An Ontology Driven ESCO LOD Quality Enhancement

Adham Kahlawi
Department of Statistics, Computer Science, Applications
University of Florence, Florence, Italy

***Abstract*—The labor market is a system that is complex and difficult to manage. To overcome this challenge, the European Union has launched the ESCO project which is a language that aims to describe this labor market. In order to support the spread of this project, its dataset was presented as linked open data (LOD). Since LOD is usable and reusable, a set of conditions have to be met. First, LOD must be feasible and high quality. In addition, it must provide the user with the right answers, and it has to be built according to a clear and correct structure. This study investigates the LOD of ESCO, focusing on data quality and data structure. The former is evaluated through applying a set of SPARQL queries. This provides solutions to improve its quality via a set of rules built in first order logic. This process was conducted based on a new proposed ESCO ontology.**

Keywords*—*ESCO; linked open data; ontology; semantic web; data quality; SPARQL; OWL; metadata

I. Introduction

Labor market governance is one of Europe's top priorities. Market governance is an important challenge because the job market is a complex network involving many diverse actors. Therefore, the European Commission has proposed European Skills, Competences, Qualifications and Occupations (ESCO)[1] (the multilingual European Skills, Competences, Qualifications, and Occupations classification) as a standard language of work. To enhance its use and reuse, ESCO has published its dataset as Linked Open Data (LOD). Meanwhile, some intelligent services have been provided by the use of LOD like entity search, personalized recommendation and so on [1][2]; Furthermore, the ability to add a language tag to different labels [3], which belongs to one Universal Resource Identifier (URI), enables the use of this system in different countries. For instance, the financial crisis of 2007–2008 increased the rate of unemployment in Europe, especially in Spain where youth unemployment exceeded 50 percent [4]. At the same time in some economic sectors such as engineering and healthcare, companies were not able to find the workforce they need [5]. The EU seeks to reduce this problem by achieving two objectives: 1) helping the jobseekers find a suitable job in another European country, and 2) enabling people to refocus on their careers with a future outlook [6]. Based on this, ESCO was born to help someone who studied in Germany, and lived in Greece to work in Italy by the linked open data that achieve semantic interoperability throughout Europe. Nevertheless, data diffusion is not the only priority to have a good knowledge system on the labor market also data quality has to be assessed. Data quality has always been the focus of researchers' attention for the many challenges it faces [7][8]. Several methodologies have been developed to enhance as well as to assess data quality [9]. For these reasons, any Linked Open Data (LOD) has to consider these aspects before being published.

In order to solve these issues, this study seeks to make the ESCO LOD more structured and more accurate in providing search results.

Section 2 in this article addresses the concept of data quality, data quality dimensions and the related methods of evaluation. Section 3 explains the ESCO structure in details. Section 4 provides a proposal to redesign the structure of ESCO ontology. Section 5 evaluates the new ontology.

II. Linked Open Data and Data Quality

The LOD has been considered as the cornerstone of the semantic web vision and as windows through which data is published in the web. Nowadays there are millions of LOD published in the web [10] at different quality. The data quality is defined as the ability to use and reuse data in a particular application or use case [11]. Data with quality problems might be useful in some cases as long as the quality is within the required range [12]. Nevertheless, it has many challenges. In particular, as explained in [13], the data is published by different providers so that a question of data confidence might be raised. Second, data increases rapidly, making its quality difficult to assess. Third, the level of data quality has been determined from the point of view of the system provider. In fact, when LOD is reused for a different purpose to the initial intention of the provider, certain difficulties are encountered due to the issue of data quality required for the new objective. Data quality has multiple dimensions [14]. In addition, these dimensions range from accessibility to completeness through comprehension. The quality dimensions pose certain challenges [15] such as: a) the issues that the quality of information is dependent solely on the data provider, b) the rapid increase of amount of data makes it more difficult to assess its quality, c) the preparation of the linked open data to be able to reused by third party in a way not expected by the provider, d) the linked open data is a dynamic environment, which requires up-to-date changes to reflect the real world.

Although data quality cannot be assessed with an absolute measurement, LOD can be considered as a useful tool to determine its fitness for reuse.

[1] https://ec.europa.eu/esco/portal

Multiple methodologies have been developed to improve the quality of linked open data; such as: using the statistical distributions to increase the quality of incomplete and noisy Linked Data sets [16]. The authors proposed a method to demonstrate the understandability problems of Resource Description Framework (RDF) data by using the different technologies provided by the semantic web.

The assessment of quality for LOD can be divided into three categories: automated [17], semiautomatic [18], and manual [19]. This article adopts the methodology used in "Test-driven Evaluation of Linked Data Quality" [20] to assess the quality LOD. The method defines some query based text cases implemented with the use of SPARQL (query language for RDF) query templates.

This article focuses on the case of the LOD of the project European Skills, Competences, Qualifications and Occupations (ESCO).

III. ESCO LOD Assessment

ESCO has published its ontology and its LOD. In Fig. 1, the ESCO ontology[2] is depicted while Fig. 2, exhibits the class structure of ESCO LOD that is represented by stardog server[3].

LOD is the new opportunity for sharing and reusing; meanwhile, the ontology forms the main joint of this LOD that weaves the data together [21]. In contrast, comparing these two structures ESCO ontology and ESCO LOD identifies some questions. In order to assess the capability of the current ESCO ontology to being exploited of retrieve valuable information from the related LOD, according to [20] we identified a number of assessment queries.

A. *Resource Description Framework Schema and Web Ontology Language Metadata in ESCO LOD are Missing*

It can be argued that the concepts of class, subclass, data property, object property, and individual lacks a clear definition.

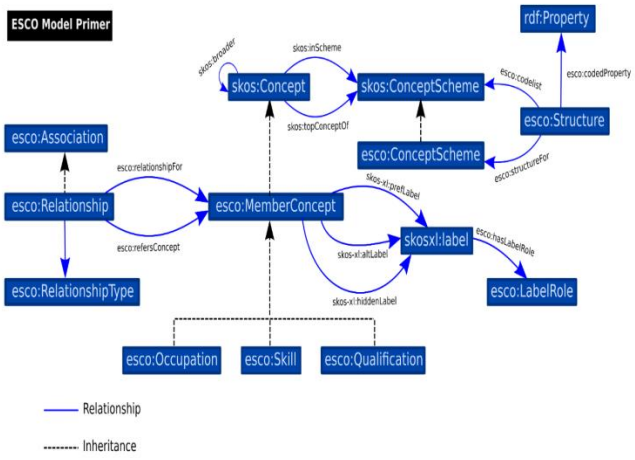

Fig 1. ESCO Ontology.

[2] https://ec.europa.eu/esco/resources/data/static/model/html/model.xhtml
[3] https://www.stardog.com/

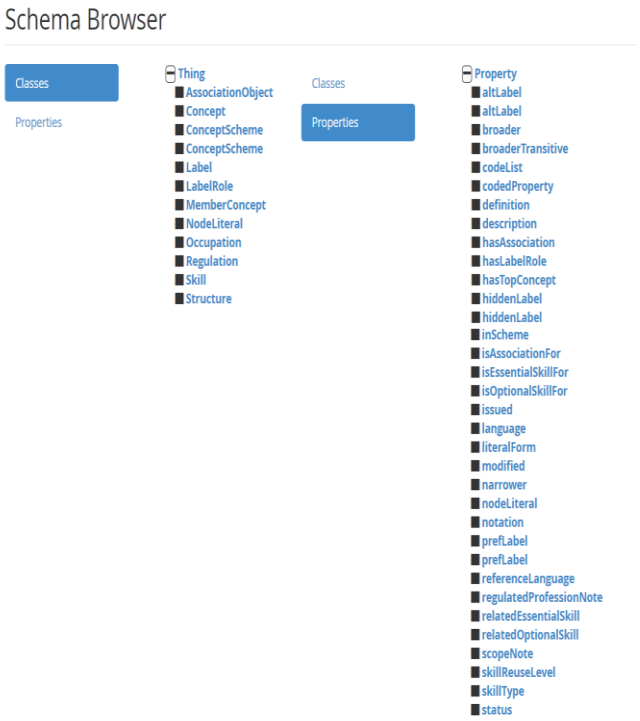

Fig 2. ESCO LOD Structure.

Binding between two resources to indicate that the first resource is sub concept of the other depends on the two properties of Simple Knowledge Organization System (SKOS) "broader and narrower". Meanwhile, one of these resources or both can be part of a classification. However, this way does not differentiate between a concept that represents a certain level of classification and the individuals contained in this level. To acquire all the skills connected with an occupation is a straightforward task:

```
SELECT DISTINCT ?skill
where{
?skill rdf:type <http://data.europa.eu/esco/model#Skill>.
?occupation rdf:type
<http://data.europa.eu/esco/model#Occupation>.
?occupation ?property ?skill.
FILTER(?property In
(<http://data.europa.eu/esco/model#relatedEssentialSkill>,
<http://data.europa.eu/esco/model#relatedOptionalSkill>))}
```

In contrast applying a SPARQL query to obtain all the skills which are not connected with an occupation is impossible. In other words, a query as the following one:

```
SELECT DISTINCT ?skill
WHERE{
?skill rdf:type <http://data.europa.eu/esco/model#Skill>.
?occupation rdf:type
<http://data.europa.eu/esco/model#Occupation>.
```

FILTER NOT EXISTS {?occupation ?property ?skill.}}

returns not only skills which are not connected with an occupation, but also all the resources which represent the hierarchical structure of the skill concept.

The benefit of using this metadata is that it facilitates the reuse [22] and supports reasoning in all profiles of Web Ontology Language (OWL). Moreover, since query answering is reduced to OWL-QL query answering, this allows queries to be run over large ontologies [23].

B. Label and Description

The label properties *altLabel, hiddenLabel* and *preflabel*, are used to provide a label to a resource. Each property has two namespaces: the first is SKOS, which links the resource to the literal object; the second one is the extension Simple Knowledge Organization (SKOS-XL), which links the resource with one or more resource type *SKOS-XL:Label* which in turn has a "*literalForm*" feature with the same role of SKOS's previous property. However, if the resource contains more than one resource from *SKOS-XL:Label*, each one belongs to label written in a specific language. Additionally, the definition and the description are properties that provide a description to a re-source where the definition property is used only 54 time concurrently with description property. Each resource is collected with one or more resource which in turn has property "*nodeLiteral*" containing a literal object that includes the description with a "*language*" property that indicates the language used to write the description. In case the resource is collected with one resource then the description is written in English. However, if the resource is collected by more than one resource, each one belongs to the description written in a specific language. Consequently, the dataset of ESCO include duplicate information. Therefore, data exploration becomes more difficult and a storage space increases.

C. The Relationship between Skill and Occupation

The relationship between skills and occupation has been built by only two predicates "*relatedEssentialSkill* and *relatedOptionalSkill*". At the same time, the skills in ESCO dataset are divided into two type "skill and knowledge" by a triple that has the skill as subject, skill type as predicate and the type of the skill as an object where each Skill belongs to only one type. The SPARQL query that returns the skills and the knowledge of an occupation, it is very complicated and is written in the following format:

```
SELECT ?essentialskill ?optionalskill ?essentialknowledge ?optionalknowledge

WHERE{

{?essentialskill rdf:type <http://data.europa.eu/esco/model#Skill>;

<http://data.europa.eu/esco/model#skilltype> <http://data.europa.eu/esco/skill-type/skill>.

?occupation <http://data.europa.eu/esco/model#relatedessentialskill> ?essentialskill.}

UNION{?essentialknowledge rdf:type <http://data.europa.eu/esco/model#Skill>;

<http://data.europa.eu/esco/model#skilltype> <http://data.europa.eu/esco/skill-type/knowledge>.

?occupation <http://data.europa.eu/esco/model#relatedessentialskill> ?essentialknowledge.}

UNION{?optionalskill rdf:type <http://data.europa.eu/esco/model#Skill>;

<http://data.europa.eu/esco/model#skilltype> <http://data.europa.eu/esco/skill-type/skill>.

?occupation <http://data.europa.eu/esco/model#relatedoptionalskill> ?optionalskill.}

UNION{?optionalknowledge rdf:type <http://data.europa.eu/esco/model#Skill>;

<http://data.europa.eu/esco/model#skilltype> <http://data.europa.eu/esco/skill-type/knowledge>.

?occupation <http://data.europa.eu/esco/model#relatedoptionalskill> ?optionalknowledge.}

FILTER(?occupation in (<http://data.europa.eu/esco/occupation/1b4e795d-6e49-4b7b-bb34-585edfd6eb18>))

}
```

This complexity in query formulation consequent to triples diversity causes slow execution of the SPARQL query [24]. The principal impediment a user faces when trying to apply a query is that he mostly has no information about the LOD underlying structure.

D. Skill and Occupation Structure

The structure of skill and occupation has been discovered within the linked open data of ESCO by applying some query and by using the information represented in class *esco:Structure*.

The occupation structure consists of six levels, the first four levels are based on International Standard Classification of Occupations (ISCO), and the last two levels can be considered as instances of the fourth level. The relation between each level is managed by some predicate like *skos:broader*, *skos:broaderTransitive* and *skos:narrower*. The resources of ESCO classification are generated from type of *skos:Concept*. However, the occupations resources are generated from type *skos:Concept, MemberConcept* and *Occupation*.

On the other hand, the skills structure has nothing to do with standard classification and not tied to a consistent classification where the classification branches have different lengths. The first two levels of the classification can be considered as classes and the rest of classification levels can be considered as instances. The relation between one level and

another is managed by some predicate like *skos:broader, skos:broaderTransitive* and *skos:narrower*.

All in all, this structure only complicates the data, making it difficult for the user to understand and manipulate.

E. The type of Concept, ConceptScheme and MemberConcept

OWL ontologies and LOD are increasing; thus, the need to give more accurate descriptions of their sources is becoming more necessary [25]. When a general type of class contains sources that only belong to this class or for other classes at the same time cause difficulty to discover their roles and their relations within the linked open data by the user; for example, each resource represents a skill is from *Skill, concept* and *MemberConcept* type; instead, each resource represents a skill reuse level is from Concept type only. SKOS classes can consider them as a representative that establishes an ‘‘indirection role’’ between lexical entities and ‘‘real-world’’ but not as a representative of the ‘‘real-world’’ [26].

IV. The Proposed Ontological Model to Reconstruct the ESCO LOD

Nowadays information and systems are growing more rapidly and becoming more complex. As a result, there has to be a method to generate the result of improving the information and the systems with shorter lead-times at less cost [27]. For the semantic data, this method is represented by the rules that define new concepts, relations and metadata which provide a real definition of each resource in the LOD [28] [29][30] All the rules included in appendix “first order logic rules”.

Fig. 3 represents the proposed ontology for ESCO. This model was built by implementing a set of rules written in first order logic. Each set of these rules has a specific task in building the model as follows

A. Classification Building

The model consists of two classifications: one represents the occupation and the other represents the skill. In terms of occupation, the structure is divided into two parts: the first part displays the hierarchical structure represented by rules from 1 to 8, and the second part shows individuals represented by rules from 9 to 16. In terms of skill, the structure is divided into two parts: the first part presents the hierarchical structure represented by rules from 17 to 24, and the second part presents individuals represented by rules from 25 to 44.

B. Give Entities to Different Resources in LOD

The proposed model encompasses classes that did not exist in the ESCO ontology to express the nature and the entity of some the sources that were under general classes. In fact, it can only be identified by relations. The rules between 45 and 54 represent the process of creating new classes and adding individuals to each one.

C. Create the Object Properties of Proposed Ontological Model

The proposed ontological model contains new object properties that represent the relations amongst the new classes. It also contains new relations that describe the relations amongst the existing classes in ESCO ontology in a more accurate manner. Rules 55 to 82 describe the process of establishing these object properties.

D. Stay away from Duplicate Data that Achieve the Same Goal

The article demonstrates that in ESCO LOD has been used the vocabulary of SKOS and the vocabulary of SKOS-XL as noted in the paragraph 3.2. The vocabulary of SKOS-XL is used when is needed to add more information to a label or a description [31]. Nonetheless, the ESCO LOD has not added any other metadata information for this reason, the vocabulary of SKOS-XL has been excluded and only used the vocabulary of SKOS.

V. Evaluation of the Proposed ESCO Ontology

The evaluation of the proposed ontology is based on three criteria:

- The ability to know the contents of the dataset and the mechanism of linking these contents through the ontological schema.

Through the ontological scheme we can understand the following issues: the individuals of class Skill have two different natures; consequently, it can be Skill or knowledge. To be able to perform an occupation, one needs to have some essential skills and knowledge and some optional skills and knowledge. Also to be able to have a skill or a knowledge, one needs to have some essential skills and knowledge and some optional skills and knowledge.

- Preventing information duplication and reducing dataset size.

The ESCO LOD uses two ways to add the labels to a resource as we see before, in spite of the pro-posed ontology use

The direct way to add the labels for a resource accordingly, it prevents the duplicate information and reduce the dataset size by more than three million and half triples.

- Easy retrieval of data through SPARQL queries.

The proposed ontology includes four object properties to connect an occupation or a skill with their essential or optional skills and knowledge. Consequently, it is easy to write a SPARQL query to know which skills or knowledge are essential and which ones are optional to perform an occupation or to obtain a new skill or knowledge.

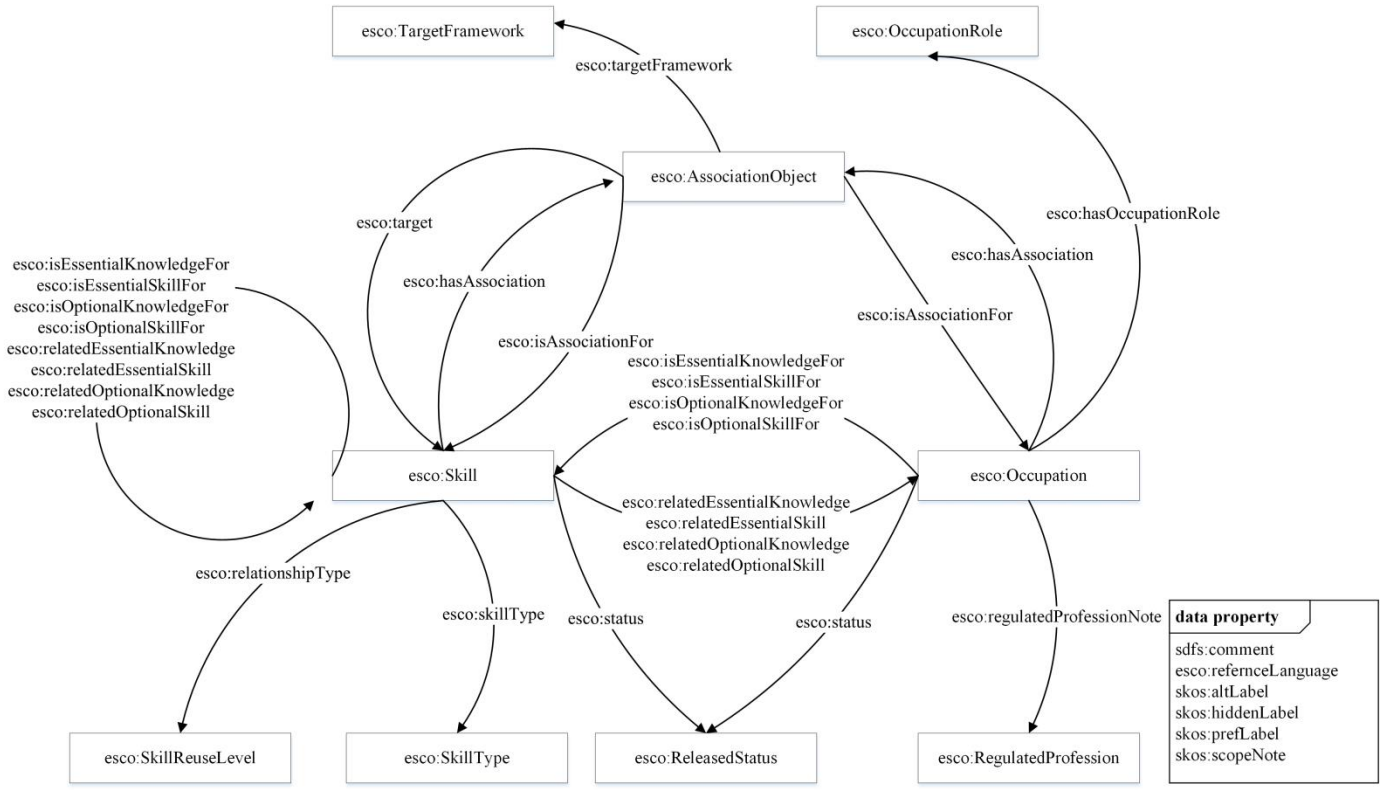

Fig 3. Proposed ESCO Ontology.

A. Getting all Skill which are Not Connected with an Occupation

Applying a SPARQL query to answer this ques-tion depending on a new ESCO ontology is as follows:

SELECT DISTINCT ?skill

WHERE{

?skill rdf:type <http://data.europa.eu/esco/model#Skill>, owl:namedindividual.

?occupation rdf:type <http://data.europa.eu/esco/model#Occupation>.

FILTER NOT EXISTS {?occupation ?property ?skill.}}

B. Acquiring All Skills and Knowledge of an Occupation

When we add other two properties to represent the relation between skills and occupations in the new ESCO ontology, the query will be more clear and more simple. For instance, get all the skills and knowledge for the occupation "footwear production machine operator" and "footwear designer"

SELECT ?essentialskill ?optionalskill ?essentialknowledge ?optionalknowledge

WHERE{

{?occupation <http://data.europa.eu/esco/model#relatedessentialskill> ?essentialskill.}

UNION{?occupation <http://data.europa.eu/esco/model#relatedessentialknowledge> ?essentialknowledge.}

UNION{?occupation <http://data.europa.eu/esco/model#relatedoptionalskill> ?optionalskill.}

UNION{?occupation <http://data.europa.eu/esco/model#relatedoptionalknowledge> ?optionalknowledge.}

FILTER(?occupation in (<http://data.europa.eu/esco/occupation/1b4e795d-6e49-4b7b-bb34-585edfd6eb18>,

<http://data.europa.eu/esco/occupation/06f89f2c-c6e9-40c5-a4a5-0e34d5fbc184>))

}

VI. Conclusion

ESCO is one of the most important European projects aimed at modeling the labor market. Its LOD is one of the most qualified LOD for reuse. Thus, it has to be clear and as easy to use as possible. In the proposed ontological model, this study relied on a set of conditions to maintain clarity, such as:

- Non-repetition data
- Using OWL and RDFS to build classifications and to identify each source and whether this source represents a class, individual, object property or data property.

- Determining the dependency of each source for a specific class illustrating the nature of this source.

One of the more significant findings to emerge from this study is that the proposed ontological model could be a pillar of a new version of the ESCO LOD in the coming years since the European Union will adopt this data at the level of all member states. The current study makes several noteworthy contributions to improve the outputs of studies that aim to use ESCO LOD as a tool for search and job matching, career management, and labor market analysis.

The methods used for this study to improve the data quality and data structure of ESCO LOD may be applied to other datasets published as LOD elsewhere in the world.

The following conclusions can be drawn from the present study. The ESCO LOD could be not only one of the most important sources of information for building job applications, but also a basis for a recommendation system for building an effective training system in all member states. This is expected to yield several benefits arising from the advantages of hierarchical structure for classifications of some classes within the data. Another benefit will result from the advantages of horizontal structure arising from relationships between the classes, as well as qualifications issued by private awarding bodies.

REFERENCES

[1] X. Niu, X. Sun, H. Wang, S. Rong, G. Qi, and Y. Yu, "Zhishi. me-weaving chinese linking open data," in International Semantic Web Conference, 2011, pp. 205–220.

[2] T. Di Noia, R. Mirizzi, V. C. Ostuni, D. Romito, and M. Zanker, "Linked open data to support content-based recommender systems," in Proceedings of the 8th international conference on semantic systems, 2012, pp. 1–8.

[3] B. Ell, D. Vrandečić, and E. Simperl, "Labels in the web of data," in International Semantic Web Conference, 2011, pp. 162–176.

[4] European Commission, Labour Market Developments in Europe. 2012.

[5] European Commission, "European Commission Digital Jobs launches Grand Coalition for," 2013.

[6] M. Le Vrang, A. Papantoniou, E. Pauwels, P. Fannes, D. Vandensteen, and J. De Smedt, "ESCO: Boosting job matching in Europe with semantic interoperability," Computer, vol. 47, no. 10, pp. 57–64, 2014.

[7] L. Cai and Y. Zhu, "The challenges of data quality and data quality assessment in the big data era," Data science journal, vol. 14, 2015.

[8] I. Taleb, H. T. El Kassabi, M. A. Serhani, R. Dssouli, and C. Bouhaddioui, "Big data quality: A quality dimensions evaluation," in 2016 Intl IEEE Conferences on Ubiquitous Intelligence & Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People, and Smart World Congress (UIC/ATC/ScalCom/CBDCom/IoP/SmartWorld), 2016, pp. 759–765.

[9] M. Färber, F. Bartscherer, C. Menne, and A. Rettinger, "Linked data quality of dbpedia, freebase, opencyc, wikidata, and yago," Semantic Web, vol. 9, no. 1, pp. 77–129, 2018.

[10] K. Jacksi, S. R. Zeebaree, and N. Dimililer, "LOD Explorer: Presenting the Web of Data," Intl. Journal of Advanced Computer Science and Applications, vol. 9, no. 1, pp. 45–51, 2018.

[11] S. Knight and J. M Burn, "Developing a Framework for Assessing Information Quality on the World Wide Web," Informing Science Journal, vol. 8, pp. 159–172, 2005.

[12] A. Zaveri, A. Rula, A. Maurino, R. Pietrobon, J. Lehmann, and S. Auer, "Quality assessment for linked data: A survey," Semantic Web, vol. 7, no. 1, pp. 63–93, 2016.

[13] H. H. Ahmed, "Data quality assessment in the integration process of linked open data (LOD)," in 2017 IEEE/ACS 14th International Conference on Computer Systems and Applications (AICCSA), 2017, pp. 1–6.

[14] L. L. Pipino, Y. W. Lee, and R. Y. Wang, "Data quality assessment," Communications of the ACM, vol. 45, no. 4, pp. 211–218, 2002.

[15] C. Batini and M. Scannapieco, Data and Information Quality: Dimensions, Principles and Techniques. Springer International Publishing, 2016.

[16] H. Paulheim and C. Bizer, "Improving the quality of linked data using statistical distributions," International Journal on Semantic Web and Information Systems (IJSWIS), vol. 10, no. 2, pp. 63–86, 2014.

[17] C. Guéret, P. Groth, C. Stadler, and J. Lehmann, "Assessing linked data mappings using network measures," in Extended semantic web conference, 2012, pp. 87–102.

[18] A. Flemming, "Quality characteristics of linked data publishing datasources," Master's thesis, Humboldt-Universität of Berlin, 2010.

[19] C. Bizer and R. Cyganiak, "Quality-driven information filtering using the WIQA policy framework," Journal of Web Semantics, vol. 7, no. 1, pp. 1–10, 2009.

[20] D. Kontokostas et al., "Test-driven evaluation of linked data quality," in Proceedings of the 23rd international conference on World Wide Web, 2014, pp. 747–758.

[21] M. C. Pattuelli, A. Provo, and H. Thorsen, "Ontology building for linked open data: A pragmatic perspective," Journal of Library Metadata, vol. 15, no. 3–4, pp. 265–294, 2015.

[22] P.-Y. Vandenbussche and B. Vatant, "Metadata recommendations for linked open data vocabularies," Version, vol. 1, pp. 2011–2012, 2011.

[23] E. Thomas, J. Z. Pan, and Y. Ren, "TrOWL: Tractable OWL 2 reasoning infrastructure," in Extended Semantic Web Conference, 2010, pp. 431–435.

[24] S. Campinas, T. E. Perry, D. Ceccarelli, R. Delbru, and G. Tummarello, "Introducing RDF graph summary with application to assisted SPARQL formulation," in 2012 23rd International Workshop on Database and Expert Systems Applications, 2012, pp. 261–266.

[25] A. Kalyanpur, B. Parsia, E. Sirin, and J. Hendler, "Debugging unsatisfiable classes in OWL ontologies," Web Semantics: Science, Services and Agents on the World Wide Web, vol. 3, no. 4, pp. 268–293, 2005.

[26] T. Baker, S. Bechhofer, A. Isaac, A. Miles, G. Schreiber, and E. Summers, "Key choices in the design of Simple Knowledge Organization System (SKOS)," Web Semantics: Science, Services and Agents on the World Wide Web, vol. 20, pp. 35–49, 2013.

[27] A. Abadi, H. Ben-Azza, and S. Sekkat, "Improving integrated product design using SWRL rules expression and ontology-based reasoning," Procedia Computer Science, vol. 127, pp. 416–425, 2018.

[28] E. Sirin, B. Parsia, B. C. Grau, A. Kalyanpur, and Y. Katz, "Pellet: A practical owl-dl reasoner," Web Semantics: science, services and agents on the World Wide Web, vol. 5, no. 2, pp. 51–53, 2007.

[29] N. Alaya, M. Lamolle, and S. Ben Yahia, "Multi-Label Based Learning for Better Multi-Criteria Ranking of Ontology Reasoners," in International Semantic Web Conference, 2017, pp. 3–19.

[30] Z. Quan and V. Haarslev, "A framework for parallelizing OWL classification in description logic reasoners," arXiv preprint arXiv:1906.07749, 2019.

[31] A. Isaac and E. Summers, "SKOS simple knowledge organization system," Primer, World Wide Web Consortium (W3C), vol. 44, 2009.

APPENDIX FIRST ORDER LOGIC RULES

A. Classification Building of Occupation

1) Class Hierarchy

Rule (1)

```
∀x,y,z (skos:narrower(x,y) ∧
(relatedEssentialSkill(y,z) ∨
relatedOptionalSkill(y,z)) ∧ ¬
(relatedEssentialSkill(x,z) ∧
relatedOptionalSkill(x,z)) → owl:Class(x)
)
```

Rule (2)

```
∀x,y,z,u (skos:narrower(x,y) ∧
(relatedEssentialSkill(y,z) ∨
relatedOptionalSkill(y,z)) ∧ ¬
(relatedEssentialSkill(x,z) ∧
relatedOptionalSkill(x,z)) ∧
skos:narrower(u,x)   → owl:Class(u) )
```

Rule (3)

```
∀x,y,z,u (skos:narrower(x,y) ∧
(relatedEssentialSkill(y,z) ∨
relatedOptionalSkill(y,z)) ∧ ¬
(relatedEssentialSkill(x,z) ∧
relatedOptionalSkill(x,z)) ∧
skos:narrower(u,x)   →
rdfs:subClassOf(u,x))
```

Rule (4)

```
∀x,y,z,u,s (skos:narrower(x,y) ∧
(relatedEssentialSkill(y,z) ∨
relatedOptionalSkill(y,z)) ∧ ¬
(relatedEssentialSkill(x,z) ∧
relatedOptionalSkill(x,z)) ∧
skos:narrower(u,x) ∧ skos:narrower(s,u)
→ owl:Class(s) )
```

Rule (5)

```
∀x,y,z,u,s (skos:narrower(x,y) ∧
(relatedEssentialSkill(y,z) ∨
relatedOptionalSkill(y,z)) ∧ ¬
(relatedEssentialSkill(x,z) ∧
relatedOptionalSkill(x,z)) ∧
skos:narrower(u,x) ∧ skos:narrower(s,u)
→ rdfs:subClassOf(s,u))
```

Rule (6)

```
∀x,y,z,u,s ,d (skos:narrower(x,y) ∧
(relatedEssentialSkill(y,z) ∨
relatedOptionalSkill(y,z)) ∧ ¬
(relatedEssentialSkill(x,z) ∧
relatedOptionalSkill(x,z)) ∧
skos:narrower(u,x) ∧ skos:narrower(s,u)
∧ skos:narrower(d,s)     →  owl:Class(d))
```

Rule (7)

```
∀x,y,z,u,s ,d (skos:narrower(x,y) ∧
(relatedEssentialSkill(y,z) ∨
relatedOptionalSkill(y,z)) ∧ ¬
(relatedEssentialSkill(x,z) ∧
relatedOptionalSkill(x,z)) ∧
skos:narrower(u,x) ∧ skos:narrower(s,u)
∧ skos:narrower(d,s)     →
rdfs:subClassOf(d,s))
```

Rule (8)

```
∀x,y,z,u,s ,d (owl:class(Occupation) ∧
skos:narrower(x,y) ∧
(relatedEssentialSkill(y,z) ∨
relatedOptionalSkill(y,z)) ∧ ¬
(relatedEssentialSkill(x,z) ∧
relatedOptionalSkill(x,z)) ∧
skos:narrower(u,x) ∧ skos:narrower(s,u)
∧ skos:narrower(d,s)     →
rdfs:subClassOf(Occupation, d))
```

2) adding individuals

Rule (9)

```
∀x,y,z (skos:narrower(x,y) ∧
(relatedEssentialSkill(y,z) ∨
relatedOptionalSkill(y,z)) ∧ ¬
(relatedEssentialSkill(x,z) ∧
relatedOptionalSkill(x,z)) →
owl:NamedIndividual(y) )
```

Rule (10)

```
∀x,y,z (skos:narrower(x,y) ∧
(relatedEssentialSkill(y,z) ∨
relatedOptionalSkill(y,z)) ∧ ¬
(relatedEssentialSkill(x,z) ∧
relatedOptionalSkill(x,z)) →
rdf:type(x,y))
```

Rule (11)

```
∀x,y,z,u (skos:narrower(x,y) ∧
(relatedEssentialSkill(y,z) ∨
relatedOptionalSkill(y,z)) ∧ ¬
(relatedEssentialSkill(x,z) ∧
relatedOptionalSkill(x,z)) ∧
skos:narrower(y,u)  →
owl:NamedIndividual(u) )
```

Rule (12)

```
∀x,y,z,u (skos:narrower(x,y) ∧
(relatedEssentialSkill(y,z) ∨
relatedOptionalSkill(y,z)) ∧ ¬
(relatedEssentialSkill(x,z) ∧
relatedOptionalSkill(x,z)) ∧
skos:narrower(y,u)  → rdf:type(x,u))
```

Rule (13)

```
∀x,y,z,u,d (skos:narrower(x,y) ∧
(relatedEssentialSkill(y,z) ∨
relatedOptionalSkill(y,z)) ∧ ¬
(relatedEssentialSkill(x,z) ∧
relatedOptionalSkill(x,z)) ∧
```

skos:narrower(y,u) ∧ skos:narrower(u,d) → owl:NamedIndividual(d)

Rule (14)

∀x,y,z,u (skos:narrower(x,y) ∧ (relatedEssentialSkill(y,z) ∨ relatedOptionalSkill(y,z)) ∧ ¬ (relatedEssentialSkill(x,z) ∧ relatedOptionalSkill(x,z)) ∧ skos:narrower(y,u) ∧ skos:narrower(u,d) → rdf:type(x,d))

Rule (15)

∀x,y,z,u,d,f (skos:narrower(x,y) ∧ (relatedEssentialSkill(y,z) ∨ relatedOptionalSkill(y,z)) ∧ ¬ (relatedEssentialSkill(x,z) ∧ relatedOptionalSkill(x,z)) ∧ skos:narrower(y,u) ∧ skos:narrower(u,d) ∧ skos:narrower(d,f) → owl:NamedIndividual(f)

Rule (16)

∀x,y,z,u (skos:narrower(x,y) ∧ (relatedEssentialSkill(y,z) ∨ relatedOptionalSkill(y,z)) ∧ ¬ (relatedEssentialSkill(x,z) ∧ relatedOptionalSkill(x,z)) ∧ skos:narrower(y,u) ∧ skos:narrower(u,d) ∧ skos:narrower(d,f) → rdf:type(x,f))

B. Classification Building of Skill

1) Class Hierarchy

Rule (17)

∀x (x = <http://data.europa.eu/esco/skill/8f18f987-33e2-4228-9efb-65de25d03330> → owl:Class(x))

Rule (18)

∀x (x = <http://data.europa.eu/esco/skill/8f18f987-33e2-4228-9efb-65de25d03330> → rdfs:subClassOf(Skill, x))

Rule (19)

∀x,y (x = <http://data.europa.eu/esco/skill/8f18f987-33e2-4228-9efb-65de25d03330> ∧ skos:narrower(x,y) → owl:Class(y))

Rule (20)

∀x,y (x = <http://data.europa.eu/esco/skill/8f18f987-33e2-4228-9efb-65de25d03330> ∧ skos:narrower(x,y) → rdfs:subClassOf(x,y))

Rule (21)

∀x,y,z (x = <http://data.europa.eu/esco/skill/8f18f987-33e2-4228-9efb-65de25d03330> ∧ skos:narrower(x,y) ∧ skos:narrower(y,z)∧¬Skill(z)→ owl:Class(z))

Rule (22)

∀x,y,z (x = <http://data.europa.eu/esco/skill/8f18f987-33e2-4228-9efb-65de25d03330> ∧ skos:narrower(x,y) ∧¬Skill(z)→ rdfs:subClassOf(y,z))

Rule (23)

∀x,y,z,a (x = <http://data.europa.eu/esco/skill/8f18f987-33e2-4228-9efb-65de25d03330> ∧ skos:narrower(x,y) ∧ skos:narrower(y,z) ∧ skos:narrower(z,a) ∧¬Skill(z) ∧¬Skill(a)→ owl:Class(a))

Rule (24)

∀x,y,z,a (x = <http://data.europa.eu/esco/skill/8f18f987-33e2-4228-9efb-65de25d03330> ∧ skos:narrower(x,y) ∧ skos:narrower(y,z) ∧ skos:narrower(z,a) ∧¬Skill(z) ∧¬Skill(a)→ rdfs:subClassOf(z,a))

2) adding individuals

Rule (25)

∀x,y,z (x = <http://data.europa.eu/esco/skill/8f18f987-33e2-4228-9efb-65de25d03330> ∧ skos:narrower(x,y) ∧ skos:narrower(y,z) ∧ Skill(z)→ owl:NamedIndividual(z))

Rule (26)

∀x,y,z (x = <http://data.europa.eu/esco/skill/8f18f987-33e2-4228-9efb-65de25d03330> ∧ skos:narrower(x,y) ∧ Skill(z)→ rdf:type(y,z))

Rule (27)

∀x,y (skos:narrower(x,y) ∧ Concept(x) ∧ Skill(y) ∧¬Skill(x) → owl:NamedIndividual(y))

Rule (28)

∀x,y (skos:narrower(x,y) ∧ Concept(x) ∧ Skill(y) ∧¬Skill(x) → rdf:type(x,y))

Rule (29)

∀x,y,z (skos:narrower(x,y) ∧ skos:narrower(y,z) ∧ Concept(x) ∧ Skill(y) ∧ Skill(z) ∧¬Skill(x) → owl:NamedIndividual(z))

Rule (30)

∀x,y,z (skos:narrower(x,y) ∧ skos:narrower(y,z) ∧ Concept(x) ∧ Skill(y) ∧ Skill(z) ∧¬Skill(x) → rdf:type(x,z))

Rule (31)

∀x,y,z,s (skos:narrower(x,y) ∧ skos:narrower(y,z) ∧ skos:narrower(z,s) ∧ Concept(x) ∧ Skill(y) ∧ Skill(z) ∧ Skill(s) ∧¬Skill(x) → owl:NamedIndividual(s))

Rule (32)

∀x,y,z,s (skos:narrower(x,y) ∧ skos:narrower(y,z) ∧ skos:narrower(z,s) ∧ Concept(x) ∧ Skill(y) ∧ Skill(z) ∧ Skill(s) ∧¬Skill(x) → rdf:type(x,s))

Rule (33)

∀x,y,z,s,a (skos:narrower(x,y) ∧ skos:narrower(y,z) ∧ skos:narrower(z,s) ∧ skos:narrower(s,a) ∧ Concept(x) ∧ Skill(y) ∧ Skill(z) ∧ Skill(s) ∧ Skill(a) ∧¬Skill(x) → owl:NamedIndividual(a))

Rule (34)

∀x,y,z,s,a (skos:narrower(x,y) ∧ skos:narrower(y,z) ∧ skos:narrower(z,s) ∧ skos:narrower(s,a) ∧ Concept(x) ∧ Skill(y) ∧ Skill(z) ∧ Skill(s) ∧ Skill(a) ∧¬Skill(x) → rdf:type(x,a))

Rule (35)

∀x,y,z,s,a,b (skos:narrower(x,y) ∧ skos:narrower(y,z) ∧ skos:narrower(z,s) ∧ skos:narrower(s,a) ∧ skos:narrower(a,b) ∧ Concept(x) ∧ Skill(y) ∧ Skill(z) ∧ Skill(s) ∧ Skill(a) ∧ Skill(b) ∧¬Skill(x) → owl:NamedIndividual(b))

Rule (36)

∀x,y,z,s,a,b (skos:narrower(x,y) ∧ skos:narrower(y,z) ∧ skos:narrower(z,s) ∧ skos:narrower(s,a) ∧ skos:narrower(a,b) ∧ Concept(x) ∧ Skill(y) ∧ Skill(z) ∧ Skill(s) ∧ Skill(a) ∧ Skill(b) ∧¬Skill(x) → rdf:type(x,b))

Rule (37)

∀x,y,z,s,a,b,c (skos:narrower(x,y) ∧ skos:narrower(y,z) ∧ skos:narrower(z,s) ∧ skos:narrower(s,a) ∧ skos:narrower(a,b) ∧ skos:narrower(b,c) ∧ Concept(x) ∧ Skill(y) ∧ Skill(z) ∧ Skill(s) ∧ Skill(a) ∧ Skill(b) ∧ Skill(c) ∧¬Skill(x) → owl:NamedIndividual(c))

Rule (38)

∀x,y,z,s,a,b,c (skos:narrower(x,y) ∧ skos:narrower(y,z) ∧ skos:narrower(z,s) ∧ skos:narrower(s,a) ∧ skos:narrower(a,b) ∧ skos:narrower(b,c) ∧ Concept(x) ∧ Skill(y) ∧ Skill(z) ∧ Skill(s) ∧ Skill(a) ∧ Skill(b) ∧ Skill(c) ∧¬Skill(x) → rdf:type(x,c))

Rule (39)

∀x,y,z,s,a,b,c,d (skos:narrower(x,y) ∧ skos:narrower(y,z) ∧ skos:narrower(z,s) ∧ skos:narrower(s,a) ∧ skos:narrower(a,b) ∧ skos:narrower(b,c) ∧ skos:narrower(c, d) ∧ Concept(x) ∧ Skill(y) ∧ Skill(z) ∧ Skill(s) ∧ Skill(a) ∧ Skill(b) ∧ Skill(c) ∧ Skill(d) ∧¬Skill(x) → owl:NamedIndividual(d))

Rule (40)

∀x,y,z,s,a,b,c,d (skos:narrower(x,y) ∧ skos:narrower(y,z) ∧ skos:narrower(z,s) ∧ skos:narrower(s,a) ∧ skos:narrower(a,b) ∧ skos:narrower(b,c) ∧ skos:narrower(c, d) ∧ Concept(x) ∧ Skill(y) ∧ Skill(z) ∧ Skill(s) ∧ Skill(a) ∧ Skill(b) ∧ Skill(c) ∧ Skill(d) ∧¬Skill(x) → rdf:type(x,d))

3) Individuals unclassified

Rule (41)

∀x,y,z (Concept(x) ∧ Skill(y) ∧ Skill(z) ∧¬Skill(x) ∧ ¬ skos:broader(y,x) ∧ ¬ skos:narrower(y,z) → owl:NamedIndividual(y))

Rule (42)

∀x,y,z (Concept(x) ∧ Skill(y) ∧ Skill(z) ∧¬Skill(x) ∧ ¬ skos:broader(y,x) ∧ ¬ skos:narrower(y,z) → rdf:type(unclassified,y))

Rule (43)

∀x,y,z,s (Concept(x) ∧ Skill(y) ∧ Skill(z) ∧¬Skill(x) ∧ ¬ skos:broader(y,x) ∧ skos:narrower(y,z) ∧ skos:broader(z,y) ∧ ¬ skos:broader(z,s) → owl:NamedIndividual(y))

Rule (44)

∀x,y,z,s (Concept(x) ∧ Skill(y) ∧ Skill(z) ∧¬Skill(x) ∧ ¬ skos:broader(y,x) ∧ skos:narrower(y,z) ∧ skos:broader(z,y) ∧ ¬ skos:broader(z,s) → rdf:type(unclassified,z))

C. Class SkillReuseLevel

Rule (45)

∀x,y (Skill(x) ∧ Concept(y) ∧ skillReuseLevel(x,y) → owl:NamedIndividual(y))

Rule (46)

```
∀x,y (Skill(x) ∧ Concept(y) ∧
skillReuseLevel(x,y) →
rdf:type(SkillReuseLevel ,y))
```

D. Class SkillType

Rule (47)

```
∀x,y (Skill(x) ∧ Concept(y) ∧
skillType(x,y) → owl:NamedIndividual(y))
```

Rule (46)

```
∀x,y (Skill(x) ∧ Concept(y) ∧
skillType(x,y) → rdf:type(SkillType ,y))
```

E. Class ReleasedStatus

Rule (47)

```
∀x,y ((Skill(x) ∨ Occupation(x)) ∧
purl:status(x,y) →
owl:NamedIndividual(y))
```

Rule (48)

```
∀x,y ((Skill(x) ∨ Occupation(x)) ∧
purl:status(x,y) →
rdf:type(ReleasedStatus ,y))
```

F. Class RegulatedProfession

Rule (49)

```
∀x,y ((Concept(y) ∧ Regulation(y)) ∧
Occupation(x) ∧
regulatedProfessionNote(x,y) →
owl:NamedIndividual(y))
```

Rule (50)

```
∀x,y ((Concept(y) ∧ Regulation(y)) ∧
Occupation(x) ∧
regulatedProfessionNote(x,y) →
rdf:type(RegulatedProfession ,y))
```

G. Class OccupationRole

Rule (51)

```
∀x,y,z (Occupation(x) ∧ LabelRole(y) ∧
Label(z) ∧ altLabel(x,z) ∧
hasLabelRole(z,y) →
owl:NamedIndividual(y))
```

Rule (52)

```
∀x,y,z (Occupation(x) ∧ LabelRole(y) ∧
Label(z) ∧ altLabel(x,z) ∧
hasLabelRole(z,y) →
rdf:type(OccupationRole ,y))
```

H. Class AssociationObject

Rule (53)

```
∀x (AssociationObject (x) →
owl:NamedIndividual(x))
```

Rule (52)

```
∀x (AssociationObject (x)
→rdf:type(AssociationObject ,x))
```

I. Class TargetFramework

Rule (53)

```
∀x,y (AssociationObject(x) ∧
ConceptScheme(y) ∧ targetFramework(x,y) →
owl:NamedIndividual(y))
```

Rule (54)

```
∀x,y (AssociationObject(x) ∧
ConceptScheme(y) ∧ targetFramework(x,y) →
rdf:type(TargetFramework ,y))
```

J. Object property

1) Relatedessentialskill and isEssentialSkillFor

Rule (55)

```
∀x,y (Skill(x) ∧ Occupation(y) ∧
relatedEssentialSkill(y,x) ∧
skillType(x,skill) →
esco:relatedEssentialSkill(y,x))
```

Rule (56)

```
∀x,y (Skill(x) ∧ Occupation(y) ∧
relatedEssentialSkill(y,x) ∧
skillType(x,skill) →
esco:isEssentialSkillFor(x,y))
```

Rule (57)

```
∀x,y (Skill(x) ∧ Skill(y) ∧
relatedEssentialSkill(y,x) ∧
skillType(x,skill) →
esco:relatedEssentialSkill(y,x))
```

Rule (58)

```
∀x,y (Skill(x) ∧ Skill(y) ∧
relatedEssentialSkill(y,x) ∧
skillType(x,skill) →
esco:isEssentialSkillFor(x,y))
```

2) Related essential knowledge and Isessentialknowledgefor

Rule (59)

```
∀x,y (Skill(x) ∧ Occupation(y) ∧
relatedEssentialSkill(y,x) ∧
skillType(x,knowledge) →
esco:relatedEssentialKnowledge (y,x))
```

Roule (60)

```
∀x,y (Skill(x) ∧ Occupation(y) ∧
relatedEssentialSkill(y,x) ∧
skillType(x,knowledge) →
esco:isEssentialKnowledgeFor(x,y))
```

Rule (61)

```
∀x,y (Skill(x) ∧ Skill(y) ∧
relatedEssentialSkill(y,x) ∧
skillType(x,knowledge) →
esco:relatedEssentialKnowledge (y,x))
```

Roule (62)

```
∀x,y (Skill(x) ∧ Skill(y) ∧
relatedEssentialSkill(y,x) ∧
```

```
skillType(x,knowledge) →
esco:isEssentialKnowledgeFor(x,y))
```

3) relatedOptionalSkill and isOptionalSkillFor

Rule (63)

```
∀x,y (Skill(x) ∧ Occupation(y) ∧
relatedOptionalSkill(y,x) ∧
skillType(x,skill) →
esco:relatedOptionalSkill(y,x))
```

Roule (64)

```
∀x,y (Skill(x) ∧ Occupation(y) ∧
relatedOptionalSkill(y,x) ∧
skillType(x,skill) →
esco:isOptionalSkillFor(x,y))
```

Rule (65)

```
∀x,y (Skill(x) ∧ Skill(y) ∧
relatedOptionalSkill(y,x) ∧
skillType(x,skill) →
esco:relatedOptionalSkill(y,x))
```

Rule (66)

```
∀x,y (Skill(x) ∧ Skill(y) ∧
relatedOptionalSkill(y,x) ∧
skillType(x,skill) →
esco:isOptionalSkillFor(x,y))
```

4) relatedOptionalKnowledge and isOptionalKnowledgeFor

Rule (67)

```
∀x,y (Skill(x) ∧ Occupation(y) ∧
relatedOptionalSkill(y,x) ∧
skillType(x,knowledge) →
esco:relatedOptionalKnowledge (y,x))
```

Rule (68)

```
∀x,y (Skill(x) ∧ Occupation(y) ∧
relatedOptionalSkill(y,x) ∧
skillType(x,knowledge) →
esco:isOptionalKnowledgeFor(x,y))
```

Rule (69)

```
∀x,y (Skill(x) ∧ Skill(y) ∧
relatedOptionalSkill(y,x) ∧
skillType(x,knowledge) →
esco:relatedOptionalKnowledge (y,x))
```

Rule (70)

```
∀x,y (Skill(x) ∧ Skill(y) ∧
relatedOptionalSkill(y,x) ∧
skillType(x,knowledge) →
esco:isOptionalKnowledgeFor(x,y))
```

5) status

Rule (71)

```
∀x,y (Skill(x) ∧ ReleasedStatus(y) ∧
purl:status(x, released) →
esco:status(x,y))
```

Rule (72)

```
∀x,y (Occupation(x) ∧ ReleasedStatus(y) ∧
purl:status(x, released) →
esco:status(x,y))
```

6) regulatedProfessionNote

Rule (73)

```
∀x,y ((Concept(y) ∧ Regulation(y)) ∧
Occupation(x) ∧
regulatedProfessionNote(x,y) →
esco:regulatedProfessionNote(x,y)
```

7) skillType

Rule (74)

```
∀x,y (Skill(x) ∧ Concept(y) ∧
skillType(x,y) → esco:skillType(x,y))
```

8) relationshipType

Rule (75)

```
∀x,y (Skill(x) ∧ Concept(y) ∧
skillReuseLevel(x,y) →
esco:relationshipType(x,y))
```

9) hasOccupationRole

Rule (76)

```
∀x,y,z(Occupation(x) ∧ LabelRole(y) ∧
Label(z) ∧ altLabel(x,z) ∧
hasLabelRole(z,y) →
esco:hasOccupationRole(x ,y))
```

10)targetFramework

Rule (77)

```
∀x,y (AssociationObject(x) ∧
ConceptScheme(y) ∧ targetFramework(x,y) →
esco:targetFramework(x ,y))
```

11)hasAssociation and isAssociationFor

Rule (78)

```
∀x,y (Occupation(x) ∧ AssociationObject
(y) ∧  hasAssociation (x,y) →
esco:hasAssociation(x ,y))
```

Rule (79)

```
∀x,y (Occupation(x) ∧ AssociationObject
(y) ∧  has Association (x,y) →
esco:isAssociationFor(y,x))
```

Rule (80)

```
∀x,y (Skill(x) ∧ AssociationObject (y) ∧
hasAssociation (x,y) →
esco:hasAssociation(x ,y))
```

Rule (81)

```
∀x,y (skill(x) ∧ AssociationObject (y) ∧
has Association (x,y) →
esco:isAssociationFor(y,x))
```

12)Target

Rule (82)

```
∀x,y (skill(x) ∧ AssociationObject (y) ∧
target(y,x) → esco:target(y,x))
```