Luttenberger, Norbert; Zedlitz, Jesper

Article — Accepted Manuscript (Postprint)

Standard International Trade Classification—from Spreadsheet to OWL-2 Ontology

Business & Information Systems Engineering

Suggested Citation: Luttenberger, Norbert; Zedlitz, Jesper (2018) : Standard International Trade Classification—from Spreadsheet to OWL-2 Ontology, Business & Information Systems Engineering, ISSN 1867-0202, Springer, Berlin, Heidelberg, Vol. 60, pp. 305–316-, https://doi.org/10.1007/s12599-017-0495-z

This Version is available at:
https://hdl.handle.net/10419/162172

Standard International Trade Classification
—from Spreadsheet to OWL-2 Ontology

Norbert Luttenberger, Jesper Zedlitz
Research Group for Communication Systems
Dept. of Computer Science
Christian-Albrechts-University in Kiel
Germany
[n.luttenberger|j.zedlitz]@email.uni-kiel.de

Abstract

Trade classifications are a necessary prerequisite for the compilation of trade statistics, and they should—beyond that—be regarded as valuable base for the definition of shared controlled vocabularies for linked business data that deal with import, export etc. The Standard International Trade Classification (SITC) provided by the UN Statistics Division is a widely used classification mostly applied for scientific and analytical purposes. SITC—as most other trade classifications—is available today only in text or spreadsheet formats. These formats reveal the inner hierarchical structure of SITC to the human reader, because SITC trade codes are built according to the decimal classification scheme, but unfortunately, SITC's inner structure is opaque to computer applications in text and spreadsheet formats. In this paper, we discuss an approach to set up an OWL-2 ontology for SITC that states subsumption relations between classes of goods. This kind of semantic underpinning of SITC is suited to ease both checking and extending SITC and to derive from it a shared controlled vocabulary for business linked data. We carefully discuss some problems of today's SITC (among them missing inner nodes of the trade code hierarchy), and we motivate several decisions that we took for ontology design. Finally, we introduce the semantic reasoner as a tool for the (at least partial) automatic derivation of structural information for SITC from the trade code building rule. We report on reasoner runtimes observed for different version of the SITC ontology and for different versions of the Pellet reasoner.

1 Introduction

Trade classifications are a necessary prerequisite for the preparation of trade statistics that are used to describe—for administrative and/or scientific purposes—domestic and international flows of goods. Not challenging this intended use, we discuss trade classifications in this paper with a different motivation: We argue that trade classifications—beyond their obvious purpose—can also be considered as valuable sources for the definition of shared controlled vocabularies. Shared controlled vocabularies—in turn—are at the foundation of Linked (Open) Data collections. With this perspective in mind, we foresee that the terminology work that is required for the provision of meaningful linked business data can profit from existing vocabularies, among them those that give trade statistics their shape.

Unfortunately, most trade classifications today are available in text or table (spreadsheet) formats. These formats address the human reader; they are not very well suited to reveal the inner, mostly hierarchical structure of trade classifications to computer processing and examination. Experience has shown that with this kind of formats structural problems may arise on several occasions, for instance when adjusting a trade classification to new demands. To avoid these problems, we therefore prefer for the development of shared controlled vocabularies a logics-based format, i.e. a format that allows us to ground "vocabulary control" on logical reasoning (and related tools).

A second argument is in favor of our proposition: Today, we see numerous trade classifications (see below), some of them covering (slightly) different purposes, some of them covering different historical epochs. It would be an endeavoring task to find out what parts of such classifications are equivalent and what parts are not. Though this task is beyond the scope of this paper, we feel free to point out that in text based formats, checking for equivalence has only string comparison as basic operation, while in logics-based formats, we can additionally take structural properties into account.

From the background as outlined above, we see our effort as an "example-driven" feasibility study for the derivation of shared controlled vocabularies from trade classifications. The paper focusses on the development of an OWL-2 ontology (term defined below) for the fourth revision of *Standard International Trade Classification* (SITC-4) that has been published in 2006 by the United Nations Statistics Division (UNSD) [1]. SITC-4—currently being published in text and spreadsheet formats—plays an important role mostly for analytical purposes in the economics area. The UNSD states: "Many countries and national and international organizations continue to use SITC for various purposes, such as for the study of long-term trends in international merchandise trade and aggregation of traded commodities into classes more suitable for economic analysis." [2] Similar to the SITC-4 with respect to its hierarchical structure is the *Combined Nomenclature* (CN) [3]; it is used in the European Union mostly for administrative purposes. The CN builds upon and extends the so-called *Harmonized System*, which has been developed by the World Customs Organization [4]. In this paper, we also take into account the correspondence table that defines mappings between SITC-4 and CN codes [5][6] (called "correspondence table" for short in the remainder of this paper). The manifold applications and the worldwide usage of both SITC-4 and CN justify the effort to improve their structure representation.

There exist numerous definitions for the term "ontology". Studer et al. give a widely accepted definition: "An ontology is a formal, explicit specification of a shared conceptualization." (Cited in [7]). Guarino, Oberle, and Staab explain: "A conceptualization is an abstract, simplified view of the world that we wish to represent for some purpose." [7] ISO 1087-1:2000 defines "concept" as a "unit of knowledge created by a unique combination of characteristics." [8]. In general, ontologies thus can be seen as collections of knowledge for some specific domain of discourse.

More specifically, formal ontologies (like e.g. OWL-2 ontologies) are ontologies that are built upon description logics [9]. Here, "concepts" are mapped to classes, and "characteristics" are mapped to properties. A formal ontology ("ontology" for short in the remainder of this paper) deals with *classes*, class subsumption, and class relationships. OWL-2 offers a rich wealth of class expressions that impose necessary and/or sufficient conditions for class membership. Besides classes, OWL-2 allows to describe *named individuals*. Named individuals carry data and/or object properties, and are assigned to classes explicitly or by a reasoning process. Ontology statements are called *axioms*. Inside an ontology, we can make a distinction between the TBox and the ABox. TBox axioms describe concepts (e.g. extensional or intensional definition of classes, class/subclass relations), while ABox axioms capture knowledge about named individuals (e.g. assign individuals to classes) [9]. Most important when discussing ontologies is the reasoning support that comes with ontologies. Reasoning—among other things—detects "hidden knowledge" in ontologies, for instance class subsumption or class membership (when these facts have not been written up by the ontology engineer) and checks for ontology consistency.

OWL-2 ontologies can be used as semantically rich representations of class hierarchies, and therefore, they lend themselves very well as format for trade classifications. The reasoning support coming with ontologies helps to detect inconsistencies and structure problems in trade classifications. To a certain extent, reasoning can also be seen as a tool for ontology generation, or in our view: a tool for generating structurally correct trade classifications from partial descriptions.

Though the approach chosen in this paper seems to be straightforward at first sight, it turned out in the course of our project that two severe problems had to be solved: First, in both SITC-4 and the correspondence table, a number of structure problems had to be patched, before a formal ontology could be developed. Second, reasoner runtimes for the initial version of our ontology were very long, i.e. several hours. We re-coded parts of the open-source Pellet reasoner to enhance parallel program execution, and we re-engineered our ontology. For a detailed discussion of the influences of different choices we took the reader is referred to section 6 of this paper.

Our paper is organized as follows: In the following section, we discuss approaches similar to ours applied to other trade classifications like, for instance, the United Nations Standard Products and Services Code (UNSPSC), or the eCl@ss cross-industry classification of products and services. In section 3 we give a short intro into SITC-4 and its problems. Section 4 explains in depth our design decisions for transforming SITC-4 into an ontology. In section 5, we briefly present the transformation workflow. A special focus is put on reasoning in the following section. The paper closes with a short summary and an outlook on further work.

2 Related Work

Tolksdorf et al. give a gross overview over semantic technologies for the business area in [10]. The authors identify "three important building blocks" for e-commerce scenarios, namely "the use of URIs as a global identification mechanism for products and traders, the RDF data model, and the Web Ontology Language (OWL) for the definition of common terms and concepts".

In a similar intention, Ding et al. discuss the "The Role of Ontologies in eCommerce" in [11]. From their perspective, ontologies may be a prerequisite for a common understanding of product catalogs in B2B transactions. In this context they discuss also the UNSPSC trade classification (see below).

Unfortunately, both papers do not cover deeper technical details. In the following, we therefore concentrate on papers that deal with trade classifications and with technical approaches to make their inherent structure explicit. The material is organized in a sequence that shows increasing formal effort for superimposing structural views on existing trade classifications.

We start with a historic trade classification that does not even use the decimal classification scheme for trade code building. In 2014, the Deutsche Zentralbibliothek für Wirtschaftswissenschaften—Leibniz-

Informationszentrum Wirtschaft (ZBW) has published on Open Access the "Statistik des Deutschen Reichs 1873–1883" (*Reichsstatistik*) [12]. The statistical data included in the 10 volumes of the Reichsstatistik refer to a historic trade classification that comprises approx. 450 terms for trade goods. Terms are ordered in a very elementary 2-layer hierarchy, but despite of this hierarchical ordering the Reichsstatistik does not care for assigning hierarchical codes to the listed goods. ZBW has published the Reichsstatistik in HTML, Excel, and other formats. A structure view on the Reichsstatistik trade classification has not been developed.

As part of its *Standard Thesaurus Wirtschaft* (STW – Thesaurus for Economics) [13], ZBW has published a sub-thesaurus for "commodities" that includes a number of entries for products of different kinds. The ZBW has published the STW in different formats, among them RDF and Turtle. The STW uses SKOS (Simple Knowledge Organization System) [14] to structure its thesaurus. SKOS provides a vocabulary that puts terms of some domain of interest into relation to each other; such relations are, for instance, "broader term", "narrower term", and "related term". By that, SKOS follows ISO 25964, the international standard for information retrieval thesauri. ISO 25964 defines a thesaurus as a "controlled and structured vocabulary in which concepts are represented by terms, organized so that relationships between concepts are made explicit, and preferred terms are accompanied by lead-in entries for synonyms or quasi-synonyms" [15]. Though using SKOS adds structural elements to an otherwise flat catalogue of words and thus is adequate for setting up a thesaurus, we do not to follow this approach for two reasons:

- We think that a trade classification does not deal with terms, but that it deals with the classification of goods themselves. Such a classification establishes real subsumption relations between classes of goods (`rdfs:subClassOf`). "Terms" play only a minor role in a classification, they are used mainly for creating annotations to be accessed by the human reader.
- SKOS aims at poly-hierarchies, i.e. a term may be, for instance, a "narrower term" to more than one "broader term". A trade classification in contrast is a strict mono-hierarchy, where each trade code belongs to only one class in the next higher layer. Enforcing a mono-hierarchy in SKOS requires additional complexity that makes the ontology less comprehensible.

Yet another direction is chosen by eCl@ss, a "cross-industry product data standard for classification and description of products and services" with "applications such as procurement, controlling and distribution" [16]. It has "40,800 product classes and 16,800 properties" covering "the majority of traded goods and services". In eCl@ss, products and services are described in the OntoML language. OntoML stands for "Product Ontology Mark-up Language" and is standardized in ISO 13584-32. OntoML is an XML schema designed for use by applications that need to exchange and process ISO 13584 PLIB (Parts LIBrary) compliant domain ontologies, possibly together with their related instances, in various Web-oriented environments. This schema—as any other XML schema—uses the element nesting capabilities provided by XML to express relations between XML elements. This is an obviously valid notation for hierarchical structures. But OntoML instance documents can only be checked syntactically by a validating parser, and not semantically, because XML Schema is not based on description logics as, for instance, OWL-2, where one can apply a reasoner to check and even establish a correct subsumption of classes.

In [17], Hepp and de Bruijn describe their attempt to develop a generic methodology for deriving OWL and RDF-S ontologies from "hierarchical classifications, thesauri, and inconsistent taxonomies". They argue that "informal hierarchical categorizations" may apply inter-class relations that are not subsumptions, and therefore they introduce a so-called "context" that installs the set of relations needed in the respective situation. They illustrate their approach by a fictitious hierarchy, where "ice cubes" are a subcategory (but not a subclass!) of "beverages", and argue that this might be the perspective of a "purchasing manager". They exemplify their very generic methodology by the construction of OWL ontologies for both eCl@ss (see above) and the United Nations Standard Products and Services Code (UNSPSC) [18]. In contrast to Hepp and de Bruijn, we focus on classifications with subsumption relations only, and by that have the benefit that the reasoner can fill in missing subsumption relations automatically, i.e. we do not need what Hepp and de Bruijn call a context.

In [19], Stolz et al. present a tool called PCS2OWL that according to the authors allows the user to transform trade classifications from various formats into OWL ontologies. Unfortunately, their approach seems to be overly complex: The mentioned tool creates from each category in the source product classification two OWL classes. "The first is a broader taxonomic class that represents the category from the [product classification] in the target ontology. The second is a context-specific class, in our case in the domain of products and services." This obviously doubles the number of concepts in the resulting ontology. Another severe concern is that in the approach of Stolz et al. the resulting ontology is not checked by a reasoner, but only by a number of SPARQL queries, comparing category and class counts. Thus, structural issues are beyond the scope of inspection.

Finally, in [20], Caracciolo et al. show the development of an OWL ontology for the fisheries domain. The ontology is built upon FAO's International Standard Statistical Classification of Fishery Commodities (ISSCFC) [21], which is an expansion of the SITC, and it is linked with the Harmonized System. Nevertheless, their ontology seems to be an ontology "from scratch", it is not just a "snippet" from SITC-4. It provides a link to SITC-4, though, by providing a per-item property that holds the SITC-4 trade code.

3 Short intro to SITC-4 and its problems

SITC-4 can be modeled as a monohierarchical classification with 5 hierarchy levels, called *tiers*. Below the unnamed root we find—in downward order—*sections* (tier 1), *divisions* (tier 2), *groups* (tier 3), and *subgroups* (tier 4). Finally, tier 5 holds so-called *basic headings*. SITC-4 has 10 sections, 66 divisions, 262 groups, 1023 subgroups, and 2652 basic headings. In the following, we are going to explain why it is advisable to add further basic headings to SITC-4.

SITC-4 trade codes are formed according to the basic rules of a decimal classification. (By "basic" we exclude auxiliary signs and auxiliary numbers as used, for instance, in the so-called Universal Decimal Classification [22].) More formally, a trade code *TC* is defined as follows:

$$0 \le c_i \le 9, c_i \in \mathbb{N}$$
$$1 \le i \le 5, i \in \mathbb{N}$$
$$TC ::= c_1 \mid c_1c_2 \mid c_1c_2c_3 \mid c_1c_2c_3c_4 \mid c_1c_2c_3c_4c_5$$

Sections have a 1-digit code, divisions have a 2-digit code, and so on. The first digit of a trade code identifies the related section, the first two digits identify the related division, and so on. In the following, we call the single digits c_i of a trade code its tier-1 code, its tier-2 code, and so on. A sample "path" through the SITC-4 hierarchy is given in table 1.

	c_1	c_2	c_3	c_4	c_5	
section	0					Food and live animals
division	0	5				Vegetables and fruit
group	0	5	7			Fruit and nuts (not including oil nuts), fresh or dried
subgroup	0	5	7	1		Oranges, mandarins, clementines and similar citrus hybrids, …
basic headings	0	5	7	1	1	Oranges, fresh/dried
	0	5	7	1	2	Mandarins (including tangerines & satsumas); clementines, …

Table 1: Sample SITC-4 trade codes

A deeper analysis of the SITC-4 shows that the classification has 318 subgroups that have no subordinate basic headings. In the following, we call these subgroups "left-alone subgroups".

In order to provide basic headings to the left-alone subgroups, we consulted the correspondence table. Indeed, the correspondence table contains 317 basic headings that are suited to fill the gaps below the left-alone subgroups. Nevertheless, one remaining subgroup keeps its status as left-alone subgroup. Figure 1 shows the result of the combination of SITC-4 and the correspondence table (bar length not proportional).

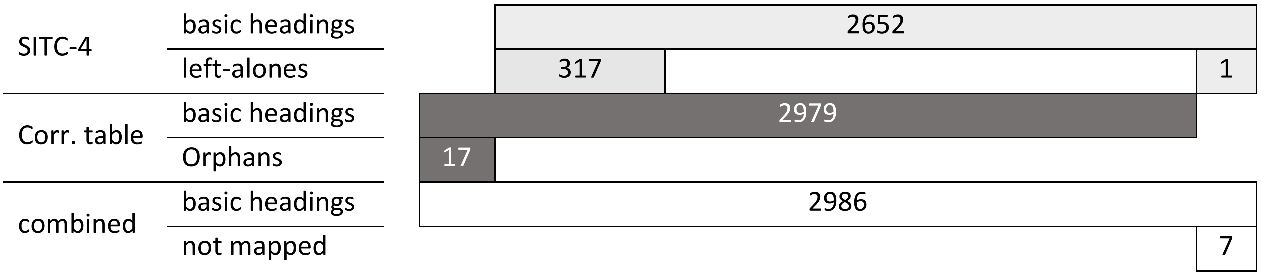

Figure 1: Combining SITC-4 and correspondence table trade codes

When taking over all non-duplicate trade codes from the correspondence table into a combined classification, we are confronted with a new phenomenon: The correspondence table delivers 17 basic headings that are not related to any subgroup in the "original" SITC-4. For some of these basic headings not only the subgroup layer is missing, but also "ancestor" layers further up in the hierarchy. We call these basic headings "orphans". Table 2 shows the orphans together with the related division, group, and subgroup codes that are needed to fully populate the hierarchy. The German version of the SITC-4, provided by the German Statistische Bundesamt (Federal Statistical Office), mentions that some of these orphans have been introduced national specifics [23].

codes missing in SITC-4	60				70		80	94		99
	600		660		700		800	941	972	998
	6000	6580	6600	6950	7000	7910	8000	9410	9720	9988
orphan basic headings from corr. table	60001	65800	66000	69500	70001	79100	80001	94100	97200	99888
	60002				70002		80002			
	60003				70003					
	60004									
	60005									

Table 2: Orphan trade codes and related gaps in the SITC-4 hierarchy

The analysis given above clearly indicates that it is hardly recommendable to convert SITC-4 on an as-is basis to an OWL-2 ontology. In following chapter, we show how we dealt with the identified problems.

4 Approach

In the following subchapters we explain our design guides for the development of an OWL-2 ontology for the SITC-4. These design guides seem very specific, but could serve "best practice" rules for solving similar problems.

4.1 Orphan basic headings and left-alone subgroups

As mentioned above, inserting trade codes of the correspondence table into SITC-4 produced orphan basic headings. With our goal in mind to make the classification structure fairly regular, we had two choices: Either prune these orphan codes, or generate artificial "ancestors", i.e. subgroups, groups, and even divisions as detailed in table 2. Even it is questionable to add this kind of pseudo-information (i.e. SITC-4 codes that do not stand for explicitly defined product groups) to SITC-4, we decided to generate ancestors for orphans, i.e. codes that play the role of stopgaps. If we had decided otherwise, we had lost a number of trade codes. Generating these artificial ancestors is easy as we can deduce their codes from the codes of the orphan basic headings.

Even after inserting trade codes of the correspondence table into SITC-4, one left-alone subgroup remained in the SITC-4. With our goal in mind to make the classification structure fairly regular, we could have considered to implement a "dummy" basic heading such that it can be subsumed under this left-alone subgroup. In fact, we decided to leave this left-alone subgroup as it is, which means that it adopts the role of a product code, similar to a basic heading. This decision was motivated by our unwillingness to add more information items than absolutely necessary to the "original" SITC-4.

4.2 SITC-4 basic headings

At first sight, it might seem rather "natural" to map all members of the five tiers of SITC-4 to OWL-2 classes and to not consider any class instances (in OWL-2 terminology: named individuals) in a trade classification. This procedure would have left the usage of the OWL-2 language construct `owl:NamedIndividual` to denote concrete instances of some product, i.e. "this box of oranges". However, we followed another approach: We mapped the members of SITC-4 tiers 1 to 4 to OWL-2 classes (`rdf:type owl:Class`), and we mapped the SITC-4 basic headings to OWL-2 named individuals (`rdf:type owl:NamedIndividual`). The reason for this design decision can be summarized as follows:

We consider SITC-4 as an instrument for the preparation of trade statistics. We assume that trade statistics are built from datasets combining e.g. product kind, value, time span, and additional information. In RDF, such a dataset (in a sample namespace abbreviated "stat") could be written (in Turtle syntax) as a set of triples for instance like: `stat:exportItem stat:ofKind sitc4:05711; stat:hasValue "1.5 M USD"; stat:inYear "2010"`. In this sample dataset, the trade code 05711 (see table 1) is used as value of an object property. For object property values, only individuals can be used in an ontology. If we had decided to map SITC-4 basic headings to OWL-2 classes, the designer of an ontology for trade statistics would have been in danger to write down the sample `stat:exportItem` dataset as follows: `stat:exportItem rdf:type sitc:05711` … A reasoner would have concluded from this modeling that `stat:exportItem` datasets were of type "Oranges, fresh/dried" which is obviously nonsense.

A second argument is valid, too. Modeling the SITC-4 basic headings as OWL-2 individuals allows us to assign further (data/object) properties to these individuals. In the next chapter, we show a sample property assignment by linking additional information from the correspondence table (CN trade code and validity period) to SITC-4 basic headings.

Find a similar discussion on the topic "class vs. individual" in [24].

4.3 CN trade codes and their validity periods

The correspondence table maps almost all SITC-4 basic headings to related CN trade codes, and additionally gives a validity period for these mappings. As can be seen from the correspondence table, SITC-4 trade codes have been mapped to different CN trade codes over time.

In order to enrich our ontology with CN trade codes and related validity periods, we need some kind of (ad-hoc) reification for statements that link a CN trade code to a SITC-4 trade code. The W3C Working Group Note on "Defining N-ary Relations on the Semantic Web" [26] defines in its Use Case 1 a pattern that can easily be applied to our problem as well. (For a more in-depth discussion on "Time-Dependent Factual Knowledge", see [27].) We constructed an ad-hoc reification for the SITC-4-to-CN mapping as follows.

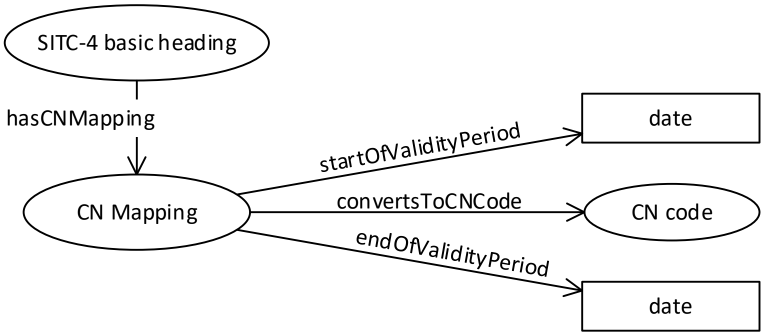

Figure 2: CN Mapping: Attaching a CN code and its validity period to a SITC-4 basic heading via a UUID-identified node

In a "pure" RDF environment, one could link reifying statements to a node via an RDF blank node. In order to enable a distinct access to any property combination, we decided to not apply the RDF blank node construct. Instead, we combine the properties `:convertsToCNCode`, `:startOfValidityPeriod`, `:endOfValidityPeriod`, and their values by named individuals that are identified by Universally Unique IDentifiers (UUID) according to ITU-T Rec. X.667, which are basically pseudo-random numbers. We call these individuals "CN Mappings" in the remainder of this paper. CN Mappings are related to SITC-4 basic headings by the object property `:hasCNMapping`. CN Mappings are collected in class `:CNMapping` (not shown in figure 2).

4.4 Inferred class subsumption

A reasoner may—at least partially—generate subsumption relations, assuming that we define suited class axioms. Thus, we have the choice to either include explicit class subsumption axioms into our ontology (using `rdfs:subClassOf` relations), or to construct class expressions such that class subsumption can be inferred by the reasoner automatically. In order to ease the insertion of new sections, divisions, groups and subgroups into SITC-4 or the deletion of such from SITC-4 or the modification of the SITC-4 structure, we think that SITC-4 tiers should be as independent from each other as possible. This prohibits the extensive use of `rdfs:subClassOf` relations. We therefore state our goal as follows: Class subsumption between SITC-4 sections, divisions, groups and subgroups is to be inferred automatically by the reasoner; the ontology should be constructed such that it is not required to state any `rdfs:subClassOf` relation between members of different tiers. From a more general perspective, we could call this "inferring knowledge from data".

We proceed as follows (see example in code snippet 1, Turtle syntax): We define functional datatype properties `:hasTier1Code`, …, `:hasTier5Code` (see lines 1–3). Next, we define necessary and sufficient conditions for class membership (`owl:equivalentClass`) by restricting the values of these properties (`owl:hasValue`) and by building an intersection of these properties, lines 4–15 of code snippet 1.

```
:hasTier1Code rdf:type owl:DatatypeProperty , owl:FunctionalProperty .
:hasTier2Code rdf:type owl:DatatypeProperty , owl:FunctionalProperty .
:hasTier3Code rdf:type owl:DatatypeProperty , owl:FunctionalProperty .
…
:S246     rdf:type owl:Class ;
          owl:equivalentClass [ rdf:type owl:Class ;
               owl:intersectionOf  (
               [rdf:type owl:Restriction ; owl:onProperty :hasTier1Code ;
                  owl:hasValue "2" ]
               [rdf:type owl:Restriction ; owl:onProperty :hasTier2Code ;
                  owl:hasValue "4" ]
               [rdf:type owl:Restriction ; owl:onProperty :hasTier3Code ;
                  owl:hasValue "6" ]
               )
          ]
.
```

Code snippet 1: Necessary and sufficient conditions for class membership

Now, we can leave it to the reasoner to find the correct positions for all classes in the class hierarchy: Classes with less conditions set are superclasses of classes with more conditions set, assuming that the property values match.

4.5 Class disjointness

SITC-4 is constructed such that SITC-4 sections do not share any SITC-4 divisions, SITC-4 divisions do not share any SITC-4 groups, and SITC-4 groups do not share any SITC-4 subgroups. In other words, the SITC-4 trade code hierarchy is not a poly-hierarchy. To represent this SITC-4 construction principle in an OWL-2 ontology we have to take care for "tier-wise" disjointness of all OWL-2 classes representing SITC-4 sections, divisions, groups, and subgroups.

We could express this by the OWL-2 language construct `owl:AllDisjointClasses`. But we decided to leave the detection of class disjointness to the reasoner:

- We made the `:hasTier1Code`, …, `:hasTier5Code` properties functional. This means that to each individual basic heading at most one distinct tier 1, …, 5 code can be assigned.
- We constructed the classes related to SITC-4 trade codes by necessary and sufficient conditions on the `:hasTier1Code`, …, `:hasTier5Code` properties.
- We assigned different sets of tier 1, …, 5 codes to the SITC-4 trade codes.

This construction enables the reasoner to detect class disjointness automatically.

4.6 Assignment of individuals to classes

A last question remains: Can we leave it to the reasoner to assign SITC-4 basic headings to SITC-4 subgroups? Or in ontological terminology: Can we leave it to the reasoner to assign individuals to classes? Obviously, the `rdf:type` language construct, which is part of OWL-2, would allow us to explicitly assign individuals representing SITC-4 basic headings to SITC-4 classes. But following the idea that a reasoner should be used not only as an instrument for checking a classification, but also as an instrument for partially generating a classification, we would prefer to leave the assignment of individuals to classes to the reasoner. Automatic assignment exploits the `:hasTier1Code`, …, `:hasTier5Code` properties and the value sets assigned to them, because they "work" both as parts of class expressions and as properties for individuals. We re-discuss this problem in more depth in chapter 6 of this paper.

5 From Excel sheet to OWL-2 ontology

Sources for the transformation of SITC-4 to an OWL-2 ontology are both the SITC-4 Excel sheet and the correspondence table Excel sheet. A large number of tools is available to transform from Excel format to RDF format; a comprehensive list of tools for this purpose is given in [25]. We decided against using any of these tools for the following reasons:

- In our case, we do not only have to solve a conversion problem, but also a combination problem.
- We do not simply transform from Excel to RDF format, but from Excel format to OWL "format".
- Some of the listed tools assume the existence of a database scheme that may serve as a conversion aid. For SITC-4 and the correspondence table such a schema is not available.

We decided to develop a dedicated tool chain that is mainly based on XSL(T). Our experience shows that this tool chain can easily be modified to carry out the intended format conversion also for other trade classifications. The tool chain is explained in the following.

After some preprocessing—splitting of SITC-4 trade codes into tier codes, converting validity dates into ISO 8601 format, converting table content from native Excel format to xml format, and other— the transformation is carried out in three steps (white boxes in figure 3):

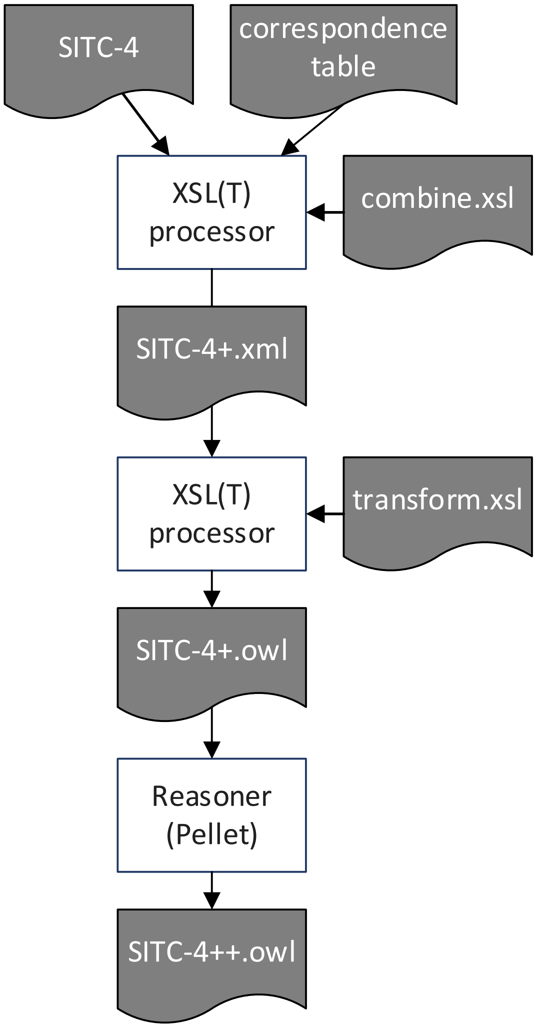

Figure 3: Workflow

1. Both Excel sheets are combined by an XSL(T) transformation (SAXON XSL(T) processor), i.e. CN Mappings are added to the ontology. Additionally, left-alone SITC-4 subgroups are equipped with basic headings from the correspondence table, and "ancestor" elements are generated for orphans from the correspondence table. The output is a file in XML format and denoted by "SITC-4+.xml" in figure 3.
2. The output of step 1 is transformed into an OWL-2 ontology by a second XSL(T) transformation. The resulting output file is denoted by "SITC-4+.owl" in figure 3. OWL is rendered in Turtle syntax.
3. Finally, the output of step 2 is processed by the Pellet reasoner. The reasoner infers additional information that is stored together with its input in a file denoted by "SITC-4++.owl" in figure 3.

The SITC-4+ ontology comprises 1,387 classes and 21,218 individuals in total. 18,231 (!) of these individuals are CN Mappings. Obviously, most SITC-4 basic headings are affected by numerous changes of the mapping CN trade code.

The next section discusses the third step of figure 3 in more detail.

6 Reasoning

To set up the class hierarchy of the SITC-4+ ontology and to assign its individuals to classes, we applied the open-source version of the Pellet reasoner (version 2.3.1). The next chapter shows reasoning benefits in general. To reduce the initially observed reasoner runtimes, we derived from the "original" Pellet reasoner a variant that takes care for parallel assignment of individuals to classes (chapter 6.2). Additionally, we examined reasoner runtimes for three different ontology versions (chapter 6.3).

6.1 Reasoner as "structure generator"

Figures 4 and 5 below show two different graphical representations of a very small snippet from our ontology dealing with "Oranges, mandarins, clementines and similar citrus hybrids …" (SITC-4 subgroup 0571). Both ontology snippets are visualized by the OntoGraf plugin of the ontology editor Protégé. Figure 4 shows the ontology snippet before reasoning, figure 5 after reasoning. A box with a circle indicates a class, a box with a diamond indicates an individual. Arrows between classes show the existence of a subsumption relation or of a domain/range relation, arrows between classes and individuals show class membership, and finally arrows between individuals show object properties.

Two differences can easily be discovered: In figure 4, all classes except class S0 (standing for SITC-4 section 0) are direct subclasses of `owl:Thing`, while in figure 5, classes S0, …, S0571 are in a subsumption relation, and only classes `:CNMapping` and `:SITC-4` are direct subclasses of `owl:Thing`. The arrow between classes `:SITC-4` and `:CNMapping` indicates that class `:SITC-4` is the `rdfs:domain` of `:CNMapping`. Without reasoning, CN Mappings are assigned to a class (namely CNMapping), and thus are rendered in the graphics, while after reasoning individuals representing SITC-4 basic headings, too, are assigned to classes, and thus are rendered. These individuals are members of classes SITC-4, S0, S05, S057, and S0571. Obviously reasoning adds a considerable amount of structuring information to the SITC-4+ ontology.

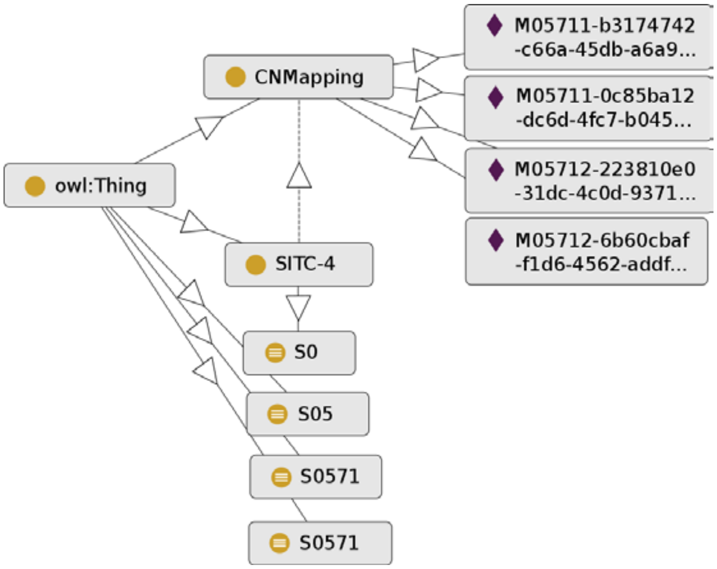

Figure 4: SITC-4 ontology before reasoning

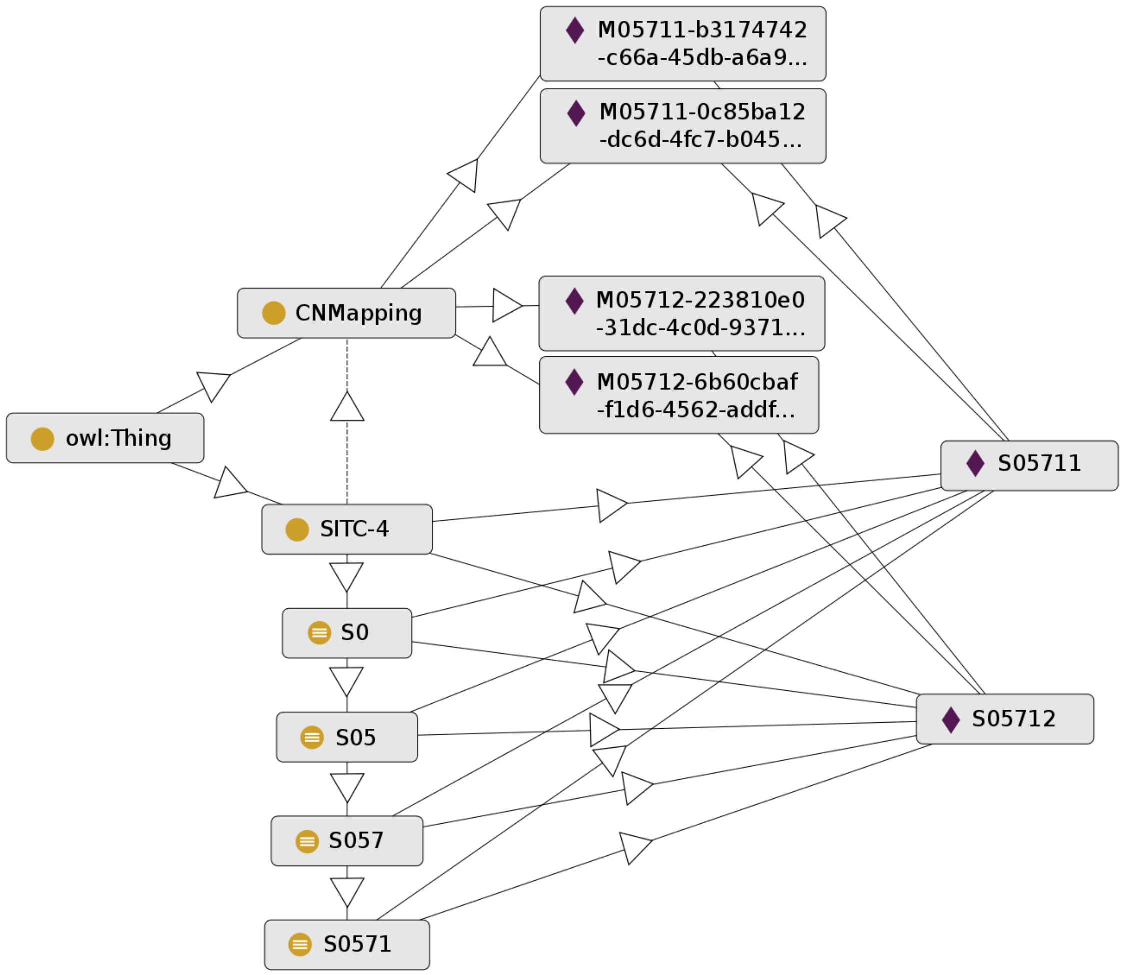

Figure 5: SITC-4 ontology after reasoning

Reasoning is a time-consuming task (see Table 3). Therefore, it is very worthwhile to carefully examine reasoner runtimes. In the following chapters, we describe several reasoning experiments that were conducted to find a good trade-off between reasoner runtime and complexity of ontology design.

6.2 Sequential vs. parallel assignment of individuals to classes

In a number of preparatory reasoning experiments, we made the following observations:

- During the whole runtime, the Pellet reasoner did not use more than approx. 4 Gbyte of main memory.
- The assignment of individuals to classes (signaled to the user by a little popup window announcing the reasoner activity "Realizing") consumes almost all of the Pellet runtime.
- Pellet makes use only of a single core, i.e. it runs strictly sequential during the "Realizing" phase.

With the goal in mind to reduce the reasoner runtime significantly, we concentrated on parallelization of the "Realizing" phase of the Pellet reasoner.

Pellet performs classification of individuals in two loops that are nested: one loop over individuals, the other loop over classes, i.e. the inner loop is performed in full per iteration step of the outer loop. If we denote by *m* the number of classes and by *n* the number of individuals, the reasoner runtime thus depends at least on $m \cdot n$. Inserting the numbers given above for classes and individuals, $m \cdot n$ = 29,429,366 holds for the SITC-4 ontology as proposed in this paper. In the extreme case, this means that the SITC-4 classification would cost approx. $30 \cdot 10^6$ possibly complex computations. It is neither possible to reduce the number of classes, nor the number of individuals.

The pellet.properties file holds a Boolean configuration variable named REALIZE_INDIVIDUAL_AT_A_TIME that lets the user choose which loop is the outer loop: When set to TRUE, the loop over individuals is the outer loop, when set to FALSE, the loop over classes is the outer loop.

For the Pellet option "Outer loop iterates over individuals" an inspection of the Pellet program code revealed that the iteration steps do not have mutual data dependencies. The absence of data dependencies allowed us to design a rather simple parallel version for the "Realizing" code, built upon the Java ExecutorService. We inserted 12 additional lines of code and modified 3 lines. We did not take care for any further optimizations like e.g. adaption of the garbage collection frequency.

Unfortunately, there are mutual data dependencies in the Pellet code for iteration steps for the option "Outer loop iterates over classes"; to avoid error-prone re-programming, we did not try to implement a parallel version for this option. We discuss observed runtimes in the following chapter together with the effects of different ontology versions.

6.3 Implicit vs. explicit assignment of individuals to classes

We analyzed also, if we could reduce reasoner runtimes by giving up the stated goal to let the reasoner do the complete job of assigning individuals to classes. We assumed that by introducing explicit `rdf:type` axioms per individual the classification task could be made easier for the reasoner. Obviously, this solution has the drawback that the reasoner's potential to generate structural relationships between ontology elements is not very well exploited.

To study the effect of inserting explicit `rdf:type` axioms, we produced three subversions of SITC-4+, namely SITC-4+$_{\text{impl}}$, SITC-4+$_{\text{map}}$, and SITC-4+$_{\text{expl}}$:

- Subversion SITC-4+$_{\text{impl}}$ ("implicitly typed individuals") is identical to the initial SITC-4+, i.e. this subversion does not contain any `rdf:type` axioms neither for individuals representing SITC-4 basic headings, nor for CN Mappings.
- In subversion SITC-4+$_{\text{map}}$ ("typed mapping individuals") we introduced `rdf:type` axioms only for CN Mappings. As mentioned before, the SITC-4 ontology comprises 18,231 individuals of this kind.
- In subversion SITC-4+$_{\text{expl}}$ ("explicitly typed individuals") we introduced `rdf:type` axioms for all individuals.

	SITC-4+$_{impl}$	SITC-4+$_{map}$	SITC-4+$_{expl}$
Sequential "Realizing"	3 h 10 min	3 h 15 min	6 min 3 s
Parallel "Realizing"	2 h 34 min	2 h 34 min	2 h 14 min

Table 3: Reasoner runtimes

Table 3 shows the measured mean runtimes, calculated from 3 runs per experiment. Pellet was running the extract subcommand with default parameters. The software environment comprised the Java SE Runtime Environment 1.7.0 provided by a Java 64-Bit Server VM on a Linux system, the hardware environment comprised an Intel Core i7 processor with 2.9 GHz clock rate and 8 cores. We discuss the shown figures as follows:

- For the SITC-4+$_{impl}$ subversion, parallelization allowed us to reduce the reasoner runtime to approx. 80% of the observed reasoner runtime with sequential "Realizing" phase.
- The SITC-4+$_{map}$ subversion does not reduce the observed runtimes, neither for the sequential, nor for the parallel "Realizing" phase.
- For the sequential "Realizing" phase, the SITC-4+$_{expl}$ subversion drastically reduces the observed runtime to approx. 3% of the SITC-4+$_{impl}$ subversion.
- The runtimes observed for the parallel "Realizing" phase are almost the same for all different SITC-4+ subversions.

Though a detailed analysis of the Pellet behavior is beyond the scope of this paper, we may assume that the Pellet option "Outer loop iterates over classes" is the conceptually better choice for our problem. We observed during several runs that the Pellet reasoner deals with the CNMapping class in the first iteration step. These allows Pellet to get rid of the 18,231 CN Mappings already in this iteration step. The reasoner does not need to take these individuals into account in following iteration steps, because all classes are disjoint: After assigning the CN Mappings to the CNMapping class, a membership in other classes is not possible for the CN Mappings. Per following iteration step, the reasoner has to deal only with the remaining 2987 individuals. The observed runtimes suggest that assigning these 2987 individuals to the 1383 SITC-4 classes is very hard, when the reasoner must infer class membership from properties, and very easy, when class membership is explicitly declared by `rdf:type` axioms.

Parallel "Realizing" for the "Outer loop iterates over individuals" option of Pellet can compensate its inherent disadvantage by applying a large enough number of cores, but it remains the fact that the number of classes in the inner loop cannot be reduced in any iteration. The full number of approx. $30 \cdot 10^6$ complex computations must be carried out.

We think that for larger ontologies—being populated with even more individuals—the Pellet "Outer loop iterates over classes" option may exhibit its advantage even clearer. An effort to design a parallel version for the "Realizing" phase for this option seems very worthwhile.

7 Summary and outlook

In this paper we presented an approach to convert the Standard International Trade Classification into a semantically rich OWL-2 ontology. We discussed some design choices, and reported from our experience with the Pellet reasoner. We think that the outcome of our work—besides the actual SITC-4 ontology—is twofold: We learned that ontology development even for seemingly regular structures may become quite a complex engineering task, and that medium to large size ontologies require careful design that takes into account not only ontological structures but also reasoning and especially reasoning runtimes. The benefit of reasoning, though, is that verifiable structure information can be added to data in an automatic procedure.

Both the developed ontology for SITC-4 and the XSL(T) style sheets will be published on the authors' website (comsys.informatik.uni-kiel.de).

In our ongoing work we concentrate on integrating trade data in the given ontology, and we extend our effort to the Combined Nomenclature.

References

[1] United Nations Statistics Division: *Standard International Trade Classification, Revision 4*. http://unstats.un.org/unsd/trade/sitcrev4.htm, retrieved 2015-09-01.

[2] http://unstats.un.org/unsd/pubs/gesgrid.asp?id=104, retrieved 2015-09-01.

[3] European Commission: *Combined Nomenclature*. http://ec.europa.eu/eurostat/ramon/index.cfm, retrieved 2015-09-01.

[4] World Customs Organization: *What is the Harmonized System*? http://www.wcoomd.org/en/topics/nomenclature/overview/what-is-the-harmonized-system.aspx, retrieved 2015-09-01.

[5] Eurostat: *Correspondence table between Standard International Trade Classification, Rev. 4 and Combined Nomenclature 2015*. http://ec.europa.eu/eurostat/ramon/other_documents/combined%20nomenclature/conversion_tables/CN2015_SITC4.zip, retrieved 2015-09-01.

[6] Eurostat: *Combined Nomenclature Ad hoc conversion tables.* http://ec.europa.eu/eurostat/ramon/other_documents/combined%20nomenclature/conversion_tables/CN_SITC_2015.zip, retrieved 2015-09-01.

[7] Staab, S., Studer, R. (Eds.): *Handbook on Ontologies, 2nd Ed.*, Springer, 2009.

[8] International Organization for Standardization: *ISO 1087-1:2000, Terminology work − Vocabulary − Part 1: Theory and application*.

[9] Krötzsch, M., Simančík, F., Horrocks, I.: *A Description Logic Primer*. arXiv:1201.4089v3 [cs.AI] 3 Jun 2013, retrieved 2015-10-20

[10] Tolksdorf, R., Bizer, Ch., Eckstein, R., Heese, R.: Business to Consumer Markets on the Semantic Web. In: *On The Move to Meaningful Internet Systems*, OTM 2003-WMS, Springer LNCS 2889, S. 816-828, Catania, Italy, November 2003.

[11] Ding, Y., Fensel, D., Klein, M., Omelayenko, B., Schulten, E.: The Role of Ontologies in eCommerce. In: Staab, S., Studer, R. (Eds.), Handbook of Ontologies, 2004, Springer, 2003.

[12] Deutsche Zentralbibliothek für Wirtschaftswissenschaften: *Digitale Reichsstatistik*. http://zbw.eu/de/ueber-uns/arbeitsschwerpunkte/forschungsdatenmanagement/digitale-reichsstatistik/, retrieved 2015-09-02.

[13] Deutsche Zentralbibliothek für Wirtschaftswissenschaften: *Standard Thesaurus Wirtschaft*. http://zbw.eu/stw/version/latest/about, retrieved 2015-09-02.

[14] SKOS Simple Knowledge Organization System. http://www.w3.org/2004/02/skos/, retrieved 2015-09-02.

[15] International Organization for Standardization: *ISO 25964 Information and documentation - Thesauri and interoperability with other vocabularies.* Citation from Wikipedia, https://en.wikipedia.org/wiki/ISO_25964, retrieved 2015-09-02.

[16] eCl@ss: *Classification and Product Description*. http://www.eclass.de/eclasscontent/index.html.en, retrieved 2015-09-02.

[17] Hepp, M., de Bruijn, J.: GenTax: A Generic Methodology for Deriving OWL and RDF-S Ontologies from Hierarchical Classifications, Thesauri, and Inconsistent Taxonomies. In: Franconi, E., Kifer, M., May, W. (Eds.): *The Semantic Web: Research and Applications*. Berlin, Heidelberg (Springer), Lecture Notes in Computer Science, vol. 4519, 2007, 129-144.

[18] UN Development Programme: *United Nations Standard Products and Services Code*. http://www.unspsc.org/, retrieved 2015-09-02.

[19] Stolz, A., Rodriguez-Castro, B., Radinger, A., Hepp, M.: PCS2OWL: A Generic Approach for Deriving Web Ontologies from Product Classification Systems. In: *Proceedings of the 11th Extended Semantic Web Conference (ESWC 2014)*, May 25-29, 2014, Crete, Greece, Springer LNCS, pp. 644-658.

[20] Caracciolo, C., Heguiabehere, J., Gangemi, A., Baldassarre, C., Keizer, J., Taconet, M.: Knowledge Management at FAO: A Case Study on Network of Ontologies in Fisheries. In: Suárez-Figueroa, M.C., Gómez-Pérez, A., Motta, E., Gangemi, A. (Eds.): Ontology Engineering in a Networked World. Berlin Heidelberg (Springer), 2012, 383-405.

[21] Food and Agriculture Organization of the United Nations: CWP Handbook of Fishery Statistical Standards. Section R: FISHERY COMMODITIES CLASSIFICATION. CWP Data Collection. In: FAO Fisheries and Aquaculture Department. Rome. Updated 10 January 2002. http://www.fao.org/fishery/cwp/handbook/R/en, retrieved 2015-10-26.

[22] UDC Consortium: *Universal Decimal Classification*. http://www.udcc.org/, retrieved 2015-09-02.

[23] Statistisches Bundesamt: Deutsche Übersetzung der Standard International Trade Classification, Revision 4, der Vereinten Nationen, Ausgabe 2006. https://www.destatis.de/DE/Methoden/Klassifikationen/Aussenhandel/DeutscheFassungSITC.pdf?__blob=publicationFile, retrieved 2015-09-02.

[24] Noy, N.F., McGuinness, D.L.: *Ontology Development 101: A Guide to Creating Your First Ontology*. http://protege.stanford.edu/publications/ontology_development/ontology101.pdf, retrieved 2015-10-15.

[25] W3C Wiki, http://www.w3.org/wiki/ConverterToRdf, retrieved 2015-10-20.

[26] Noy, N., Rector, A. (eds.), Hayes, P., Welty, C.: *Defining N-ary Relations on the Semantic Web.* W3C Working Group Note 12 April 2006, http://www.w3.org/TR/swbp-n-aryRelations/, retrieved 2015-10-15.

[27] Krieger, H.-U., Declerck, Th.: An OWL Ontology for Biographical Knowledge. Representing Time Dependent Factual Knowledge. In: *Proceedings of the First Conference on Biographical Data in a Digital World 2015*, http://ceur-ws.org/Vol-1399/paper16.pdf, retrieved 2015-10-15.