GECCO: GERMAN CRAFTS & CRAFTSMEN ONTOLOGY
A Common Crafts Ontology

Andrea Horch and Maximilien Kintz
Institute of Human Factors and Technology Management, IAT University of Stuttgart
Nobelstr. 12, 70569 Stuttgart, Germany

Keywords: Ontology, Semantic Web, Crafts, Craftsmen.

Abstract: In this paper we introduce GeCCO (German Crafts & Craftsmen Ontology), an ontology to establish a common crafts classification, to be used for example in craftsmen search systems. Current standardized categorization systems, which order crafts, do not include all needed logical connections between crafts and craftsmen as well as the synonyms for a sufficient use in a search system. Actual craftsmen search systems have their own categorizations of crafts and craftsmen, but they have the same problems as the standardized systems.
The crafts ontology proposed in this paper solves the mentioned problems by creating an extensible class hierarchy as well as by defining synonyms and all needed logical connections between the classes. For quality assurance the ontology was evaluated by crafts experts.

1 INTRODUCTION

There are several search platforms and online marketplaces like MyHammer[1] or blauarbeit.de[2] for searching craftsmen. Most of them are organized by keyword search combined with a search by various categories. These platforms are not using a consistent and standardized form to categorize crafts.

Currently in Germany there are two popular categorization systems ordering crafts:

1. The German Crafts Code[3] (Handwerksordnung), which lists more than 100 crafts and crafts similar trades.
2. The eCl@ss[4] classification system for products and services.

These systems are not sufficient for use in crafts search services since they do not include enough logical connections between a craft and the associated craftsmen. The German Crafts Code does not describe the overlap of craftsmen having common activities. eCl@ss shows only some connections between a craft and its performing craftsmen in the form of keywords.

[1]http://news.myhammer.com/
[2]http://www.blauarbeit.de/
[3]http://www.gesetze-im-internet.de/hwo/
[4]http://www.eclass.de/

Thus there are no existing crafts categorization systems sufficient for the use in search systems. The ontology proposed in this paper shall close this gap. We named it *GeCCO (German Crafts & Craftsmen Ontology)* since it is modeled on the structure of the german crafts like given in the German Crafts Code.

The paper is structured as follows: section 2 explains the limitations of current crafts classifications and search systems. Section 3 gives some examples for related work. Section 4 introduces the approach for the ontology development. The proposed concepts of the ontology are presented in section 5. The evaluation is shown in section 6. We conclude and give some future prospects in section 7.

2 MOTIVATION

The common crafts ontology proposed in this paper focuses on two aspects: the creation of a consistent crafts categorization and the development of a crafts model including all logical connections between crafts and the associated craftsmen by using semantic technologies.

Testing five of the most popular online craftsmen search systems (see table 1) we identified the following current problems in their crafts categorizations:

1. **Redundancy of Classes:** a craftsman class has more than one master category. There is only

Horch A. and Kintz M..
GECCO: GERMAN CRAFTS & CRAFTSMEN ONTOLOGY - A Common Crafts Ontology.
DOI: 10.5220/0003902003550360
In *Proceedings of the 8th International Conference on Web Information Systems and Technologies* (WEBIST-2012), pages 355-360
ISBN: 978-989-8565-08-2

a hierarchical connection between the craftsman class (presented in each master class) and its direct master class, but there are no logical connections between the craftsman class and all of its master classes (see figure 1) or between these craftsman classes among each other.

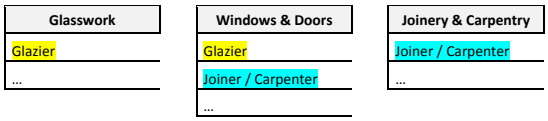

Figure 1: Redundancy of craftsman classes.

This kind of classification returns different search results for each instance of the craftsman class.

2. **Fuzzy Classification:** a craftsman class is not assigned to the expected master class. Instead, it is categorized in a fuzzy named class like "Miscellaneous" or "Other" (see figure 2).

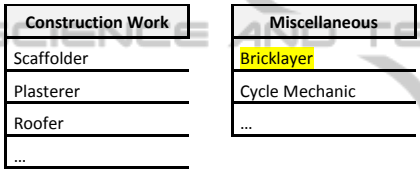

Figure 2: Fuzzy classification.

Thus, the craftsman class is very hard to find.

3. **Disregard for Equivalent Classes:** equivalent craftsman classes are assigned to different master classes and there is no logical connection between them (see figure 3).

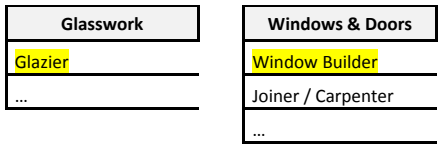

Figure 3: Disregard for equivalent classes.

That means the equivalent classes are completely separated from each other therefore they return different search results.

Other problems are missing synonyms and unregarded changes in the structure of a craft like the consolidation of some craft trades to one trade (in Germany the trades "forger" and "locksmith" were combined to the trade "metal worker" in the 1980s (Berufe-Lexikon, 2011)).

For solving all introduced problems a system representing the objects of the crafts domain (classification) and the relationships among them is needed. For this purpose an ontology was created and is proposed in this paper. An ontology is defined as *"an explicit specification of a conceptualization"* (Gruber, 1993), which represents the domain knowledge including all objects and their relationships.

Table 1: Tested crafts search systems.

Platform Name	URL
MyHammer	http://www.myhammer.com
Find a Craftsman	http://www.findacraftsman.com
blauarbeit.de	http://www.blauarbeit.de
handwerker24	http://www.handwerker24.de
jobdoo	http://www.jobdoo.de

3 RELATED WORK

The ASTECH project (Pralon and Million-Rousseau, 2011) defines a craft ontology and terminologies for knowledge sharing in the sector of the renewable energies area across Europe. The Web Ontology for Products and Services (eClassOWL[5]) the semantic model of the eCl@ss classifcation system (see section 1) is an approach to classify products and services, which also includes some crafts services.

4 DESIGN APPROACH

(Uschold and King, 1995) introduce a methodology that presents several recommendations and steps for building ontologies. Some analysis of this methodology done in (López, 1999) results in the finding, that the methodology does not propose a life cycle for ontology design, which could be a problem when extending or reengineering the ontology for purposes of quality assurance.

Nevertheless we have chosen the mentioned methodology for creating our ontology since it is a purpose driven approach of ontology design. In some steps we adapted it a bit and integrated a cycle for the purposes of quality assurance. After including our adjustments the methodology results in the following steps for ontology design (see figure 4):

1. **Identification of Purpose(s).**
   (Uschold and King, 1995) determine that clarifying the purpose(s) of the ontology (use cases and users) is important for the ontology design.
2. **Ontology Capture.**
   This step covers the identification of the key concepts like classes, class hierarchies and the rela-

[5]http://www.heppnetz.de/projects/eclassowl/

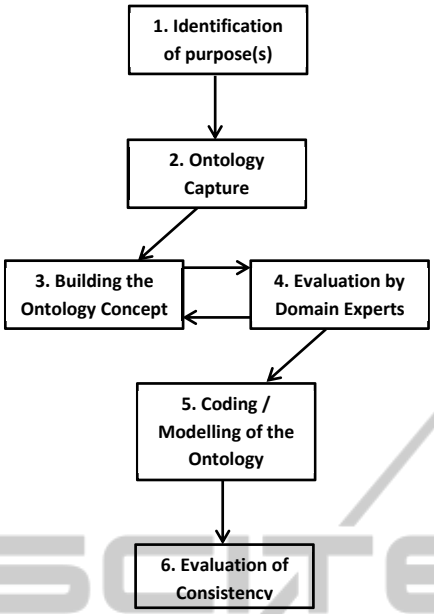

Figure 4: Ontology design steps.

tionships among them. Additionally it includes the definition of clear textual names for the classes and relations (Uschold and King, 1995).

3. **Building the Ontology Concept.**
   This part of ontology design contains the creation of triples (subject, predicate, object; for more information see (Segaran et al., 2009)) to sketch the ontological concept (classes, class hierarchies, relationships) in a form readable and understandable for domain experts (who usually are no ontology experts); like e.g. tables of an Excel sheet.
4. **Evaluation by Domain Experts.**
   Some domain experts evaluate the created concept. If necessary, they do some corrections and provide suggestions for improvement. This step and step No. 3 can be repeated till the experts confirm the correctness of the ontology concept.
5. **Coding/Modeling of the Ontology.**
   After the first evaluation step the ontology becomes coded or modeled in an ontology description language like OWL (W3C, 2009) or DAML + OIL (Stevens et al., 2003) including all logical relationships.
6. **Evaluation of Consistency.**
   Check the ontology for conceptual consistency; e.g. by using a reasoner like HermiT (Shearer et al., 2008) or Pellet (Sirin and Parsia, 2004).

5 CRAFTS ONTOLOGY

To model the ontology we decided to use Protégé since it is an open source editor, which underlies a continuous development process. Furthermore Protégé offers support as well as a lot of features like several reasoners (Horridge et al., 2007).

The ontology was built in the standardized Web Ontology Language (OWL 2 DL) (W3C, 2009). Some parts of the ontology we want to describe in detail in this paper are held in Manchester OWL Syntax (W3C, 2009) (Horridge et al., 2006) and some description logics (Baader et al., 2008) for clarity.

The basis of the ontology is built by the four main classes *Craft*, *Craftsman*, *Object* and *TypeOfLoss* (see figure 5), which are disjoint from each other.

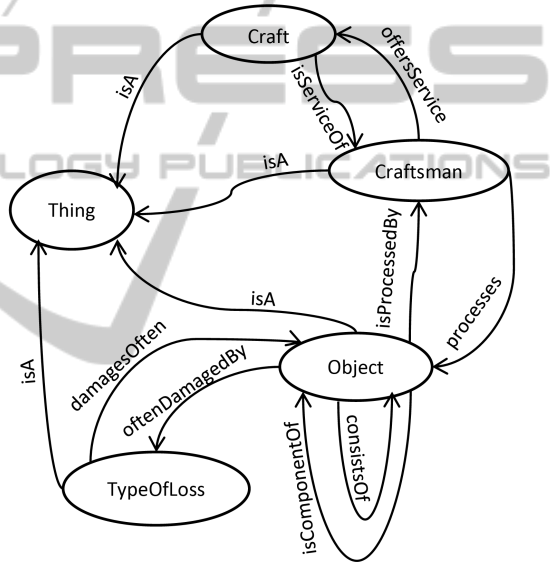

Figure 5: Ontology structure.

These four main classes have all the same structure. They are subclasses of the class *Thing* and have an English and a German label.

```
Class: Craft

    Annotations:
        label "craft"@en
        label "Handwerk"@de,

    SubClassOf:
        Thing

    DisjointWith:
        Craftsman, Object, TypeOfLoss
```

Listing 1: Class craft.

The class *Craft* (see listing 1) is the main class of all different crafts like masonry, roofing work, etc. The class *Craftsman* is the main class of all craftsmen like a mason or a roofer. The class *Object* comprises the classes of objects, which can be processed by a craftsman. *TypeOfLoss* is the main class of the different types of losses, which can damage an object; e.g. the breakage of glass or a fire loss.

Like shown in figure 5 the classes of the ontology are restricted by eight different object properties. Each two of the defined object properties are inverse. The object property *processes* and its inverse *isProcessedBy* define which craftsman processes which object(s) (see listing 2).

```
ObjectProperty: processes

    Annotations:
        label "processes"@en,
        label "bearbeitet"@de

    Characteristics:
        Asymmetric,
        Irreflexive

    Domain:
        Craftsman

    Range:
        Object

    InverseOf: 'is processed by'
```

Listing 2: Object property processes.

```
ObjectProperty: 'is component of'

    Annotations:
        label "is component of"@en,
        label "ist Bestandteil von"@de

    Characteristics:
        Transitive

    Domain:
        Object

    Range:
        Object

    InverseOf: 'consists of'
```

Listing 3: Object property isComponentOf.

The object property *offersService* and its inverse *isOfferedBy* describe which craftsman offers which craft.

Which type of loss often damages which object(s) is given by the object property *damagesOften* and its inverse *oftenDamagedBy*.

The structure of the object properties *offersService*, *damagesOften* and their inverses is equal to the structure of the object property *processes* and its inverse. The six depicted object properties are all defined as asymmetric and irreflexive.

The object property *isComponentOf* (see listing 3) and its inverse *consistsOf* is a special property, because it defines which object consists of which other object(s). That means, the object property is transitive; so domain as well as range refer to the class *Object*.

All crafts are defined as subclasses of the class *Craft*. The class *Masonry*, for instance, describes the characteristics of the craft masonry (see equation 1).

Looking at the definition of the class *Masonry* one can see that masonry is a craft offered by a mason or a concrete worker (object property: 'is service of'). The keyword "some" defines that it exists at least one mason (class *Mason*) and at least some concrete worker (class *ConcreteWorker*) whose offered services include masonry.

$$\begin{aligned} \text{Masonry} \equiv \text{Craft} \sqcap \exists \mathit{isServiceOf}.\text{Mason} \\ \sqcap \exists \mathit{isServiceOf}.\text{ConcreteWorker} \end{aligned} \quad (1)$$

It is impossible to claim that there are no other craftsmen offering masonry work. Craft is very complex, it is imprecisely defined which craftsmen offer which crafts and some crafts overlap in some activities. Thus, class restrictions can only be defined by the quantifier $\exists$, which defines existential restrictions, and may not be defined by $\forall$ defining universal restrictions.

The subclasses of the main class *Craftsman* define the different craftsmen, e.g. the class *Mason*. The logical definition of the class *Mason* is given by the following term:

$$\begin{aligned} \text{Mason} \equiv \text{ConcreteWorker} \equiv (\text{Craftsman} \\ \sqcap \exists \mathit{offersService}.\text{Masonry} \\ \sqcap \exists \mathit{offersService}.\text{NaturalStoneProcessing} \\ \sqcap \exists \mathit{offersService}.\text{InsulationWork} \\ \sqcap \exists \mathit{offersService}.\text{DiaphragmWallWork} \\ \sqcap \exists \mathit{offersService}.\text{ScreedWork}) \end{aligned} \quad (2)$$

StormLoss is an example for a subclass of the main class *TypeOfLoss*. These subclasses define the types of loss, which often damage some objects (equation 3). For determining the objects a type of loss often damages, one also has to use an existential restriction, because a type of loss can damage much more objects than can be defined by the ontology.

From the subclasses of the main class *Object* one can get lots of information like which types of loss often damages the object, which craftsmen process it and if it is a component of another object or if it consists of other objects. An example is given by the class *WindowGlass* (see equation 4).

$$\begin{aligned} \text{StormLoss} \equiv (&\text{TypeOfLoss} \\ &\sqcap \exists damagesOften.\text{Fence} \\ &\sqcap \exists damagesOften.\text{Roof} \\ &\sqcap \exists damagesOften.\text{Window} \\ &\sqcap \exists damagesOften.\text{RoofCovering} \\ &\sqcap \exists damagesOften.\text{WindowGlass}) \end{aligned} \tag{3}$$

$$\begin{aligned} \text{WindowGlass} \equiv (&\text{BuildingElement} \\ &\sqcap \exists isProcessedBy.\text{Glazier} \\ &\sqcap \exists isProcessedBy.\text{WindowFitter} \\ &\sqcap \exists isComponentOf.\text{Window} \\ &\sqcap \exists oftenDamagedBy.\text{BreakageOfGlass} \\ &\sqcap \exists oftenDamagedBy.\text{StormLoss} \\ &\sqcap \exists oftenDamagedBy.\text{Vandalism} \\ &\sqcap \exists oftenDamagedBy.\text{Burglary} \\ &\sqcap \exists oftenDamagedBy.\text{Earthquake} \\ &\sqcap \exists oftenDamagedBy.\text{Landslide} \\ &\sqcap \exists oftenDamagedBy.\text{FireLoss} \\ &\sqcap \exists oftenDamagedBy.\text{Avalanche}) \end{aligned} \tag{4}$$

The object property *isComponentOf* and its inverse *consistsOf* are transitive. For example: a tile consists of clay and a tile roof consists of tiles, which implies that a tile roof also consists of clay.

$$\begin{aligned} \text{SolarEnergySpecialist} \equiv (&\text{Craftsman} \\ &\sqcap (\text{AirHeatingFitter} \sqcup \text{Electrician} \\ &\sqcup \text{ElectronicsTechnician} \sqcup \text{HeatingEngineer} \\ &\sqcup \text{Roofer} \sqcup \text{StoveEngineer}) \\ &\sqcap \exists offerService.\text{SolarEngineering} \\ &\sqcap \exists processes.\text{SolarHotWaterSystem} \\ &\sqcap \exists processes.\text{PhotovoltaicSystem}) \end{aligned} \tag{5}$$

GeCCO models synonyms as labels (classes have different English and German labels to describe synonyms). Examples for synonyms are the terms *joiner* and *cabinet maker* or *masonry* and *walling*. Special craftsmen are subclasses of other craftsman classes; e.g. *solar expert* is subclass of roofer or electrician (equation 5).

GeCCO defines a total of 524 classes and 8 object properties. The main class *Craft* has 106 subclasses. The main class *Craftsman* has 185 subclasses. 211 classes are subclasses of the main class *Object*, whereas the last main class *TypeOfLoss* counts 17 subclasses.

6 EVALUATION

From the design criteria described in section 4 and the ones mentioned in (Gruber, 1993) we can derive the following criteria for doing the evaluation:

1. **Structural/Technical Consistency.**
   This first point affects the technical correctness of the ontology. That means, if it is structurally as well as logically correct. GeCCO was tested for its consistency by using three different reasoners: HermiT[6], Pellet[7] and FaCT++[8]. These evaluation steps meet the criteria of *Clarity*, *Coherence* and *Extendibility* described in (Gruber, 1993) and the *Ontology Capture* step of section 4.
2. **Functional/Textual Consistency.**
   Functional / textual consistency means the accuracy of the ontology content, or rather the ontology concept. That affects the ontology class hierarchy, the logical connections between the classes as well as the design of synonyms.

   For these purposes we created an Excel sheet including the crafts and craftsman classes of the ontology as well as the connections between them in form of different tables. We designed it in the form of triples (subject, predicate, object). The first design was made in Excel since it was the format of choice of our domain experts. These domain experts, who supported the creation of the ontology domain concept, were employees of the Chamber of Crafts of Stuttgart. Ontology modeling activities in OWL did not start before the evaluation of the Excel version of the domain concept was finished.

   This evaluation approach fits the design criteria of *Minimal encoding bias* and *Minimal ontological commitment* as introduced in (Gruber, 1993) and the *Ontology Concept* step of section 4.
3. **Practical Suitability.**
   In the first design step of section 4 we mentioned the proposition of (Uschold and King, 1995) , who determine that it is essential for ontology design to clarify the purpose(s) of the ontology. From this determination we derive the importance for ontology evaluation to examine the suitability of the created ontology for the purpose of its creation defined in the first step of the ontology design.

   Therefore we developed a semantic craftsmen search based on GeCCO, which we tested for different search queries. The craftsmen search

[6] http://hermit-reasoner.com/
[7] http://clarkparsia.com/pellet/
[8] http://owl.man.ac.uk/factplusplus/

is connected to a test-database containing real craftsmen data.

The main problem GeCCO focuses on is the fact that laymen often do not know which craftsman is processing which damage. GeCCO helps a user of a craftsmen search system to find the right craftsman for eliminating a damage. Therefore a user can just type in the name of the damaged item or the craft he is searching for as a keyword into the semantic search and GeCCO will find the right type(s) of craftsmen for doing the database search.

The evaluation of GeCCO has shown that GeCCO is structural and technical as well as functional and textual consistent. The test for the practical suitability for the use case of a craftsmen search system demonstrated the comfort of a semantic system based on a domain ontology.

7 CONCLUSIONS/FUTURE WORK

We have introduced GeCCO, an ontology describing crafts. GeCCO delivers a good class basis for crafts search services or crafts translation services, but does not claim to be complete.

In this paper we have also shown how to evaluate the ontology. We have checked the ontology for structural consistency and evaluated the concept by the support of domain experts. The ontology was tested as basis of a semantic craftsmen search and has passed the test. The test has shown the comfort of an ontology-based user guidance for web searches, especially for complex search domains like crafts.

Further work should focus on the maintanance and the extension of GeCCO. One way to extend the ontology could be the use of GeCCO in an online search system in conjunction with an automatism, which learns from user entries and extends GeCCO automatically by using the acquired knowledge.

Furthermore GeCCO could be extented by other modules like eClassOWL (see section 3), e.g. by embedding the product-classes of eClassOWL as object-classes into GeCCO.

ACKNOWLEDGEMENTS

The work published in this article was partially funded by the openXchange project of the German Federal Ministry of Economy and Technology under the promotional reference 01MQ09011.

REFERENCES

Baader, F., Horrocks, I., and Sattler, U. (2008). Description Logics. In van Harmelen, F., Lifschitz, V., and Porter, B., editors, *Handbook of Knowledge Representation*, chapter 3, pages 135–179. Elsevier B.V.

Berufe-Lexikon (2011). Berufsbild Schlosser/in @http://www.berufe-lexikon.de/berufsbild-beruf-schlosser.htm. http://www.berufe-lexikon.de/berufsbild-beruf-schlosser.htm.

Gruber, T. R. (March 1993). Toward principles for the design of ontologies used for knowledge sharing. In *In Formal Ontology in Conceptual Analysis and Knowledge Representation*. Kluwer Academic. Available as Stanford Knowledge Systems Laboratory Report KSL-93-04.

Horridge, M., Drummond, N., Goodwin, J., Rector, A., Stevens, R., and Wang, H. H. (2006). The manchester owl syntax. Technical report, WebOnt.org - A WEB ONTOLOGY PORTAL, The University of Manchester.

Horridge, M., Jupp, S., Moulton, G., Rector, A., Stevens, R., and Wroe, C. (2007). A practical guide to building owl ontologies using protégé 4 and co-ode tools.

López, M. F. (1999). Overview of Methodologies for Building Ontologies. In *Proceedings of the IJCAI-99 Workshop on Ontologies and Problem Solving Methods (KRR5) Stockholm, Sweden, August 2, 1999*.

Pralon, S. and Million-Rousseau, C. (2011). Astech project: craft ontology and terminologies to share knowledge in renewable energies area across europe. *International Journal of Reasoning-based Intelligent Systems (IJRIS), Volume 3 - Issue 3/4 - 2011*.

Segaran, T., Evans, C., and Taylor, J. (2009). *Programming the Semantic Web*. O'Reilly, Wiesbaden.

Shearer, R., Motik, B., and Horrocks, I. (2008). Hermit: A highly-efficient owl reasoner. In Dolbear, C., Ruttenberg, A., and Sattler, U., editors, *OWLED*, volume 432 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Sirin, E. and Parsia, B. (2004). Pellet: An owl dl reasoner. In *Description Logics*.

Stevens, R., Wroe, C., Bechhofer, S., Rector, A., and Goble, C. (2003). Building Ontologies in DAML+OIL. *Comparative and Functional Genomics*, 4(1):133–141.

Uschold, M. and King, M. (1995). Towards a methodology for building ontologies. In *Workshop on Basic Ontological Issues in Knowledge Sharing, held in conjunction with IJCAI-95*, Montreal, Canada.

W3C (2009). *OWL 2 Web Ontology Language Document Overview*.

W3C (2009). OWL 2 web ontology language manchester syntax. http://www.w3.org/TR/owl2-manchester-syntax/.