DBFOschemafy – A DB-Driven, BFO annotated Ontology Generator for Streamlined OWL

Edit Hlaszny, PhD
Dr Hlaszny Bioystems Engineering
Mail: edit@edithlaszny.eu
http://www.edithlaszny.eu/
Phone: +36 30 3116516

Abstract

DBFOschemafy is an advanced Java-based ontology engineering tool that leverages a relational database for input, overcoming the scalability limitations of its predecessor, OWLschemafy (http://www.edithlaszny.eu/ontology/OWLschemafy/), which relied on text files. This approach enables efficient management and manipulation of large-scale ontologies. DBFOschemafy generates OWL ontologies with comprehensive annotations: Class entities are annotated in English, German, and Dutch, and full BFO 1.0 [1] annotations provide a robust and standardized framework for understanding the ontological relationships between entities.

Keywords

Ontology Development, OWL-Generator, BFO Annotation, Relational Database, Java Application.

1. Introduction

1.1 The Significance of Ontologies in the Digital Age

In today's data-rich world, the ability to effectively organize and represent knowledge is crucial. Ontologies, with their formal and structured representation of concepts and their relationships, play a pivotal role in enabling knowledge sharing, interoperability, and intelligent data analysis.

This chapter introduces DBFOschemafy, an innovative ontology engineering tool that leverages a relational database (RDB) to streamline the creation of OWL [3][4][5] ontologies.

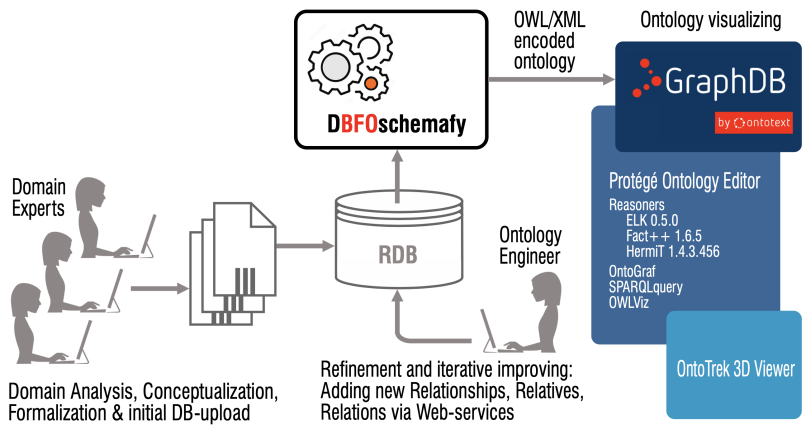


Figure 1: DBFOschemafy is an innovative software tool designed to streamline the process of ontology development [6][7].

DBFOschemafy is an innovative software tool designed to streamline the process of ontology development. It leverages a relational database (RDB) for efficient data management and manipulation, overcoming the limitations of traditional text-based approaches. This database-driven approach allows for the creation, management, and refinement of large-scale ontologies with greater ease and scalability.

The tool incorporates a collaborative workflow involving domain experts and ontology engineers. Domain experts contribute their knowledge during the initial stages of domain analysis, conceptualization, and formalization.

Ontology engineers utilize DBFOschemafy to construct OWL ontologies [8] based on the information gathered from domain experts. The tool provides a user-friendly interface for defining classes, properties, and relationships within the ontology. DBFOschemafy ensures that the generated ontologies adhere to the Basic Formal Ontology (BFO) standard, enhancing their consistency and interoperability.

The developed ontologies can be exported in OWL/XML, for use in different applications. Additionally, the tool integrates with a range of ontology visualization and reasoning tools, such as GraphDB, Protégé, and OntoTrek 3D Viewer, enabling users to explore, analyze, and reason with the created ontologies.

DBFOschemafy supports an iterative development process, allowing for continuous refinement and improvement of the ontology. Users can add new relationships, refine existing definitions, and incorporate new knowledge through web-based services.

In summary, DBFOschemafy is a powerful tool that streamlines ontology development by combining the efficiency of a relational database with the flexibility and expressiveness of OWL. It empowers domain experts and ontology engineers to collaborate effectively and create high-quality, BFO-compliant ontologies for a wide range of applications.

2. Adding new relationships, relatives, relations via Web-services

2.1 Dedicated Web services are used to modify and extend the initial DB content

- **Class Annotation Control (CAC):** This service enables the modification or creation of annotations associated with classes within the ontology. Furthermore, it facilitates the management of database update and insert commands generated during these modifications.
- **Class Definition Service (CDS):** This service empowers users to transform the entire OWL class structure, including modifications to subclass, superclass, and disjoint class relationships. The subsequent generation of individuals within the ontology is then based on this revised class structure, ensuring consistency across the entire knowledge representation.
- **Data Property Definition (DPD):** This service enables the creation of data properties and the specification of their attributes, such as subproperty/superproperty relationships, property annotations, and data property types. These data properties can then be assigned to the subject and object entities within the defined triplets.
- **Header Annotation Control (HAC):** This service facilitates the optional supplementation of existing header annotations with properties derived from external ontologies, such as those defined in the Dublin Core (dc), Friend-of-a-Friend (foaf), or other relevant vocabularies.
- **Object Property Definition (OPD):** This service allows for the creation of new object properties. For each newly created object property, users can specify attributes such as super object properties, inverse object properties, and associated object annotation properties, including the annotation itself, its type, and language.
- **Triplet Definition (TD):** This service provides a mechanism for defining the subject-predicate-object relationships within the ontology. Users can specify the entities involved in each triplet, identify the associated data properties, and define the corresponding data property values.

While the maintenance Web services are not included in the DBFOschemafy software product, two of the most commonly used services (CAC and OPD) are presented here for illustrative purposes.

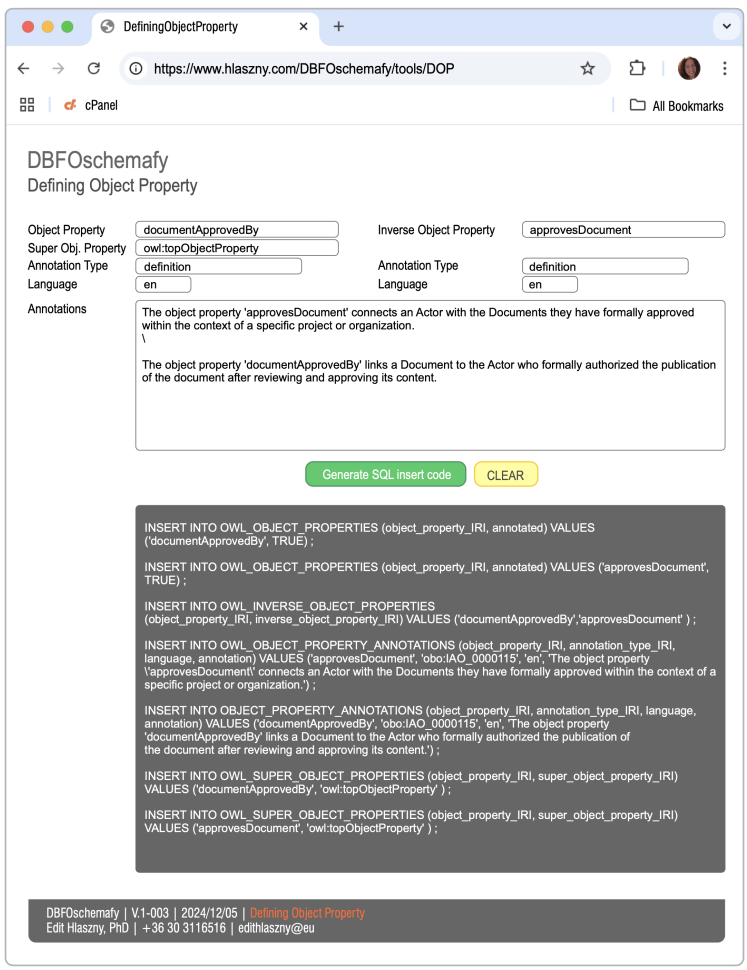


Figure 2: The *Object Property Definition* Web service.

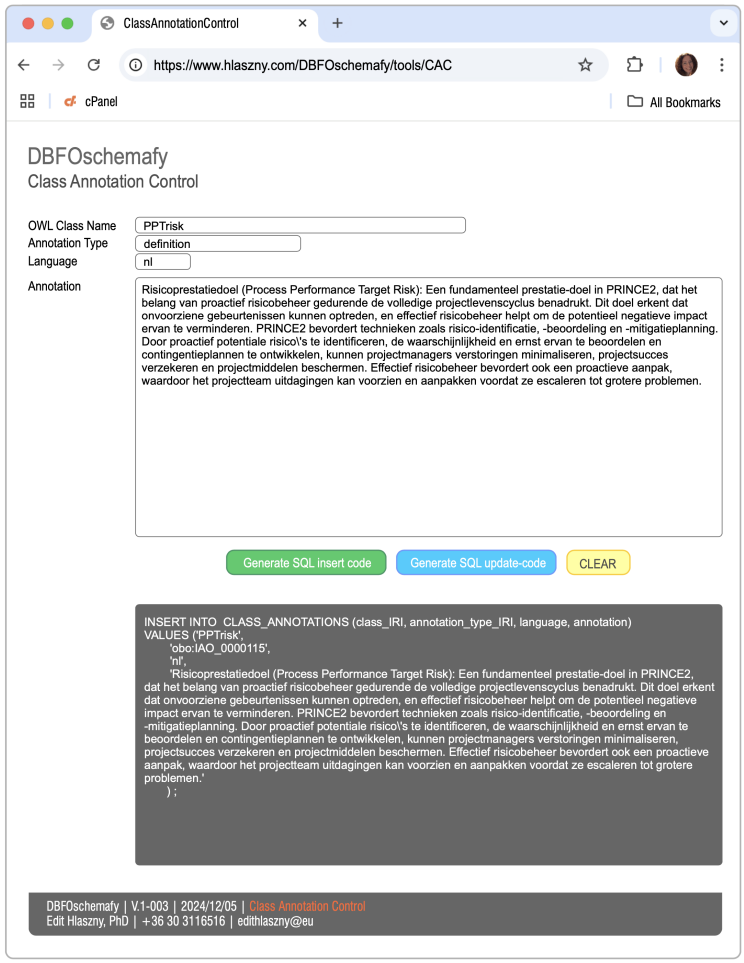


Figure 3: The *Class Annotation Control* Web service.

2.2 Class annotations

Class annotations are provided in three languages, further enriched by a referential classification based on BFO 1.0 (Basic Formal Ontology).

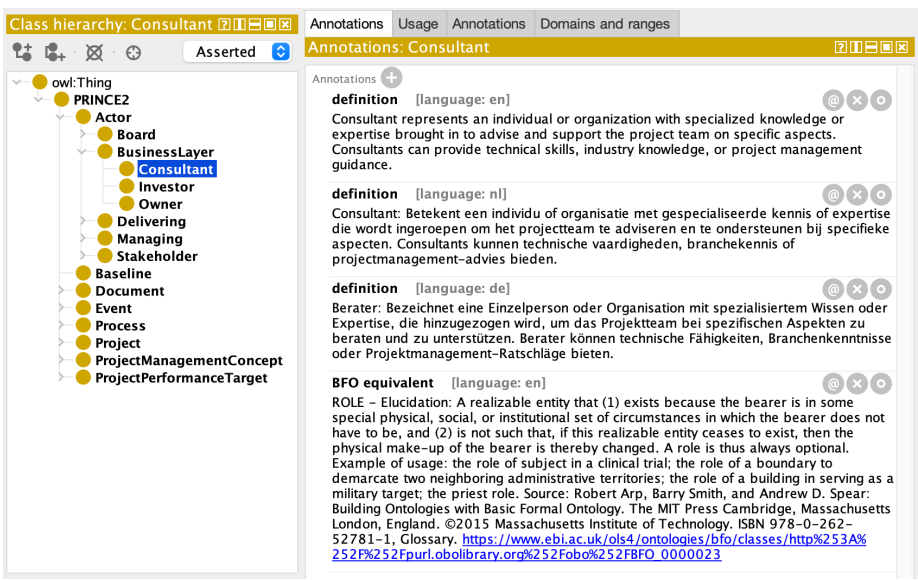
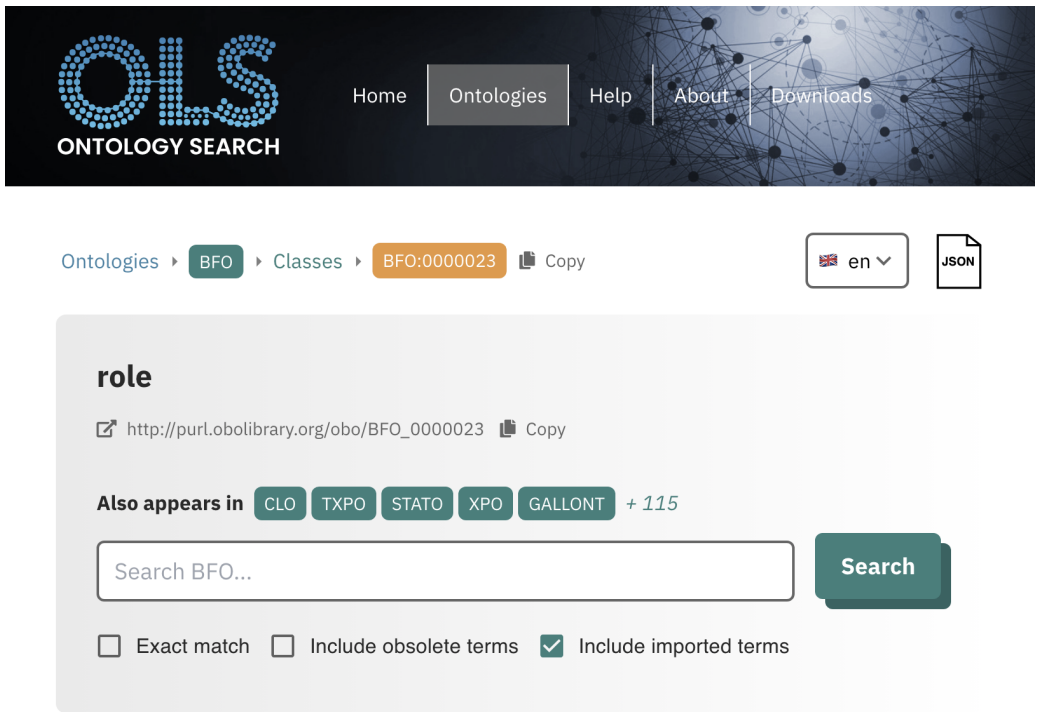


Figure 4: Class Annotation and BFO reference

As illustrated in Figure 4, each OWL class in the ontology is linked to the corresponding BFO entity, with further details available through the EMBL-EBI OLS. The EMBL-EBI OLS (European Molecular Biology Laboratory, European Bioinformatics Institute: Ontology Lookup Service) provides further details on the referenced BFO entity, as depicted in Figure 5:



▼ Class Information

BFO CLIF specification label
Role

BFO OWL specification label
role

editor note
BFO 2 Reference: One major family of examples of non-rigid universals involves roles, and ontologies developed for corresponding administrative purposes may consist entirely of representatives of entities of this sort. Thus 'professor', defined as follows,b instance_of professor at t =Def. there is some c, c instance_of professor role & c inheres_in b at t.denotes a non-rigid universal and so also do 'nurse', 'student', 'colonel', 'taxpayer', and so forth. (These terms are all, in the jargon of philosophy, phase sortals.) By using role terms in definitions, we can create a BFO conformant treatment of such entities drawing on the fact that, while an instance of professor may be simultaneously an instance of trade union member, no instance of the type professor role is also (at any time) an instance of the type trade union member role (any more than any instance of the type color is at any time an instance of the type length).If an ontology of employment positions should be defined in terms of roles following the above pattern, this enables the ontology to do justice to the fact that individuals instantiate the corresponding universals – professor, sergeant, nurse – only during certain phases in their lives.

elucidation
b is a role means: b is a realizable entity & b exists because there is some single bearer that is in some special physical, social, or institutional set of circumstances in which this bearer does not have to be& b is not such that, if it ceases to exist, then the physical make-up of the bearer is thereby changed. (axiom label in BFO2 Reference: [061-001])

Figure 5: **EMBL-EBI OLS** BFO reference

The database structure of the DBFOschemafy is depicted on the in Figure 6. The design was made with the Exterprise Architect modelling tool.

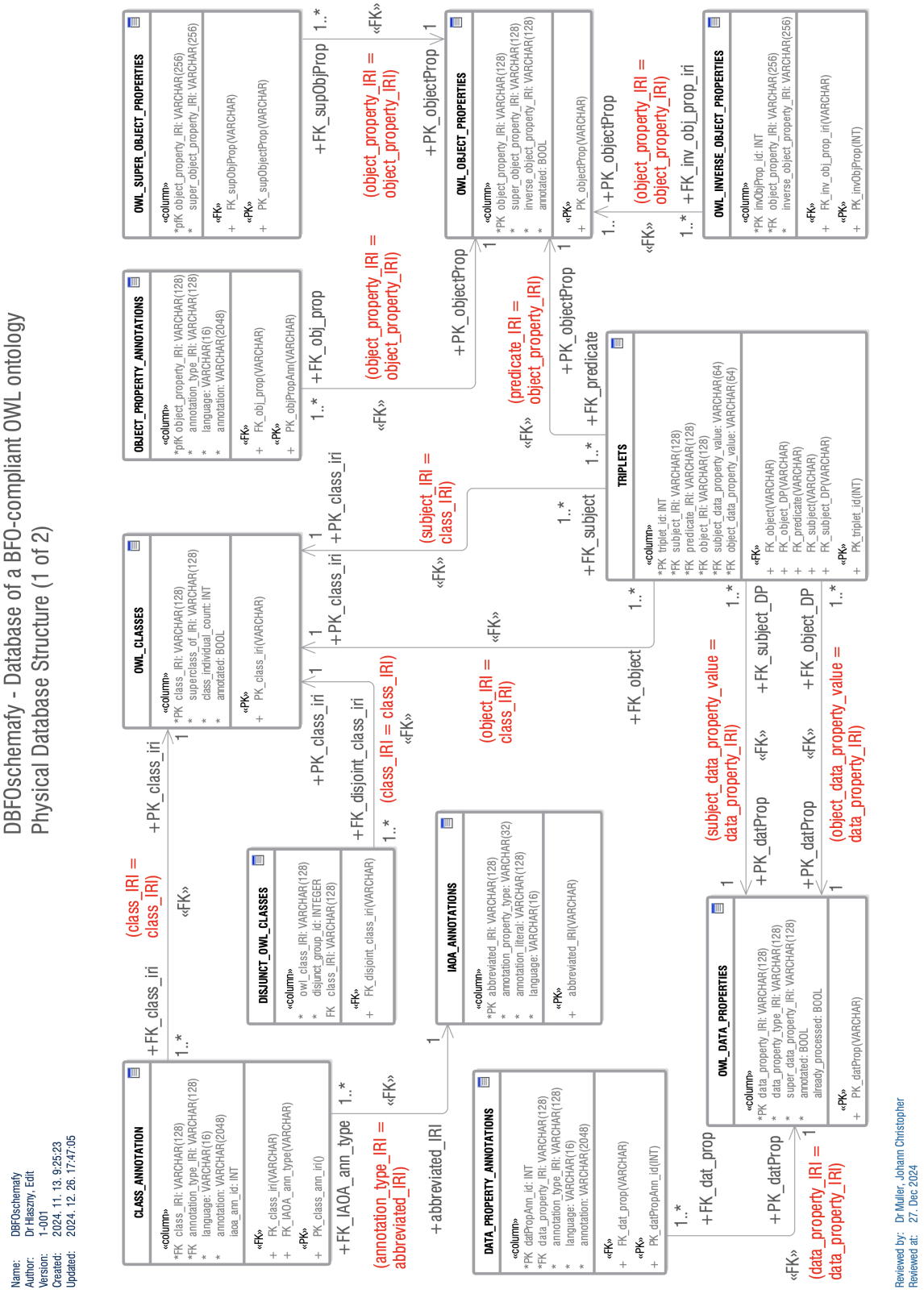


Figure 6: DBFOschemafy Database Structure (1)

The DBFOschemafy database structure includes four independent tables that store information such as ontology header and trailer data, as well as elucidations of BFO annotations, as depicted in Figure 7.

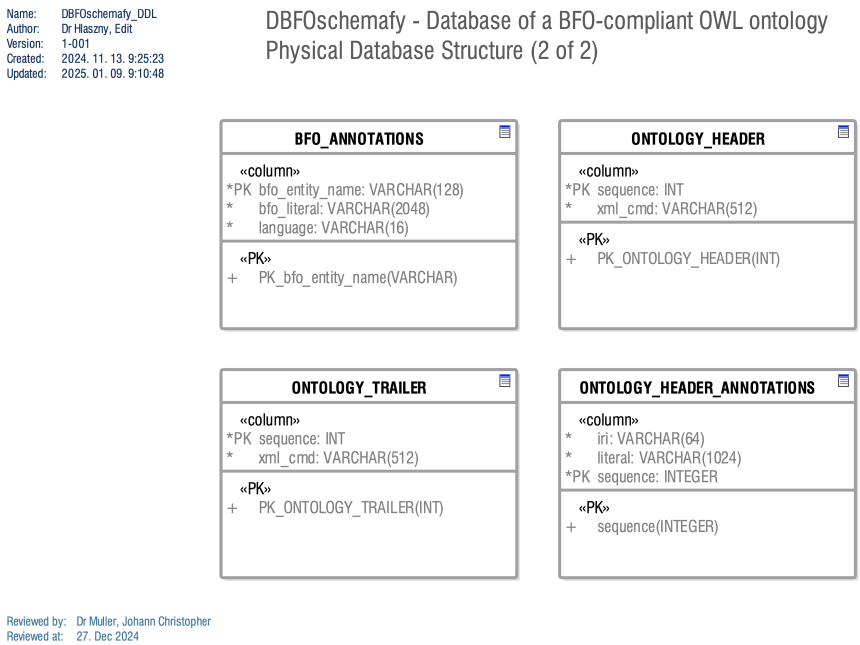


Figure 7: DBFOschemafy Database Structure (2)

3. Future Directions: Enhancing Development with Description Logics

The ongoing development of DBFOschemafy aims to incorporate more advanced features. A key step is the introduction of Description Logics [2] (DLs) to enhance the expressivity and formal rigor of the generated ontologies. The initial focus will be on the Attributive Language with Concept Negation ($\mathcal{ALC}$), a foundational Description Logic. However, depending on the complexity of the challenges encountered during development, the implementation may be extended to support the more expressive $\mathcal{SROIQ}$ Description Logic.

DLs provide a structured and well-defined framework for representing knowledge. As a subset of First-Order Logic (FOL), DLs offer a balance between expressivity and computational tractability.

To support the integration of DLs, the following challenges need to be addressed:

- *User Interface for DL Constructs:* Developing user-friendly interfaces for handling DL symbols, including their insertion, editing, and display within the DBFOschemafy environment.
- *Basic Syntactic Checking:* Implementing basic syntactic checks to ensure the validity of DL expressions entered by users.
- *OWL Code Generation from DL Expressions:* Generating OWL code from the defined DL expressions, ensuring accurate translation and preserving the intended meaning.

By addressing these challenges, DBFOschemafy will evolve into a more powerful and sophisticated tool for building high-quality, formally grounded ontologies.

4. Software Background The software development environment consists of the following components:

- *Operating System:* macOS Monterey, (Version 12.5.1) running on an iMac (Retina 5k, 27-inch, Late 2015)
- *RDB:* MySQL Server (Version 8.0.30), MySQL Workbench (Version 8.0.31 build 2235049 CE (64 bits) Community)
- *Java Runtime:* Java™ SE Runtime Environment (build 13.0.2+8)
- *Java Virtual Machine:* Java HotSpot™ 64-Bit Server VM (build 13.0.2+8, mixed mode, sharing)
- *Integrated Development Environment:* Eclipse IDE for Java Developers (includes Incubating components), Version: 2021-12 (4.22.0), Build id: 20211202-1639
- *Protégé:* Open-source ontology editor and framework for building intelligent systems, Version 5.6.3

5. DBFOschemafy as a Stand-Alone Application

DBFOschemafy is not merely a theoretical concept but a fully functional Java application. To facilitate further research and development, all components of the system are made available to the research community, including the Java source code, data models, and example database content.

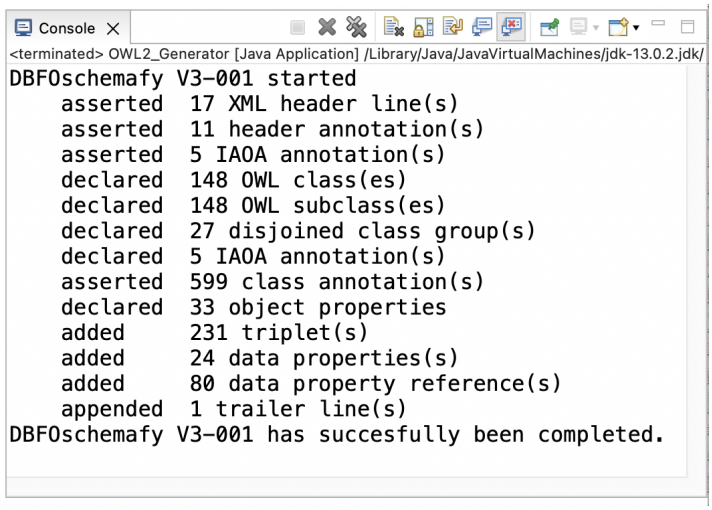


Figure 8: DBFOschemafy as a running stand alone application

6. References

[1] Robert Arp, Barry Smith, and Andrew D. Spear: Building Ontologies with Basic Formal Ontology. The MIT Press Cambridge, Massachusetts London, England. (2015). Massachusetts Institute of Technology. ISBN 978-0-262-52781-1.

[2] Bader, F., Horrocks, I., Lutz, C., Sattler, Uli. (2017). An Introduction to Description Logic. Cambridge University Press, ISBN 978-0-521-69542-8.

[3] Allemang, D., Hendler, J. A. (2012). Semantic web for the working ontologist effective modeling in RDFS and Owl. Dean Allemang, Hendler. J.A.

[4] Arp, R., Smith, B., Spear, A. D. (2015). Building ontologies with basic formal ontology. MIT Press

[5] Curé, O., Blin, G. (2015). RDF database systems: Triples Storage and SPARQL query processing. Morgan Kaufmann

[6] DuCharme, B. (2011). Learning Sparql: Querying and updating with SPARQL 1.1. O'Reilly

[7] Horridge, M. (2011). A Practical Guide To Building OWL Ontologies Using Protégé 4 and CO-ODE Tools. The University Of Manchester Press

[8] Owl. OWL - Semantic Web Standards. (n.d.) `https://www.w3.org`

Online Content

Any control and source data, extended data, supplementary information, and author details are available electronically.

Base URL: http://www.edithlaszny.eu/ontology/DBFOschemafy/
Present paper: paper/DBFOschemafy_V_3_001.pdf
LaTeX source: paper/DBFOschemafy_V_3_001.tex
Figures: paper/figures/
RDB content: RunnableEnv/*.sql
Java sources: eclipseWorkSpacePRINCE2/*
Javadoc: javadoc/index.html
OWL Ontology: RunnableEnv/resultOntology/DBFOprince2_ontology.owl
Author's data: authorsData/*.pdf

The author declares no conflicts of interest, including any competitive interests.
This paper has been typeset from a TEX/LATEX file prepared by the author.