Football Ontology Construction using Oriented Programming

Abstract

According to the W3C, the semantic web is the future of the www. The data that is based on the semantic web can be understood by machines and devices. The main component of the semantic web is the ontology, which is known as the backbone of the semantic web. There are many tools used to edit and create an ontology, however, few kinds of research construct an ontology using oriented programming. SPARQL and API OWL are used to access and edit ontologies, though they are not using oriented programming. The main objective of this paper is to build an ontology using oriented programming and allowable to access OWL entities. Owlready module is effectively used in sport ontology for football in 11 European Leagues.

Keywords: *Owlready, Ontology, Semantic Web, Protégé, RDF and OWL, Oriented-Programming*

I. Introduction

Nowadays, the amount of information on the internet is very massive and not organized. So, applying semantic web technology is very important to organize the information and link the data as well [1], [2]. The ontology is the backbone of the semantic web technology. Ontology is widely used in many fields such as e-learning, medical, industrial, and so on. The term ontology refers to the relationships between concepts where these concepts are related to a specific domain. All the terms and the concepts in the ontology can be represented visually to easily represent this data semantically. The relationship between the concepts and the terms are described formally in the ontology [3]–[5]. There are logical rules used to interpret the concepts and terms in the ontology. Authors in [6] clarified that some reasons such as unavailable methodology and related tools, can make the development of ontology a big challeng. To illustrate the data in the World Wide Web Consortium (W3C), Resource Description Framework-Schema (RDFS) and Web Ontology Language (OWL) are proposed for delivering the foundation of the ontology [7].

In the web, as a huge amount of data is distributed across several fields or domains in different subjects, ontology is used to interpret the knowledge of any domain into key terms [8]. Hence, the development of the ontology is very important in many fields. Due to unavailability of integrated tools and methodologies, the development of the ontology is a challenge [9]. According to Abbas in [10] representing the information on the web is still not formal, this means that there is a gap between the software engineering in different fields.

As mentioned that many fields or domains are useing ontologies to formalize and organize the structure [11]. The ontology term has been used widely in artificial intelligence and representing knowledge in the semantic web. The concepts of the particular domain, properties of the concepts and the individuals of each concept will be described formally [12]. Nowadays, the sport domain is one of the complex domains because of the huge number of football teams and the staff of each team such as players, coaches and so on. So, because of the huge amount of information about the football sport, building an ontology about the football teams will be helpful to extract information and knowledge. In addition to that, there is a difficulty to extract or get the exact information because of the availability of multi resources about football teams over several countries.

Numerous tools and methodologies have been proposed to build and construct ontologies such as protégé plugin and reasoners such as HermiT. This work uses an oriented-programming language to create an ontology for a sport teams in football. The proposed ontology in this paper has been created using python3 as the main language supporting OWL 2, Owlready, NLP and protégé application to read the ontology,

and it is published online[1] . All the related data are linked to each other such as object property, data property, and individuals. The ontology is visualized using OntoGraph tools which is available as a plugin in the protégé tool in addition to the DL query to search for classified concepts in the ontology.

This paper is organized as follows: the section II reviews the related works, section III discusses the methodology of the proposed ontology. Section IV presents the applications and tools used to create the ontology. Finally, conclusions and future suggestions are presented in section V.

II. RELATED WORKS

Many tools are used to present or create ontologies based on RDF or OWL ontologies which are listed in W3C[2]. In this section, the researches that are related to sport domain are reviewed. Each research is using different approaches to create an ontology for a specific domain of the sport. Many researchers applied the semantic web on sports domains with different technologies. In [13], retrieval information is used to build an ontology related to sport by extracting public news from different sports websites. The researchers in [13] focused on football league as the main target. In this research, logical rules are applied to implement the model of information retrieval, in addition to First-Order Logic rules. RDF query language and Jena SPARQL are used to retrieve information. Jena is a java application programming that helps to write java code in the semantic web that deals with RDF and OWL [14].

In [15] an approach of (La Liga) the Spanish league football has been evaluated and illustrated based on the Knowledge Base (KB). Two layers are presented for the ontology based on KB where the contents are smoothly selected, and the structure is in Natural Language text Generation (NLG). The application of the first layer is independent and based on the ontology which is not based on NLG. The second layer is located on the top level of the ontology which is based on semantical relation between the individuals.

Nguyen et al., implemented a recommender system which is based on sports events available on the Web [16]. To use the huge amount of data that available on the web the authors proposed and implement a system to collect the data from different sources on the internet that is related to sports events, by applying machine learning tools. Later, semantic web technologies is applied on the data so as to build the ontology (web engineering). The main contribution was to create an ontology for different vocabularies selected from multiple sources on the Internet. SPARQL query is used to get the information from the different users and sources. The proposed ontology in is created using protégé as the main application and the form of the ontology is RDF.

Ramkumar developed an ontology in sports domain as well [17]. The developed ontology consists of five different sport games like Football, Crickets, Athletics, Tennis and Rugby. The proposed system includes six phases to develop the sport ontology which are: (1) specify the sport domain; (2) important conceps that are related to sport ontology; (3) the main classes of the domain; (4) the properties of the classes of the sport ontology; (5) specify the restriction of the properties; (6) create the instances. The protégé tool is used to create the developed ontology, and its language is OWL which has been converted into a Jena model using eclipse as IDE so as to cluster the documents.

The author in [18] designed an approach of the e-learning system for the sports domain. The system is based on searching for specific keywords which makes the system more accurate to search for specific information, and the response of the system is much better than the traditional developed ones. When the query is generated, it uses the Constraint Satisfaction Problem (CSP) to retrieve the data. When the number of the documents in the system is 256, the result of the search with the keywords showed is 99.6, recall is 256 precision. However, in the traditional systems the recall is 87.5 and 93.7 with precision. Protégé application has been used to design the ontology. The author concluded that based on CSP, the goal is achieved and the result based on CSP was more accurate.

The literature review revealed that, most of the ontologies that are created in the sport domain were not using basic rules to construct the ontology about a specific sports domain. Most of the authors were using protégé tool as the main application to read the ontology and to visualize it, this is because they are open sources sofrware and available from Standford university for free.

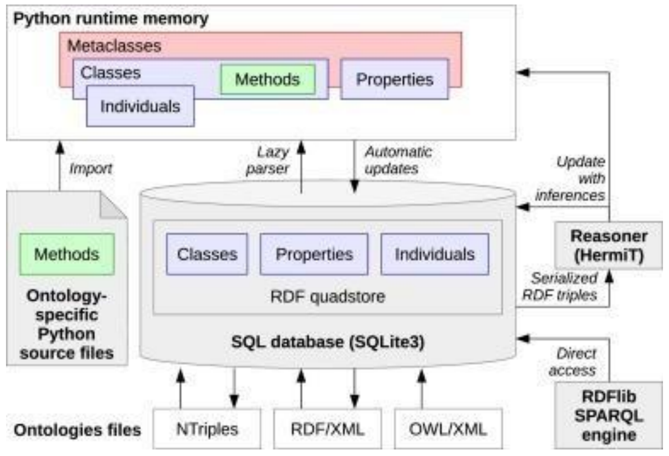


Fig. 1. General architecture of Owlready

III. OWLREADY DESCRIPTION

OWLready is used for oriented-programming to create an ontology using Python language. The ontology is loaded as an object of python and is saved in OWL XML and performed using HermiT. In addition to that, accessing the ontology in Owlready is much more easier and transparent than in Java API. The version of the Owlready that is used in this work is version 2. The general structure of the Owlready is shown in Fig. 1. where the 5 components in Owlready are shown: 1) Implement and optimize the RDF and save it either in memory or disk, the RDF is implementing it with SQL; 2) the classes of the OWL is based on meta concept; 3) optional ontology-specific Python source files, defining methods to insert into OWL classes ;4) classification of the concepts of the ontology using HermiT reasoned [19]; and 5) RDFlib used to import SPARQL engine [11].

The ontology is imported into quadstore and their concepts and/or entities are saved in python objects which is the responsibility of Owlready. additionally, the reasoner is executed there as well. The programmer can use method definitions to mix the OWL and Python statements with the unified interface.

The entities of the ontology are stored in a memory form RDF. When the Owlready parser parsing the entities, the entity of the ontology is wrapped in an object of Python, stored in the cache, then it is returned. This parser is specified as lazy. However, if the objects in Python are modified, the RDF quadstore will be updated automatically. In case if the cache is full the wrapper will be destroyed. Once the program wants to access the entities, the RDF is loaded again and the ontology is linked into the Python module because the programmer can use the definition methods in Python. In the end, the HermiT reasoned used to classify the entities (classes, properties and individuals) of the ontology [20]. This architecture of the Owlready supports ontology and store it in quadstore, which is allowed a fast accessing to entities of the ontology.

OWL/XML, RDF/XML, and NTriples are the three types of ontology that supported by Owlready. All the ontology will be loaded automatically when Owlready is used. The programmer defines a global variable for the ontology path "onto_path". In case the ontology is not found in the local directory or path, it is downloaded from the internet. When Owlready loads the ontologies sometimes an error occurred. In case the ontology is loaded it and the error occurred the ontology is removed from the quadstore. Otherwise, If the load is failed no change occurred in the quadstore. As each ontology has IRI it should be available online, But if ontology can not be available online an HTTP error will occur.

The classification in Owlready in automatically executed as protégé application. Nevertheless, in Owlready calling sync_reasoner() function. The function is defined as a global which is used to export the RDF in a temporary file to run the HermiT reasoned on that file.

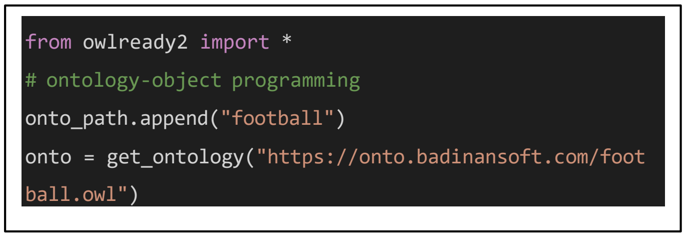
The classes of the ontology with Owlready is defined as Python class. The classes can be defined directly or in a separate Python file.

IV. IMPLEMENTATION OF THE PROPOSED SYSTEM

The main aim of this paper is to create an ontology and develop an Owlready using oriented-programming in Python 3 to access OWL ontology. As reviewed in many sports ontologies in this paper, most of the classes represented as classes not individuals. Nonetheless, the available tools that are used to construct the ontology do not permit classes as individuals. So, a Python module is designed using oriented programming. As mentioned in the previous section, the classes (entities) of the ontology are defined as objects so using oriented programming will allow accessing the entities of OWL.

The implementation of Owlready was in Python 3. In this work the Owlready which is used is Owlready V2, because it is used to optimize the RDF and also used to support the big ontologies. Owlready is free of charge and available in Python Package[3].

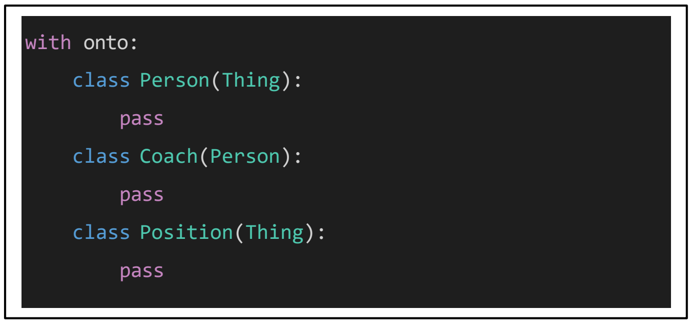
The first step that Owlready loading the ontology from local storage or the internet. The script code shows the method of loading the ontology and export the ontology or save it.



```
from owlready2 import *
# ontology-object programming
onto_path.append("football")
onto = get_ontology("https://onto.badinansoft.com/foot
ball.owl")
```

Fig. 2. Get the ontology

As it is clear from the script code the *onto.path* is defined as a global variable. Owlready first searches for the local storage in case it is not available, it tries to get it from the internet. The function *get_ontology* will create the new ontology in case it is not available in local memory. The current Owlready 2 is supporting only OWL/XML format.

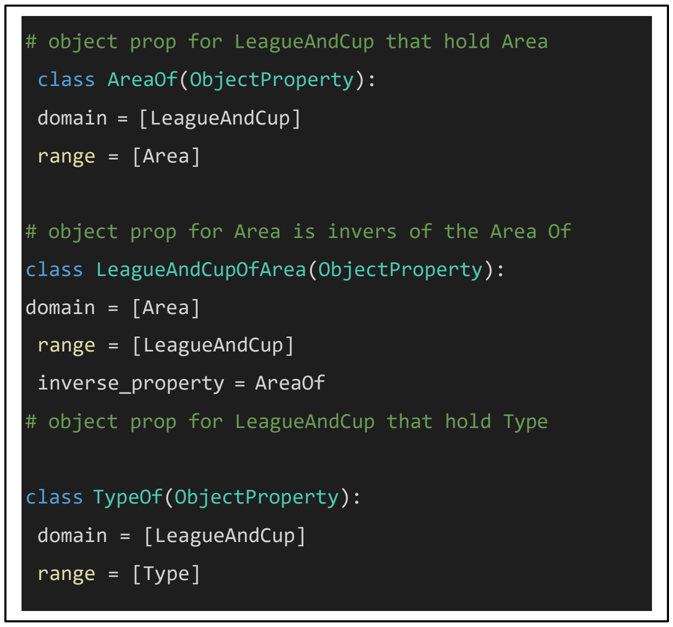


```
with onto:
    class Person(Thing):
        pass
    class Coach(Person):
        pass
    class Position(Thing):
        pass
```

Fig. 3. Creating class in Python

The first entity of the ontology created is the class. The parent class of the ontology is Thing which is defined as the main class and the other classes inherited from the Thing class as shown in the next script. The subclass can be created as an inherited class.

After the classes are created the next step is to create the property of the ontology. When the property has defined the domain and range is specified for each property. There are two types of properties, object property, and data property. Owlready allows the user to specify multiple domains or ranges if required. The below code shows how the domain and range can be specified for an object property. Where the domains and ranges are classes.



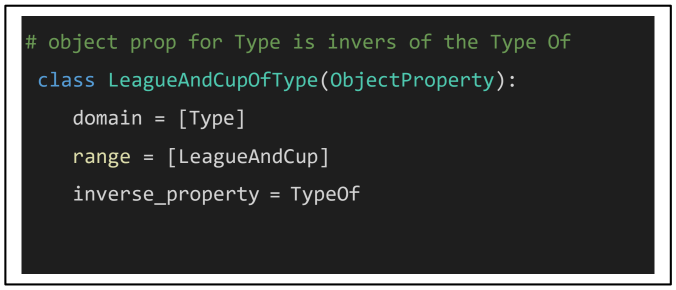
```
# object prop for LeagueAndCup that hold Area
 class AreaOf(ObjectProperty):
 domain = [LeagueAndCup]
 range = [Area]

# object prop for Area is invers of the Area Of
class LeagueAndCupOfArea(ObjectProperty):
domain = [Area]
 range = [LeagueAndCup]
 inverse_property = AreaOf
# object prop for LeagueAndCup that hold Type

class TypeOf(ObjectProperty):
 domain = [LeagueAndCup]
 range = [Type]
```

Fig. 4. Object Property

As mentioned that everything in Python is defined as an object, the object property and data property in is not defined in Owlready. However, the data property defines the range as a data type. The data types that are supported by Owlready are int, float, bool, str (string), Owlready.normstr, and DateTime.date. In case if there is an inverse property, the "inverse_proerty" attribute wihin Owlreay is used to define the inverse property. As shown in the following script code Fig. 5.

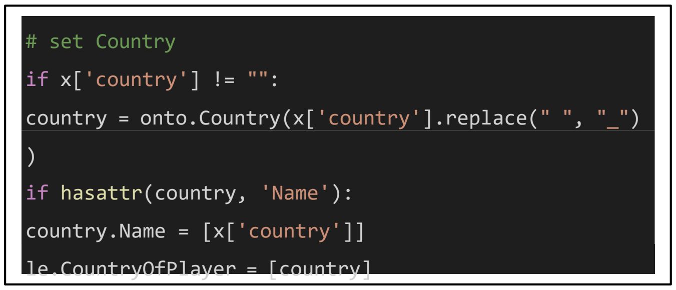


```
# object prop for Type is invers of the Type Of
 class LeagueAndCupOfType(ObjectProperty):
    domain = [Type]
    range = [LeagueAndCup]
    inverse_property = TypeOf
```

Fig. 5. Inverse of Object Property

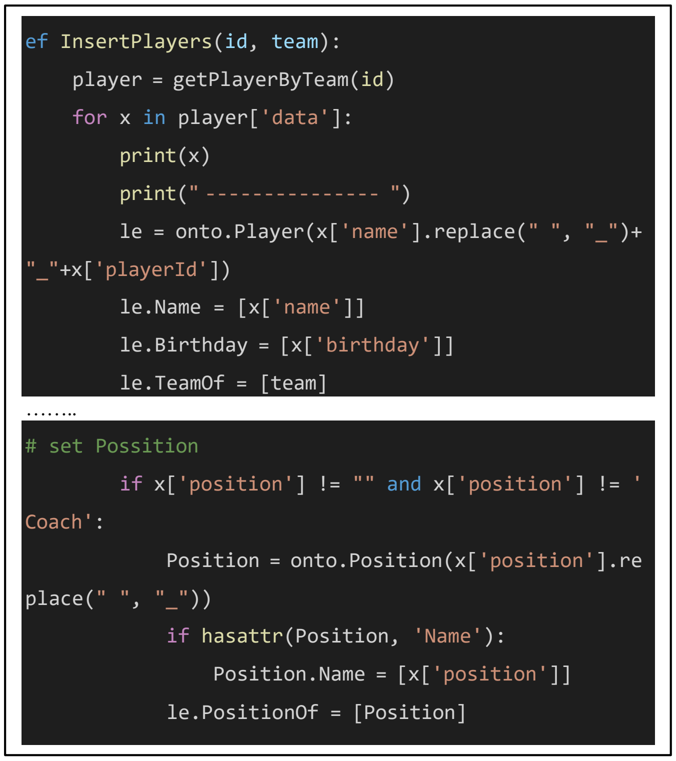
The inverse property will be updating the relations automatically. But, Owlready is not updating the property automatically in this version.

This work is focusing on the 11 European leagues (Austerian, Belgian, Denmark, Finland, France Ligue 1, France Ligue 2, German 1, German 2, Ireland, Poland, and Rusin), So, it is important to set the country of each league in the ontology as individuals with the object property (CountryOf, AreaOf, and TypeOf) and Data property Name. Once the league has been set it the next step is to set each team for it is related league and country as well. For instance: the team (RB_Leipzig) is set it for German_Bundesliga. Fig 6. showed the piece code of the country.



```
# set Country
if x['country'] != "":
country = onto.Country(x['country'].replace(" ", "_")
)
if hasattr(country, 'Name'):
country.Name = [x['country']]
le.CountryOfPlayer = [country]
```

Fig. 6. The piece code of the country

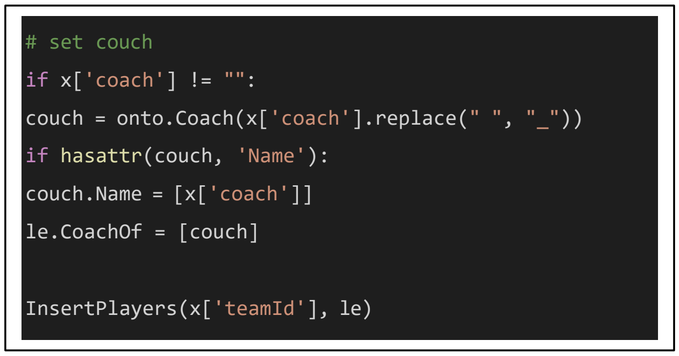


```
ef InsertPlayers(id, team):
    player = getPlayerByTeam(id)
    for x in player['data']:
        print(x)
        print(" --------------- ")
        le = onto.Player(x['name'].replace(" ", "_")+
"_"+x['playerId'])
        le.Name = [x['name']]
        le.Birthday = [x['birthday']]
        le.TeamOf = [team]
……..
# set Possition
        if x['position'] != "" and x['position'] != '
Coach':
            Position = onto.Position(x['position'].re
place(" ", "_"))
            if hasattr(Position, 'Name'):
                Position.Name = [x['position']]
            le.PositionOf = [Position]
```

Fig. 7. player and his position in the team

The final step for creating the proposed ontology in Python is to set the player for his related team with his position in the team and some important object and data properties such as (TeamOf, CountryOfPlayer, and PositionOf) and data property (Name and Birthday).

One of the important individuals is to set a coach for each team or to display his name. The only data property is set for the coach which is his name. As shown in the below code the set of each coach for his team.

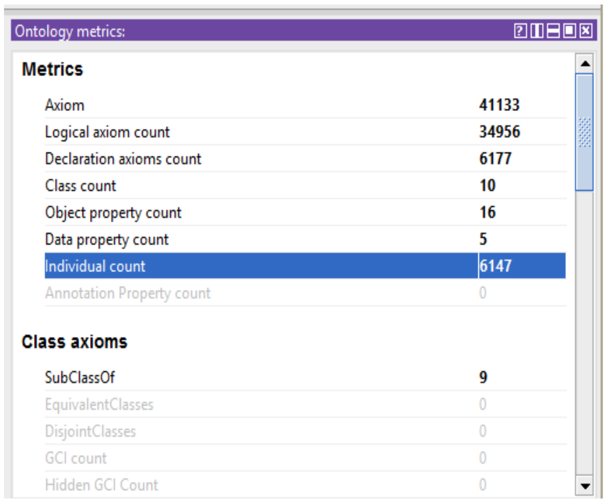


```
# set couch
if x['coach'] != "":
couch = onto.Coach(x['coach'].replace(" ", "_"))
if hasattr(couch, 'Name'):
couch.Name = [x['coach']]
le.CoachOf = [couch]

InsertPlayers(x['teamId'], le)
```

Fig. 8. set the coach for a team

The next step is to load or read the ontology that has been constructed by Python 3 as the main programming language. Protégé application used to read the ontology and visualize it as well. Protégé is an open source to edit and read the ontology and providing a graphical interface as well. As mentioned in the previous section that this ontology is related to the football team in some Europe league. The main IRI of the ontology is (*https://onto.badinansoft.com/football.owl*). The main classes for this ontology are (Area, LeagueAndCup, Person, Position, Team, and Type). The number of individuals in this ontology is about 6147, this is a huge number because it includes 11 leagues in Europe, with 10 classes, object properties, and data properties are 16 and 5 respectively. As shown in Fig. 9. the main metrics in the ontology.



Ontology metrics:

Metrics

Axiom	41133
Logical axiom count	34956
Declaration axioms count	6177
Class count	10
Object property count	16
Data property count	5
Individual count	6147
Annotation Property count	0

Class axioms

SubClassOf	9
EquivalentClasses	0
DisjointClasses	0
GCI count	0
Hidden GCI Count	0

Fig. 9. Metrics of football ontology

As it can be seen in Fig. 10. the main structure of the implemented ontology about the football team. The main reason for taking the football team in Europe is because a lot of people interesting in European League and do not have enough information about each team. This ontology will clarify the main information related to each team such as players, coaches, and country.

The object and data properties are one of the important parts of the ontology, where object Properties are used to define the relationship between the classes and set domain and range for each property. While the data properties used to set the relationship between the instances (individuals). Table 1 and 2 shows the main properties in football ontology which their range and domain for each property.

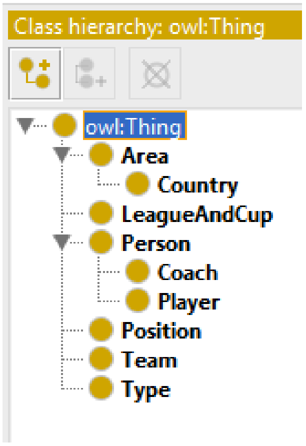


Fig. 10. The main classes of the Implemented ontology

The individuals of the ontology are created and load it into protégé because of the huge number of the individuals of this ontology it is difficult to display all of them, but in the next section of visualization will display some of them.

As shown in Fig. 11. the main classification of the classes in football ontology. as can be seen that the main classes of the ontology in linked with the parent class which is Thing. The class Type is connected with LeagueAndCup class and has two individuals (League and Cup).

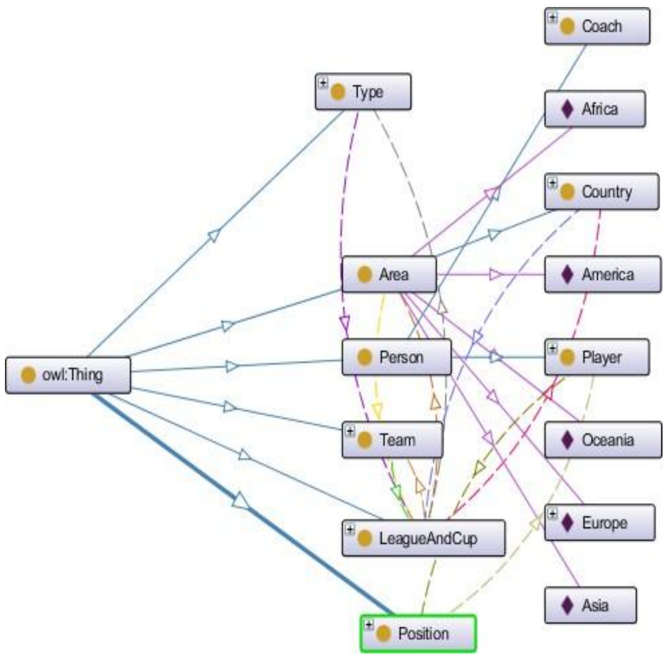


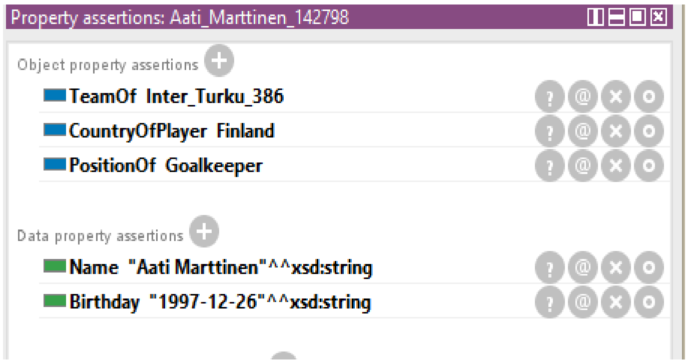
Fig. 11. classification of football ontology classes

TABLE I. THE OBJECT PROPERTIES ON FOOTBALL ONTOLOGY

Object Property	**InverseOf**	**Domain**	**Range**
TeamOfCoach	CoachOf	Coach	Team
AreaOf	LeagueAndCup	LeagueAndCup	Area
CoachOf	TeamOfCoach	Team	Coach
CountryOf	LeagueAndCup	LeagueAndCup	Country
CountryOfPlayer	PlayerOfCountry	Player	Country
LeagueAndCupOfAreaa	AreaOf	Area	LeagueAndCup
LeagueAndCupOfCountry	CountryOf	Country	LeagueAndCup
LeagueAndCupOfType	TypeOf	Type	LeagueAndCup
LeagueOf	TeamOfLeagueAndCup	Team	LeagueAndCup
PlayerAtPosition	PositionOf	Position	Player
PlayerAtTeam	TeamOf	Team	Player
PlayerOfCountry	CountryOfPlayer	Country	Player
PositionOf	PlayerAtPosition	Player	Position
TeamOf	PlayerAtTeam	Player	Team
TeamOfLeagueAndCup	LeagueOf	LeagueAndCup	Team
TypeOf	LeagueAndCupOfType	LeagueAndCup	Type

TABLE II. DATA PROPERTIES OF FOOTBALL ONTOLOGY

Data Property	**Domain**	**Range**
Address	Team	String
Birthday	Player	String
Capacity	Team	Integer
FoundingDate	Team	String
Name	Team, Player, Country, Coach, and LeagueAndCup	String

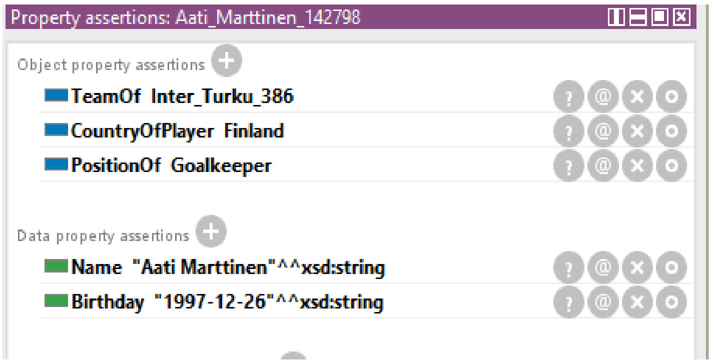

Property assertions: Aati_Marttinen_142798

Object property assertions
- TeamOf Inter_Turku_386
- CountryOfPlayer Finland
- PositionOf Goalkeeper

Data property assertions
- Name "Aati Marttinen"^^xsd:string
- Birthday "1997-12-26"^^xsd:string

Fig. 12. object and data properties Aati_Marttinen

The property assertion of the ontology for each individual has been set according to related data. For example, the individual Aati_Marttinen is a type of player class with the object and data properties as shown in Fig. 13.

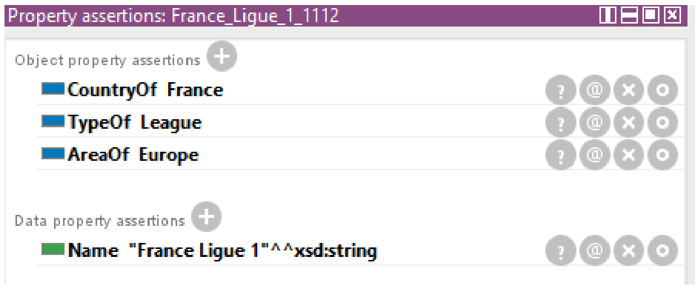

Property assertions: Aati_Marttinen_142798

Object property assertions
- TeamOf Inter_Turku_386
- CountryOfPlayer Finland
- PositionOf Goalkeeper

Data property assertions
- Name "Aati Marttinen"^^xsd:string
- Birthday "1997-12-26"^^xsd:string

Fig. 13. object and data properties Aati_Marttinen

The individual France_Ligue_1 is a type of LeagueAndCup class. This individual set it with the related object and data properties as can be seen in Fig. 14.


Property assertions: France_Ligue_1_1112

Object property assertions
- CountryOf France
- TypeOf League
- AreaOf Europe

Data property assertions
- Name "France Ligue 1"^^xsd:string

Fig. 14. object and data properties of France_Ligue_1

As mentioned previously that the last step in the ontology is to create the individuals and set it in a specific class of the ontology. the individual then set it with the object and data properties as shown in Fig. 14.

V. CONCLUSIONS AND FUTURE WORK:

In this paper, a sport ontology has been created using ontology-oriented programming by Python version 3. Several researches within the field of ontology creation have been reviewed, however, most of them are constructing ontologies using common methods in the protégé application. To the best of our knowledge, none of them were using oriented programming to construct an ontology. Owlready module, which is described in details, has been used for ontology oriented programming.

Witin this work, a football ontology for 11 European Leagues has been created, and is loaded in protégé application to be read and visualize it. This ontology has 10 classes and more than 6000 individuals with object and data properties.

The future plan of this work is to integrate the ontology with a semantic search engine so as to search for specific information and link the related data and individuals. Furthermore, additional leagues around the world to be added to the ontology, and use the Linked Open Data (LOD) technology to link the related data from different leagues around the world.

References

[1] A.-Z. S. R. Zeebaree, A. Adel, K. Jacksi, and A. Selamat, 'Designing an ontology of E-learning system for duhok polytechnic university using protégé OWL tool', *J Adv Res Dyn Control Syst Vol*, vol. 11, pp. 24–37.

[2] K. Jacksi, S. R. Zeebaree, and N. Dimililer, 'LOD Explorer: Presenting the Web of Data', *Intl J. Adv. Comput. Sci. Appl.*, vol. 9, no. 1, pp. 45–51, 2018.

[3] K. Jacksi, N. Dimililer, and S. R. M. Zeebaree, 'A Survey of Exploratory Search Systems Based on LOD Resources', in *PROCEEDINGS OF THE 5TH INTERNATIONAL CONFERENCE ON COMPUTING & INFORMATICS*, COLL ARTS & SCI, INFOR TECHNOL BLDG, SINTOK, KEDAH 06010, MALAYSIA, 2015, pp. 501–509.

[4] S. R. M. Z. Adel AL-Zebari Karwan Jacksi and Ali Selamat, 'ELMS–DPU Ontology Visualization with Protégé VOWL and Web VOWL', *J. Adv. Res. Dyn. Control Syst.*, vol. 11, no. 1, pp. 478–485, 2019.

[5] R. Ibrahim, S. Zeebaree, and K. Jacksi, 'Survey on Semantic Similarity Based on Document Clustering', *Adv. Sci. Technol. Eng. Syst. J.*, vol. 4, no. 5, pp. 115–122, 2019, doi: 10.25046/aj040515.

[6] A. Akerman and J. Tyree, 'Using ontology to support development of software architectures', *IBM Syst. J.*, vol. 45, no. 4, pp. 813–825, 2006.

[7] K. Jacksi, N. Dimililer, and S. Zeebaree, 'State of the art exploration systems for linked data: a review', *Int J Adv Comput Sci Appl IJACSA*, vol. 7, no. 11, pp. 155–164, 2016.

[8] K. Jacksi and S. M. Abass, 'Development History Of The World Wide Web', *Int. J. Sci. Technol. Res.*, vol. 8, pp. 75–79, 2019.

[9] K. Jacksi, 'Design and Implementation of E-Campus Ontology with a Hybrid Software Engineering Methodology', *Sci. J. Univ. Zakho*, vol. 7, no. 3, pp. 95–100, 2019.

[10] M. A. Abbas, 'A Unified Approach for Dealing with Ontology Mappings and their Defects', 2016.

[11] J.-B. Lamy, 'Owlready: Ontology-oriented programming in Python with automatic classification and high level constructs for biomedical ontologies', *Artif. Intell. Med.*, vol. 80, pp. 11–28, 2017.

[12] M. Horridge, H. Knublauch, A. Rector, R. Stevens, and C. Wroe, 'A practical guide to building OWL ontologies using the Protégé-OWL plugin and CO-ODE tools edition 1.0', *Univ. Manch.*, 2004.

[13] N. N. Aung and T. T. Naing, 'Sports Information Retrieval with Semantic Relationships of Ontology', presented at the 3rd International Conference on Information and Financial Engineering, 2011, vol. 12.

[14] M. Grobe, 'Rdf, jena, sparql and the'semantic web'', presented at the Proceedings of the 37th annual ACM SIGUCCS fall conference: communication and collaboration, 2009, pp. 131–138.

[15] N. Bouayad-Agha, G. Casamayor, L. Wanner, F. Díez, and S. L. Hernández, 'FootbOWL: Using a generic ontology of football competition for planning match summaries', presented at the Extended Semantic Web Conference, 2011, pp. 230–244.

[16] Q. Nguyen, L. N. Huynh, T. P. Le, and T. Chung, 'Ontology-Based Recommender System for Sport Events', presented at the International Conference on Ubiquitous Information Management and Communication, 2019, pp. 870–885.

[17] D. B. Sudha Ramkumar, 'Development of Ontology for Sports Domain'.

[18] M. S, 'DESIGN AND DEVELOPMENT OF ONTOLOGY BASED E-LEARNING SYSTEM FOR SPORTS DOMAIN', PhD Thesis, ANNA UNIVERSITY, 2014.

[19] B. Motik, R. Shearer, and I. Horrocks, 'Hypertableau reasoning for description logics', *J. Artif. Intell. Res.*, vol. 36, pp. 165–228, 2009.

[20] 'Introduction — Owlready 0.2 documentation'. https://pythonhosted.org/Owlready/intro.html (accessed Mar. 27, 2020).