Production Systems and Information Engineering
Volume 11 (3), pp. 88–103.
https://doi.org/10.32968/psaie.2023.3.6

ANALYSIS OF COMPANIES, FILMFESTIVAL AND FUNNY SALAD ONTOLOGY

ANITA AGÁRDI
University of Miskolc
Hungary Institute of Information Technology
agardianita@iit.uni-miskolc.hu

Abstract: This article presents the analysis and evaluation of three ontological systems. The first ontology represents a company, the second presents the film festival, and the third is about a salad bar. These three ontologies are presented in the article. The article also evaluates ontologies based on metrics, for which metrics adapted from the UML. The ontologies are not my own implementations, all three systems were available on GitHub. The article shows that UML metrics can also be used to evaluate ontological systems.

Keywords: *ontology, metrics, UML, companies, filmfestival, funny salad*

1. Introduction

Comparing and evaluating systems and objects is a common technique. Humanity has always strived to create the best possible tools and systems. Over the years, objective measures have also been introduced for this reason. The existence of software metrics and their application are very important, they help software developers in the early phase of software development in preparing the system and validating its goodness.

Baroni & Abreu [1] discuss definitions of object-oriented design metrics. A UML metamodel has been created and some concrete examples are presented.

Chen, Boehm, Madachy & Valerdi [2] present a study of 14 projects with three different types of metrics. These metrics cover different parts of the software lifecycle. These lifecycles are: requirement, architecture and implementation.

Yi, Wu & Gan [3] compare some typical metrics of UML class diagrams. The authors used the following comparison: complexity, theoretical and empirical validation, advantage and disadvantage. The following metric types are reported by the authors: M. Marchesi, Genero, In, Rufai. The following M. Marchesi metrics are reported by the authors: number of classes (OA1), number of inheritance hierarchies (OA2), average weighted method of classes (OA3), standard deviation of the number of weighted methods of classes (OA4), dependencies of classes (OA5) , the standard deviation of the number of direct dependencies (OA6) and the percentage of inherited dependencies in relation to their total number (OA7). Genero's metrics are: number of classes (NC), number of attributes (NA), number of methods (NM), association relations (NAssoc), aggregation relations (NAgg), dependency relations (NDep), generalization relations (NGen), generalization hierarchies (NGenH), aggregation hierarchies (NAggH). In metrics: The output indicators are total number of classes (TNC), inheritance relations (TNIR), usage relations (TNUR), association relations (TNA), roles (TNR), operation (TNO), parameters (TNP) and the attributes (TNCA). Rufai's metrics: Shallow Semantic Similarity Metric (for class names) (SSSM) and Deep Semantic Similarity Metric (for attribute and method names) (DSSM). The second type of metric is the signature similarity metric (SBSM). However, a third approach is the usage of the relationships between the classes of a class model as a criterion for comparing the models to be compared (relation-based similarity metric RBSM).

2. Ontology systems

In this chapter, I present the ontology itself and OWL as a Web Ontology Language, followed by the three selected ontology systems.

2.1. Ontology

Ontology is the representation of knowledge. One of the most well-known languages is the OWL (Web Ontology Language), which is used to describe ontologies. OWL has an XML-like syntax. Classes are one of the most important building blocks of ontologies. For example, classes can be created with the following syntax in OWL [4]:

```
<owl:Class rdf:ID="professor">
<rdfs:subClassOf rdf:resource="#academicStaff"/>
</owl:Class>
```

Classes can contain properties. The object property that connects two classes can be created with the following syntax [4]:

```
<owl:ObjectProperty rdf:ID="isTaughtBy">
<owl:domain rdf:resource="#course"/>
<owl:range rdf:resource="#academicStaff"/>
</owl:ObjectProperty>
```

The datatype property associates a class with a datatype value, which can be created with the following syntax [4]:

```
<owl:DatatypeProperty rdf:ID="year">
<rdfs:range rdf:resource="http://www.w3.org/2001/XLMSchema
#nonNegativeInteger"/>
</owl:DatatypeProperty>
```

The creation of OWL does not necessarily have to be written by ourselves, ontology editors have been created where a new ontology and its OWL elements can be created. OWL elements can be viewed graphically, modified or deleted. Protége [5] is such an ontology editor.

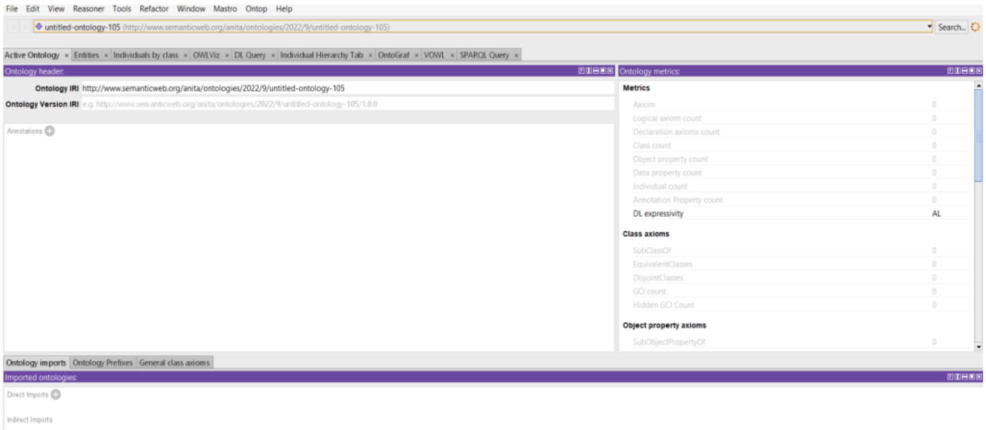

Figure 1. *Protége main page*

On the main page (Figure 1), we can see the ontology IRI, ontology version IRI on the Active Ontology tab. In the entities section, we can see the following (each in a separate sub-tab): classes, object properties, data properties, annotation properties, datatypes, individuals. Individuals by classes is a part, which presents the system from another aspect. OWLViz displays the classes of the system in a class-subclass hierarchy, in a graph structure.

2.2. Companies Ontology

Companies [6] is an ontology representing companies. Its main classes are the followings: 'Energy', 'Financial_Services', 'Food', 'Foreign_Corporations', 'Health_Care', 'Hospitality', 'Manufacturing', 'Retail_Stores' and 'US_Corporations'. These classes were further specialized by the authors. Within the 'Energy' class are oil and solar ('Oil_ Exploration', 'Solar_Energy'). The financial area is subdivided into commercial bank, investment, financial advisor, etc. ('Commercial_Bank', 'Consulting', 'Financial_Ad visor', 'Investment_Bank').

Companies from foreign countries were also divided into sub-classes, the following sub-classes were created: British, French, German, Italian, Japanese.

The companies dealing with health ('Healt_Care') were also divided into categories. There are two categories here which are as follows: 'Medical_Equipment' and 'Pharmaceutical_Manufacturer'.

The ontology describing companies also includes the 'Manufacturing' class. This class includes the following subclasses: aircraft production, automobile production, electrical equipment production and medical equipment.

The system does not contain properties, neither data type nor object properties.

However, it contains individuals of companies. It contains 26 individuals, which are as follows: 'Bank_Of_America' (as commercial bank), 'BMW' (as automotive company and German company), 'Capital_One' (as commercial bank), 'Charles_Schwabb' (as financial advisor), 'Chase_Bank' (as commercial bank), 'Chrysler' (as an automotive manufacturer), 'Citibank' (as a commercial bank), 'Credit_Suisse' (as an investment bank), 'Deloitte' (as a consulting firm), 'Exxon_Mobil' (as an oil company), 'Fidelity' (as a financial advisor), 'Ford_Motor_Company' (as a car manufacturer), 'Franklin_ Templeton' (as a financial advisor), 'General_Electric' (as an electrical equipment company), 'General_Mills' (as a food company), 'General_Motors' (as a car manufacturer), 'Goldman_Sachs' (as investment bank), 'JP_Morgan_Chase' (as an investment bank), 'Kraft' (as a food company), 'McKinsey' (as a consulting company), 'Medtronix' (as a medical equipment company), 'Mitsubishi' (as a Japanese automotive company), 'Morgan_Stanley' (as a financial consulting firm), 'Price_Waterhouse_ Cooper' (as a consultant company), 'Toyota' (as a Japanese automotive company), 'Wells_Fargo' (as a commercial bank). The VOWL representation of the ontology is presented in Figure 2.

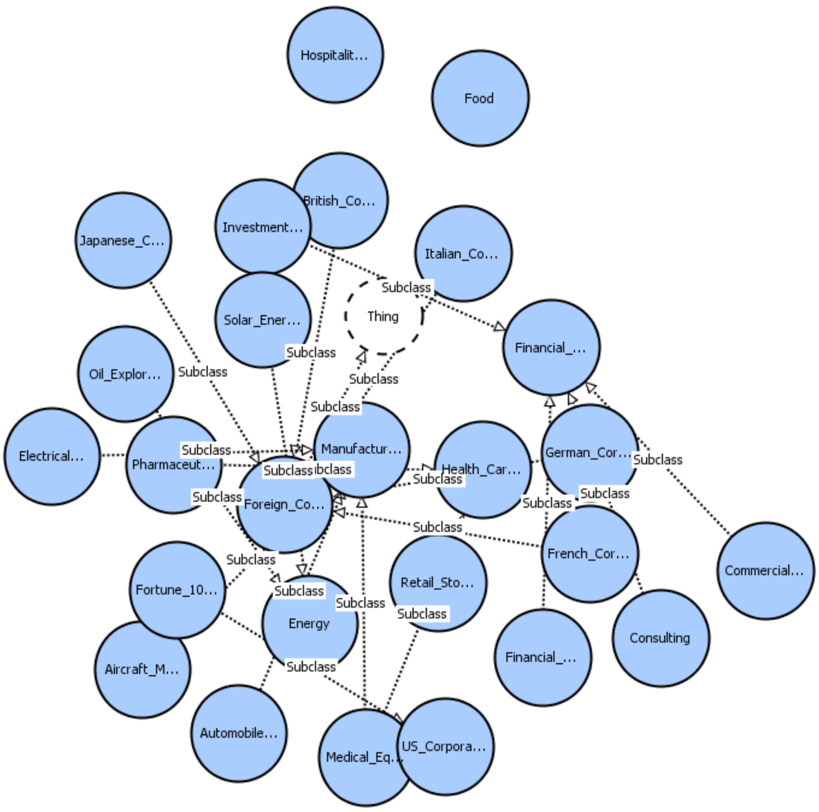

Figure 2. *Companies ontology VOWL representation*

In the following, I will present the UML diagram of Companies OWL. During the conversion, I did not convert all OWL classes into UML classes. In my opinion, the subclasses of each class function better as an enumeration than as a class. Thus, the number of classes in UML has decreased. The OWL does not contain properties: neither object nor datatype properties, so there is no connection between the individual classes in UML. I have connected the UML classes with each enumeration with a ‘type’ data member.

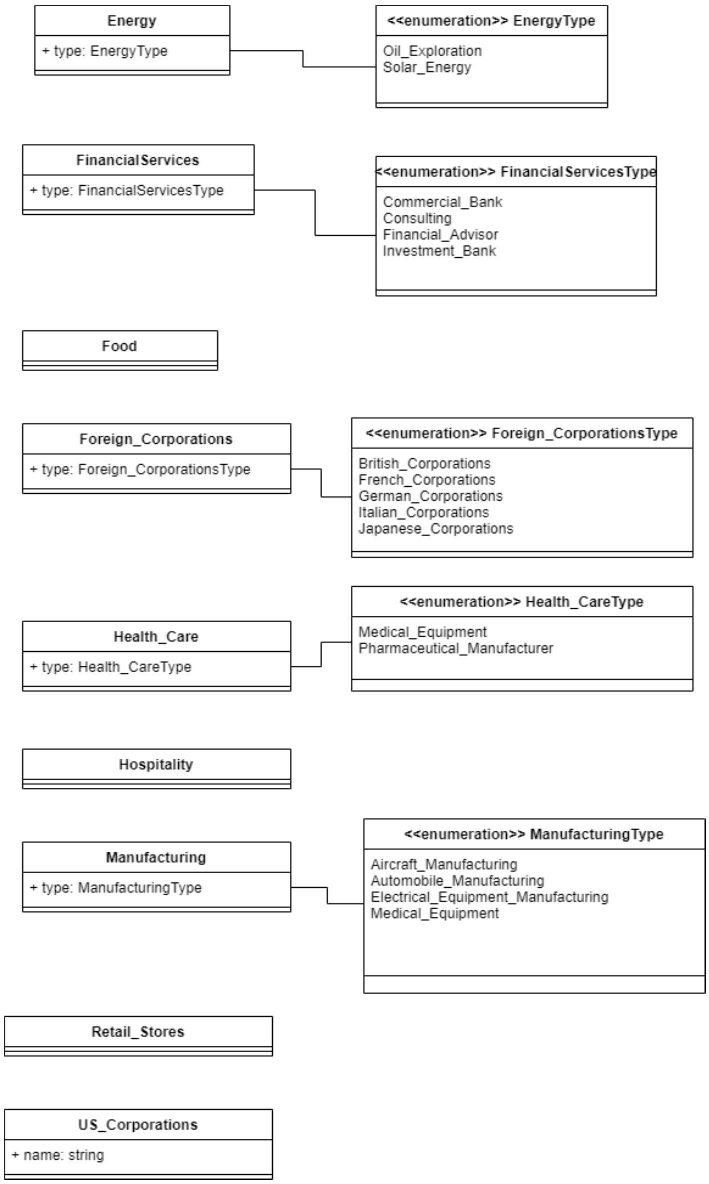

Figure 3. *Companies ontology UML representation*

2.3. Filmfestival Ontology

The film festival [7] ontology is used to model a film festival. This ontology contains not only classes, but also object and data type properties. The ontology contains the following classes: ‘dbo:FilmFestival’ (which represents a film festival), ‘Event’ (represents an event which has three entities: ‘Glastonbury Festival 2017’, ‘Ham Festival 2017’ and ‘London Movie Festival 2017’ and an object property which is ‘hasEvent’). ‘Festival’ is a class describing a festival. This class has a subclass called ‘FilmFestival’. The ‘Festival’ class also has an object property, ‘hasEvent’.

Also for an equivalentClass, ‘foaf:Person’ is equal to the class ‘Person’. The ‘Genre’ class also has an object property ‘hasGenre’. The ‘Movie’ class represents a movie. It has an instance: ‘movie1’. It also has the following object properties: ‘hasActor’, ‘hasActress’, ‘hasDirector’, ‘hasGenre’, ‘hasMaleActor’, ‘isActorIn’ and ‘isDirectedBy’. The class also has a subclass called ‘DramaMovie’. The ‘Person’ class represents a person. This class has two subclasses, which are ‘Actor’ and ‘Director’. The ‘Person’ class has data type properties, which are: ‘birthDate’, ‘firstname’, ‘isAlive’ and ‘lastname’. The ‘Place’ class represents a place. This class also has data type properties, which are the following: ‘maximumAttendeeCapacity’ and ‘smokingAllowed’. There is also an object property, the ‘adjacentPlace’. The class ‘schema:Festival’ is the same as the class ‘Festival’ and ‘schema:Movie’ is the same as the class ‘Movie’. The VOWL presentation of the ontology is illustrated in Figure 3.

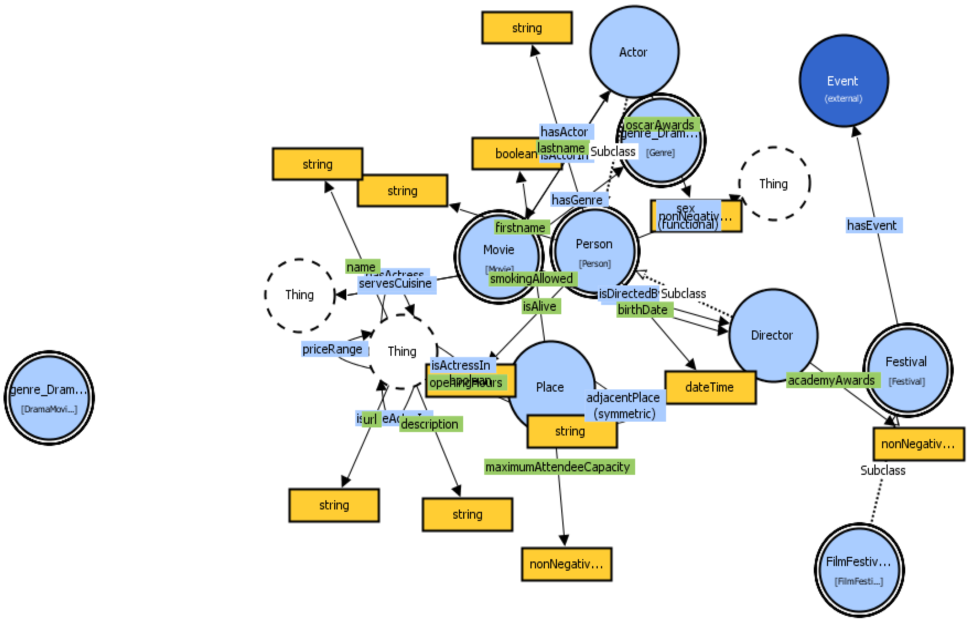

Figure 4. *Filmfestival ontology VOWL representation*

In the following, I will present the UML conversion of the FilmFestival ontology. The ontology contains object properties, which are converted as UML properties. In some places, it is an array, because for these I assumed one or more connections. There are also class-subclass relationships between OWL classes, and these remain class-subclass relationships in UML as well.

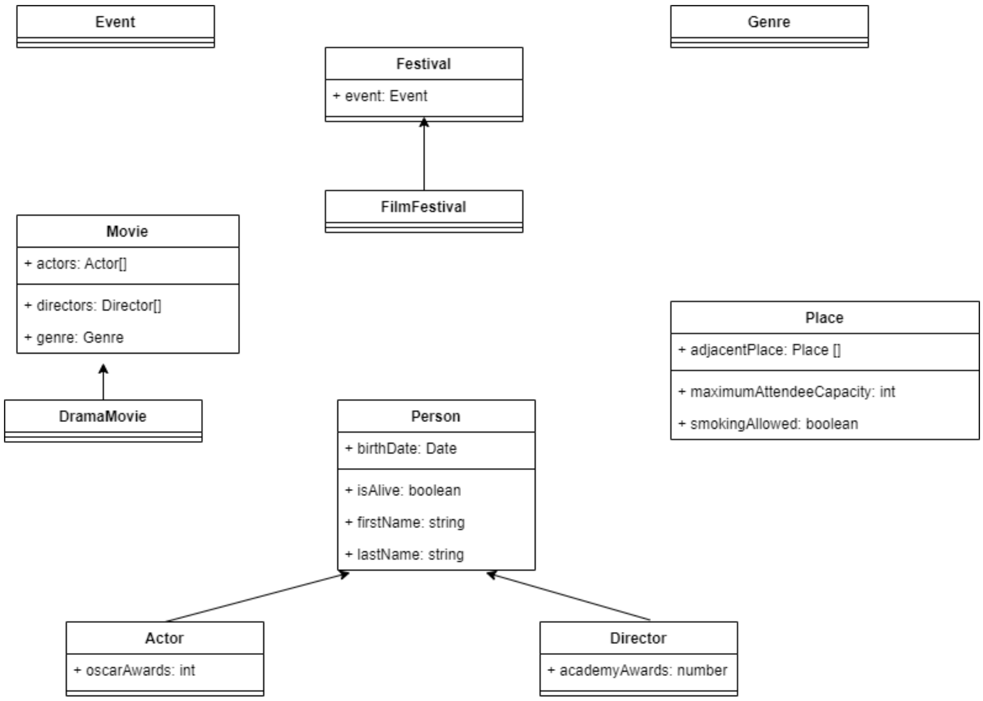

Figure 5. *Filmfestival ontology UML representation*

2.4. Funny salad ontology

The funny salad ontology [8] is an ontology representing a salad bar. The ontology represents different salad types. It represents the country and food. In meals, only salads. Inside the salad are ‘funnySalad’ and ‘namedSalad’. The system contains the following salad ingredients: ‘crackedWheat’ and ‘oliveOil’. The salad can have the following toppings: ‘saladTopping’, ‘herbTopping’ or ‘nutTopping’. It also includes the system spices, which are: ‘cinnamon’, ‘pepper’ and ‘salt’.

The system contains an object property, but no data type property. The following object contains properties: ‘hasCountryOfOrigin’, ‘hasIngredient’, ‘hasSpices’, ‘hasTopping’, ‘hasVegetables’, ‘isCountryOfOrigin’, ‘isIngredientOf’ and ‘isSpicesOf’. The VOWL presentation of the ontology is illustrated in Figure 4.

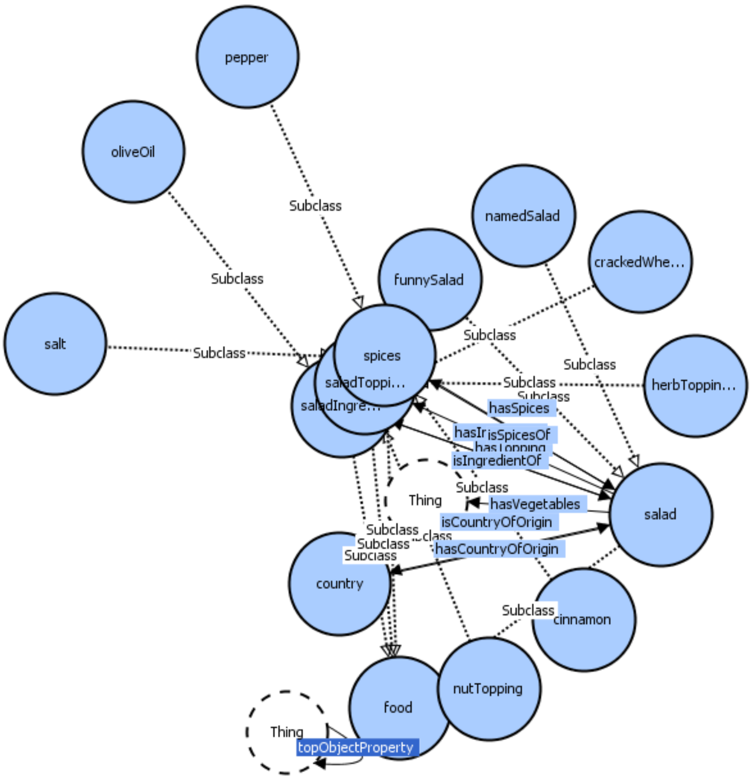

Figure 6. *Funny salad ontology VOWL representation*

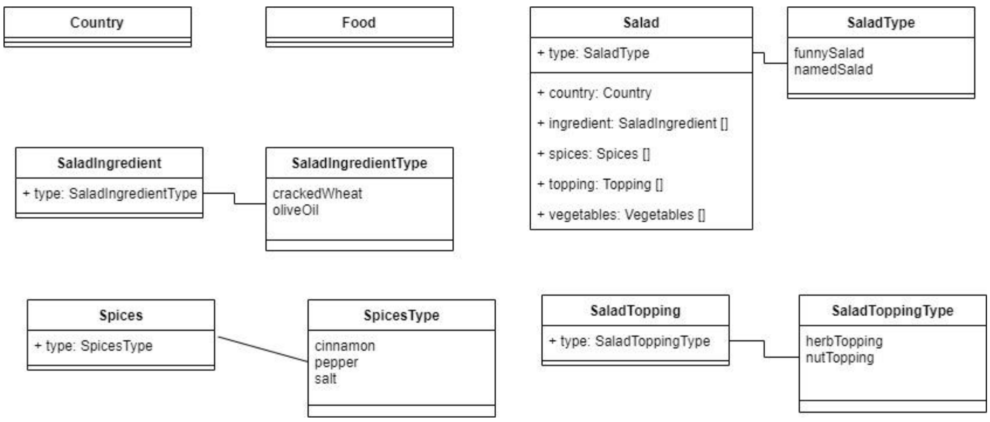

Figure 7. *Funny salad ontology UML representation*

The Funny salad ontology contains many classes and object properties. It does not contain a datatype property. During the conversion to UML, I converted certain OWL subclasses to enum because, in my opinion, this is a better conversion because

UML contains fewer classes. If I had created classes instead of enums, they would have been empty classes, they would not have contained data members. In my opinion, the UML model could be expanded with a few more parts, such as the class-subclass relationship of the Food-Salad class.

3. Evaluation of the ontologies

This chapter evaluates the presented ontological systems. I have already presented the metrics in the publication [9]. The following metrics are used in this article:

- WMC (Weighted Methods per Class) and Average WMC
- DIT (Depth of Inheritance) and Average DIT
- NOC (Number Of Childrens) and Average NOC
- DAC and Average DAC
- OA1
- OA2

3.1. Companies ontology

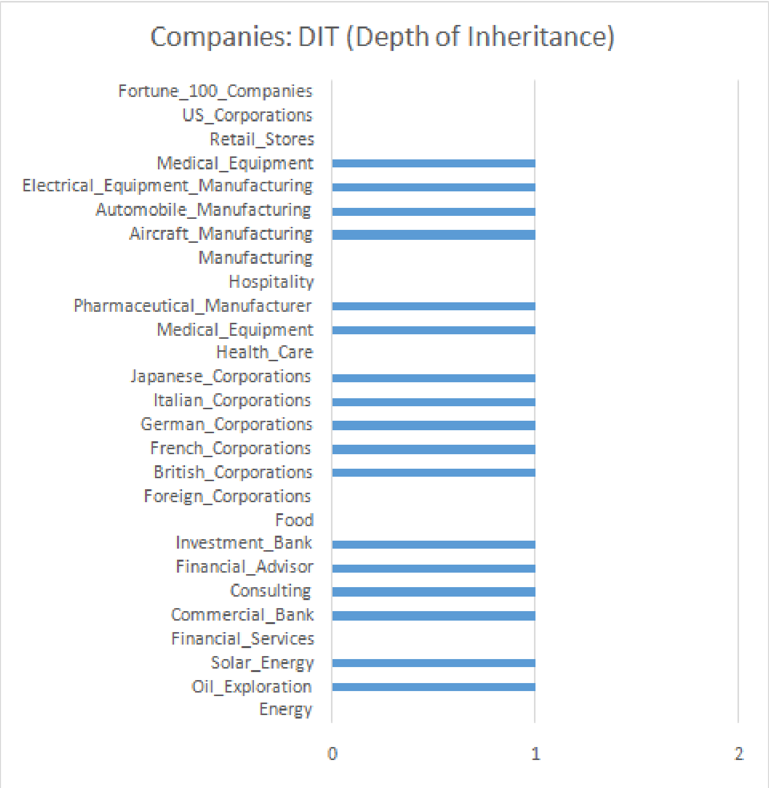

Figure 8. *Companies: DIT (Depth of Inheritance)*

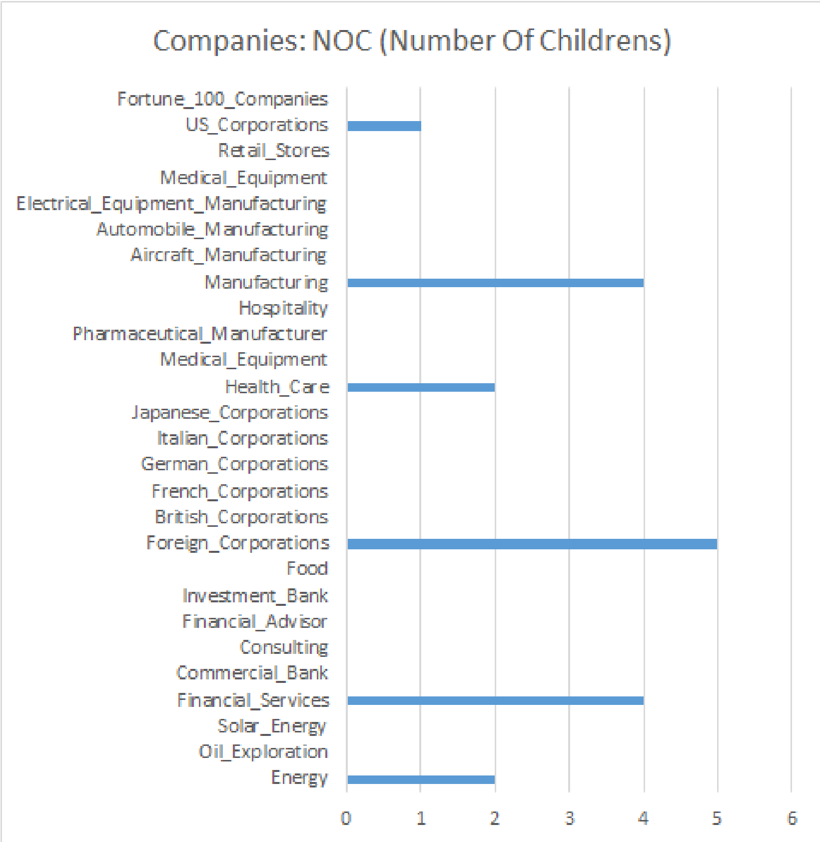

Figure 9. *Companies: NOC (Number Of Childrens)*

The Companies ontology has no property, so the WMC (Weighted Methods per Class) and Average WMC (Weighted Methods per Class) values were not calculated. Companies: DIT (Depth of Inheritance) values are 0 or 1, respectively, which means that the class inheritance level is 1. The DIT values are presented in Figure 5. Average DIT (Depth of Inheritance) is 0.62963, not a high-level ontology that would branch class-subclass hierarchy considering. According to the Companies NOC (Number Of Childrens) diagram (Figure 6.), many classes have no children. However, there is also a class that has 5 children. Average NOC (Number Of Childrens) is 0.666666667. OA1 value (Total number of classes) is 27, OA2 value (Total number of inheritance hierarchies) is 2.

3.2. Filmfestival ontology

The Filfestival WMC (Weighted Methods per Class) values range from 0 to 7. This means that some classes have no properties, other classes have 3, 4, or 7 properties. Average WMC (Weighted Methods per Class) is 2.5. The WMC diagram is illustrated in Figure 7.

The DIT (Depth of Inheritance) values are 0 and 1 for each class. This means that those with a value of 0 are at the top level, and those with a value of 1 are one below them. Average DIT (Depth of Inheritance) is 0.4. The DIT values are presented in Figure 8.

The NOC (Number Of Childrens) values are between 0 and 2, that is how many children each class has. According to this value, there are not many subclasses in the system. Average NOC (Number Of Childrens) is 0.6. The NOC diagram is illustrated in Figure 9.

Average DAC value is 1.7. The DAC diagram is illustrated in Figure 10. The OA1 (Total number of classes) value is 10, and the OA2 (Total number of inheritance hierarchies) value is 1. The ANA (Average Number of Ancestors) value is 1.

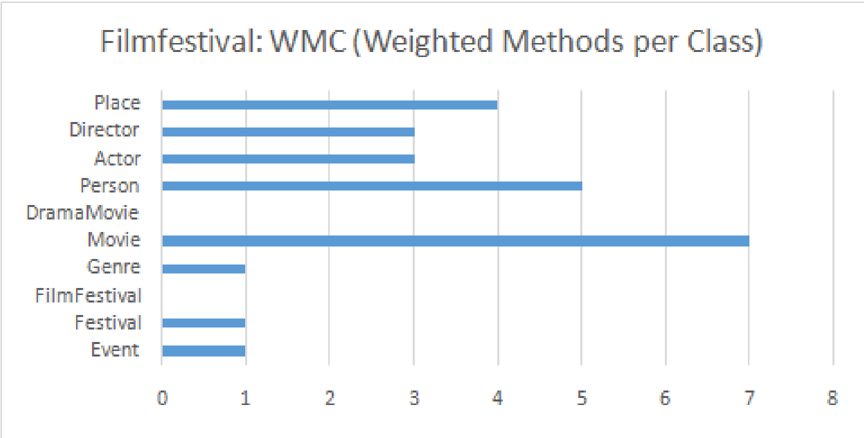

Figure 10. *Filmfestival: WMC (Weighted Methods per Class)*

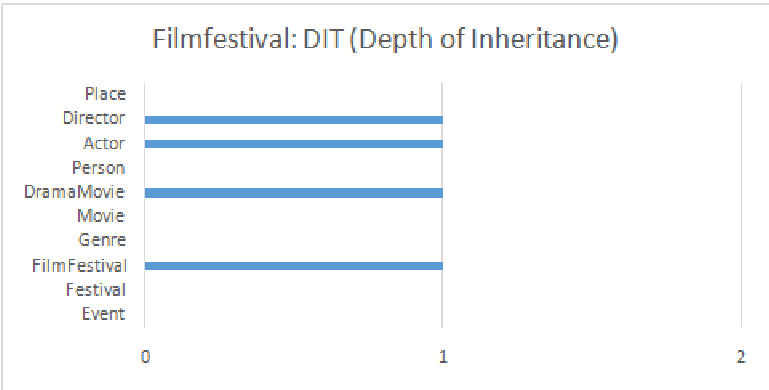

Figure 11. *Filmfestival: DIT (Depth of Inheritance)*

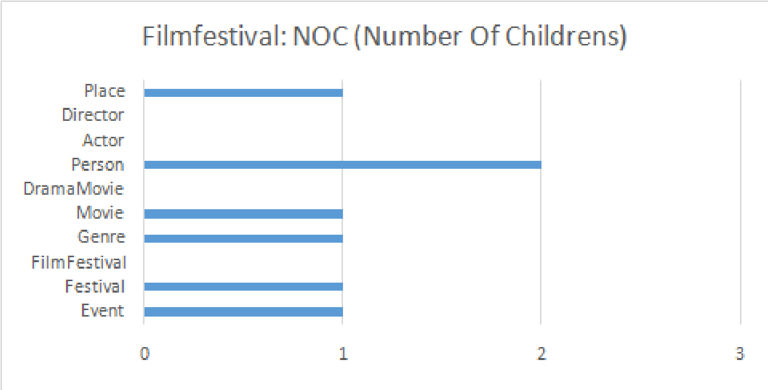

Figure 12. *Filmfestival: NOC (Number Of Childrens)*

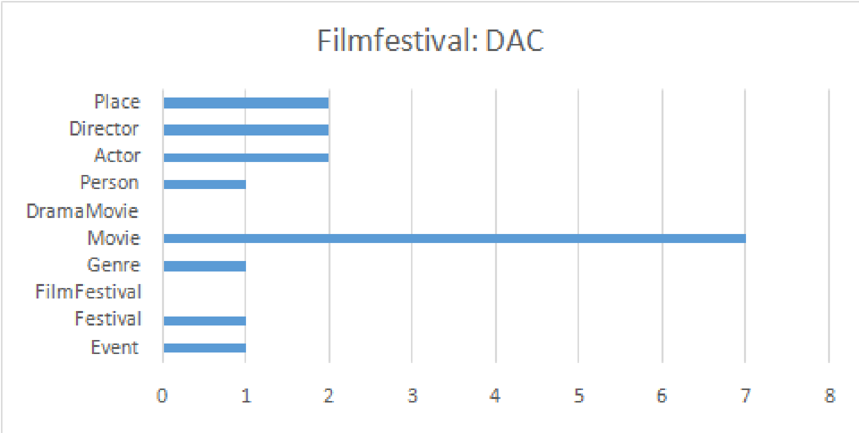

Figure 13. *Filmfestival: DAC*

3.3. Funny salad ontology

The WMC (Weighted Methods per Class) values of the Funny salad ontology are between 0 and 8, which means that the number of properties of each class varies. Average WMC (Weighted Methods per Class) is 1. The WMC diagram is illustrated in Figure 14. The DIT (Depth of Inheritance) values are between 0 and 2, many classes have a value of 2, so they are at level 2, they have no children. Average DIT (Depth of Inheritance) is 1.466666667. The DIT diagram is illustrated in Figure 15. The NOC values are between 0 and 3, that is how many children each class has. Its average value is 0.666666667. The NOC diagram is illustrated in Figure 16. The DAC values are between 0 and 8. This means that the number of object properties for most classes is 0, there is one class, ‘salad’, where this value is 8, which can be considered a main class in the system. The value of Average DAC is 1, on average this is how many object properties a class has. The DAC diagram is illustrated in Figure 17. The OA1 (Total number of classes) value is 15, while the OA2 (Total number of inheritance hierarchies) value is 2.

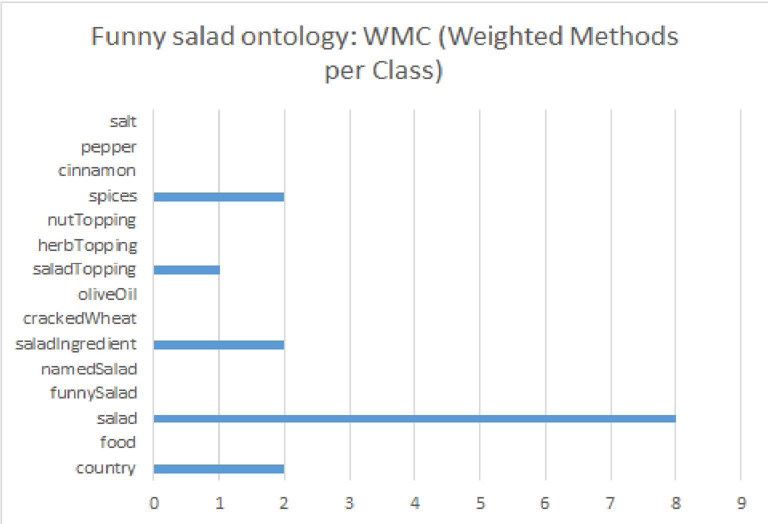

Figure 14. *Funny salad ontology: WMC (Weighted Methods per Class)*

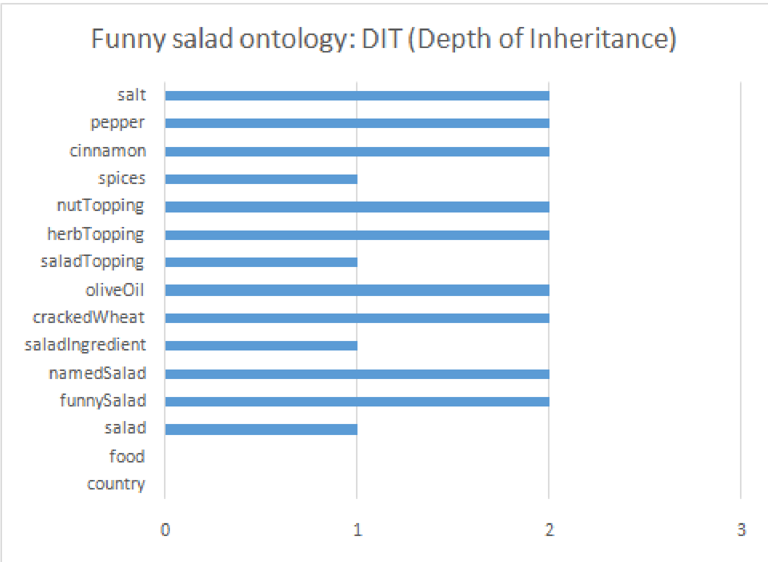

Figure 15. *Funny salad ontology: DIT (Depth of Inheritance)*

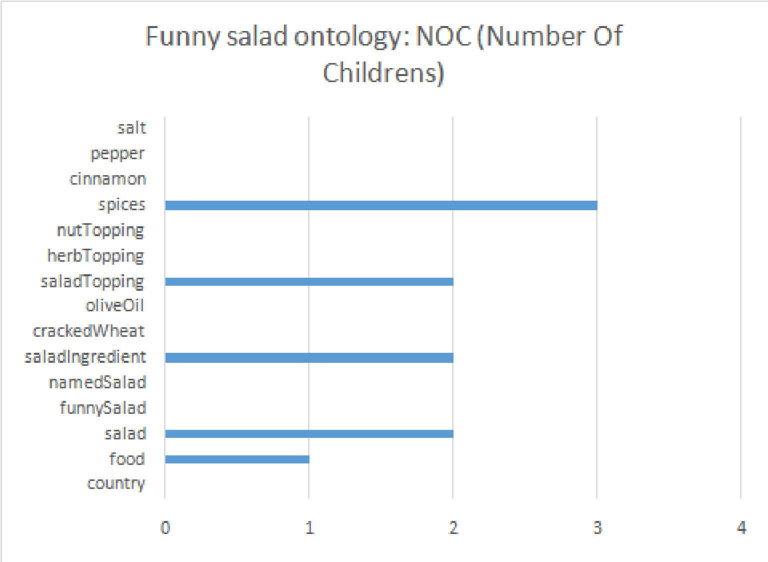

Figure 16. *Funny salad ontology: NOC (Number Of Childrens)*

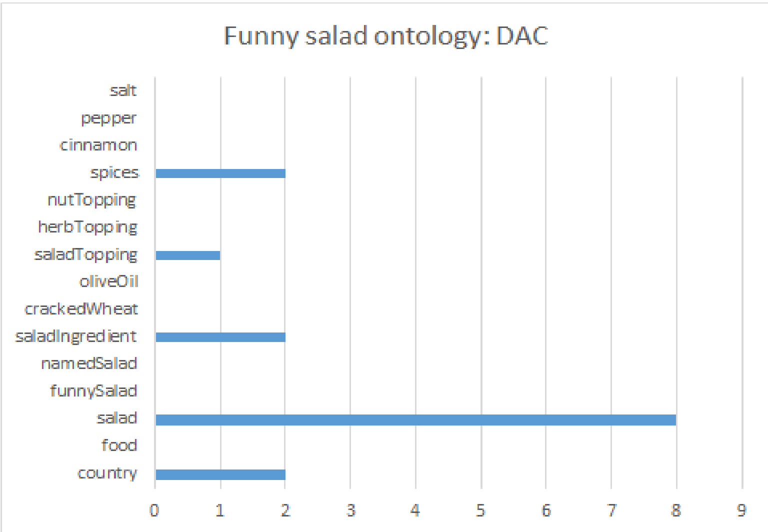

Figure 17. *Funny salad ontology: DAC*

3.4. Summary result

In this chapter, there is a summary evaluation of the presented ontologies. The summary diagram is illustrated in Figure 18.

The average WMC values are low for all three ontologies, this value was the highest for the Filmfestival ontology. The average DIT is also low, as are the NOC and DAC averages. The number of classes is high, the Companies ontology contains the most classes. The average number of inheritance is low for all ontologies. This means that ontologies contain many classes, but there is little class-subclass organization, and the classes have few properties.

The Companies ontology has low DIT and NOC values, which shows a relatively simple and linear structure. The Filmfestival ontology shows varying degrees of inheritance depth and number of children, which results in a more varied and balanced structure. The low WMC and moderate DIT of the Funny Salad ontology suggests that the ontology has a simple structure but little hierarchy.

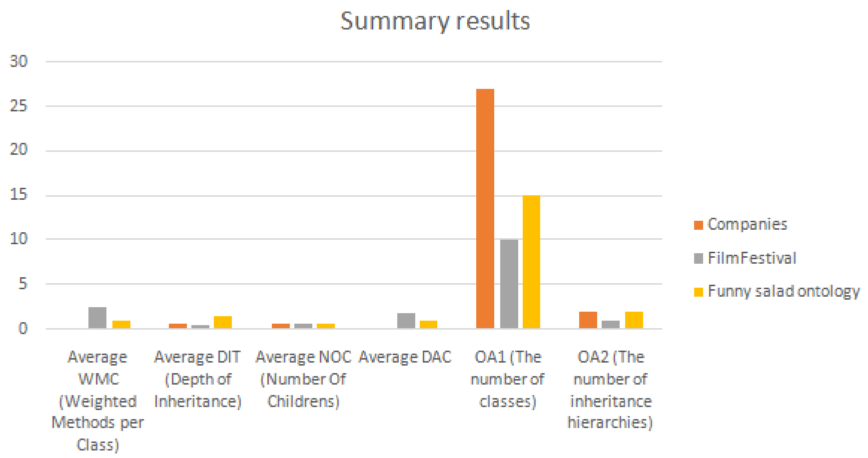

Figure 18. *Summary results*

4. Conclusions and future work

The article presents three ontologies and evaluates them from a UML metric point of view. These three ontologies are the followings: Companies, Filmfestival and the Funny salad ontology. The Companies ontology is an ontology describing a company. Filmfestival is a system describing a film festival, while Funny salad ontology describes a salad bar. After the presentation of the systems, their analysis from a metric point of view was also presented. The systems contain many classes, but few properties, and the class-subclass hierarchy is not strong either. Future work is the evaluation of additional ontological systems and the adaptation of new metrics.

Acknowledgement

Supported by the ÚNKP-22-3 New National Excellence Program of the Ministry for Culture and Innovation from the source of the National Research, Development and Innovation Fund.

References

[1] Baroni, A. L. & Abreu, F. B. (2002, October). Formalizing object-oriented design metrics upon the UML meta-model. In: *Brazilian Symposium on Software Engineering*, Gramado-RS, Brazil.

[2] Chen, Y., Boehm, B. W., Madachy, R. & Valerdi, R. (2004, August). An empirical study of eServices product UML sizing metrics. In: *Proceedings. 2004 International Symposium on Empirical Software Engineering, 2004*. ISESE'04, pp. 199–206, IEEE. https://doi.org/10.1109/ISESE.2004.1334907

[3] Yi, T., Wu, F. & Gan, C. (2004). A comparison of metrics for UML class diagrams. *ACM SIGSOFT Software Engineering Notes*, 29 (5), pp. 1–6. https://doi.org/10.1145/1022494.1022523

[4] Antoniou, G. & Harmelen, F. V. (2004). Web ontology language: Owl. In: *Handbook on ontologies*. Springer, Berlin, Heidelberg, pp. 67–92. https://doi.org/10.1007/978-3-540-24750-0_4

[5] Protégé: https://protege.stanford.edu/, accessed: 29. 11. 2022.

[6] *Companies owl. https://github.com/detnavillus/rdf-owl-ontologies.git, accessed: 29. 11. 2022.*

[7] *filmfestival owl.* https://github.com/thodoris/FilmFestival-OWL-Ontology.git, accessed: 29. 11. 2022.

[8] *funny-salad-ontology owl*. https://github.com/basselkassem/funny-salad-ontology.git, accessed: 2022. 11. 29.

[9] Agárdi, A. (2022). Ontology metrics as UML metrics aspect. *Production Systems and Information Engineering*, 11 (3), pp. 128-139 https://doi.org/10.32968/psaie.2023.3.10