Dipartimento di Ingegneria e Scienza dell'Informazione

Modeling Conceptual Models – Step 2

Document data:	**Reference persons:**

13.11.2017	Tovo Alessia 189192;
	Luca Massimiliano 173497;
	Tavonatti Stefano 183846

Index

Revision History

Revision	Date	Author	Description of Changes
0.0	13.11.2017	Luca Massimiliano; Tovo Alessia; Tavonatti Stefano	Document created
0.1	14.11.2017	Luca Massimiliano; Tovo Alessia; Tavonatti Stefano	Review of EER model
0.2	16.11.2017	Tavonatti Stefano; Tovo Alessia	Assignment description section written and readjusted in respect with the first assignment. EER rebuilded
0.3	16.11.2017	Tovo Alessia; Luca Massimiliano	Informal model section written and readjusted in respect with the first assignment
0.4	18.11.2017	Tovo Alessia; Tavonatti Stefano; Luca Massimiliano	Model checking and ontology possible mapping
1.0	20.11.2017	Luca Massimiliano; Tovo Alessia; Tavonatti Stefano	General review and scheduling
1.1	22.11.2017	Tavonatti Stefano	EER diagrams for section 4
1.2	22.11.2017	Luca Massimiliano; Tovo Alessia	Top-level ontology section written
1.3	24.11.2017	Tavonatti Stefano	Top-level grounding section written

2.0	27.11.2017	Luca Massimiliano; Tovo Alessia; Tavonatti Stefano	General review and scheduling
2.1	29.11.2017	Luca Massimiliano	Document and grammar review
2.2	30.11.2017	Tovo Alessia	Diagrams validation
2.3	03.12.2017	Luca Massimiliano; Tovo Alessia; Tavonatti Stefano	DERA methodology and DERA evaluation
3.0	04.12.2017	Luca Massimiliano; Tovo Alessia; Tavonatti Stefano	General review and scheduling
3.1	05.12.2017	Tavonatti Stefano	General review
3.2	07.12.2017	Luca Massimiliano	References
3.3	08.12.2017	Luca Massimiliano; Tovo Alessia; Tavonatti Stefano	General review
3.4	09.12.2017	Luca Massimiliano; Tovo Alessia; Tavonatti Stefano	General review cont.
3.5	10.12.2017	Luca Massimiliano; Tovo Alessia; Tavonatti Stefano	Last check before delivery

1. Assignment description

In this project, our team would like to provide a brand new solution to allow people to find out the best semi-formal way to model something. Our idea is to provide a top-level ontology and an open source search engine that allow people to look up for the best conceptual model to use. It is not always simple to choose the right model to use and actually, there are no tools that support student, researcher or professors or whoever want to semi-formalize something and so we decided to build it. There are basically six way to model things: data-flow modeling (DFM), entity-relationship modeling (ERM), event-driven process chain (EPC), joint application development (DSDM), place/transition net and finally state-transition modeling. A big issue is that there is not a tool that allows designers to choose the best conceptual model and we also find out that in most of the cases there is more than one model that can provide a valid solution and choose between them should not be simple. In order to provide a valid and valuable solution we would like to do basically two things: focus on the versioning of ER (it has been done only on ER just because we do not have enough resources - time - to look also at other conceptual models) and show how the EER provided at the end of this document and the followed ontology can be easily used also for all the other conceptual models. For this reason, if you look at the EER you can find out that the name of all the entities is as generic as possible and precise and meaningful at the same time.

To build this tool we found out that there are not structured or semi-structured datasets that can help us and so all the data has been retrieved manually or automatically (RapidMiner ©) by web pages or papers (pdf). We decided to try to map in our EER schema the ER model and to do the versioning on all the different standard of representation.

2. The (informal) model

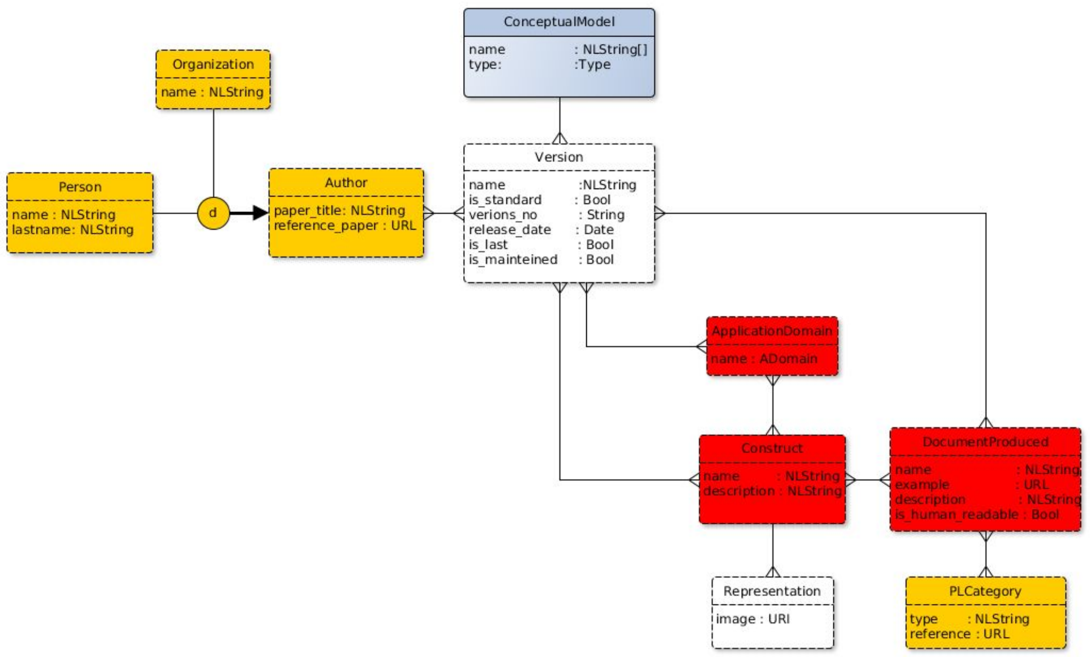


Fig. 1. The EER diagram used to model the problem

The color meaning is:

- White : complex attributes
- Red : auxiliary entities
- Blu : core entities
- Yellow : common entities

There is one core entity which is called **ConceptualModel** and it is the entity that we always return. In order to provide more meaningful information, we actually return the join over all the table but everything is based on the **ConceptualModel** table. All the other entities are used to apply the right filters to the query that will be launched by the user. **Version** has to be considered as a complex attribute of **ConceptualModel** and is crucial for our system. It contains information about the conceptual model version and it allows us to create a filter that is fundamental. Suppose you are looking for a specific version of ER: Chen's representation allows you to represent entities, relationships and attributes while the min-max allows you to represent also the direction of the relationship. It means that, thanks to this table, you will be able to do two things: take a look on how the model has evolved and also know which version of ER model use to define create your diagram. For this reason, as you can see from the diagram all the other filter are not directly linked to **ConceptualModel** but pass through **Version**. Given the domain, we can also describe which programming language paradigm (the **PLCategory** common entity) can be mapped on it (if there is one). For example, when the result is ER and UML is the version, the mappable programming languages are the object oriented programming languages while when the version is Chen, as long as the domain is “Database Design”, the mappable languages are all the Query Languages. Of course we provide also the **constructs** linked to a specific **Version** of a **ConceptualModel** and we also assume that the same constraint can have multiple representation as long as some conceptual model does not have a standard. Just think about ER: there couple of ways to represent entities and relationship (particularly the problem is on the representation of the cardinality).

There is also a common entity which is shared among different schemas: **Author**. We use this entity to define who takes care of the standard definition or the development of the Conceptual Model. Author is further specialized in **Organization or Person**

As said, all the information that we retrieve for building this schema are non-structured so all the decisions are based on the analysis of texts and reviews that are done in weekly meetings. Some of the decision taken for modeling the database schema are assumptions:

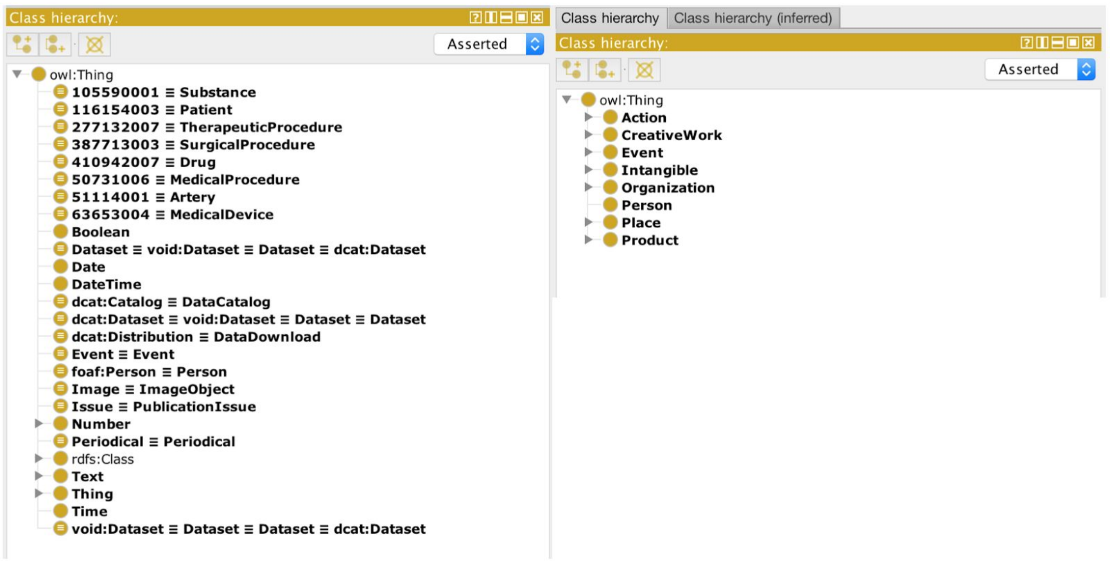
- If you look to something you do not specify the version, then you are referring to the Chen’s version as long as it is the most widely used.
- The same Author take care of a **ConceptualModel** from the very beginning to the end and this implies that there is only one organization linked to a conceptual model and this organization can not change in time.

3. Top-level ontology

It has been decided to use schema.org as the top-level ontology to map with. Schema.org “is a collaborative, community activity with a mission to create, maintain, and promote schemas for structured data on the Internet, on web pages, in email messages, and beyond.” [1].

The decision has been driven by two main aspects: firstly, schema.org has an official website ([2]) where the community ask and experts answer to question. Moreover, schema.org is the most widely used top-level ontology.

Unfortunately, some problem in importing Schema.org in Protege can emerge. The owl file produces some errors. A version of it has been delivered via the mailing list by doctor Gianluca Apriceno but there were some inconsistencies inside and the ontology was not completely original-like. A possible solution to achieve this problem is to download the RDF version of schema.org from [3] and adapt it to the specification of Protège ©. By reading the documentation of the software, it can be seen that Protège use rdfs:range and rdfs:include to build the internal model while schema.org uses its own structure: schema:domainIncludes and schema:rangeIncludes. By opening the RDF with a text editor (Atom has been used to produce this work) and by replacing schema:domainIncludes with rdfs:include and schema:rangeIncludes with rdfs:include the original version of schema.org can be used. In the next two images, the differences between the two version can be seen (classes tab).



```
Class hierarchy:                                   Asserted
owl:Thing
    105590001 ≡ Substance
    116154003 ≡ Patient
    277132007 ≡ TherapeuticProcedure
    387713003 ≡ SurgicalProcedure
    410942007 ≡ Drug
    50731006 ≡ MedicalProcedure
    51114001 ≡ Artery
    63653004 ≡ MedicalDevice
    Boolean
    Dataset ≡ void:Dataset ≡ Dataset ≡ dcat:Dataset
    Date
    DateTime
    dcat:Catalog ≡ DataCatalog
    dcat:Dataset ≡ void:Dataset ≡ Dataset ≡ Dataset
    dcat:Distribution ≡ DataDownload
    Event ≡ Event
    foaf:Person ≡ Person
    Image ≡ ImageObject
    Issue ≡ PublicationIssue
  ▸ Number
    Periodical ≡ Periodical
  ▸ rdfs:Class
  ▸ Text
  ▸ Thing
    Time
    void:Dataset ≡ Dataset ≡ Dataset ≡ dcat:Dataset
```

```
Class hierarchy | Class hierarchy (inferred)
Class hierarchy:                                   Asserted
owl:Thing
  ▸ Action
  ▸ CreativeWork
  ▸ Event
  ▸ Intangible
  ▸ Organization
    Person
  ▸ Place
  ▸ Product
```

Fig. 2: on the left the version sended, on the right the version used by Sojourner

Only some classes of the schema.org TLO has been used to map our ontology:

- Intangible
- Person
- Organization
- CreativeWork
- ImageObject

The mapping with the TLO has been done by using a "subclass" relations. In section 5, some other details will be provided.

4. Top-level grounding

As asked, in chapter 3, the model used for the first assignment has been plotted. That is not the one used for this assignment. The newest version of Sojourner EER will be provided in *Fig. 3* while the mapping with schema.org will be provided in *Fig. 4*

As you can see, the new EER drastically change with respect to the one of the first assignment. Particularly, our team found out that is was impossible to design **Version** as a complex attribute as long as it was implied to lose consistency and expressivity power. In order to achieve our needs, rdfs:Datatype statements the entity **Version** became an auxiliary entity.

The same happened to the entity **Representation.** It became a relational attribute of **Construct.** In order to better map everything to schema.org, all the relationships involved in the diagram became hierarchical.

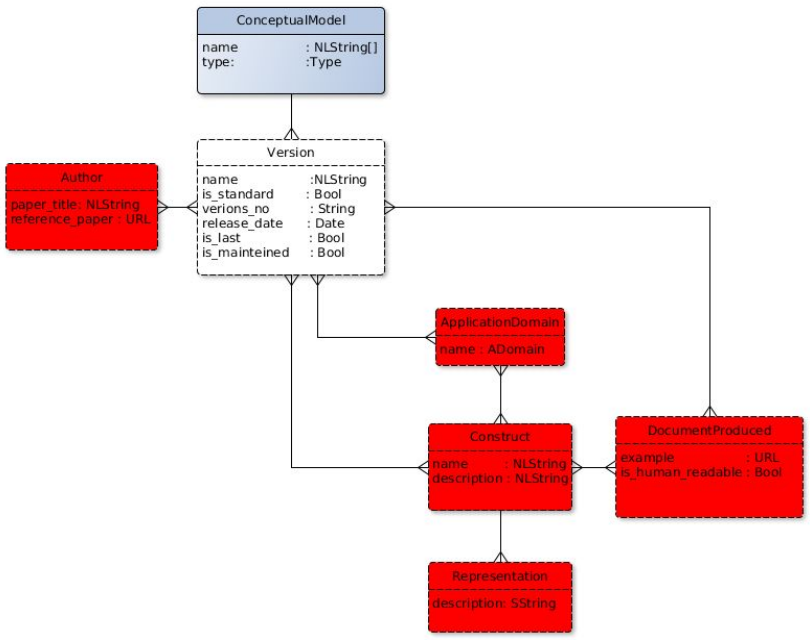


Fig. 3: the latest version of the EER diagram

In *Fig. 4* also the entities coming from schema.org have been added and are represented in green. As you can see, each relation used to map our model to schema.org is a “specialization-mapping” relationship which is translated with “subclass-of” in Protège.

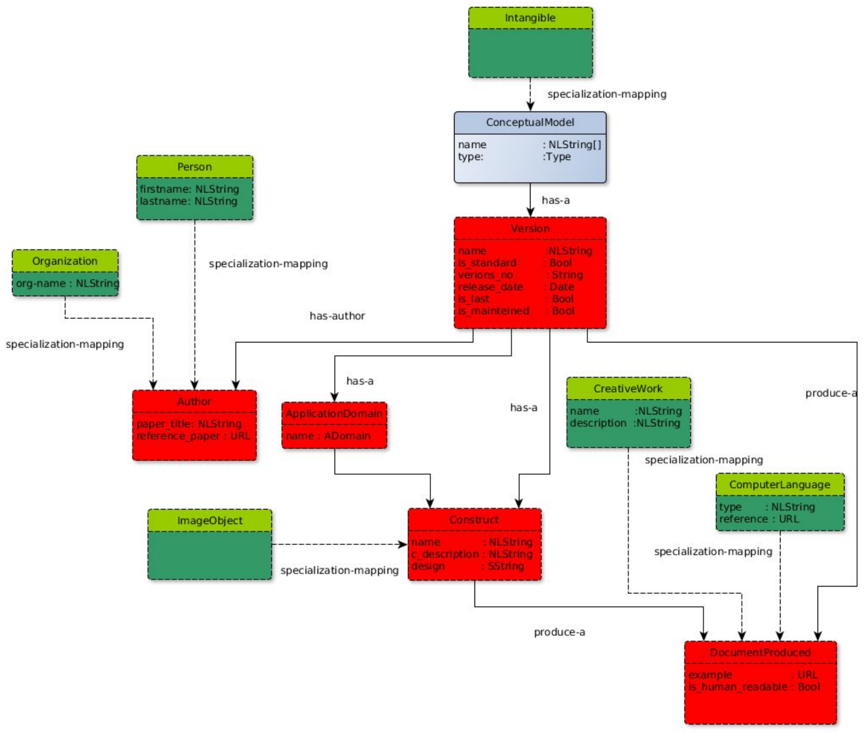


Fig. 4: the EER mapped to the TLO schema.org

The problem founded during the mapping process has been achieved as said at the beginning of this chapter. The biggest problem has been faced with creating the entity **Version.** As far as it was designed as a complex attribute, to model it in Protège, the idea was to include all the attribute and relationships of **Version** in **ConceptualModel.** The point is that all the relationships of **Version** become meaningless if linked to **ConceptualModel** and to design some particular aspect become impossible (i.e. the application domain strictly depends on the version of a conceptual model and not directly from a conceptual model). There was two solution to achieve this problem: one is to design **Version** as an entity (the one was chosen) while the other one is to have duplicate entities in **Conceptual Model**. We chose the second one also because the second is strongly not suggested (see also [4]).

Another problem was to map **ApplicationDomain** with the **TLO** but after launching the reasoner in Protège we find out that it was inferred to be **Intangible.**

At the end, **Sojourner** is 100% mapped with schema.org and we strongly believe that it is a great result.

"The Semantic Web community has achieved a good standing within the last years. As more and more people get involved, many individual ontologies are created. Interop-erability among different ontologies becomes essential to gain from the power of the Semantic Web. Thus, mapping and merging of ontologies become a core question." [5]

5. Facets creation

According to what has been defined in [6], this table has been produced.

Entity	**Relation**	**Attribute**
ConceptualModel	instance-of	name
(is-a) Version	concerne	type
(is-a) AppDomain	has-a	(value-of)DFM
(is-a) Construct	produce-a	(value-of) ERM
(is-a) CMDiagram	has-representation	(value-of) EPC
	has-author	(value-of) DSDM
	(is-a) is-person	is-standard
	(is-a) is-organization	(value-of) true
		(value-of) false
		version-no
		release-date
		is-last
		(value-of) true
		(value-of) false
		is_mainteined
		(value-of) true
		(value-of) false
		appdom_name
		(value-of) general
		(value-of) business
		(value-of) database
		(value-of) OOP
		(value-of) graph
		(value-of) flow control
		c_description
		design
		example
		is-human-readable
		(value-of) true
		(value-of) false
		cw-name
		cw-description
		cl-type
		cl-reference
		paper-title
		reference-paper
		firstname
		lastname
		org-name

6. The final ontology

- The final ontology: Sojourner

Formalization issues:

1. In order to be able to import data correctly, without loss of meaning and representations of data, we have modified and removed some attributes of different entities.
   First of all, we made Version as an entity and not Complex Attribute, as represented in Figure 3. This decision has been made because mapping Version as an attribute of ConceptualModel may cause loss of consistency and expressivity, since many entities, like DocumentProduced, Construct, ApplicationDomain are mainly related to the version of a particular conceptual model instead of the conceptual model itself.
   After this mainly change, entity Representation has been removed. The attribute “description” has been added to the entity Construct with the name “design”, and the Construct’s attribute “description” become “c_description” in order to disambiguate. The removal of entity Representation has been done so the mapping of the entity Construct could be done more efficiently. In fact, with this adjust, Construct is easily mappable to ImageObject of Schema.org. Doing this, we have also reduced the redundancy problem of attribute values.

1. As said, schema.org is not fully compatible with a direct importing on Protège. Some work has to be done before using it. As far as we decided to use schema.org as TLO, we replaced schema:domainIncludes with rdfs:include and schema:rangeIncludes with rdfs:include. Furthermore, we replaced schema:DataType with rdfs:Datatype.

2. Also some extension of schema.org have been designed during the last few years, i.e. auto.schema.org
   At the beginning of this assignment, we were in the position to decide whether to build a schema.org extension or a domain-specific ontology. **Sojourner** is the first ontology to model conceptual models all over the world. It makes sense to build it as an extension of schema.org but, anyway, we decided to build an SDO. We consider a possible submission to the schema.org community as a future work.

3. As mentioned above, we are not able to found structured or semi-structured data. So the data had been collected manually from various paper and website and temporarily stored in an excel file.

4. The excel file which contains the data has been changed and restructured several times in order to achieve a better mapping with the ontology and to simplify the data importing with the Cellfie plugin.

7. References

[1] Schema.org website, *schema.org/docs*

[2] Blog of schema.org, *blog.schema.org*

[3] Download entry-point for schema.org rdf *http://schema.org/docs/datamodel.html*

[4] Raghu Ramakrishnan, Johannes Gehrke, *Database Management Systems*

[5] Marc Ehrig, York Sure, *Ontology Mapping - An Integrated Approach*

[6]Giunchiglia F. & Dutta B.. *DERA: A FACETED KNOWLEDGE ORGANIZATION FRAMEWORK*