdfs:label	It is important to consider labels when performing the matching process.
<b>Label format</b>	“lower camel case” with spaces (e.g. “post code”)	“lowerCamelCase” (e.g. “postalCode”)	It is important to consider the label format when performing the matching process.
<b>Definitions</b>	shacl:description (within SHACL shapes)	rdfs:comment	Definitions are considered to refine the mapping candidates.
<b>Version</b>	Per vocabulary, around 1 or 2 releases per year	15.0 (25/10/2022), 2 releases per year	The release schedule sets the minimum frequency for

			which mappings should be updated.
--	--	--	-----------------------------------

Schema.org is considered as the target model of a mapping. This brings important considerations as it implies the direction of the mappings that will need to be expressed via well-defined relations; see section “2.5 Align and map” for more details.

Setting Core Vocabularies as the source and Schema.org as the target will facilitate reusers that would want to transform data expressed with Core Vocabularies into Schema.org.

## 2.3 Reuse

Before creating any mapping, 2 types of existing mappings have been considered:

1) Direct mappings to create up-to-date ontology mappings:

Mapping From	Mapping To	Location	Date	Format
Core Location (23 March 2015)	Schema.org	<a href="http://Geodcat-ap.semic.eu">Geodcat-ap.semic.eu</a>	-	HTML table with SKOS mapping relations
	Schema.org	<a href="https://github.com/SEMICEu/locn-mapping">https://github.com/SEMICEu/locn-mapping</a>	03/06/2018	RDF described as: •Alignment Format (+ADMS) for overall •SKOS mapping relations
	<a href="http://Schema.org/v3.6">Schema.org v3.6</a> (01/05/2019)	<a href="https://semiceu.github.io/locn-mapping">https://semiceu.github.io/locn-mapping</a>	13/09/2019	•HTML table •SPARQL construct query per entity

Core Vocabularies (including Core Location), DCAT-AP, ADMS, CPSV-AP	Schema.org	Previous mappings created within SEMIC action	05/09/2017	Excel with SKOS mapping relations
---	------------	---	------------	-----------------------------------

2) Indirect mappings to create new ontology mappings:

Mapping From	Mapping To	Location	Date	Format
Schema.org	Dublin Core	<a href="https://github.com/dcmi/schemaorg_2012">https://github.com/dcmi/schemaorg_2012</a>	25/05/2012 (public archive)	•HTML table •RDF (using RDFS and OWL assertions)
DBpedia	Schema.org	<a href="https://dbpedia.org/sparql">https://dbpedia.org/sparql</a>	-	•HTML •RDF (using RDFS and OWL assertions)
Wikidata	Schema.org	<a href="https://github.com/okfn-brasil/schemaOrg-Wikidata-Map">https://github.com/okfn-brasil/schemaOrg-Wikidata-Map</a>	25/10/2017	•CSV
Wikidata	Schema.org	<a href="https://query.wikidata.org/sparql">https://query.wikidata.org/sparql</a>	-	•RDF

## 2.4 Matching

In addition to existing mappings, candidate mappings have been found by:

1. Enriching Core Vocabularies terms with existing synonyms, via a dedicated [script](#), using the following data sources:
  - API such as [Datamuse](#) and [Altevista](#);
  - RDF files:
    - Thesauri such as [Wordnet](#) and [Wiktionary](#)
    - Taxonomies such as [Unesco](#), [LCSH](#), [STW](#), [Eurovoc](#)
    - Ontologies such as [FIBO](#)
    - Registries such as [LOV](#)

2. Comparing terms and synonyms between Core Vocabularies and Schema.org via the open source [Silk Workbench](#);

The process, repeated for each Core Vocabulary, can be described as the following:

1. A [script](#) takes as input from a Core vocabulary (e.g Core Person):
  - a. The script uses a configuration file (config.yaml) that allows to point out where to find synonyms and what kind of output generate;
  - b. The script queries the API directly and it queries the RDF files stored in a local graph database.
2. The script can generate 2 types of output:
  - a. The Core Vocabulary enriched with synonyms (skos:altLabel);
  - b. The Core Vocabulary enriched with synonyms (skosxl:altLabel) and the data sources; useful to evaluate the data sources over time. On this aspect, it has been noted that the API, Wordnet and LOV were providing most of the synonyms;
3. Silk Workbench takes the SKOS and Schema.org files as input and generates an alignment.xml file which is analysed later.

The image below describes the matching process:

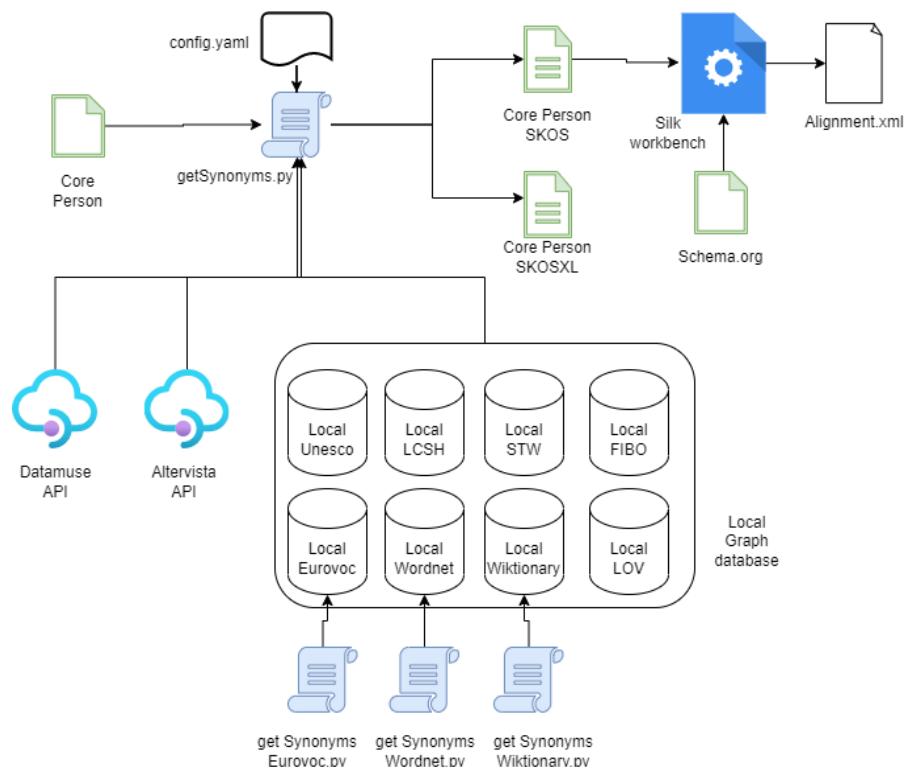


Figure 2: Enhancing matching process with synonyms

In the image below, it is possible to see the workflow setup within Silk Workbench. The workflow has 2 main streams:

- 

Each stream is at the end compared via a similarity string algorithm chosen, in this case “[Q-gram](#)” (Q-gram is one of the string algorithms proposed by Silk), then both are aggregated in order to generate candidate mappings..

links	START	WORKSPACE	ACTIVITIES	EDITOR	EVALUATE	EXECUTE	REFERENCE LINKS
Finished in 5,570s							
<div> <div>▼</div> <div>▲</div> </div> <div>FIRST PREVIOUS 1 NEXT LAST</div> <div>Filter:</div>							
Source: coreperson	Target: schema	Score	Correct				
> http://purl.org/dc/terms/identifier	https://schema.org/identifier	100.0%	✓ ? ✕				
> http://data.europa.eu/m8g/gender	https://schema.org/gender	100.0%	✓ ? ✕				
> http://data.europa.eu/m8g/deathDate	https://schema.org/deathDate	86.1%	✓ ? ✕				
> http://purl.org/dc/terms/alternative	https://schema.org/alternateName	60.8%	✓ ? ✕				
> http://purl.org/dc/terms/alternative	https://schema.org/alternativeOf	60.8%	✓ ? ✕				
> http://www.w3.org/ns/person#placeOfDeath	https://schema.org/deathPlace	47.9%	✓ ? ✕				
> http://www.w3.org/ns/person#placeOfBirth	https://schema.org/birthPlace	47.9%	✓ ? ✕				
> http://data.europa.eu/m8g/birthDate	https://schema.org/birthDate	44.4%	✓ ? ✕				
> http://xmlns.com/foaf/0.1/familyName	https://schema.org/familyName	33.3%	✓ ? ✕				
> http://xmlns.com/foaf/0.1/givenName	https://schema.org/givenName	33.3%	✓ ? ✕				
> http://data.europa.eu/m8g/gender	https://schema.org/lender	33.3%	✓ ? ✕				
> http://data.europa.eu/m8g/gender	https://schema.org/sender	33.3%	✓ ? ✕				
> http://purl.org/dc/terms/alternative	https://schema.org/alternativeHeadline	20.0%	✓ ? ✕				
> http://data.europa.eu/m8g/birthDate	https://schema.org/deathDate	16.7%	✓ ? ✕				
> http://data.europa.eu/m8g/deathDate	https://schema.org/birthDate	16.7%	✓ ? ✕				
> http://data.europa.eu/m8g/domicile	https://schema.org/domiciledMortgage	12.7%	✓ ? ✕				

Figure 4: Candidate mappings found in Silk

## 2.5 Align and map

The result of the matching process, together with the existing (direct and indirect) mappings, is manually refined and collected in a structured spreadsheet that allows to generate the alignment in a machine-readable (RDF) format via the [SKOS Play! Convert](#) service; this process is repeated for all Core Vocabularies.

The spreadsheet is based on a custom [AF-AP](#) data model, developed around the [Alignment Format](#) and [SSSOM](#) vocabularies (see section “3. Data models”), thus the RDF generated is compatible with the Alignment Format with the advantage to be reused by ontology matching tools (such as Silk Workbench) or editors (such as [Vocbench](#)).

The spreadsheet is divided into 2 sheets:

1. **Alignment**, with alignment metadata such as provenance and licensing information.
2. **Mapping**, where mappings have been defined as 1:1 relations, choosing, when appropriate, the closest terms, together with comments and descriptions to clarify the mapping.

The mapping relations expressed in the mapping sheet, are two-folds:

- **Equivalent**: the entities are supposed to be equivalent, i.e., they are supposed to have the same meaning. These are represented via OWL **equivalentClass** and **equivalentProperty** relations.
- **Hyponym/subclass**: the first entity is supposed to be narrower than the second one, i.e., it is a hyponym, or a subclass of the second. These are represented via RDFS **subClassOf** and **subPropertyOf** relations.

This approach allows to generate mappings that can be used to transform directly from Core Vocabularies into Schema.org, for example, by using SPARQL queries. In this sense, hypernym/narrower relations have been excluded as they don't allow generating direct mappings.

In the image below we can see:

- on the left, that the meaning of an entity in Core Vocabularies fits (equivalent or hyponym) into an entity of Schema.org (target model), therefore it is possible to reuse such mappings to perform automatic transformations;
- on the right, the meaning of an entity of Core Vocabularies does not fit a priori into the one in Schema.org, thus such mapping does not allow automatic transformation and manual analysis must be done every time.

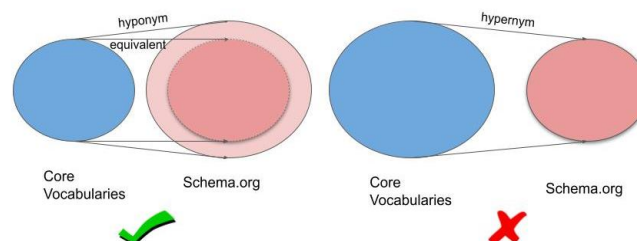


Figure 5: relations used for the mapping



## 2.6 Application

The RDF generated out of the alignment process can be then converted to the AF in XML thanks to a custom [XSLT](#), that could be used as reference, see the overall process described in section “2. Process”.

In this way it is already possible to support ontology matchers tools that can import mappings expressed as AF such as Silk Workbench.

In addition, it would be possible to reuse the alignments in the following way:

1. Store all the mappings in a central database that could be used to find mappings; this could be supported by the Virtuoso instance already setup on [mapping.semic.eu](#);
2. Import such alignment in editor tools, such as Vocbench, to allow editors editing manually such mappings by [creating an EDOAL project](#).

Such alignments could be represented by:

1. A tabular representation of the direct relations that could be embedded as HTML representation next to a vocabulary specification to support reusers of Core Vocabularies;
2. RDF triples (e.g. “X rdfs:subClassOf Y”) that could be imported into other ontologies or used by reasoners (see for example documentation in [Virtuoso](#), [Apache Jena](#) or [GraphDB](#)) to generate new relations to support data modellers when designing ontologies;
3. SPARQL constructs queries, in order to automate transformations of data instances from Core Vocabularies to Schema.org, like the approach taken in the [mapping of Core Location with Schema.org](#).

These types of representation are the most used in different specifications as the below comparative analysis highlights.

As it is possible to see, the most used representations are Text/Table, RDF triples (RDFS/OWL/SKOS), SPARQL queries or just as image. It is observed that most of those specifications prefer to put the mappings in a separate page from the specification itself.

Specification	Alignment						Mappings in Separate page
	Vocabulary					Code list	
	RDFS / OWL	SKOS	Text / Table	SPARQL	Image		
<a href="#">Core Location</a>		<a href="#">Vcard and Schem</a>	<a href="#">Loc-mapping</a>	<a href="#">Loc-mapping</a>			yes

		<a href="#">a.org</a>					
<a href="#">INSPIRE Guidelines</a>	<a href="#">Core Location, GeoSPA RQL</a>						no
<a href="#">W3C DCAT</a>	<a href="#">Schema.org</a>						no
<a href="#">DCAT-AP</a>			<a href="#">Schema.org</a>				yes
<a href="#">GEO-DCAT-AP</a>			<a href="#">Mapping INSPIRE</a>				no
<a href="#">DCAT-AP_IT</a>						<a href="#">Eurovoc and INSPIRE</a>	yes
<a href="#">DCAT-AP_CH</a>			<a href="#">GMD - DCAT</a>				yes
<a href="#">W3C PROV-O</a>	<a href="#">DCTERMS</a>						yes
<a href="#">W3C TIME</a>			<a href="#">Alignment of PROV-O with OWL-Time</a>				no
<a href="#">W3C SHACL</a>			<a href="#">SPARQL</a>				no
<a href="#">W3C Data Quality Vocabulary</a>		<a href="#">ISO IEC 25012</a>					no
<a href="#">DCTERMS</a>			<a href="https://github.com/dcmi/schemaorg_2012">https://github.com/dcmi/schemaorg_2012</a>				yes
<a href="#">FOAF</a>			<a href="#">dct:Agent</a>				
<a href="#">STIR Data</a>						<a href="#">NACE (skos)</a>	no

						<a href="#">example)</a>	
<a href="#">Schema.org</a>	<a href="#">RDF</a> , <a href="#">RDFS</a> , <a href="#">DCTERMS</a> , <a href="#">DCMITY</a> <a href="#">PE</a> , <a href="#">FOAF</a> , <a href="#">ELI</a> , <a href="#">DCAT</a> , <a href="#">SNOMED</a> , <a href="#">BIBO</a> , <a href="#">VOID</a> , <a href="#">MO</a>	<a href="#">ELI</a> , <a href="#">REGO</a> <a href="#">RG</a>	<a href="#">FIBO</a>				no
<a href="#">FHIR</a>			<a href="#">PROV</a>				yes
<a href="#">ELI</a>	<a href="#">Schema.org</a>						yes
<a href="#">RDA Registry</a>	<a href="#">Various</a> <a href="#">including</a> <a href="#">DCTERMS</a>		<a href="#">Alignment</a> <a href="#">are given</a> <a href="#">as tables</a> <a href="#">using rdfs</a> <a href="#">and skos</a> <a href="#">naming</a>				yes
<a href="#">CIDOC CRM</a>			<a href="#">Dublin</a> <a href="#">Core</a>				yes
<a href="#">DATACITE</a>			<a href="#">DCTERMS</a>				
<a href="#">BioPortal</a>			<a href="#">Mapping</a> <a href="#">between</a> <a href="#">ontologies</a>				yes
GLEI	<a href="#">GLEI</a> <a href="#">Mapping</a> <a href="#">to FIBO</a> <a href="#">rdf</a>			<a href="#">GLEI</a> <a href="#">Mapping</a> <a href="#">to FIBO</a> <a href="#">rdf</a>	<a href="#">GLEI</a> <a href="#">Mapping</a> <a href="#">g to</a> <a href="#">FIBO</a> <a href="#">rdf</a>		
<b>TOTAL</b>	<b>7</b>	<b>3</b>	<b>14</b>	<b>2</b>	<b>1</b>	<b>2</b>	<b>10</b>

## 3. Data models

### 3.1 Alignment Format and EDOAL

The [Alignment Format](#) (AF) is a data structure designed to express an alignment between ontologies. Many ontology matching tools produce alignment structured according to AF, such alignment can be used as input for further alignment methods.

The AF allows N to N mappings, and it is expressed as DTD (for XML) and as RDF/XML.

In the image below, it is possible to see the class diagram of the AF extracted from the RDF/XML representation. The main class, Alignment, keeps the metadata of the Alignment between 2 ontologies, while the Cell class keeps the mappings between entities of the 2 ontologies. The cardinalities associated to each property are not expressed in the RDF/XML and they are coming from the DTD instead.

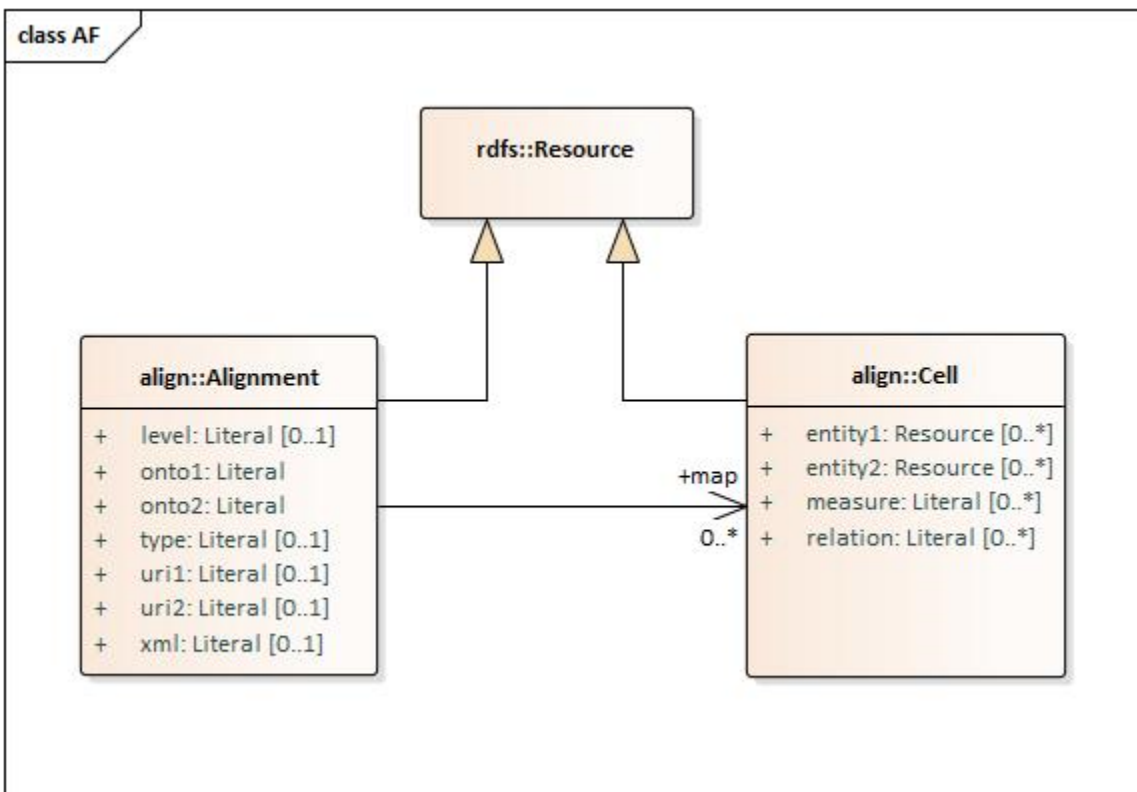


Figure 6: Alignment Format data model

Like AF, the [Expressive and Declarative Ontology Alignment Language](#) (EDOAL) allows for representing correspondences between the entities of different ontologies focusing on complex correspondences to precisely describe the relation between the entities. EDOAL extends the [Alignment Format](#) with focus on ontologies expressed as RDF/XML.

Many tools are able to read or generate AF or EDOAL, such as VocBench that can [create an EDOAL project](#) and validate it.

The [Alignment API](#) is an API (supporting SOAP and REST) to implement services about ontology alignments, such as storing, finding and sharing alignments.

The Alignment API is just an API interface to access alignments, the way alignments are determined must be done upfront (manually or automatic). The Alignment API is based on the AF.

The [Ontology Alignment Evaluation Initiative](#) (OAEI) is a coordinated international initiative to forge consensus on alignment. Annual campaigns have been created since 2005, to measure tools/algorithms performing alignments; tools are expected to generate alignments according to the AF or indirectly via the Alignment API, see the [framework](#) used.

## 3.2 SSSOM and SEMAPV

The [Simple Standard for Sharing Ontology Mappings \(SSSOM\)](#) is a community ontology expressed as RDF provided mainly by people in the Biological sector to describe alignments, needed as there are hundreds of ontologies in this sector.

Developers of SSSOM created an application “[SSSOM Toolkit](#)” that represent mappings in TSV format, import from the Alignment Format as XML, and export in RDF. Linked to SSSOM, the [Semantic Mapping Vocabulary \(SEMAPV\)](#) describes how mappings have been determined; this vocabulary is expressed as OWL.

In the image below, the structure of SSSOM model is extracted from the website, highlighting the mandatory properties:

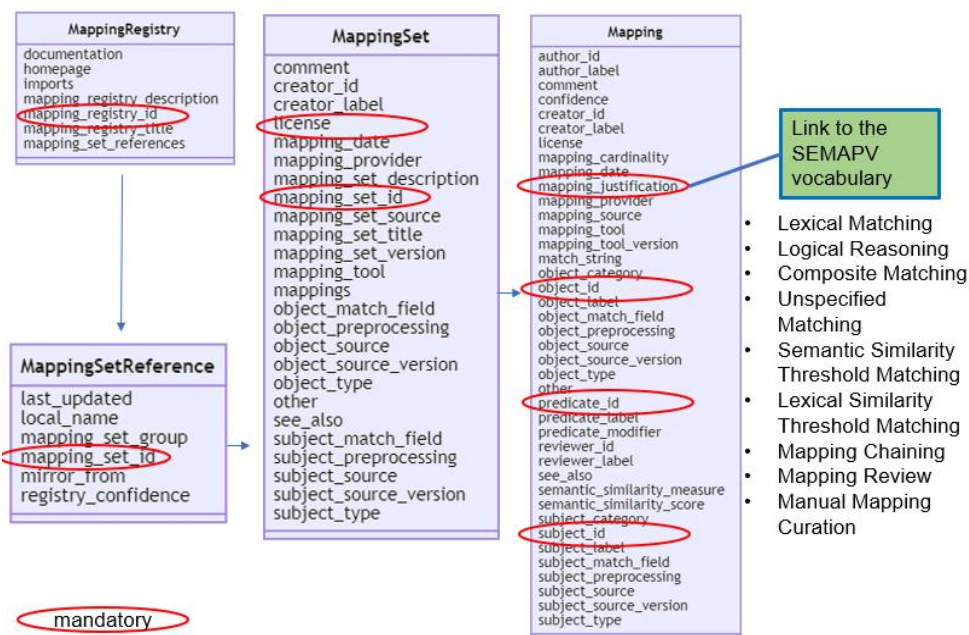


Figure 7: SSSOM data model

### 3.3 AF-AP

The AF-AP (Alignment Format Application Profile) is based on the Alignment Format and includes the SSSOM data model.

In the image below it is possible to find the current version of the AF-AP:

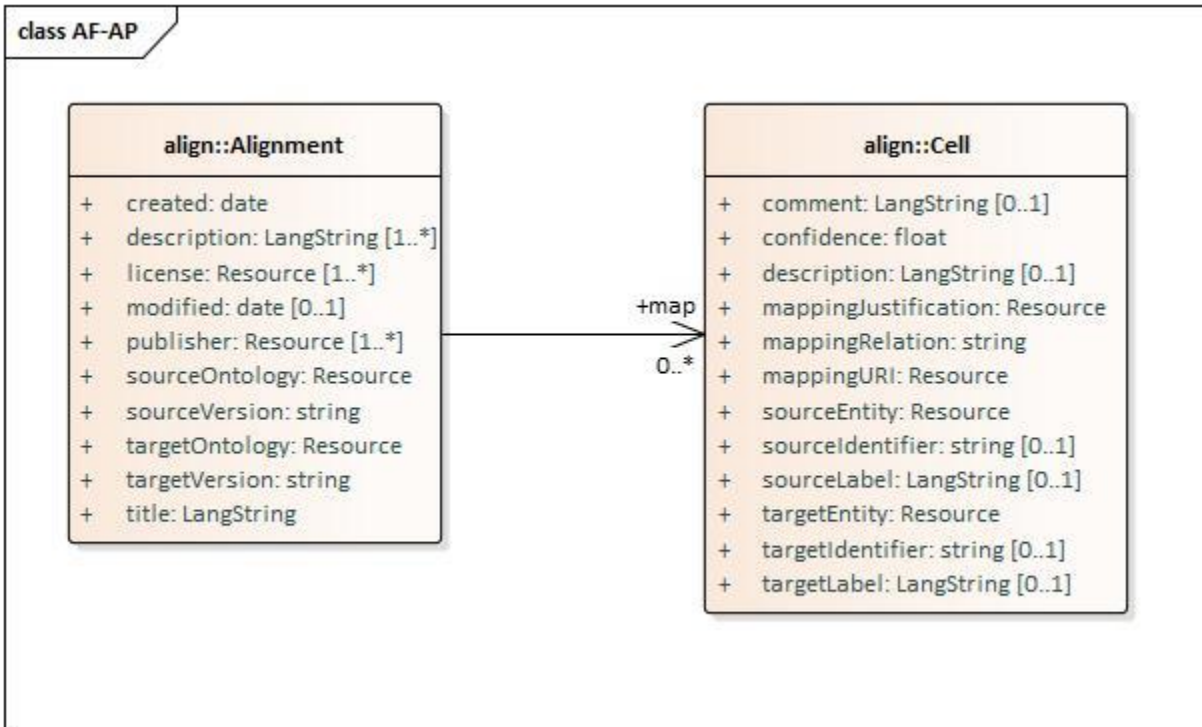


Figure 8: AF-AP data model

#### 3.3.1 Mapping with AF

The following table shows how the AF has been reused in the AF-AP, highlighting in orange the differences:

AF			AF-AP		
URI	Label	Data type	URI	Label	Data type
align:Alignment	Alignme nt		align:Alignment	Alignment	
align:onto1	onto1	rdfs:Literal	align:onto1	Source Ontology	rdfs:Resource
align:onto2	onto1	rdfs:Literal	align:onto1	Target Ontology	rdfs:Resource

align:map	map	align:Cell	align:map	map	align:Cell
align:Cell	Cell		align:Cell	Cell	
align:entity1	entity1	rdfs:Resource	align:entity1	Source Entity	rdfs:Resource
align:entity2	entity2	rdfs:Resource	align:entity2	Target Entity	rdfs:Resource
align:measure	measure	rdfs:Literal	align:measure	confidence	xsd:float
align:relation	relation	rdfs:Literal	align:relation	Mapping Relation	xsd:string

### 3.3.2 Mapping with SSSOM

The following table shows how the SSSOM can be mapped to AF-AP, highlighting in orange the differences:

SSSOM			AF-AP		
URI	Label	Data type	URI	Label	Data type
sssom:MappingSet	MappingSet		align:Alignment	Alignment	
sssom:mappings	Mappings	owl:Axiom	align:map	map	align:Cell
dct:title	mapping_set_title	xsd:string	dct:title	title	rdfs:LangString
dct:description	mapping_set_description	xsd:string	dct:description	description	rdfs:LangString
dct:creator	creator_id	rdfs:Resource	dct:publisher	publisher	rdfs:Resource
dct:license	license	xsd:anyURI	dct:license	license	rdfs:Resource
sssom:subject_source	subject_source	rdfs:Resource	align:onto1	Source Ontology	rdfs:Resource
sssom:subject_source_version	subject_source_version	xsd:string	sssom:subject_source_version	Source version	xsd:string

sssom:object_source	object_source	rdfs:Resource	align:onto2	Target ontology	rdfs:Resource
sssom:object_source_version	object_source_version	xsd:string	sssom:object_source_version	Target version	xsd:string
http://purl.org/pav/authoredOn	mapping_date	xsd:date	dct:created	created	xsd:date
owl:Axiom	Mapping		align:Cell	Cell	
owl:Annotated Source	subject_id	rdfs:Resource	align:entity1	Source Entity	rdfs:Resource
sssom:subject_label	subject_label	xsd:string	sssom:subject_label	Source label	rdfs:LangString
owl:annotated Property	predicate_id	rdfs:Resource	owl:annotated Property	Mapping URI	rdfs:Resource
sssom:predicate_label	predicate_label	xsd:string	align:relation	Mapping Relation	xsd:string
owl:Annotated Target	object_id	rdfs:Resource	align:entity2	Target Entity	rdfs:Resource
sssom:object_label	object_label	xsd:string	sssom:object_label	Target label	rdfs:LangString
sssom:mapping_justification	mapping_justification	rdfs:Resource	sssom:mapping_justification	Mapping justification	rdfs:Resource
sssom:confidence	confidence	xsd:double	align:measure	confidence	xsd:float
rdfs:comment	comment	xsd:string	rdfs:comment	comment	rdfs:LangString



## 4.Tools

### 4.1 Silk Workbench

The [Silk Framework](#) is an open-source framework (published with Apache licence) for integrating heterogeneous data sources, and it was supported in part by the EU FP7 project LOD2 with the objective to perform ontology matching.

The [Workbench](#) component is a web application that allows generating links between data items coming from different data sources.

In order to do so, it allows to design workflows providing transformation functions (such as lower case, split words) and string similarities methods off the shelf (such as Jaccard, Jaro, JaroWinkler, Levenshtein, Q-grams).

These workflows can be exported in the [Silk - Link Specification Language \(Silk-LSL\)](#), an XML syntax describing the matching process.

The Workbench allows previewing the links found, as result of the workflow execution, so that the workflow can be adjusted later.

In addition, the links found can be exported in N-triples or in Alignment Format; the latter can be also imported as reference for the matching process.

### 4.2 SKOS Play! Converter

[SKOS Play! Convert](#) is a free service that allows users to convert an Excel spreadsheet, with a predefined structure, into RDF, with some focus on SKOS vocabularies. Behind the service there is an open-source tool, [XLS2RDF](#) released under the LGPL licence.

The development was partly founded by the government of Luxembourg, and it is still actively maintained. In particular, the conversion to RDF is due to the famous open-source Java library [RDF4J](#) which is updated to the latest RDF specifications.

The predefined structure is quite flexible, for example it allows to specify the language of the content in a cell or link spreadsheet cells between each other.

### 4.3 VocBench

[VocBench](#) is a web-based, multilingual, collaborative development platform for managing OWL ontologies, SKOS(/XL) thesauri, Ontolex-lemon lexicons and generic RDF datasets. The software is released as open source under BSD 3-clause license.

Funded by the European Commission ISA<sup>2</sup> programme, the development of VocBench is managed by the Publications Office of the EU.

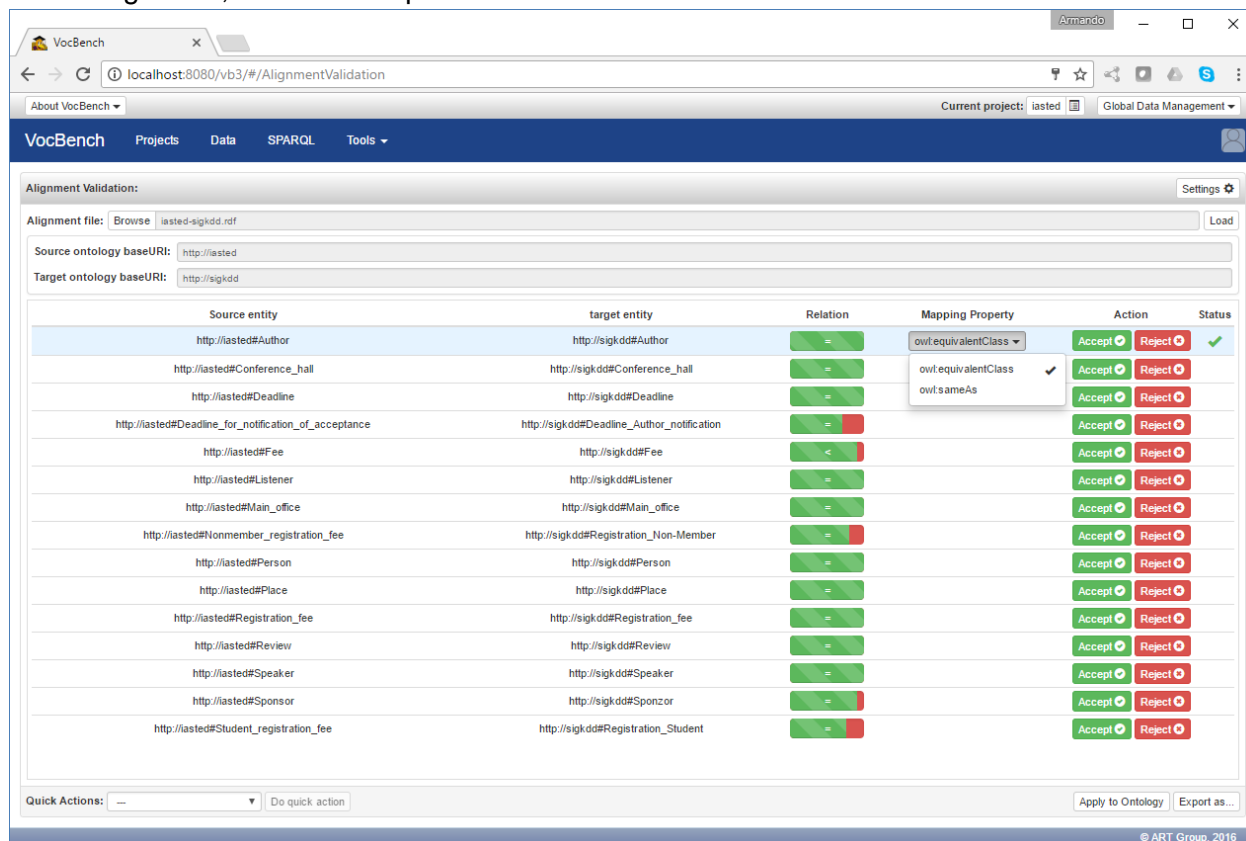
In the context of mapping, VocBench offers 2 functionalities:

- Editing [Egoal Alignments](#), where an Alignment is expressed by 2 entities, the relation between the 2 and the measure as in the AF (and not EDOAL as it could be understood).
- Perform a manual [Alignment Validation](#), either by uploading a file expressed in the AF or via a remote service.

In the context of the alignment with Schema.org, SKOS Play! Converter is used to perform the alignment as it contains metadata accordingly the AF-AP, see section “3.3 AF-AP”.

The RDF generated by the SKOS Play! Converter is then transformed in XML (accordingly to the AF format) via a custom XSLT, as explained in the section “2. Process”.

The XML file could be then imported by tools such as VocBench for a manual review/validation of the Alignment, see an example taken from the VocBench website:



The screenshot displays the VocBench web application interface for Alignment Validation. The browser address bar shows the URL `localhost:8080/vb3/#/AlignmentValidation`. The application has a navigation bar with 'VocBench', 'Projects', 'Data', 'SPARQL', and 'Tools'. The main content area is titled 'Alignment Validation' and includes a 'Settings' gear icon. Below the title, there are input fields for 'Alignment file' (with a 'Browse' button and the file 'lasted-sigkdd.rdf'), 'Source ontology baseURI' (set to 'http://lasted'), and 'Target ontology baseURI' (set to 'http://sigkdd'). A 'Load' button is also present. The central part of the interface is a table with the following columns: 'Source entity', 'target entity', 'Relation', 'Mapping Property', 'Action', and 'Status'. The table lists 15 rows of alignments. A dropdown menu for 'Mapping Property' is open, showing options: 'owl:equivalentClass' (selected), 'owl:equivalentClass' (with a checkmark), and 'owl:sameAs'. At the bottom, there are 'Quick Actions' and 'Do quick action' buttons, and 'Apply to Ontology' and 'Export as...' buttons. The footer indicates '© ART Group, 2016'.

Source entity	target entity	Relation	Mapping Property	Action	Status
http://lasted#Author	http://sigkdd#Author	=	owl:equivalentClass	Accept  Reject	
http://lasted#Conference_hall	http://sigkdd#Conference_hall	=	owl:equivalentClass	Accept  Reject	
http://lasted#Deadline	http://sigkdd#Deadline	=	owl:sameAs	Accept  Reject	
http://lasted#Deadline_for_notification_of_acceptance	http://sigkdd#Deadline_Author_notification	=		Accept  Reject	
http://lasted#Fee	http://sigkdd#Fee	=		Accept  Reject	
http://lasted#Listener	http://sigkdd#Listener	=		Accept  Reject	
http://lasted#Main_office	http://sigkdd#Main_office	=		Accept  Reject	
http://lasted#Nonmember_registration_fee	http://sigkdd#Registration_Non-Member	=		Accept  Reject	
http://lasted#Person	http://sigkdd#Person	=		Accept  Reject	
http://lasted#Place	http://sigkdd#Place	=		Accept  Reject	
http://lasted#Registration_fee	http://sigkdd#Registration_fee	=		Accept  Reject	
http://lasted#Review	http://sigkdd#Review	=		Accept  Reject	
http://lasted#Speaker	http://sigkdd#Speaker	=		Accept  Reject	
http://lasted#Sponsor	http://sigkdd#Sponsor	=		Accept  Reject	
http://lasted#Student_registration_fee	http://sigkdd#Registration_Student	=		Accept  Reject	

Figure 9: Alignment Validation in VocBench

Note that VocBench introduces some constraints:

1. The source ontology must be already loaded in VocBench, so the baseURI defined in the alignment file must be the same;
2. The “Mapping Property” can be chosen among a set of values predefined (so it cannot be typed manually) and it depends on the “Relation” value;
3. The export of the alignment in RDF, introduces a URI for the “Mapping Relation” which does not exist in the AF.

## 5. Acronyms

Term	Acronym
AF	Alignment Format
AF-AP	Alignment Format Application Profile
API	Application Programming Interface
CSV	Comma Separated Value
DTD	Document Type Definition
HTML	Hypertext Markup Language
JSON	JavaScript Object Notation
JSON-LD (context)	JSON-based Serialization for Linked Data; the context is a JSON file that can be used to enrich a JSON serialisation as Linked Data.
OAEI	Ontology Alignment Evaluation Initiative
Ontolex	Ontology Lexicon
OWL	Web Ontology Language
RDF	Resource Description Framework
RDFS	RDF Schema
REST	Representational state transfer
SHACL	Shapes Constraint Language
SILK LSL	Silk - Link Specification Language
SOAP	Simple Object Access Protocol
SPARQL	SPARQL Protocol and RDF Query Language
SKOS	Simple Knowledge Organization System
SKOS-XL	SKOS eXtension for Labels
XML	Extensible Markup Language
XSLT	Extensible Stylesheet Language Transformations

UML	Unified Modeling Language
-----	---------------------------