# SAIKS Report for the 2022 summer semester

Authors: Bogdan Bordeianu, Filip Kovacevic, Nino Ziegelwanger
Group number: 10
E-mails: e11836796@student.tuwien.ac.at;; filip.kovacevic@tuwien.ac.at; e1029033@student.tuwien.ac.at;

## Step 1: Advanced Ontology Engineering

### Introduction

In the first step we create an ontology from scratch in the application domain of European football. The semantic system should be an information system, where you can retrieve different information about football, with an opportunity for betting on different markets from different betting providers.
Our main concepts are *Person*, *Game*, *Market* and *Contract*. A *Person* has a nation and can be a natural person or a legal entity.
A natural person can be a football associate or a user. A user has the opportunity to place bets. A legal entity can be an association, a bookmaker or a sponsor. A bookmaker offers markets for betting. A sponsor sponsors a team. An association has some infrastructure, minimum one team  and can be a club or a selection.
A market  can only be a match result, has an underlying game and some bets from the users.
A game belongs to a season, has a home and an away participant, takes place on a playground, and belongs to a division. A *Playground* is an infrastructure like an *Office* and can be a stadium or a training center. A *Division* is a part of a competition. A competition can be a selection competition or a club competition, which can be a domestic club competition or an international club competition.
A contract belongs to a football associate, defines a role and can be an association contract or a referee contract. An association contract belongs to an association and is valid for at least one season. A referee *Contract* is valid for at least one *Game*. Additionally there are transfers, which are transitions from an old *Contract* to a new *Contract*.

### Competency questions

The semantic system should answer the following competency questions:

1. Which associations have more than 20 players?
2. Which stadiums have a capacity of more than 20.000?
3. How many associations have a mental coach?
4. How many selection competitions were played in the last season?
5. Which match result has the highest draw quote?
6. Which player plays the most games?
7. Which transfer was the most expensive?

8. Which team has the most sponsors?
9. Are there different types of competitions?
10. Is a bookmaker a legal entity?

## OWL features

We have used the following OWL2 features:

*Table 1 - OWL features*

| **Feature** | **Usage (Examples)** |
|---|---|
| Union Disjoint Classes | Subtypes of person, competition and contract |
| Disjoint Properties | Old contract can not be the new contract of a transfer,<br>home participant can not be away participant in a game |
| Cardinality | A bet is placed by exactly one user,<br>An association contract is valid for minimum one season,<br>A market has some bets |

## Design patterns

We have used the following design patterns:

*Table 2 - Design patterns*

| **Pattern** | **Usage (Examples)** |
|---|---|
| part Of | A division is part of a competition |
| n-ary relations | A game takes place in a season, belongs to a division, takes place on a playground, has a home team and has an away team. |
| roles | An association contract belongs to an association, is valid for at least one season, belongs to on football associate and defines one role |

## Process

In the beginning we were sitting together and implemented a first draft of the ontology. During step 2 we modify our ontology to align our concepts with the data for betting. After step 3 we modify the ontology again to fix the critical and important issues found by Oops.

# Step 2: KG creation through ODBA

In this step we employ ETL techniques to integrate football-related datasets found on the web into our previously defined knowledge structure and create a knowledge graph, which we then also explore and visualize a subgraph using GraphDB's visualization feature. We

illustrate our process in Figure 1. We start from four datasets, three found on the web and a handcrafted one. We use a python script to transform and load the data into a postgre database for which we also design a database schema. We containerize this step using docker in order to make it reproducible. We then use Ontop-mappings to first prepare the data by using terms from our ontology (Ontop source) and then build our knowledge graph (KG) by writing triple statements in the Ontop-target section considering reasonable IRIs. Finally, we materialize our KG and extract it together with the ontology into a new .owl file. We only target those concepts and properties of our ontology for which we have data.

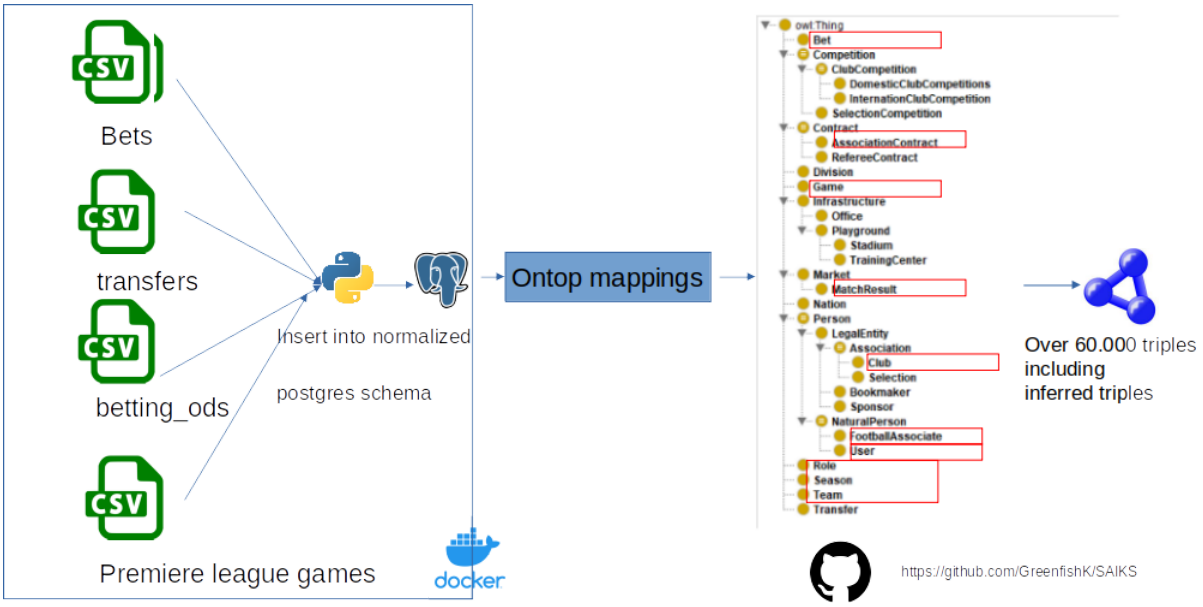



*Figure 1* - Overview of our ETL process to construct the KG.

## Extracting data

After having designed the first version of our ontology we looked for datasets to fit both - the betting-related and association-related concepts of our ontology. We sourced three dataset from three different websites including Github, Kaggle and football-data.co.uk. We used the df_full_premiereleague.csv[1] dataset to extract Premiere League games and information such as teams, results, goals and game dates. From the E0.csv[2] dataset we extracted betting-related data, such as betting providers, odds and offering dates. To be able to integrate data into our *Contract* concept which a.o. tells about players and the seasons they played in, we used data from the player transfer market which were available in a github directory with subdirectories structured by years. For the sake of this lecture, we only used transfer data from 2021. Additionally, we created a synthetic dataset bets.csv where we constructed records about bet placements from users on certain games.

## Creating the database schema

After having dug into the data and searching online for relational database schemas for football we got an idea how our relational database schema could look like. We used a schema from Github[3] as a starting point and extended & modified it to cover all the data we

[1] https://www.kaggle.com/datasets/pablohfreitas/all-premier-league-matches-20102021?select=df_full_premierleague.csv
[2] https://www.football-data.co.uk/mmz4281/2021/E0.csv
[3] https://github.com/plkpiotr/soccer-league

wanted to integrate. At that point we raised a fundamental question whether our constructed database schema should be closer to the ontology or to the datasets. For the sake of this lecture, we stayed closer to the data with our schema design[4] so that we could explore data transformation functions in the Ontop mappings.

## Transforming data

In this step we subdivided the data transformation process into three steps including an intermediary loading process, i.e a staging process.
In the first step, we used a python script[5] to load the datasets into dataframes, align club names, rename a few columns to fit the schema and load them all together into a postgre database (staging). In order to make this step reproducible, we employed docker[6] to containerize the transformation and loading process.
In the second step, we used the “source part” of Ontop to create surrogate keys, concatenate columns, change date formats and rename columns and thereby bring the data closer to our ontology and prepare them for the construction of our knowledge graph.
In the third step, we used the “target part” of Ontop to fit the prepared data from the “source part” into our knowledge graph with concepts from our ontology. We created five Ontop mappings[7] with a varying number of concepts and properties. Roughly, each mapping targets 1-3 concepts and up to 11 properties from our ontology. Inhere, we also tried to think of reasonable IRI paths, e.g. football:club/{id_home}/team/first-team_squad would be the locator for the first team squad of, say, *label*(id_home) = ‘Manchester United’. Manchester United could also have a women's team squad.
We also used the label property from the well-known rdfs: namespace so that the nodes in GraphDB would have human-readable labels.

## Loading data - Creating the KG

To create the knowledge graph we simply used Protege’s Ontop plugin to materialize the triples by first selecting and starting the Ontop 4.2.1 reasoner and then materializing them in turtle syntax as defined in the Ontop-mappings. Throughout the engineering process of our ontology we used this feature several times and tried out all the materialization possibilities, which include serializing the KG in different formats, saving the resulting triples into a separate or materializing them into our ontology. What worked best for us was to materialize them into the active ontology but then save it as a new file in order to have two files - one with the ontology only and one with ontology + KG. Sometimes this materialization process would get stuck which required us to restart Protege and retry. Our KG at the time of assignment submission contains 66.771 triples including inferred ones and 43 ontology entities.
Upon KG creation we imported it into GraphDB to inspect a subgraph of it visually. Figure 2 shows three users with their bets and one market which all three users have placed a bet on. This market is a MatchResult market, offered by the bookmaker B365 and includes the Newcastle United vs West Bromwich Albion match.

[4] https://github.com/GreenfishK/SAIKS/blob/main/scripts/create.sql
[5] https://github.com/GreenfishK/SAIKS/blob/main/scripts/import_data.py
[6] https://github.com/GreenfishK/SAIKS/blob/main/docker-compose.yml
[7] https://github.com/GreenfishK/SAIKS/blob/main/Workspace/saiks2022ss_football_desktop_with_inds.obda

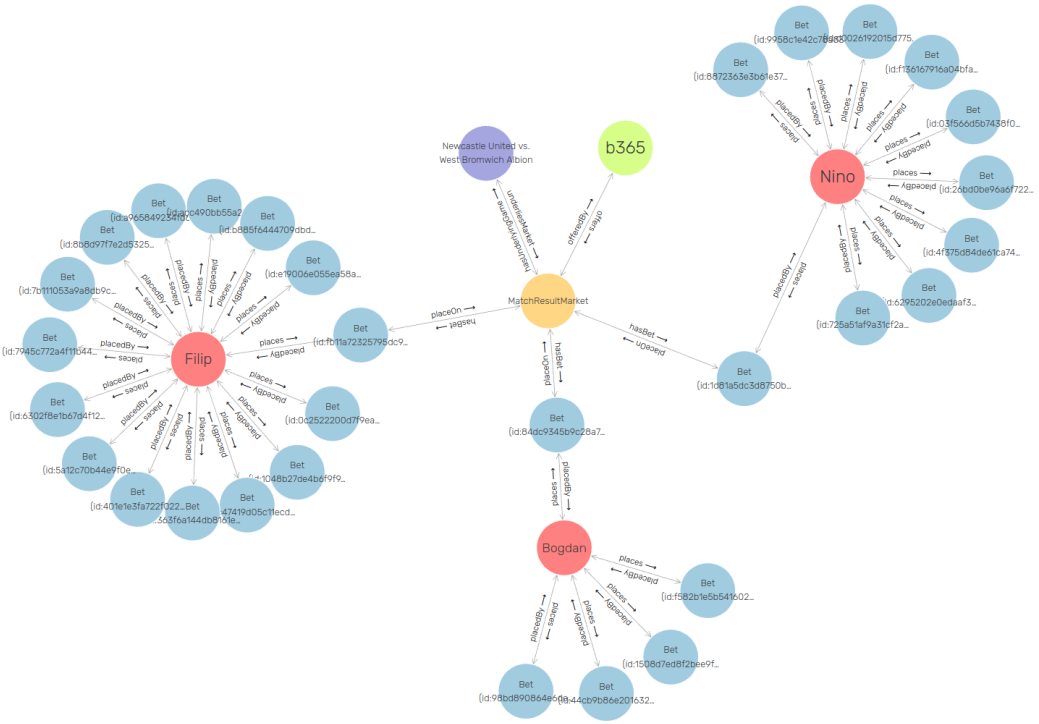


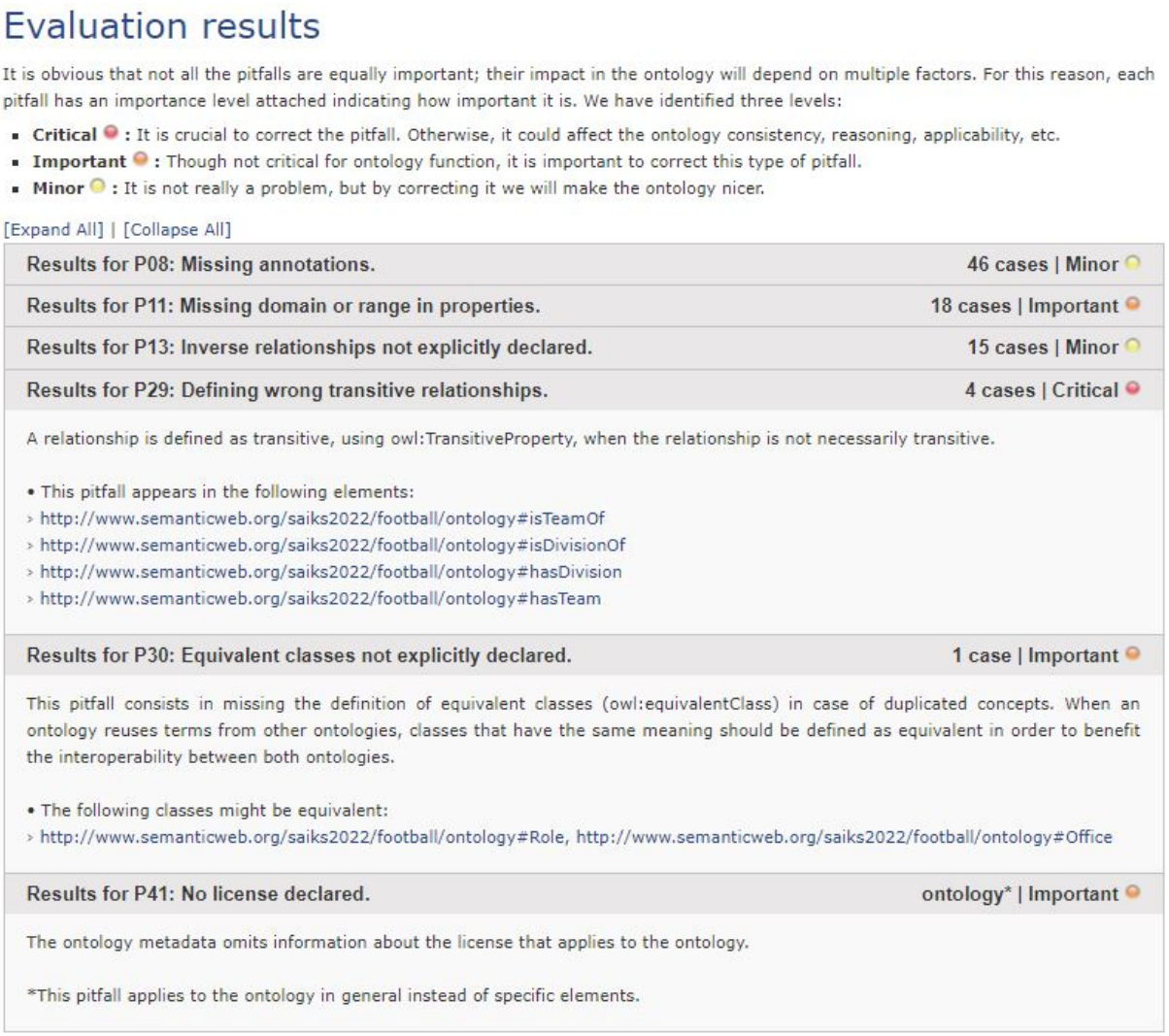

*Figure 2*: A subgraph of the football KG showing three users and a shared betting market among them.

# Step 3: Verification of Semantic Structures

## Ontology verification (OOPS!)

For this step first we had to save our ontology from “Turtle Syntax” to “RDF-XML Syntax” so that we can use “Scanner by direct input” in the online tool.

After running the scanner on our ontology, the tool detected some pitfalls, minor, important and critical.



## Evaluation results

It is obvious that not all the pitfalls are equally important; their impact in the ontology will depend on multiple factors. For this reason, each pitfall has an importance level attached indicating how important it is. We have identified three levels:

- **Critical** : It is crucial to correct the pitfall. Otherwise, it could affect the ontology consistency, reasoning, applicability, etc.
- **Important** : Though not critical for ontology function, it is important to correct this type of pitfall.
- **Minor** : It is not really a problem, but by correcting it we will make the ontology nicer.

[Expand All] | [Collapse All]

**Results for P08: Missing annotations.** — 46 cases | Minor

**Results for P11: Missing domain or range in properties.** — 18 cases | Important

**Results for P13: Inverse relationships not explicitly declared.** — 15 cases | Minor

**Results for P29: Defining wrong transitive relationships.** — 4 cases | Critical

A relationship is defined as transitive, using owl:TransitiveProperty, when the relationship is not necessarily transitive.

• This pitfall appears in the following elements:
› http://www.semanticweb.org/saiks2022/football/ontology#isTeamOf
› http://www.semanticweb.org/saiks2022/football/ontology#isDivisionOf
› http://www.semanticweb.org/saiks2022/football/ontology#hasDivision
› http://www.semanticweb.org/saiks2022/football/ontology#hasTeam

**Results for P30: Equivalent classes not explicitly declared.** — 1 case | Important

This pitfall consists in missing the definition of equivalent classes (owl:equivalentClass) in case of duplicated concepts. When an ontology reuses terms from other ontologies, classes that have the same meaning should be defined as equivalent in order to benefit the interoperability between both ontologies.

• The following classes might be equivalent:
› http://www.semanticweb.org/saiks2022/football/ontology#Role, http://www.semanticweb.org/saiks2022/football/ontology#Office

**Results for P41: No license declared.** — ontology* | Important

The ontology metadata omits information about the license that applies to the ontology.

*This pitfall applies to the ontology in general instead of specific elements.

*Figure 3 - OOPS results*

The first important pitfall detected by the tool was “P11: Missing domain or range in properties - Object and/or datatype properties without domain or range (or none of them) are included in the ontology”. This pitfall was fixed via Protégé very easy.

The only critical pitfall “P29: Defining wrong transitive relationships - A relationship is defined as transitive, using owl:TransitiveProperty, when the relationship is not necessarily transitive.” was fixed also using Protégé. Example for Object property “isTeamOf” that was wrongly ticked as “Transitive”.

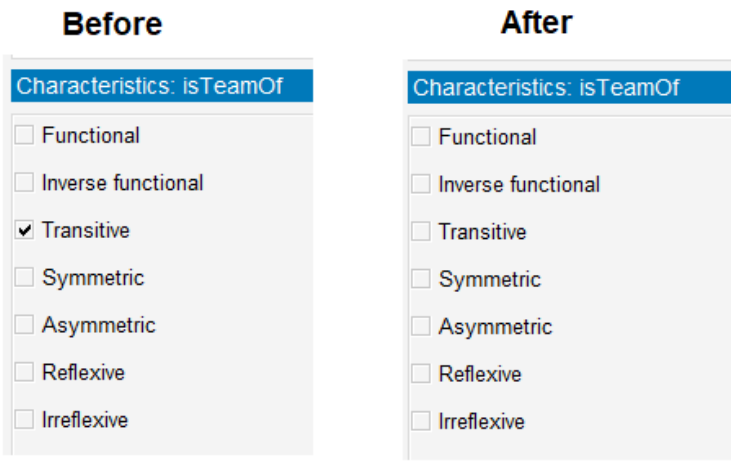
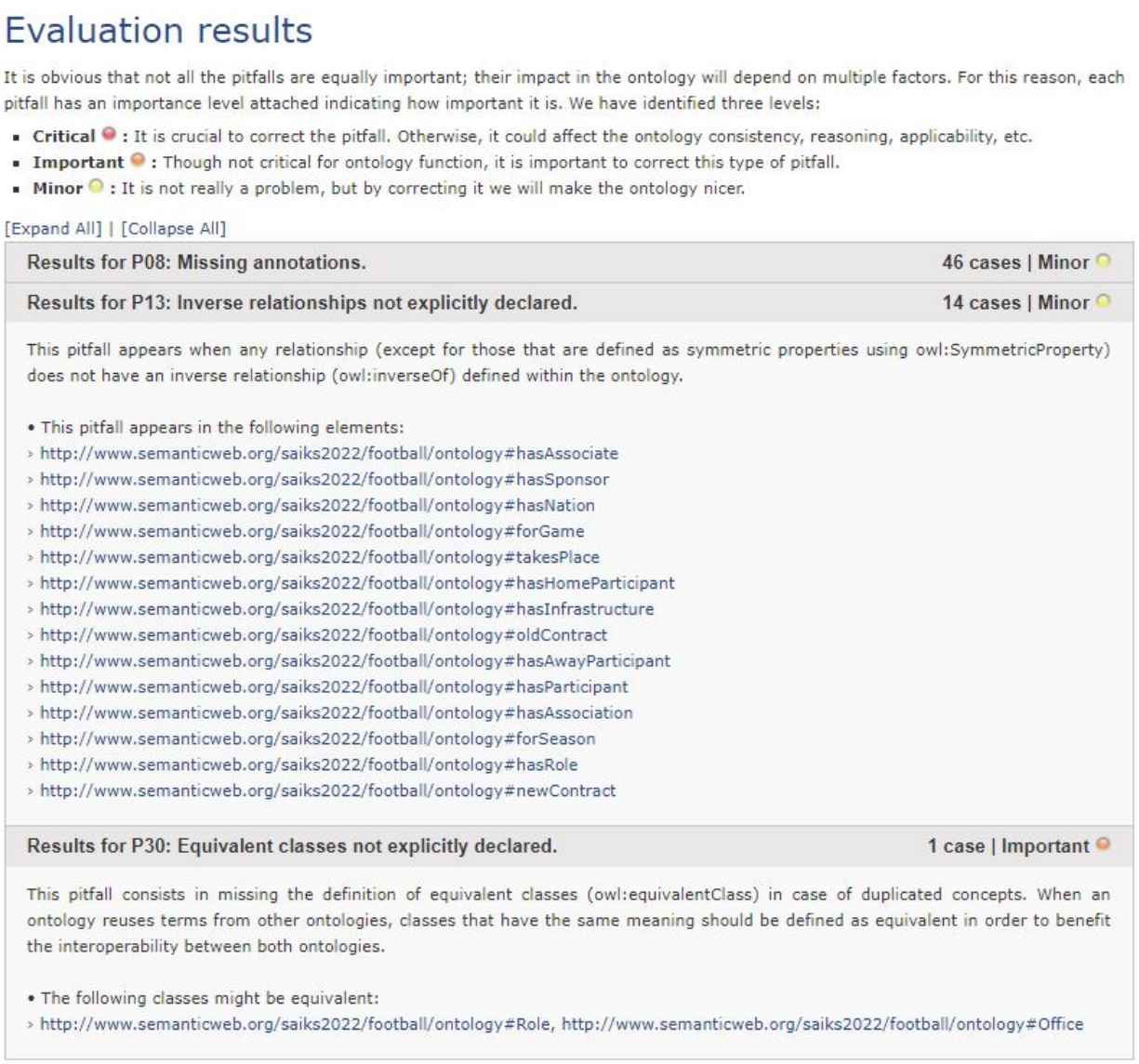



*Figure 4 - Changes made to the isTeamOf property in our ontology*

Another important pitfall detected by the toll was “P41: No license declared - The ontology metadata omits information about the license that applies to the ontology.” Our football ontology omitted information about the licenses that applies to it. To fix this pitfall we edited the metadata file of our ontology and added the necessary license information which clearly stated that this football ontology is licensed under Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International (CC BY-NC-SA 4.0). In Protégé, the annotation property “dcterms:license” was created and then used to conduct this modification. https://creativecommons.org/licenses/by-nc-sa/4.0/

The last important pitfall detected by the toll was “P30: Equivalent classes not explicitly declared - This pitfall consists in missing the definition of equivalent classes (owl:equivalentClass) in case of duplicated concepts. When an ontology reuses terms from other ontologies, classes that have the same meaning should be defined as equivalent in order to benefit the interoperability between both ontologies.” This pitfall regarding equivalent classes/duplicated concepts (Role class and Office class) we didn’t fix because in our ontology they are completed different classes, different concept, with different data and object properties.

After solving the pitfalls and running again the tool with fixed ontology this was the result:



## Evaluation results

It is obvious that not all the pitfalls are equally important; their impact in the ontology will depend on multiple factors. For this reason, each pitfall has an importance level attached indicating how important it is. We have identified three levels:

- **Critical** : It is crucial to correct the pitfall. Otherwise, it could affect the ontology consistency, reasoning, applicability, etc.
- **Important** : Though not critical for ontology function, it is important to correct this type of pitfall.
- **Minor** : It is not really a problem, but by correcting it we will make the ontology nicer.

[Expand All] | [Collapse All]

**Results for P08: Missing annotations.** — **46 cases | Minor**

**Results for P13: Inverse relationships not explicitly declared.** — **14 cases | Minor**

This pitfall appears when any relationship (except for those that are defined as symmetric properties using owl:SymmetricProperty) does not have an inverse relationship (owl:inverseOf) defined within the ontology.

• This pitfall appears in the following elements:
› http://www.semanticweb.org/saiks2022/football/ontology#hasAssociate
› http://www.semanticweb.org/saiks2022/football/ontology#hasSponsor
› http://www.semanticweb.org/saiks2022/football/ontology#hasNation
› http://www.semanticweb.org/saiks2022/football/ontology#forGame
› http://www.semanticweb.org/saiks2022/football/ontology#takesPlace
› http://www.semanticweb.org/saiks2022/football/ontology#hasHomeParticipant
› http://www.semanticweb.org/saiks2022/football/ontology#hasInfrastructure
› http://www.semanticweb.org/saiks2022/football/ontology#oldContract
› http://www.semanticweb.org/saiks2022/football/ontology#hasAwayParticipant
› http://www.semanticweb.org/saiks2022/football/ontology#hasParticipant
› http://www.semanticweb.org/saiks2022/football/ontology#hasAssociation
› http://www.semanticweb.org/saiks2022/football/ontology#forSeason
› http://www.semanticweb.org/saiks2022/football/ontology#hasRole
› http://www.semanticweb.org/saiks2022/football/ontology#newContract

**Results for P30: Equivalent classes not explicitly declared.** — **1 case | Important**

This pitfall consists in missing the definition of equivalent classes (owl:equivalentClass) in case of duplicated concepts. When an ontology reuses terms from other ontologies, classes that have the same meaning should be defined as equivalent in order to benefit the interoperability between both ontologies.

• The following classes might be equivalent:
› http://www.semanticweb.org/saiks2022/football/ontology#Role, http://www.semanticweb.org/saiks2022/football/ontology#Office

*Figure 5 - OOPS results after correction*

## KG verification

In this step we wrote 5 SHACL rules to verify our knowledge graph:

1. *Passwort Constraint*: This is a constraint for the password of a user. A user must have exactly one password and the password must have at least 8 characters. The password must be a string.
2. *Quotes Constraint*: This is a constraint for the quotes of a match result. Each quote must be minimum exclusive 1 and each quote must be a double.
3. *Tip Constraint*: This constraint checks the tip of a bet. If the bet is placed on a match result, the tip must be A (away), H (home) or d (draw). If the bet is placed on any other market, the tip can be any string.

4. *Association Contract Constraint*: This is a constraint for the association contract. Each association contract must have exactly 1 role, exactly 1 football associate, exactly 1 association and at least one season.
5. *Nation Constraint*: This is a constraint for the nation of an association. Each association must have exactly one nation. The nation be either a string or an instance of the class nation.

# Step 4: Ontology Alignment

We did a google search to find ontologies from the football domain. First, we found three ontologies within scientific papers[8][9][10] as figures or appendix and one ontology available on Github[11] as text representation. We took two out of these four ontologies into closer consideration, namely, the ones that were available as text in OWL/XML representation. One of them comprised only a T-box  and the other one a T-Box and A-Box. We first decided to pick the one with both boxes so that the ontology alignment tools could use the extensional concept similarity to conclude the similarity of their concepts by comparing their instances. To align our ontology (=source ontology) with the chosen premier league ontology we employed AML and LogMap. While LogMap found only one similar concept AML found two similar concepts. Due to the low number of mappings and for the sake of the alignment report we decided to try with a football ontology from the previous semester (=target ontology, T-Box only), which also was our stepping stone for Step 1 and therefore shares more similar entities. We executed the ontology alignment with two tools - AML and LogMap but got results only from AML. In Table 3 we show some metadata about our two ontologies. We describe our experience, settings and results with ALM in the following.

*Table 3* - Ontologies' metadata

| | Source ontology | Target ontology |
|---|---|---|
| Number of classes | 33 | 20 |
| Number of properties | 46 | 22 |

## AML

AML found 8 common classes with the automatic matching. The automatic matching has a seemingly high threshold as only concepts around 98% certainty were matched. So we executed the "custom match" with a 60% threshold. In this setting AML found 17 common entities and we labeled 15 of them as correct, as shown in Figure 6. One mismatch was, however, interesting. The object property "has role" was mapped to "has referee". This is on the second thought not entirely wrong as we modeled referees via a role-based pattern in our new ontology. However, our domain for "has role" our new ontology is broader and also

[8] https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5556663
[9] researchgate.net/publication/221140988_A_Semantic_Framework_for_Personalized_Ad_Recommendation_based_on_Advanced_Textual_Analysis
[10] https://www.diag.uniroma1.it/~degiacom/papers/2007/calv-etal-OWLED-2007.pdf
[11] https://github.com/apurvann/RDF-OWL-Soccer/blob/master/soccer.owl

encompasses players. So the low certainty of 60.56% is reasonable. In Table 4 we summarize our results from AML in the two different settings.

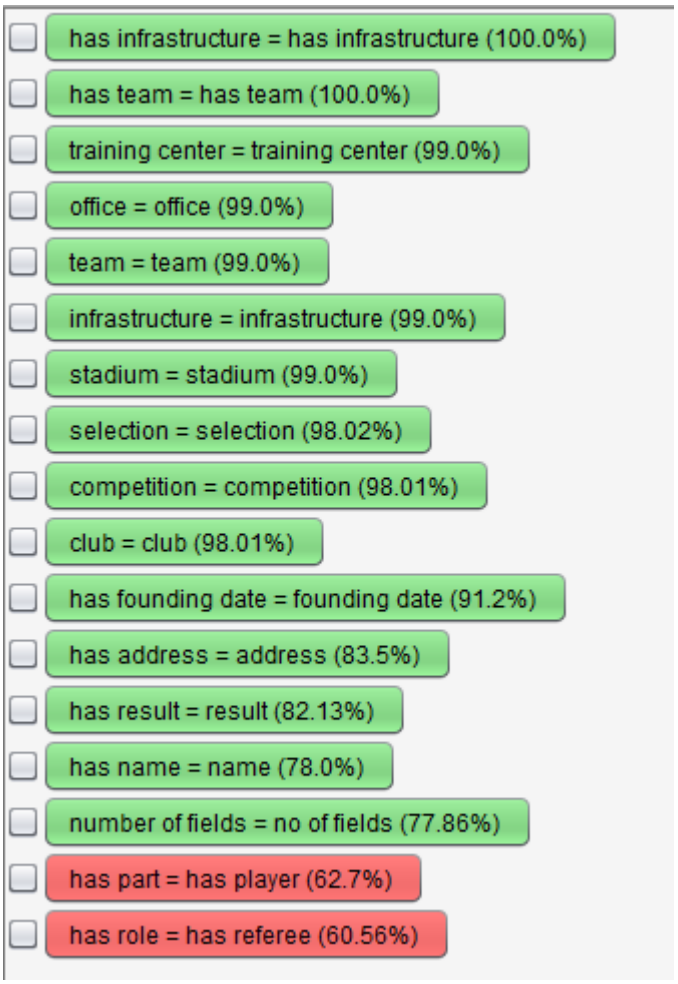



*Figure 6 - Mappings and correct/not correct color-encoded labels*

*Table 4* - alignment results

| | Mappings | Threshold |
|---|---|---|
| AML automatic match | 8/8 right mappings | > 84% |
| AML custom match | 15/17 right mappings | 60% |

# Appendix

*Table 5* - Deliverables

| **Requirement** | **File in archive** | **Online link** |
|---|---|---|
| OWL ontology | saiks2022ss_football_desktop.owl | https://github.com/GreenfishK/SAIKS/blob/main/Workspa |

| | | |
|---|---|---|
| | | ce/saiks2022ss_football_desktop.owl |
| Database dump | database_dump.zip | https://github.com/GreenfishK/SAIKS/blob/main/Datasets/database_dump.zip |
| Knowledge graph | saiks2022ss_football_desktop_with_inds.owl | https://github.com/GreenfishK/SAIKS/blob/main/Workspace/saiks2022ss_football_desktop_with_inds.owl |
| SHACL definitions (.ttl) | shacl_constraints.ttl | https://github.com/GreenfishK/SAIKS/blob/main/Workspace/SHACL/shacl_constraints.ttl |
| SHACL reports (.ttl) | shacl_report.ttl | https://github.com/GreenfishK/SAIKS/blob/main/Workspace/SHACL/shacl_report.ttl |
| Alignment result files | aml_alignment_report_automatic_match.rdf | https://github.com/GreenfishK/SAIKS/blob/main/Workspace/OntologyAlignment/aml_alignment_report_automatic_match.rdf |
| | aml_alignment_report_custom_match.rdf | https://github.com/GreenfishK/SAIKS/blob/main/Workspace/OntologyAlignment/aml_alignment_report_custom_match.rdf |
| Report including the description of your work for all 4 steps of the assignment and the rationale for key decisions you took. | Saiks2022ss_report_group10.pdf | https://github.com/GreenfishK/SAIKS/blob/main/deliverables/Saiks2022ss_report_group10.pdf |