Assignment 2 - RDF and Sparql

Zefan Liang

School of Electrical Engineering and Computer Science

University of Ottawa

27th January, 2019

CSI 5180 Topics in AI - Ontologies and Semantic Web

Caroline Barrière

Table of content

1 Introduction..... 3

2 Domain Definition and Competency Questions.....3

3 Define the terms important in your domain 5

4 Define the classes and subclasses..... 5

5 Define the properties.....8

6 Define the property restrictions..... 9

7 Define some instances..... 9

8 Explore many of OWL possibilities..... 10

9 References.....23

1 Introduction

This report contains the whole content of the requirements.

My chosen ontology topic is the ontology of Borussia Dortmund. Borussia Dortmund is a German sports club based in Dortmund, North Rhine-Westphalia. The reason why I choose it as my topic is because Borussia Dortmund is my favorite soccer club. I am a fans for this soccer club since I was a middle school student. I think I am very familiar with the players, honors, team formation and the culture of this club, so I can build a better ontology with my knowledge. The ontology will be designed by the ontology editor Protege and the design process will follow the ontology design 101 approach.

2 Domain Definition and Competency Questions

The first step is to think of 10 questions it should be able to answer.

So the ten questions and answers are as follow:

(1) .Q: What is the domain that the ontology will cover?

A: Representation of the soccer club is the domain of the ontology.

(2) .Q:For what we are going to use the ontology?

A:I want to build up the ontology of the soccer club to let people know the consistence of a soccer club and its honor history. By comparing the different abilities of player, choosing different line-ups in different situations.

(3). Q:For what types of questions the information in the ontology should provide answers?

A:First of all, the ontology will describe the consistence of the soccer club, Borussia

Dortmund. What's more, the ontology will make the soccer players in different positions(such as midfielder, forward) to make up different football lineups and assess the overall ability of the team. In my opinion, it is a great idea to make the ontology related to soccer.

(4). Q:How can I get the abilities of different soccer players?

A:The capability value of soccer players will refer to fifa2019, the most famous soccer games in the world. The data is very accurate after evaluation by fifa company. Every soccer player has 4 abilities, including attack ability, defence ability, power ability, speed ability. Given A,B,C,D levels according to different ability values.

(5).Q:What should we choose for main lineups when the match opponent has a strong attack ability?

A:If our opponent has a strong attack ability, we can choose defensive counterattack tactics. At this time, we should choose the defenders with high defence and power abilities while we also need to choose some midfielders and forwards with high speed abilities.

(6).Q:What main classes do we use to describe the soccer club?

A:The advanced class in the ontology includes honors, people, sponsorship, stadium and team formation. The subclass of people include CEO, coaches, player and sport manager. The honors include the champion and runner-up in different type of league and cup.

(7).Q:What properties does every player have?

A:Every player has attack, defence, speed and power ability. They also have the other 3 properties, height, player number and nationality.

(8) .Q:What team lineups does the ontology have for instance?

A:The ontology will define some different team lineups with different soccer players. The lineups refer to Borussia Dortmund's past matches.

(9) .Q:How many classes are appropriate to define this ontology ?

A:Because I want to reflect many aspects of a team, so I want to define over 40 classes (including classes and subclasses).

(10).Q:What are the main individuals in this ontology ?

A:The main individuals in this ontology should be the people of this soccer club. Because the core part is built around people.

3 Define the terms important in your domain.

Before starting to design the ontology, some nouns and verbs can be very important to determine the framework.

The terms are as follow:

(1) .The soccer club holds soccer matches.

(2) .The soccer club has its own soccer stadium.

(3) .The soccer club has many people, including CEO, coaches, player and sports manager.

(4) .The soccer club has only one CEO.

(5) .The soccer club has only one sport manager.

(6) .There are different types of coaches in the team, including athletik-trainer, chef-coach, co-trainer, fitness coach.

(7) . For soccer players, there are 4 positions containing defender, forward, goalkeeper, midfielder.

(8) . For every soccer player, they must have 4 properties to show his ability, including attack, defence, speed and power ability.

(9) .By reference to the data provided by fifa company, their abilities can be divided into 4 levels, including A, B, C and D.

(10). If the ability value is higher than 83, we define their ability A.

(11). If the ability value is between 75 and 83, we define their ability B.

(12). If the ability value is between 65 and 75, we define their ability C.

(13). If the ability value is lower than 65, we define their ability D.

(14). The height of soccer player is also every important.

(15). Every soccer player in the club has one number.

(16). For every soccer match, one team just needs one goalkeeper.

(17). For the famous soccer club, there must be some rental players.

(18). The soccer player comes from different countries.

(19). For the famous Germany soccer club, it must have many honors in different kinds of league and cup.

(20). The famous league match includes Bundesliga.

(21). The famous cup match includes German Cup and UEFA Champions League.

(22). The Bundesliga season, German Cup season and UEFA Champions League season can be expressed by different years.

(23). The honors which club gain can be divided into champion and runner-up.

(24). The kit manufacturer, shirt sponsor and sleeve sponsor are sponsored by sponsors.

(25). A soccer team has different sponsors.

(26). The soccer team has different team formations.

(27). Different team formation has different properties. For example, 4-3-3 formation focus on attack because there are some forward in the team.

(28). Every team formation is made up by different soccer players.

(29). Some soccer players won some club honors before.

(30). One team needs to have one team leader.

4 Define the classes and subclasses.

Refer to the above terms, it is important to define different classes and subclasses.

I think the main classes in this ontology are Honors, People, Sponsorship, Stadium and Team formation.

Because the honors include different cup and league, so the subclasses of Honors are Bundesliga, German Cup and UEFA Champions League. Different cup match seasons or league match seasons must be the subclasses of Bundesliga, German Cup and UEFA Champions League. Meanwhile, Ranking should be defined to be the subclass of Honors, including championship and runner-up.

CEO, Coaches, Player and Sports manager must be the subclasses of People. Coaches contains subclass Coach type while Player should have subclass Position. The subclasses of Position are Forward, Midfielder, Defender and Goalkeeper. Nationality can also be the subclass of People, because just the people have different nationalities.

The subclasses of Sponsorship include Sponsor Type and Sponsors. The Sponsor Type includes Kit manufacturer, Shirt sponsor and Sleeve sponsor.

There are no subclasses of Stadium, because I'd like to define the stadium as the individual.

The subclasses of team formation include different team formations.

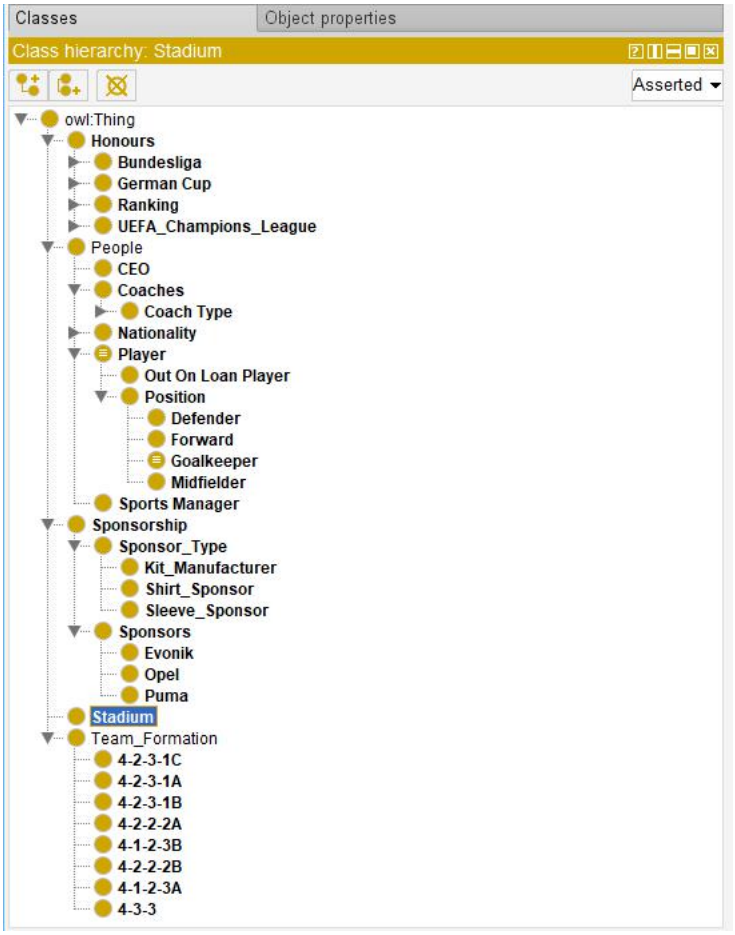

Figure 1 - The classes and subclasses of the ontology.

5 Define the properties.

For the object properties, I'd like to define many of them.

Because the soccer players have their own stadium, so I define hasHomeField here. For the main class People, every player has their own position, so I define hasPosition.Objected property isPositionof is inverse of hasPosition. Every soccer player has 4 abilities, so there are 4 object properties, containing hasAbilityofAttack, hasAbilityofDefence, hasAbilityofPower,

hasAbilityofSpeed. For the coaches, there are different coach types, so I define hasCoachType whose inverse property is isCoachTypeof. The players have their teammates, so I define hasTeammates here. Some players won the honors before, so I define Win here.What's more, different sponsors have different sponsor type, so I define hasSponsorType.

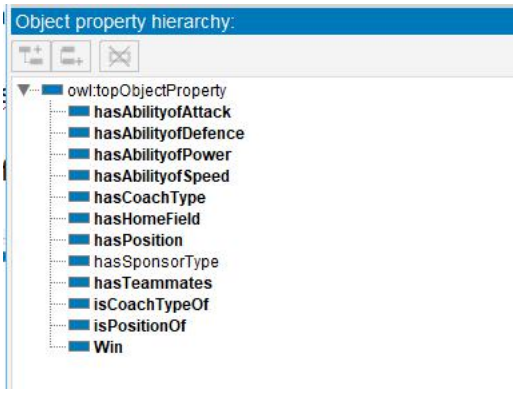

Figure 2 - The object properties of the ontology.

For the data properties, I define No. And height. They are numbers and height of players.

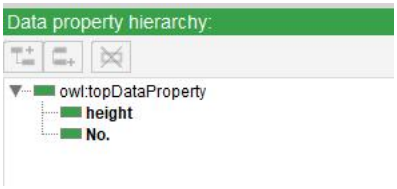

Figure 3 - The data properties of the ontology.

6 Define the property restrictions.

For object properties, hasTeammates is Transitive property and Symmetric property.

For data property, No. is Functional property.

7 Define some instances.

I define the whole team members name in Individuals. Because I think people should be individuals. And I also define A,B,C,D four levels here as well as the stadium name.

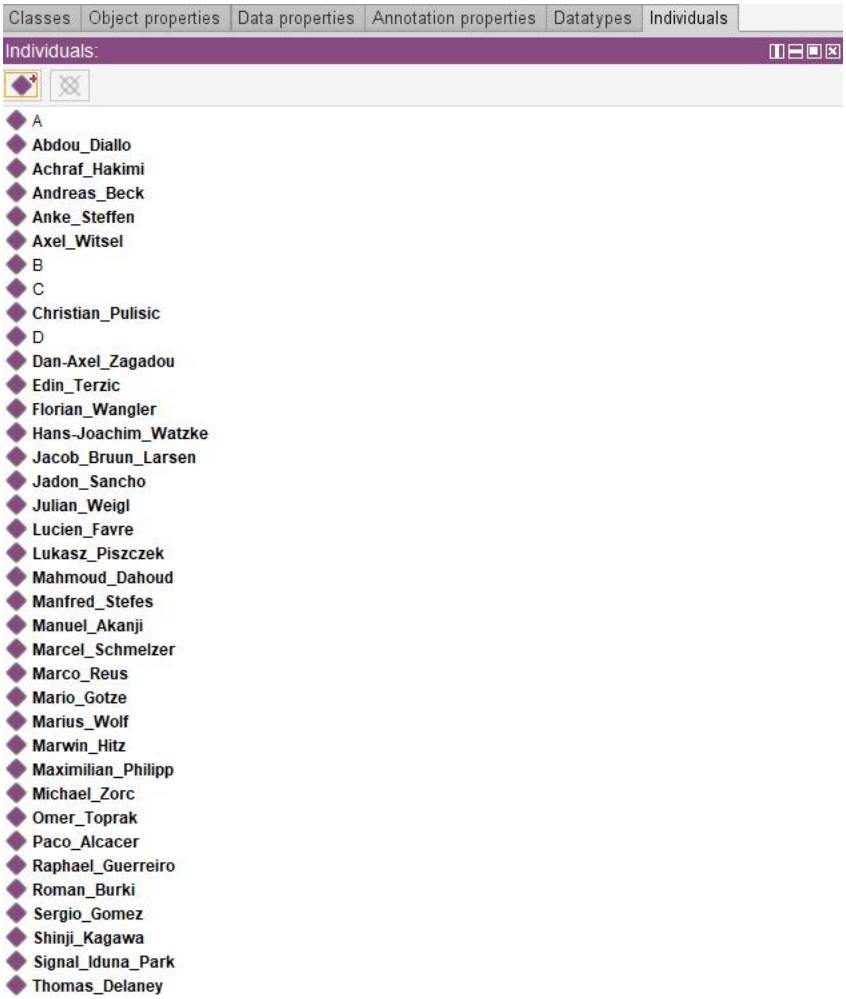

Figure 4 - The individuals of the ontology.

8 Explore many of OWL possibilities.

(1). Class / Subclass (rdfs:subClassOf)

CEO is the subclass of People.

Figure 5 - The example of Class / Subclass

The turtle format is as follow:

<http://www.semanticweb.org/pc/ontologies/2019/1/CEO>

rdf:type owl:Class ;

rdfs:subClassOf <http://www.semanticweb.org/pc/ontologies/2019/1/People> ;

(2). Domain / Range (rdfs:domain, rdfs:range)

The domain of the objected property hasCoachType is Coaches and the range of hasCoachType is Coach Type.

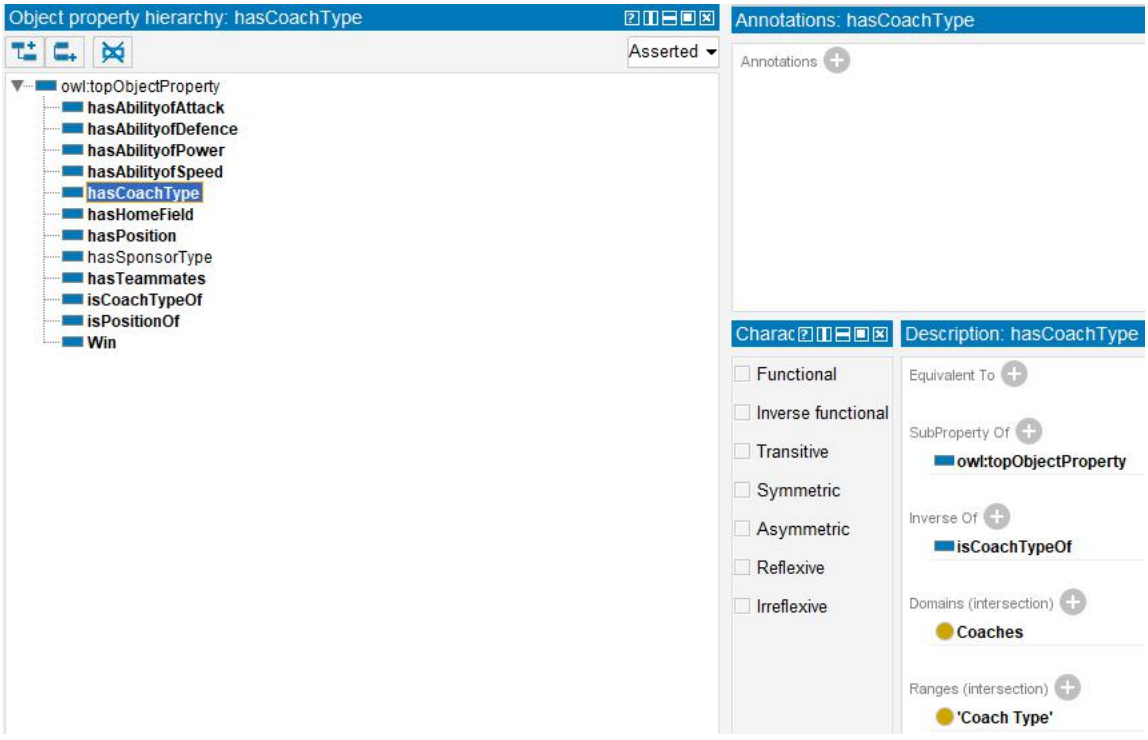

Figure 6 - The example of Domain / Range

The turtle format is as follow:

<http://www.semanticweb.org/pc/ontologies/2019/1/hasCoachType>

rdf:type owl:ObjectProperty ;

rdfs:subPropertyOf owl:topObjectProperty ;

owl:inverseOf <http://www.semanticweb.org/pc/ontologies/2019/1/isCoachTypeOf> ;

rdfs:domain <http://www.semanticweb.org/pc/ontologies/2019/1/Coaches> ;

rdfs:range <http://www.semanticweb.org/pc/ontologies/2019/1/CoachType> .

(3). Object property linking two classes (owl:ObjectProperty)

Defender has teammates Forward and Goalkeeper and Midfielder. So the object property links two classes together.

Figure 7 - The example of object property linking two classes.

The turtle format is as follow:

<http://www.semanticweb.org/pc/ontologies/2019/1/hasTeammates>

rdf:type owl:ObjectProperty ,owl:SymmetricProperty ,owl:TransitiveProperty .

(4). Data property linking a class to a literal (owl:DatatypeProperty)

For example, the height of soccer player Marco Reus is 180.

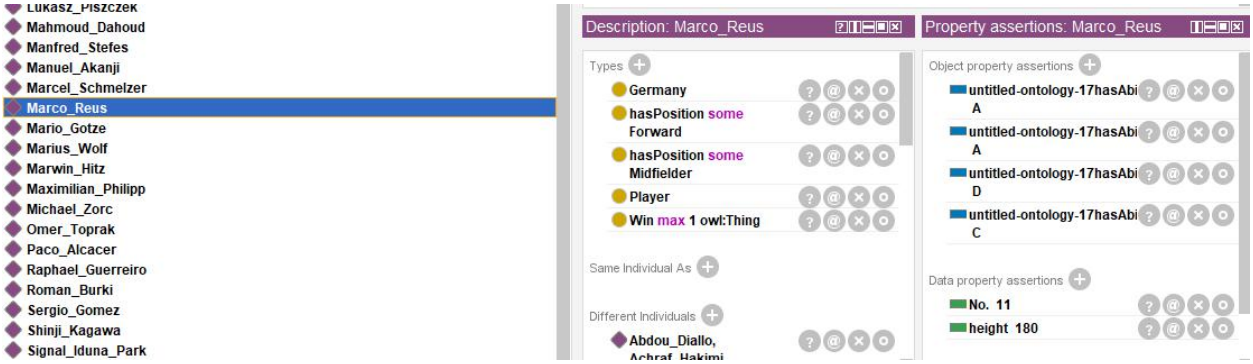

Figure 8 - The example of data property linking a class to a literal.

The turtle format is as follow:

<http://www.semanticweb.org/pc/ontologies/2019/1/height>

rdf:type owl:DatatypeProperty ,owl:FunctionalProperty ;

rdfs:domain <http://www.semanticweb.org/pc/ontologies/2019/1/Player> ;

rdfs:range xsd:integer ;

owl:propertyDisjointWith <http://www.semanticweb.org/pc/ontologies/2019/1/No.> .

(5) .Many disjoint classes (owl:AllDisjointClasses)

For different Coach Type, Athletik-Trainer is disjoint with Chef-Coach, Co-trainer, Fitness Coach.

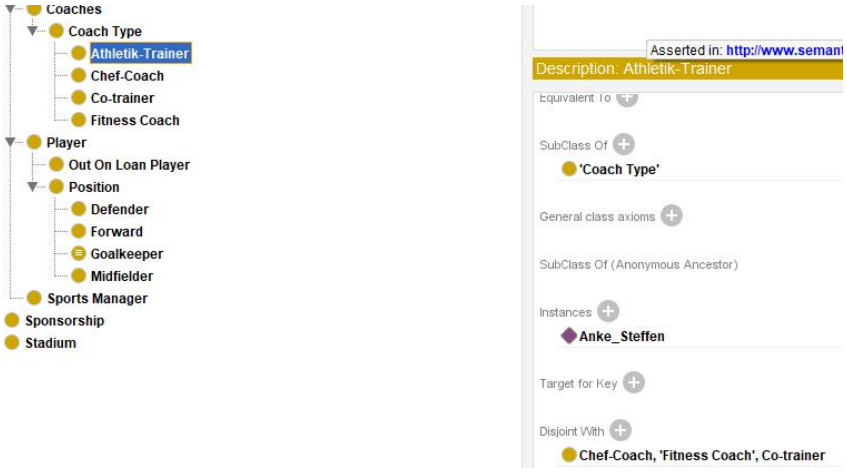

Figure 9 - The example of many disjoint classes.

The turtle format is as follow:

[rdf:type owl:AllDisjointClasses ;

owl:members (<http://www.semanticweb.org/pc/ontologies/2019/1/Athletik-Trainer>

<http://www.semanticweb.org/pc/ontologies/2019/1/Chef-Coach>

<http://www.semanticweb.org/pc/ontologies/2019/1/Co-trainer>

<http://www.semanticweb.org/pc/ontologies/2019/1/Fitness_Coach>)] .

(6) .Different individuals (owl:differentFrom)

Soccer players are different individuals.

Figure 10 - The example of many different individuals.

The turtle format is as follow:

[rdf:type owl:AllDifferent ;

owl:distinctMembers (<http://www.semanticweb.org/pc/ontologies/2019/1/Abdou_Diallo>

<http://www.semanticweb.org/pc/ontologies/2019/1/Achraf_Hakimi>

<http://www.semanticweb.org/pc/ontologies/2019/1/Andreas_Beck>

<http://www.semanticweb.org/pc/ontologies/2019/1/Anke_Steffen>

<http://www.semanticweb.org/pc/ontologies/2019/1/Axel_Witsel>

<http://www.semanticweb.org/pc/ontologies/2019/1/Christian_Pulisic>

<http://www.semanticweb.org/pc/ontologies/2019/1/Dan-Axel_Zagadou>

<http://www.semanticweb.org/pc/ontologies/2019/1/Edin_Terzic>

<http://www.semanticweb.org/pc/ontologies/2019/1/Florian_Wangler>

<http://www.semanticweb.org/pc/ontologies/2019/1/Hans-Joachim_Watzke>

<http://www.semanticweb.org/pc/ontologies/2019/1/Jacob_Bruun_Larsen>

<http://www.semanticweb.org/pc/ontologies/2019/1/Jadon_Sancho>

<http://www.semanticweb.org/pc/ontologies/2019/1/Julian_Weigl>

<http://www.semanticweb.org/pc/ontologies/2019/1/Lucien_Favre>

<http://www.semanticweb.org/pc/ontologies/2019/1/Lukasz_Piszczek>

<http://www.semanticweb.org/pc/ontologies/2019/1/Mahmoud_Dahoud>

<http://www.semanticweb.org/pc/ontologies/2019/1/Manfred_Stefes>

<http://www.semanticweb.org/pc/ontologies/2019/1/Manuel_Akanji>

<http://www.semanticweb.org/pc/ontologies/2019/1/Marcel_Schmelzer>

<http://www.semanticweb.org/pc/ontologies/2019/1/Marco_Reus>

<http://www.semanticweb.org/pc/ontologies/2019/1/Mario_Gotze>

<http://www.semanticweb.org/pc/ontologies/2019/1/Marius_Wolf>

<http://www.semanticweb.org/pc/ontologies/2019/1/Marwin_Hitz>

<http://www.semanticweb.org/pc/ontologies/2019/1/Maximilian_Philipp>

<http://www.semanticweb.org/pc/ontologies/2019/1/Michael_Zorc>

<http://www.semanticweb.org/pc/ontologies/2019/1/Omer_Toprak>

<http://www.semanticweb.org/pc/ontologies/2019/1/Paco_Alcacer>

<http://www.semanticweb.org/pc/ontologies/2019/1/Raphael_Guerreiro>

<http://www.semanticweb.org/pc/ontologies/2019/1/Roman_Burki>

<http://www.semanticweb.org/pc/ontologies/2019/1/Sergio_Gomez>

<http://www.semanticweb.org/pc/ontologies/2019/1/Signal_Iduna_Park>

<http://www.semanticweb.org/pc/ontologies/2019/1/Thomas_Delaney>)] .

(7) .Closes classes (owl:oneOf)

The Goalkeeper is one of Marwin Hitz and Roman Burki. Because one match just needs one goalkeeper.

Figure 11 - The example of closes classes.

The turtle format is as follow:

<http://www.semanticweb.org/pc/ontologies/2019/1/Goalkeeper>

rdf:type owl:Class ; owl:equivalentClass [rdf:type owl:Class ;

owl:oneOf (<http://www.semanticweb.org/pc/ontologies/2019/1/Marwin_Hitz>

<http://www.semanticweb.org/pc/ontologies/2019/1/Roman_Burki>)] ;

(8) .Complex class construction with intersection (owl:intersectionOf)

In 2012 season, Borussia Dortmund win the champion in Bundesliga and German Cup. So 2012 is equivalent to the two classes.

Figure 12 - The example of complex class construction with intersection.

The turtle format is as follow:

<http://www.semanticweb.org/pc/ontologies/2019/1/untitled-ontology-17#2012>

rdf:type owl:Class ; owl:equivalentClass [owl:intersectionOf

(<http://www.semanticweb.org/pc/ontologies/2019/1/Bundesliga>

<http://www.semanticweb.org/pc/ontologies/2019/1/German_Cup>) ;

rdf:type owl:Class] ;

(9).Property restriction: specific value (owl:hasValue)

Players have matches in the home field , Signal Iduna Park.

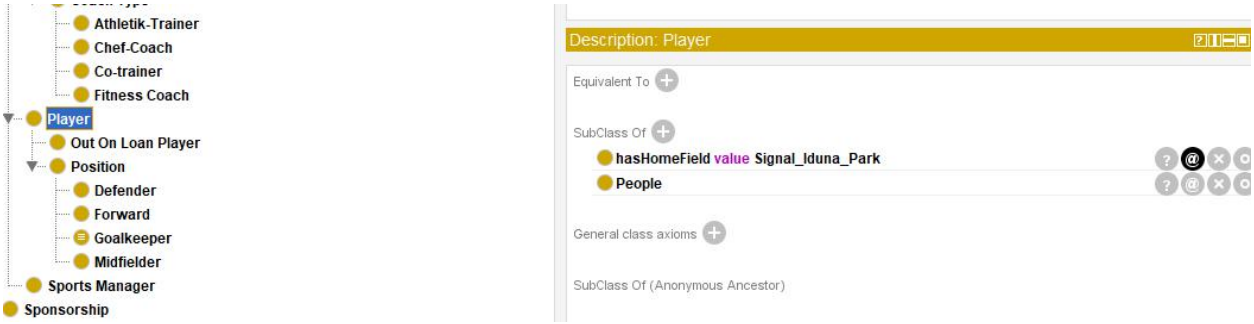

Figure 13 - The example of property restriction: specific value.

The turtle format is as follow:

<http://www.semanticweb.org/pc/ontologies/2019/1/Player>

rdf:type owl:Class ;

rdfs:subClassOf <http://www.semanticweb.org/pc/ontologies/2019/1/People> ,

[rdf:type owl:Restriction ;

owl:onProperty <http://www.semanticweb.org/pc/ontologies/2019/1/hasHomeField> ;

owl:hasValue <http://www.semanticweb.org/pc/ontologies/2019/1/Signal_Iduna_Park>] .

(10).Property restriction: existential (owl:someValuesFrom)

Figure 14 - The example of property restriction: existential.

The turtle format is as follow:

The soccer player Marco Reus can have position midfield while he can also play the role of forward.

<http://www.semanticweb.org/pc/ontologies/2019/1/Marco_Reus>

rdf:type owl:NamedIndividual ,

<http://www.semanticweb.org/pc/ontologies/2019/1/Player> ,:Germany ,

[rdf:type owl:Restriction ; owl:onProperty

<http://www.semanticweb.org/pc/ontologies/2019/1/hasPosition> ;

owl:someValuesFrom <http://www.semanticweb.org/pc/ontologies/2019/1/Forward>] ,

[rdf:type owl:Restriction ; owl:onProperty

<http://www.semanticweb.org/pc/ontologies/2019/1/hasPosition> ;

owl:someValuesFrom

<http://www.semanticweb.org/pc/ontologies/2019/1/Midfielder>]

(11).Property restriction: universal (owl:allValuesFrom)

For the soccer player Christian Pulisic, he just has one position as Forward.

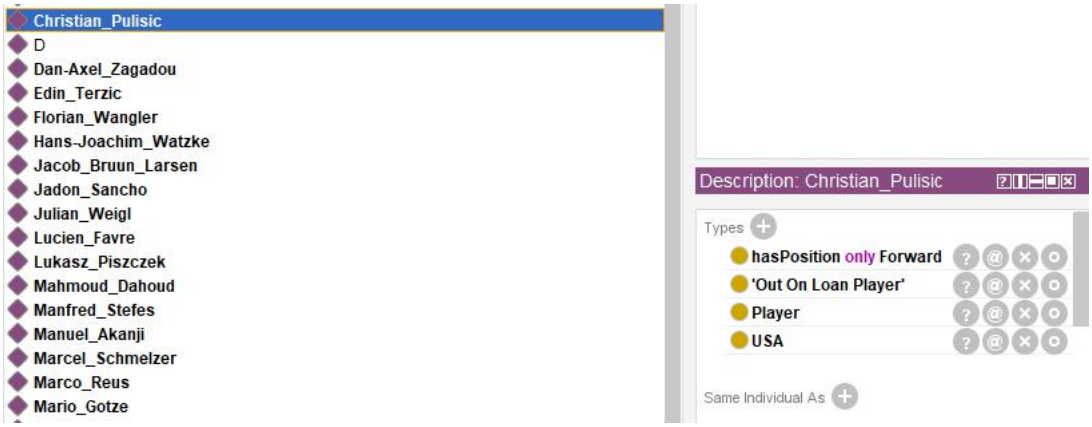

Figure 15 - The example of property restriction: universal.

The turtle format is as follow:

<http://www.semanticweb.org/pc/ontologies/2019/1/Christian_Pulisic>

rdf:type owl:NamedIndividual ,

<http://www.semanticweb.org/pc/ontologies/2019/1/Out_On_Loan_Player> ,

<http://www.semanticweb.org/pc/ontologies/2019/1/Player> ,:USA ,

[rdf:type owl:Restriction ;

owl:onProperty <http://www.semanticweb.org/pc/ontologies/2019/1/hasPosition> ;

owl:allValuesFrom <http://www.semanticweb.org/pc/ontologies/2019/1/Forward>] ;

:hasAbilityofAttack <http://www.semanticweb.org/pc/ontologies/2019/1/B> ;

:hasAbilityofDefence <http://www.semanticweb.org/pc/ontologies/2019/1/D> ;

:hasAbilityofPower <http://www.semanticweb.org/pc/ontologies/2019/1/D> ;

:hasAbilityofSpeed <http://www.semanticweb.org/pc/ontologies/2019/1/A> ;

<http://www.semanticweb.org/pc/ontologies/2019/1/height> 173 ;

<http://www.semanticweb.org/pc/ontologies/2019/1/untitled-ontology-17#No.> 20 .

(12) . Cardinality restriction (owl:cardinality OR owl:minCardinality OR owl:maxCardinality)

One player has at most two positions.

Figure 16 - The example of cardinality restriction.

The turtle format is as follow:

<http://www.semanticweb.org/pc/ontologies/2019/1/Player>

rdf:type owl:Class ;owl:equivalentClass [owl:intersectionOf

(<http://www.semanticweb.org/pc/ontologies/2019/1/Player>

[rdf:type owl:Restriction ; owl:onProperty

<http://www.semanticweb.org/pc/ontologies/2019/1/hasPosition> ;

owl:maxCardinality "2"^^xsd:nonNegativeInteger]) ;

rdf:type owl:Class] ;

(13) Transitive property (owl:TransitiveProperty)

The objected property hasTeammates is TransitiveProperty.

Figure 17 - The example of transitive property.

The turtle format is as follow:

<http://www.semanticweb.org/pc/ontologies/2019/1/hasTeammates>

rdf:type owl:ObjectProperty ,owl:SymmetricProperty ,owl:TransitiveProperty .

(14) Symmetric property (owl:SymmetricProperty)

The objected property hasTeammates is SymmetricProperty.

Figure 18 - The example of symmetric property.

The turtle format is as follow:

<http://www.semanticweb.org/pc/ontologies/2019/1/hasTeammates>

rdf:type owl:ObjectProperty ,owl:SymmetricProperty ,owl:TransitiveProperty .

(15) Functional property (owl:FunctionalProperty)

The data property No. is functional property.

Figure 19 - The example of symmetric property.

The turtle format is as follow:

<http://www.semanticweb.org/pc/ontologies/2019/1/untitled-ontology-17#No.>

rdf:type owl:DatatypeProperty ;rdfs:subPropertyOf owl:topDataProperty ;

rdf:type owl:FunctionalProperty ;

rdfs:domain <http://www.semanticweb.org/pc/ontologies/2019/1/Player> ;

rdfs:range xsd:integer .

9 References.

[1] CSI 5180 Topics in AI - Ontologies and Semantic Web -Assignment 2 - Ontology Design.

Retrieved from

https://uottawa.brightspace.com/d2l/le/content/102243/viewContent/2169219/View

[2] Borussia Dortmund-Wikipedia. Retrieved from

https://en.wikipedia.org/wiki/Borussia_Dortmund#First_national_titles

[3] Borussia Dortmund official website. Retrieved from

https://www.bvb.de/eng

[4] [4] Borussia Dortmund player data from fifa 2019. Retrieved from

https://sofifa.com/team/22/borussia-dortmund/19/159376/