

Foundational Ontology Interchangeability with the Repository of Ontologies for MULTiple USEs (ROMULUS)

by
Zubeida C. Khan

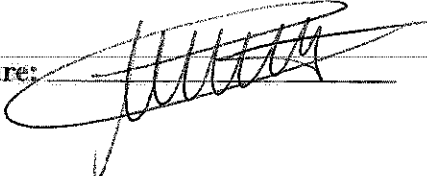
A dissertation submitted in fulfilment of the academic requirements for the degree of Master of
Science in Computer Science at the University of KwaZulu-Natal



Student Name: Zubeida C. Khan **Date:** 04 JULY 2013

Student number: 208509140 **Signature:** 

Supervisor Name: Dr. C. Maria Keet **Date:** 4 july 2013

Signature: 

Declaration 1 -Plagiarism

I, Zubeida Casmod Khan, declare that

1. The research reported in this thesis, except where otherwise indicated, is my original research.
2. This thesis has not been submitted for any degree or examination at any other university.
3. This thesis does not contain other persons data, pictures, graphs or other information, unless specifically acknowledged as being sourced from other persons.
4. This thesis does not contain other persons' writing, unless specifically acknowledged as being sourced from other researchers. Where other written sources have been quoted, then:
 - a) Their words have been re-written but the general information attributed to them has been referenced.
 - b) Where their exact words have been used, then their writing has been placed in italics and inside quotation marks, and referenced.
5. This thesis does not contain text, graphics or tables copied and pasted from the Internet, unless specifically acknowledged, and the source being detailed in the thesis and in the References sections.

Date 04 JULY 2013

Place PRETORIA

Signature 

Declaration 2 -Publications

DETAILS OF CONTRIBUTION TO PUBLICATIONS that form part and/or include research presented in this thesis (include publications in preparation, submitted, in press and published).

Publication 1

KHAN, Z., KEET, C. M. 2012. ONSET: Automated foundational ontology selection and explanation. *18th International Conference on Knowledge Engineering and Knowledge Management (EKAW'12)*. A. ten Teije et al. (Eds.). Springer, Lecture Notes in Artificial Intelligence LNAI 7603, 237-251. Oct 8-12, 2012, Galway, Ireland. Status: published (after peer-review)¹.

Publication 2

KHAN, Z., KEET, C. M. 2013. Toward semantic interoperability with linked foundational ontologies in ROMULUS. *Seventh International Conference on Knowledge Capture (K-CAP13)*. ACM proceedings. 23-26 June, Banff, Canada. (poster/demo) Status: published (after peer-review).

Publication 3

KHAN, Z., Keet, C.M. 2013. Addressing issues in foundational ontology mediation. *5th International Conference on Knowledge Engineering and Ontology Development, KEOD'13*. SciTePress. 19-22 Sep, Vilamoura, Algarve, Portugal. Status: in press (after peer-review).

Publication 4

KHAN, Z., Keet, C.M. 2013. The foundational ontology library ROMULUS. *3rd International Conference on Model & Data Engineering (MEDI'13)*. Springer, Lecture Notes in Computer Science LNCS. 25-27 Sep, Amantea, Calabria, Italy. Status: accepted (after peer-review).

Date 04 JULY 2013

Place PRETORIA

Signature 

Abstract

The notion behind foundational ontologies is to have a single foundational ontology to serve as a basis for providing high-level entities and relations that are common between all ontologies in order to facilitate interoperability among heterogeneous systems. However foundational ontologies alone do not suffice in solving the problem of interoperability, due to the fact that many foundational ontologies exist, each with conflicting philosophies. The WonderWeb Foundational Ontologies Library (WFOL) was envisioned to facilitate interoperability, but not implemented, possibly due to a lack of: ontology mediation (alignment, mapping and merging) techniques, documentation and comparisons between foundational ontologies and modularisation techniques. In order to solve this problem, three widely used foundational ontologies: DOLCE, BFO and GFO were selected and a web-based repository, ROMULUS was created. Ontology mediation was performed to assist in achieving foundational ontology interchangeability between the selected foundational ontologies. Modularity was performed to simplify ontologies in order to easily perform mediation and to create modules for specific functions. ROMULUS provides the user with access to: new foundational ontology modules, mappings between foundational ontologies, merged foundational ontologies, a higher level foundational ontology containing only the most general entities common to the three foundational ontologies and a method to assist the user with performing foundational ontology interchangeability. The new modules in ROMULUS (separate enduring/perdurable modules, OWL 2 profile modules, and more/less-detailed ontology modules) are useful when one wants to perform functionality specific to the module type. The mapping and merged ontologies, which may be used together with the method for performing foundational ontology interchangeability, allow a user to convert between the three foundational ontologies and to link an ontology using a particular foundational ontology to a different ontology that uses another foundational ontology, thereby achieving transparency. The higher level foundational ontology may assist in interoperability because it is a single ontology that encompasses entities that are common between all three foundational ontologies. ROMULUS has been evaluated in terms of its foundational ontology interchangeability, accuracy of alignments and by comparing it to other repositories. From the evaluations, we realised the following: While barely 50% of the participants agreed with the alignments, real disagreement was less than 10%; foundational ontology interchangeability may be achieved using the merged ontologies; ROMULUS offers advanced functionality for most criteria when compared to other repositories. Therefore there is reason to believe that ROMULUS does assist with foundational ontology interoperability.

Acknowledgement

Foremost, I would like to express my deepest gratitude to my supervisor Dr. C. Maria Keet for her continuous support on this thesis and in my academic pursuits. This thesis would not have been possible without her guidance, motivation, advice, and expertise. I would also like to thank the Centre for Artificial Intelligence Research (CAIR) for the funding provided to conduct this research. I am grateful for my visit to the Digital Enterprise Research Institute (DERI), partially funded by the European FP7 IRSES project Net 2 “A Network for Enabling Networked Knowledge”. My sincere thanks goes to the members of DERI for participating in the evaluations. Last but not least, I wish to thank Dr. Aidan Hogan from DERI for his suggestions on the evaluation design and for his assistance in conducting the evaluation.

Contents

List of Figures	viii
List of Tables	x
List of Abbreviations	xii
1. Introduction	1
1.1. Foundational ontologies and the Semantic Web	2
1.2. Problem statement	3
1.3. Research objectives and tasks	4
1.4. Structure of thesis	4
2. Literature Review	6
2.1. Foundational ontology philosophies and choices	6
2.2. Official foundational ontology deliverables	7
2.3. Comparison of popular foundational ontologies	12
2.4. Existing ontology repositories	16
2.5. Ontology modularisation	18
2.5.1. Evaluating modularity tools	20
2.6. Ontology mediation	21
2.7. Ontology metadata	23
2.8. Ontology browsing tools	23
2.8.1. Evaluating ontology browsing tools	26
2.9. Ontology view tools	27
2.9.1. Evaluation of ontology view tools	28
3. Materials and Methods	29
3.1. Methodology	29
3.2. Evaluation technique	31
3.3. Foundational ontologies for the repository	32

4. Ontology Mediation	34
4.1. Foundational ontology content comparison	35
4.1.1. Similarities and differences between DOLCE and BFO	35
4.1.2. Similarities and differences between BFO and GFO	36
4.1.3. Similarities and differences between GFO and DOLCE	36
4.2. Alignment	37
4.2.1. Accurate alignments	38
4.2.2. Approximate alignments	47
4.3. Mapping and Merging	49
4.3.1. Logical inconsistencies	50
4.3.2. A higher-level foundational ontology	59
4.3.3. FFO Mappings	61
4.4. Foundational ontology interchangeability method	61
5. Web-based ROMULUS	64
5.1. Requirements	64
5.1.1. Functional requirements	64
5.1.2. Non-functional requirements	65
5.2. Design	65
5.2.1. Meeting functional requirements	68
5.3. Implementation	69
5.4. ROMULUS's features	70
5.4.1. Browse ontologies	70
5.4.2. Ontology comparison	73
5.4.3. Search	73
5.4.4. Ontology views	73
5.4.5. Ontology mediation	75
5.4.6. Ontology selection	75
5.4.7. Ontology metadata	77
5.4.8. Downloads	79
6. Evaluation and discussion	85
6.1. Evaluating foundational ontology interchangeability	85
6.2. Evaluating ontological alignments by users	87
6.3. Evaluating functionality by comparison with other ontology repositories	88
6.4. Summary of evaluation	89
6.5. Discussion	91
7. Conclusions and future work	93
7.1. Future Work	94
7.2. Summary of contributions	95
Bibliography	96

A. Complete set of accurate alignments	104
B. Alignments from existing tools and documentations	122
B.1. H-Match's alignments	122
B.2. PROMPT's alignments	125
B.3. LogMap's alignments	126
B.4. YAM++'s alignments	127
B.5. HotMatch's alignments	128
B.6. Hertuda's alignments	129
B.7. Optima's alignments	131
B.8. GFO documentation's alignments	133
B.9. Temal et al.'s alignments	134
C. ROMULUS documentation	135

List of Figures

1.1. Using a foundational ontology to align entities from heterogenous ontologies. . .	2
2.1. The DOLCE taxonomy.	8
2.2. A portion of the OCHRE taxonomy.	8
2.3. The BFO taxonomy.	9
2.4. A portion of the SUMO taxonomy.	10
2.5. A portion of the GFO taxonomy.	10
2.6. The top level categories of YAMATO.	11
2.7. The core classes of GIST.	11
2.8. A summary of BFO metadata from SOCoP.	17
2.9. The layered architecture of COLORE.	18
2.10. Modularisation by axioms in Protégé.	20
2.11. The flow of the PROMPT algorithm.	22
2.12. An overview of the OMV model.	24
2.13. The OM ² R metadata model fields.	25
2.14. BFO in jOWL.	25
2.15. Mappings between OWL and ACE constructs from OWL verbalizer.	27
3.1. Flow of the materials and methods.	31
4.1. Inconsistent alignment due to the OWL <code>DisjointClasses</code> class axiom.	52
4.2. Annotating the 3D entity in FFO.	60
4.3. The FFO class taxonomy.	61
5.1. A conceptualisation of ROMULUS's front-end system.	66
5.2. The interaction of ROMULUS's components.	66
5.3. The ontology layers upon which FFO may be used.	67
5.4. ER diagram of ROMULUS's database regarding ONSET's saved data.	67
5.5. ER diagram of ROMULUS's database regarding ontology metadata and mediation.	68
5.6. The ontology browsing feature in ROMULUS.	70
5.7. Ontology alignment search.	73

5.8. Ontology metadata search.	74
5.9. Ontology view: DL view.	74
5.10. Ontology view: Natural language view.	75
5.11. Output from the new version of ONSET (version 2.0).	77
6.1. The three main entities in the new ontology.	86
6.2. Evaluating an ontological alignment.	87
6.3. Entity annotations that are not clearly defined.	88
A.1. Alignments between BFO and GFO ontologies.	104
A.2. Alignments between BFO and GFOBasic ontologies.	105
A.3. Alignments between DOLCE-Lite and BFORO ontologies.	105
A.4. Alignments between FunctionalParticipation and BFORO ontologies.	106
A.5. Alignments between BFORO and GFO ontologies.	107
A.6. Alignments between BFORO and GFOBasic ontologies.	108
A.7. Alignments between SpatialRelations and BFORO ontologies.	109
A.8. Alignments between TemporalRelations and BFORO ontologies.	110
A.9. Alignments between DOLCE-Lite and BFO ontologies.	110
A.10. Alignments between DOLCE-Lite and GFO ontologies.	111
A.11. Alignments between DOLCE-Lite and GFOBasic ontologies.	112
A.12. Alignments between FunctionalParticipation and BFO ontologies.	113
A.13. Class alignments between FunctionalParticipation and GFO ontologies.	113
A.14. Object property alignments between FunctionalParticipation and GFO ontologies.	114
A.15. Alignments between FunctionalParticipation and GFOBasic ontologies.	115
A.16. Alignments between SpatialRelations and BFO ontologies.	116
A.17. Alignments between SpatialRelations and GFO ontologies.	117
A.18. Alignments between SpatialRelations and GFOBasic ontologies.	118
A.19. Alignments between TemporalRelations and BFO ontologies.	119
A.20. Class alignments between TemporalRelations and GFO ontologies.	119
A.21. Object property alignments between TemporalRelations and GFO ontologies.	120
A.22. Alignments between TemporalRelations and GFOBasic ontologies.	121
C.1. ROMULUS's menu bar with different functions.	135
C.2. ROMULUS's home page.	135
C.3. ROMULUS's browse ontology page.	136
C.4. Browsing through the BFO ontology in ROMULUS.	136
C.5. The Software Engineering Properties comparison page.	136
C.6. Ontology view pages.	137
C.7. A table of ontological alignments.	137
C.8. Ontology alignment search.	138
C.9. Metadata list for BFO-Continuants.	139
C.10. Ontology metadata search.	139

List of Tables

2.1.	Comparison of ontological commitments for each foundational ontology.	14
2.2.	Comparison of representation languages and software engineering properties for each foundational ontology.	15
2.3.	An evaluation of the ontology browsing tools	26
4.1.	Comparison of manually performed alignment accuracies of the GFO documentation, related works, and ours, and aggregates for mappings.	39
4.2.	Comparison of alignment accuracies of the matching tools and aggregates for mappings.	40
4.3.	False positives caused by syntactic matching generated by the alignment tools; the terms in italics represent the strings that are common between aligned entities.	41
4.4.	Common alignments between DOLCE-Lite, BFO and GFO.	42
4.5.	Equivalence alignments between DOLCE-Lite and GFO ontologies.	43
4.6.	Equivalence alignments between BFORO and GFO ontologies.	44
4.7.	Equivalence alignments between DOLCE-Lite and BFORO ontologies.	45
4.8.	Transitivity in the alignments between the three foundational ontologies.	48
5.1.	A comparison of the accuracy of ontology selection by Group A and Group B.	77
5.2.	List of metadata for DOLCE-Lite.	80
5.3.	List of metadata for BFO.	81
5.4.	List of metadata for GFO.	82
5.5.	List of metadata for DOLCE-Lite-EL module; the criteria in boldfont is ROMULUS's new metadata criteria.	83
5.6.	List of metadata for BFOGFOMappings; the criteria in boldfont is ROMULUS's new mediation criteria.	84
6.1.	A comparison of alignment evaluation responses.	88
6.2.	Comparison of ROMULUS's features with those of other repositories.	90
B.1.	H-Match's equivalence alignments between DOLCE-Lite and BFO ontologies.	122
B.2.	H-Match's equivalence alignments between DOLCE-Lite and GFO ontologies.	123

B.3. H-Match’s equivalence alignments between BFO and GFO ontologies.	124
B.4. PROMPT’s equivalence alignments between DOLCE-Lite and BFO ontologies. .	125
B.5. PROMPT’s equivalence alignments between DOLCE-Lite and GFO ontologies. .	125
B.6. PROMPT’s equivalence alignments between BFO and GFO ontologies.	126
B.7. LogMap’s equivalence alignments between DOLCE-Lite and BFO ontologies. . .	126
B.8. LogMap’s equivalence alignments between DOLCE-Lite and GFO ontologies. . .	126
B.9. LogMap’s equivalence alignments between BFO and GFO ontologies.	127
B.10. YAM++’s equivalence alignments between DOLCE-Lite and BFO ontologies. . .	127
B.11. YAM’s equivalence alignments between BFO and GFO ontologies.	127
B.12. YAM++’s equivalence alignments between DOLCE-Lite and GFO ontologies. . .	128
B.13. HotMatch’s equivalence alignments between DOLCE-Lite and BFO ontologies. .	128
B.14. HotMatch’s equivalence alignments between DOLCE-Lite and GFO ontologies. .	129
B.15. HotMatch’s equivalence alignments between BFO and GFO ontologies.	129
B.16. Hertuda’s equivalence alignments between DOLCE-Lite and BFO ontologies. . .	129
B.17. Hertuda’s equivalence alignments between DOLCE-Lite and GFO ontologies. . .	130
B.18. Hertuda’s equivalence alignments between BFO and GFO ontologies.	130
B.19. Optima’s equivalence alignments between DOLCE-Lite and BFO ontologies. . .	131
B.20. Optima’s equivalence alignments between BFO and GFO ontologies.	131
B.21. Optima’s equivalence alignments between DOLCE-Lite and GFO ontologies. . .	132
B.22. Equivalence alignments between DOLCE-Lite and GFO ontologies from the GFO documentation [30].	133
B.23. Equivalence alignments between DOLCE-Lite and BFO ontologies from Temal et al.’s alignment [82].	134

List of Abbreviations

BFO	Basic Formal Ontology
C-OWL	Context Web Ontology Language
CLIF	Common Logic Interchange Format
COLORE	Common Logic Ontology Repository
DL	Description Logics
DOL	Distributed Ontology Language
DOLCE	Descriptive Ontology for Linguistic and Cognitive Engineering
GEM	General Extensional Mereology
GFO	General Formal Ontology
KIF	Knowledge Interchange Format
MAFRA	MApping FRAmework
OBO	Open Biological Ontologies
OCHRE	Object-centered high-level reference ontology
OMV	Ontology Metadata Vocabulary
OOR	Open Ontology Repository
OWL	Web Ontology Language
RO	Relational Ontology
ROMULUS	Repository of Ontologies for MULTiple USes

SAO	Subcellular Anatomy Ontology
SOCOP	Spatial Ontology Community of Practice
SUMO	Suggested Upper Merged Ontology
SUO-KIF	Standard Upper Ontology Knowledge Interchange Format
SWOOP	Semantic Web Ontology Editor
TONES	Thinking ONtologiES
WFOL	WonderWeb Foundational Ontology Library
YAMATO	Yet Another More Advanced Top-Level Ontology

Introduction

A foundational ontology is one which describes the general high-level entities that are common across all domains. Foundational ontologies serve as building blocks in ontology development. By using a foundational ontology, the developer has an idea of how to model the entities of a domain. The need for foundational ontology usage is increasing especially with the growth of the Semantic Web. In order to realise the functionality of the Semantic Web, interoperability is required among all systems. An important function of a foundational ontology is that it aims to assist in semantic interoperability among a number of systems. Other reasons to use foundational ontologies are that they may be used to align domain ontologies, speed up ontology development, assist in developing ontological applications, and improve the quality of the proposed system.

Foundational ontologies have been used to align domain ontologies e.g., biomedical ontologies [76]. By integrating different domain ontologies in a common foundational ontology, one is able to identify which entities are equivalent according to their classification in the foundational ontology. Using foundational ontologies in alignment also ensures that incorrect alignments are avoided. Fig. 1.1 illustrates how two entities from heterogenous domain ontologies may be aligned by using the DOLCE foundational ontology.

It has been proven in an empirical assessment by Keet [42] that ontology development efforts are improved when a foundational ontology is included. Since high-level entities are already defined and properly axiomatised in the ontology, the time taken in ontology development is decreased and the properly axiomatised entities leave less room for modelling errors.

Foundational ontologies are a dependency for applications such as: ontologies for natural language processing, the Semantic Web [41], database integration, and more. For instance, DOLCE [53] has been applied to database integration and information retrieval [22]. BFO [53] has been applied to natural language processing and database integration [77, 79]. Some applications of GFO [30] include domain specific semantic wikis [34], ontological foundation of conceptual modelling [28] and ontology modelling for software applications [33]. SUMO [64] has been used in knowledge reasoning [8] and natural language processing [65, 71].

Scientific ontologies such as those used in the biomedical [25, 11, 10], environment [59] and

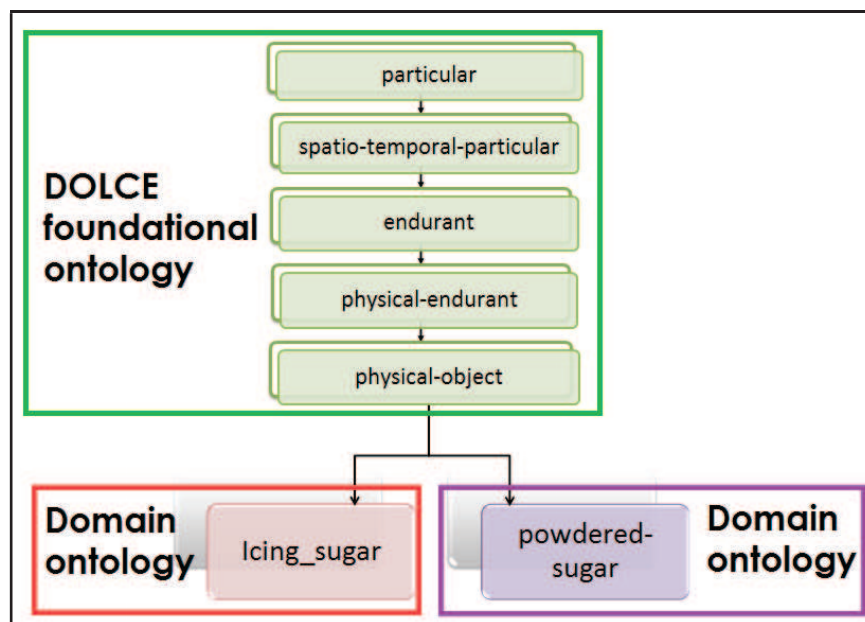


Figure 1.1.: Using a foundational ontology to align entities from heterogeneous ontologies.

life sciences [6, 44, 40] domains mainly use BFO and GFO. There is an increase in BFO usage due to the fact that the OBO foundry [79] has recommended that ontologies registered on the OBO foundry use BFO. However, some granularity issues are encountered when aligning life sciences ontologies with BFO [73]. DOLCE and SUMO have been applied to a variety of subject domains including engineering [37, 19], biomedical [3], government and military [74], landscape [78], and more. YAMATO [56] has been applied to several diverse projects such as a medical ontology [58], functional ontology [57] and an ontology of genomics [54].

1.1. Foundational ontologies and the Semantic Web

One of the main reasons to use foundational ontologies is to realise the functionality of the Semantic Web and its applications. The Semantic Web is an improved extension of the web which is meaningful to machines. The integrated web of linked data which make up the Semantic Web automates many operations. However, Semantic Web system developers use their preferred ontologies. The semantics of each foundational ontology differs causing a problem in semantic interoperability. Heterogeneous systems on the Semantic Web are restricted to committing to a single foundational ontology in order to promote interoperability. However, no single foundational ontology is used across all systems, therewith preventing interoperability. In order to enable such semantic operations, there is a need for infrastructure which integrates foundational ontologies. One such library was envisioned in the WonderWeb Foundational Ontologies Library (WFOL) [53]. However, this library was not implemented due to theoretical and usage gaps. According to their ontological commitments, Semantic Web applications should be able

to commit to different but systematically related modules of the envisioned library. The advantage of having such a library is that developers may use their preferred foundational ontology, and will be able to translate it to a common foundational ontology for Semantic Web systems. The underlying philosophy behind the WFOL, as described in WonderWeb deliverable D18 [53], involves creating a library of heterogeneous foundational ontologies, such as DOLCE, BFO and OCHRE [72], each with different underlying philosophies and ontological criteria thereby satisfying the different requirements and use cases of ontology developers. The main goals of the WFOL include the following:

- The library is to be used as a starting point for building new ontologies. The library aids in this by classifying the things that are to be modelled in the domain. This is performed by providing an integrated, high level view of the implemented foundational ontologies in order to assist ontology developers.
- It is to provide a reference for comparisons between ontological approaches.
- It is to provide a general framework for critically analysing and integrating foundational ontologies.

In order to realise the goals of the WFOL, it is required for there to be some guidelines, such as content comparison and ontological alignments that aid in achieving foundational ontology interchangeability. By content comparison, we mean comparing the structure, entities and relational properties of ontologies to one another. To the best of our knowledge, there has not been any work done on comparing foundational ontology content or providing complete alignment sets between foundational ontologies. The GFO documentation offers some rough mappings between itself and DOLCE, but it is not completely usable at this stage in that both foundational ontologies have been changed since release. Furthermore it does not include alignments between relational properties.

Seeing that foundational ontologies enable semantic interoperability but the infrastructure to allow for this is lacking, there is a need to solve this problem. The rationale behind the unimplemented WFOL was to assist with this issue of semantic interoperability by providing an infrastructure that would assist with foundational ontology interchangeability. We hope to realise the solution by creating a novel web-based repository of foundational ontologies.

1.2. Problem statement

There has been an exponential growth in ontology development for the Semantic Web. This causes ontology interoperability issues. Different foundational ontologies are used rather than committing to a single foundational ontology. In order for Semantic Web applications to share and process information correctly there is a need for ontology integration, so that ontology developers committing to a preferred foundational ontology will achieve seamless linking to domain ontologies. However, to the best of our knowledge, infrastructure to perform ontology integration for foundational ontologies do not exist. Such infrastructure includes content comparison between foundational ontologies, and alignments or mappings between foundational ontologies.

1.3. Research objectives and tasks

To solve this problem, we will select three foundational ontologies to which we perform a content comparison and thereafter mediation which includes the processes of alignment, mapping and merging. This will be implemented in a repository of foundational ontologies, such as the proposed WFOL, with the aim of supporting foundational ontology interoperability. The main objective of this research is to investigate and overcome the issues posed in foundation ontology interchangeability with a focus on mediation and modularity. The development of a tool will facilitate this. In order to achieve this, the following subtopics must be investigated:

- Compare the foundational ontologies of the library to one another. In order to perform this research objective, we must first perform the task of selecting foundational ontologies to be included in the library after extensive research and careful consideration.
- Provide a content comparison of the foundational ontologies. In order to obtain this, we must perform the task of identifying similarities and differences between foundational ontologies, regarding structural organisation, naming convention and related entities. For instance, one may assume that a DOLCE enduring is similar to a BFO Continuant but the term names are different.
- Perform the task of creating functional modules of foundational ontologies, where applicable.
- Identify and devise a way to deal with the conflicting theories between foundational ontologies. For instance, the BFO ontology takes on a philosophy of realism and as such abstract entities are not allowed, while the DOLCE ontology follows a descriptive philosophy allowing common-sense notions such as abstract entities to be modelled.
- Perform ontology mediation:
 - Perform alignment, mapping and merging of the foundational ontologies.
 - Take into consideration approximate alignments and mapping inconsistencies that may arise.
 - Create a higher-level foundational ontology.
- Promote reuse of the foundational ontologies, its modules and mapping and merged ontologies by performing the task of creating extensive metadata for each ontology in the repository.
- Perform experimental evaluation of the theory by creating a tool to assist with foundational ontology interchangeability.

1.4. Structure of thesis

The remainder of the thesis is structured as follows: Chapter 2 is a review of literature on: the proposed WFOL, official foundational ontology publications, comparative studies of pop-

ular foundational ontologies, existing ontology repositories, ontology modularisation, ontology mediation, ontology metadata, ontology browsing tools, and ontology verbalisation and view. In Chapter 3 we introduce and describe the materials and methods used to solve the problem of foundational ontology interchangeability. Chapter 4 describes the processes and outcomes of foundational ontology mediation, i.e. content comparison, alignment, mapping, and merging. This chapter also introduces a novel method to be used to perform foundational ontology interchangeability. Chapter 5 presents the design and features of the web-based repository, ROMULUS. In Chapter 6, we present and summarise the results of each evaluation performed. In this Chapter, we also compare ROMULUS to the envisioned WFOL. Lastly, in Chapter 7, we conclude the problem, propose future direction for foundational ontology interchangeability and present a summary of contributions.

Literature Review

In this chapter, we explore the theoretical and practical aspects to be considered for creating a functional repository of foundational ontologies. We begin by introducing the general philosophical stances and concepts that foundational ontologies undertake. Thereafter we explore widely-used foundational ontologies and classify them in terms of these philosophical distinctions and criteria. We then conduct comparisons of the foundational ontologies. Existing ontology repositories are then introduced with the hope of assisting with the development of the proposed repository. We look at ontology modularisation techniques and tools, and the possible types of modules that can be created in order to create functional modules of the foundational ontologies. In order to achieve foundational ontology interchangeability and integration, we look at foundational ontology mediation: its main processes and outcomes, and possible guidelines and tools that could facilitate foundational ontology mediation. We look at existing metadata models to select an appropriate one to be used for documenting the foundational ontologies of the repository. Ontology browsing tools, to facilitate online ontology browsing are then considered. Lastly, we explore ontology view tools to provide human-readable formats of the ontologies. These fields have seen an exponential growth over the years. It should be within reach to realise the goals of the WFOL.

2.1. Foundational ontology philosophies and choices

Foundational ontologies are based on philosophy. In order to understand and apply a foundational ontology, it is necessary to have an understanding of general philosophical choices world views and choices. In this section, we introduce and elucidate the common philosophical stances and choices that foundational ontologies undertake and their effects on modelling in an ontology.

Common world views that foundational ontologies commit to are descriptive, realist and newtonian. A descriptive foundational ontology is one that is based on human cognition. It is not restricted to modelling entities that exist in the real world only, and can include human common-sense and perception in the ontology. In contrast to this, a realist foundational ontology models

the world exactly as is and as such does not consider human common-sense concepts and abstract entities. Different to these two approaches, is the newtonian view of the world. A newtonian foundational ontology models the world as being composed of 3D space and time, while 3D and 4D objects exist mutually-dependent from each other.

Objects in the world are considered as 3D or 4D entities. 3D entities are simply those entities that are wholly present at all time. e.g., a soccer ball is a 3D entity because it is entirely present. 4D entities, are those that unfold in time. e.g., the process of kicking the soccer ball is a 4D entity because it occurs in time. Foundational ontologies commonly use the words *endurants* and *continuants* to describe 3D entities, and *perdurants* and *occurrents* to describe 4D entities.

In philosophy, and likewise in foundational ontologies, a distinction is made on whether an entity is concrete or abstract in nature. Concrete entities are those that exist in space and time e.g., a bar of chocolate which we are able to see and feel; while abstract entities are those that do not exist in space nor time e.g., number sets and propositions are abstract in nature; we cannot see or physically interact with them nor do they have a physical location.

A choice undertaken by foundational ontologies is that of universals or particulars. Universals are entities that can be instantiated e.g., the term ‘baker’ is a universal because it can be instantiated. In contrast to this, particulars are entities that cannot be instantiated e.g., the individual, ‘Alice’ is a baker, is a particular because it cannot be further instantiated.

An ontology can have a multiplicative or reductionist stance. In a multiplicative stance, different related entities can be co-localised in the same space-time e.g., both the bookshelf, and the wood that it is constituted occupy the same space-time model in a multiplicative ontology. In a reductionist ontology, different related entities cannot exist in the same-space time. Each space-time entity is restricted to just one entity.

In the following sections, we introduce foundational ontologies and their philosophical choices and views. Thereafter we compare these distinctions in foundational ontologies.

2.2. Official foundational ontology deliverables

The WonderWeb deliverable [53] provided much material about widely used foundational ontologies. The envisioned WonderWeb library includes 3 foundational ontologies at present: DOLCE, OCHRE and BFO. DOLCE is to be a starting point foundational ontology for comparing relationships with other foundational ontologies of the WonderWeb library. It is a descriptive ontology, based on common-sense principles. DOLCE uses the axioms of General Extensional Mereology (GEM) [87]. The taxonomy of DOLCE is displayed in Fig. 2.1. DOLCE allows properties to be represented by using quality and qualia. Qualities are basic entities such as colour or width, and qualia are the corresponding values for these basic entities.

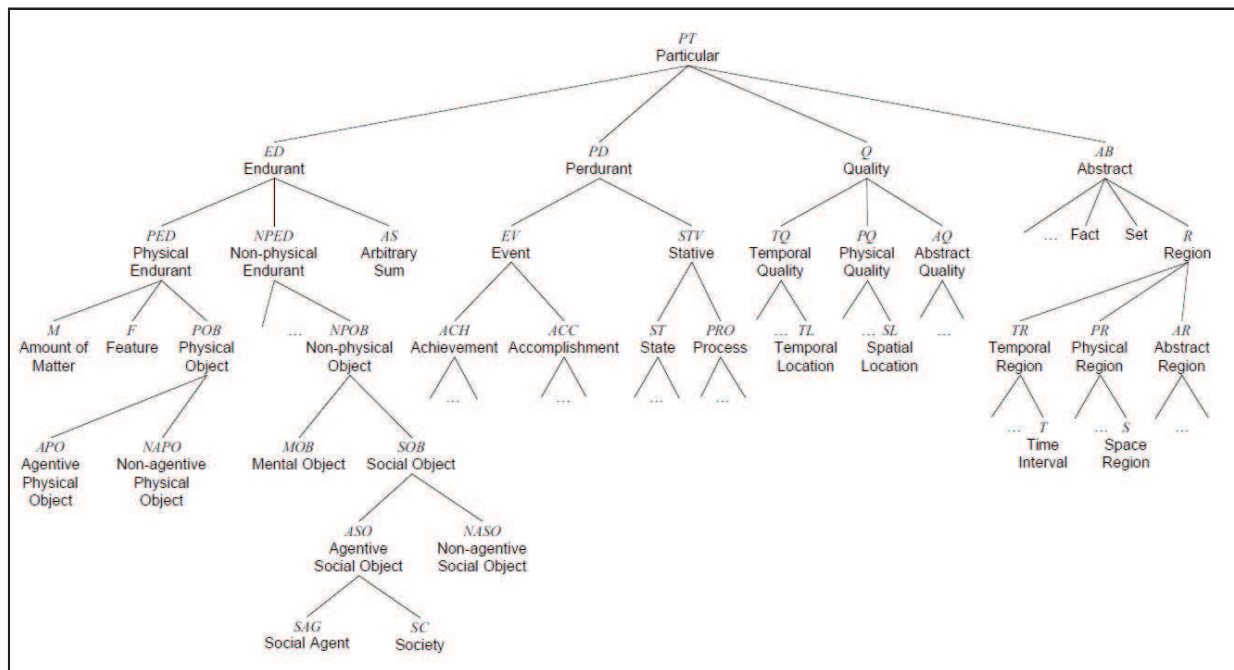


Figure 2.1.: The DOLCE taxonomy. Source: [53].

The second ontology of the WonderWeb library is OCHRE. It differs from DOLCE in a sense as it takes on revisionary view of the world, capturing it exactly as is without considering cognitive artifacts such as human perception, and other abstract entities. A portion of the taxonomy of OCHRE is displayed in Fig. 2.2.

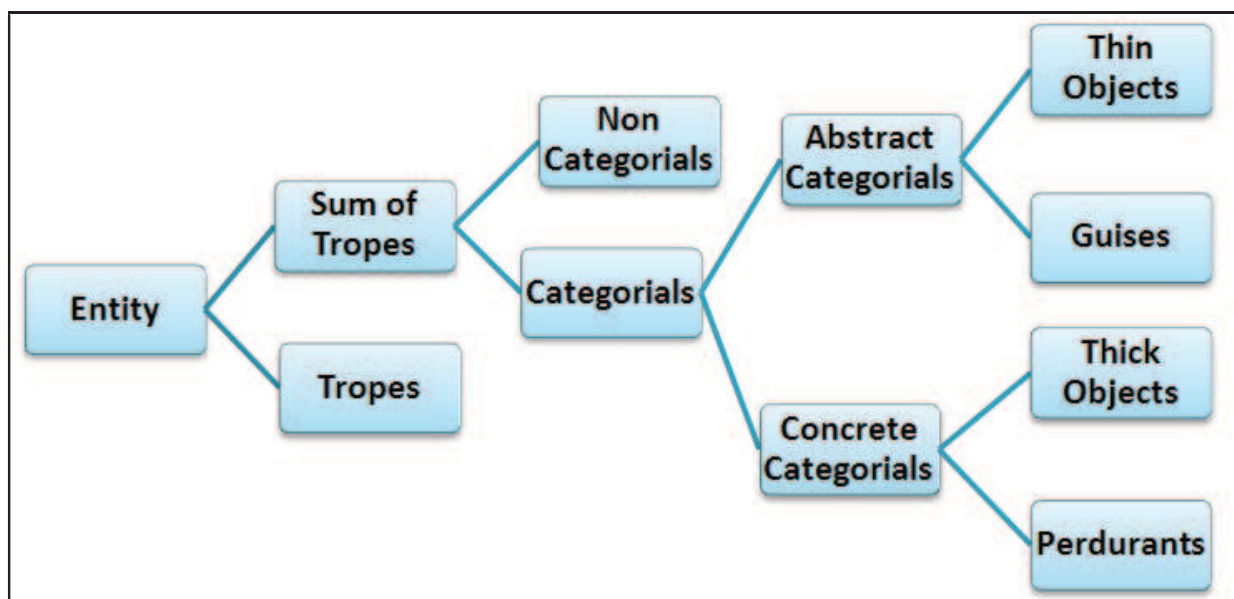


Figure 2.2.: A portion of the OCHRE taxonomy.

BFO, the third module of the WonderWeb library, is a relatively small taxonomy commonly used for scientific research and data integration purposes. The taxonomy of BFO is displayed in Fig. 2.3. The universals in BFO are connected with the *is_a* relation. BFO is a simple taxonomy and as such has no relational properties.

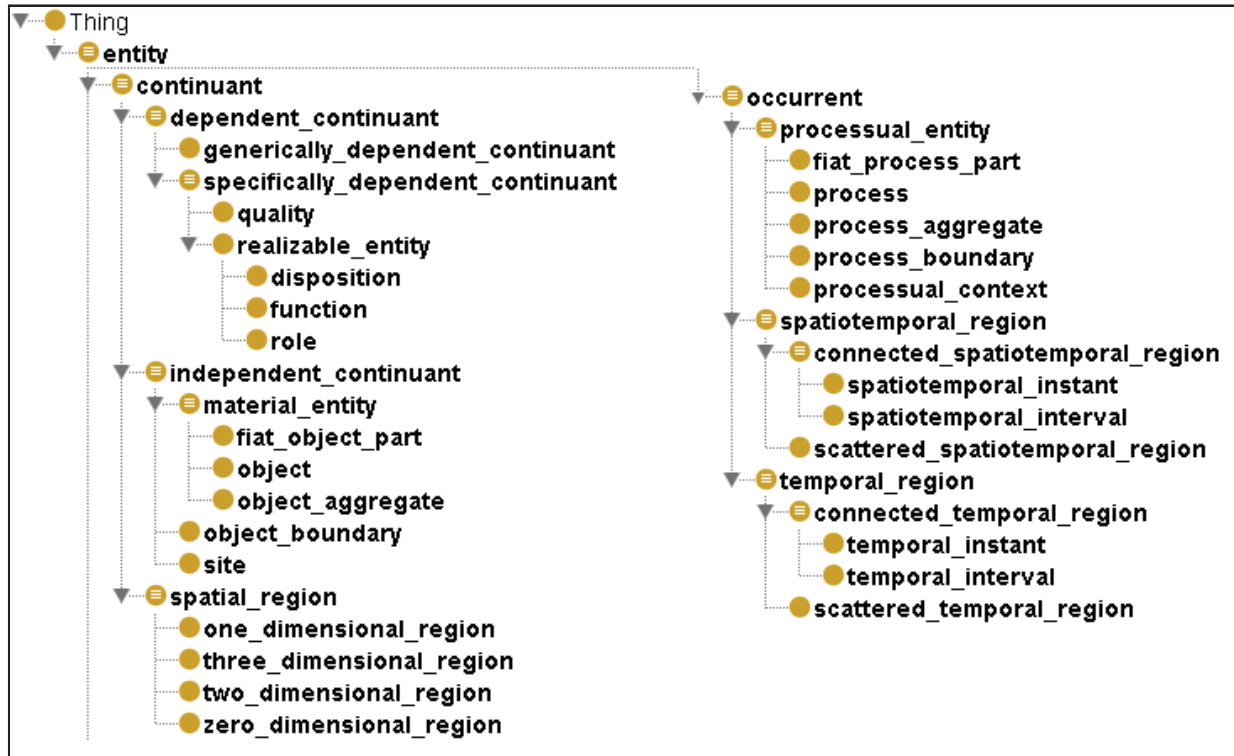


Figure 2.3.: The BFO taxonomy.

SUMO is an ontology of universals and particulars. It is descriptive in nature and may be used in a number of applications such as the Semantic Web [21] and ontologies for natural language processing [65]. Nevertheless, it is a massive ontology with thousands of terms resulting in it being time consuming to understand and adapt to applications. A portion of the taxonomy of entities in SUMO is displayed in Fig. 2.4.

GFO is an ontology of universals, concepts and symbols. It has a model for space and time, and is used mainly in the health-care/medical field. A portion of the taxonomy of GFO is displayed in Fig. 2.5.

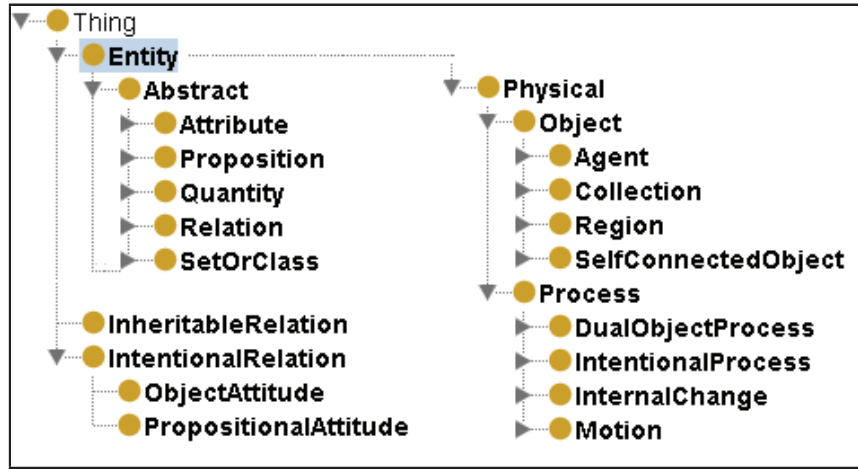


Figure 2.4.: A portion of the SUMO taxonomy.

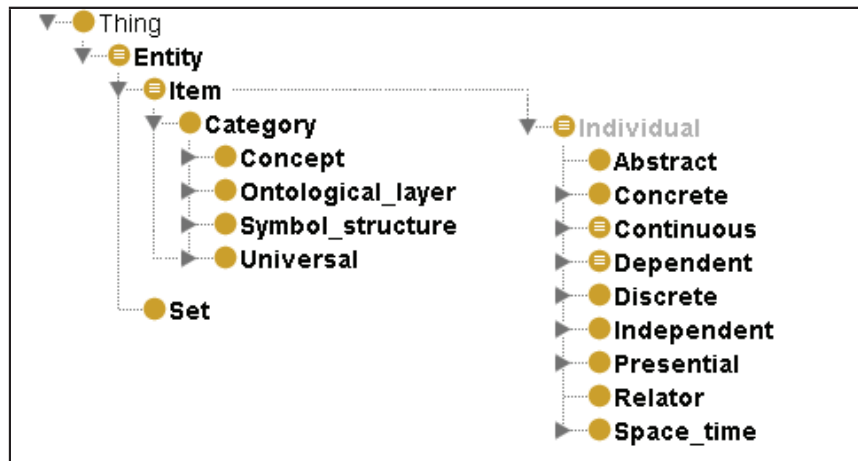


Figure 2.5.: A portion of the GFO taxonomy.

One of the most recent foundational ontologies, YAMATO was created to fill in the gaps with respect to quality and quantity representation, representation (information objects) and views with respect to processes, objects and events, thereby improving existing foundational ontologies. The world view or standpoint that YAMATO is based on is the newtonian world view. In the newtonian world view, the world is perceived to be composed of 3D space with absolute time. Both 3D and 4D entities exist with equal importance in a mutually-dependent manner. The top level categories of YAMATO are displayed in Fig. 2.6. Relations are not included in YAMATO because it is embedded into the Hozo [48] tool. Included in YAMATO, is a separate ontology for accurately representing quality and quantity.

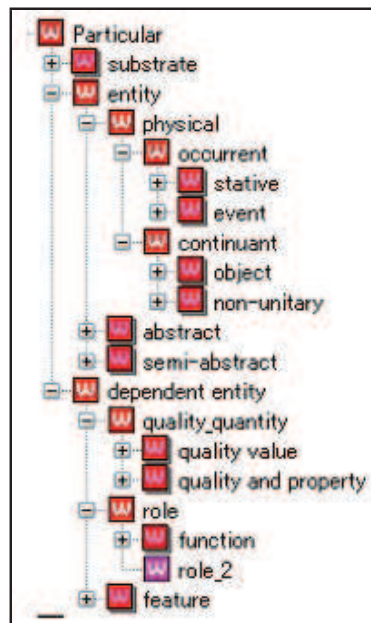


Figure 2.6.: The top level categories of YAMATO. Source: [56].

GIST¹, is a foundational ontology aimed for the business domain. Unlike the other foundational ontologies, it is not based on philosophical commitments but rather on common business concepts. Fig. 2.7 displays GIST's core classes.

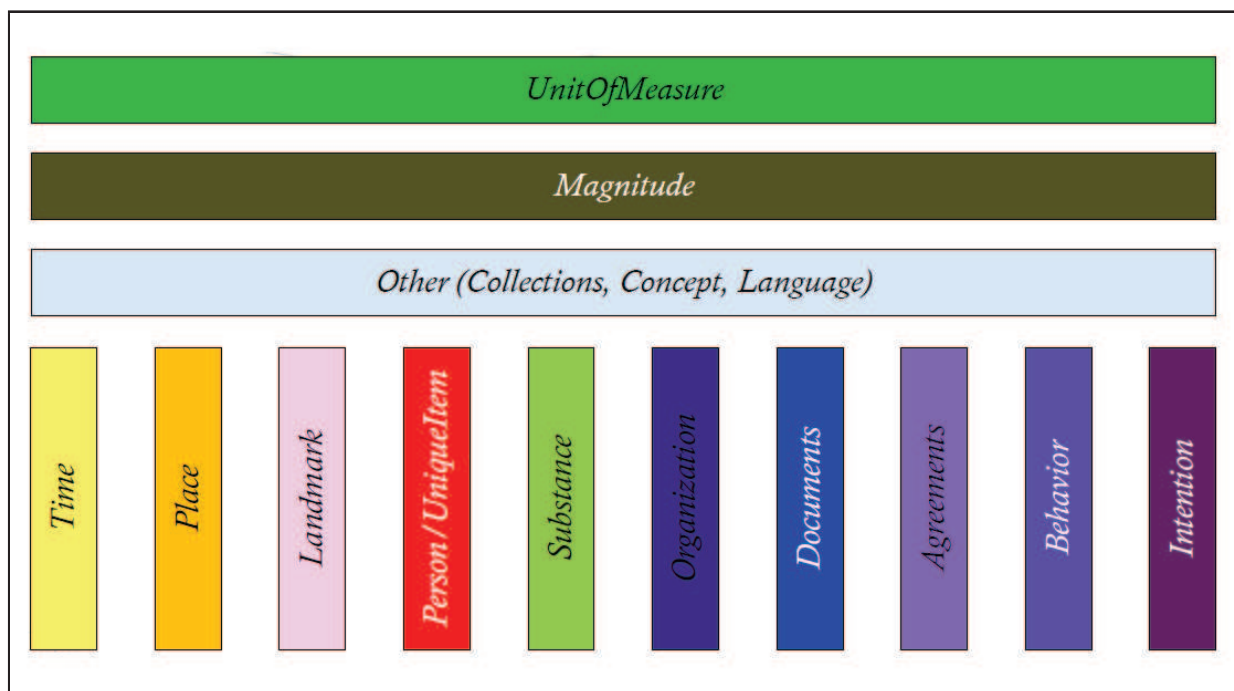


Figure 2.7.: The core classes of GIST. Source: [55].

¹<http://semanticarts.com/gist/>

2.3. Comparison of popular foundational ontologies

A number of existing works [45, 67, 52, 24, 37] critically analyse and compare existing foundational ontologies. Information such as technical aspects, available languages and the building blocks of the compared foundational ontologies are discussed. Based on these comparative studies and other documentation [74, 3, 45], the following was concluded:

In terms of the philosophies undertaken by each foundational ontology; DOLCE, GFO and SUMO are descriptive in nature, meaning ontological categories underlying natural language as well as common sense are captured. BFO, on the other hand is realist in a sense as it aims to capture the world exactly as is. YAMATO takes on a newtonian world view, allowing objects and processes to exist in mutual dependence. GIST's world view is not defined.

A number of languages used to represent ontologies exist. Some of them are KIF, CLIF, OBO, OWL and DOL. Knowledge Interchange Format (KIF) is a language designed for use in the interchange of knowledge among different computer systems. Common Logic Interchange Format (CLIF) is a logic-based language which has the purpose of standardizing syntax and semantics for semantic interoperability. OBO began from the Gene Ontology and is a directed acyclic graph. The Web Ontology Language (OWL) [4] is a W3C recommendation [32]. The DL-based OWL species are decidable fragments of first order logic used for publishing and sharing ontologies on the World Wide Web. The Distributed Ontology Language (DOL) [60] is a metalanguage which consists of levels that work together to allow ontologies to be formalised in heterogeneous logics, provide support for modular ontologies and assist in linking and annotating ontologies.

There are versions of DOLCE in OWL DL, OWL 2 DL and KIF languages. The OWL version of DOLCE (DOLCE-Lite) does not contain modality, temporal indexing, relation composition. DOLCE-Lite is made up of 37 entities and 70 relational properties. The simplicity of BFO allows it to be represented in all OWL species and in OBO. BFO in OWL is made up of 39 entities. BFO in OBO is made up of 39 entities. BFO together with RO in OWL is made up of 42 entities and 25 relational properties. The core of BFO is represented in Isabelle (First- Order based), and is made up of 18 theories. This version consists of a non-extensional temporal mereology with collections, sums, and universals. There are versions of GFO in OWL DL, OWL 2 DL and KIF. The full version of GFO is made up of 78 entities and 67 relational properties. The basic version of GFO has 45 entities and 41 relational properties. There are versions of SUMO in OWL DL and SUO-KIF. SUMO is a large ontology, made up of 1000 terms, 4000 axioms and 750 rules. There are versions of YAMATO in OWL DL and Hozo. YAMATO is made up of 540 entities and 48 relational properties. GIST contains 105 classes and 128 relational properties, and there are versions of it in OWL DL and OWL 2 DL.

Modularity is said to be required in an ontology when one needs to hide knowledge which is unnecessary to the task at hand [68]. DOLCE, BFO, GFO, SUMO, YAMATO and GIST are all modular ontologies. By modular, we mean any of the following: lighter/detailed versions of the ontology exists, the ontologies offer built-in domain ontologies or that the ontologies have separate branches of 3D and 4D entities. Section 2.5 provides further details about ontology

modularisation. DOLCE, GFO, SUMO and GIST offer lighter and more-detailed versions of the ontologies. GIST's modules are, however, available to clients only and not freely available. A tactic used in ontology development [40] is to separate 3D and 4D entities in an ontology. DOLCE, BFO and YAMATO have separate branches of 3D and 4D entities. This allows one to easily modularise them by creating separate modules for 3D and 4D entities. In BFO, Continuant and Occurrent are found in separate sub-ontologies, and as such do not co-exist in the same ontology, making it simple to create separate modules. In DOLCE, enduring and perduring are linked by a participation relation. This may cause some difficulty when modularising because ontology modularity tools and techniques preserve the local completeness of the module, further discussed in Section 2.5.1. In YAMATO, continuant and occurrent are found in separate hierarchies but exist in a mutually dependent manner according to its underlying newtonian philosophy. DOLCE, BFO, GFO, SUMO, YAMATO and GIST are all actively used and maintained.

There has not been much work done in comparing content of the foundational ontologies. By this, we mean comparing their classes, properties and relations. Work has been performed where primitive relations of BFO (formalised with Relation Ontology (RO)) and DOLCE are compared [75]. To a certain extent, the philosophies behind the foundational ontologies affect the way the relations are represented, e.g., BFO is a realist ontology, and has no abstract entities while GFO is both descriptive and realist in nature and allows abstract entities in an ontology. BFO's parthood relation `has_part` does not consider abstract entities, while GFO has a parthood relation `abstract_has_part` that considers abstract entities at a higher-level than its `has_part` relation.

Temal et al. [82] created an alignment between DOLCE and BFO in order to integrate medical information. The entities are mapped with equivalence and subsumption relations. Based on the version of BFO where it was divided into SNAP (Continuant) and SPAN (Occurrent) ontologies, they found that all BFO entities were successfully mapped to DOLCE, but not all DOLCE entities could be mapped to BFO. These alignments were based on the First Order Logic (FOL) version of the ontologies, where the SNAP-BFO has, e.g., `Boundary`, that the current BFO version in OWL does not have, and DOLCE is claimed to have `Collection`, which appears neither in the principal documentation [53] nor in the OWL version of DOLCE. These alignments were not mapped on consistency. Some of their alignments are useful, however, which we will return to during mediation processes in Chapter 4.

We have performed initial work on comparing DOLCE, BFO, GFO and SUMO foundational ontologies as part of a BSc Honours project. This comparison is based on the following categories: ontological commitments, representation language, software engineering properties, subject domain, and applications. Thereafter, we have extended these comparisons to include new foundational ontologies YAMATO and GIST, and to include new criteria for some categories. Refer to Table 2.1 to view a comparison for the category of ontological commitments. Table 2.2 provides a comparison of representation languages and software engineering properties. These comparisons will aid in selecting foundational ontologies to be implemented in the proposed repository.

Table 2.1.: Comparison of ontological commitments for each foundational ontology.

Term (and very brief descriptions of its meaning)	DOLCE	BFO	GFO	SUMO	YAMATO	GIST
Universals vs. Particulars (Universals can have instances, particulars do not)	Particulars	Universals	Universals, particulars	Universals, particulars	Particulars	Unclear
Descriptive vs. Realist (Descriptive: represent the entities underlying natural language and human common-sense; Realist: represent the world as is)	Descriptive	Realist	Descriptive, realist	Descriptive	Realist	Descriptive, realist
Multiplicative vs. Reductionist (Multiplicative: different objects can be co-located at the same time; Reductionist: only one object may be located at the same region at one time)	Multiplicative	Reductionist	Unclear	Multiplicative	Multiplicative	Multiplicative
Endurantism vs. Perdurantism (Endurantism: an object is wholly present at all times; Perdurantism: an object has temporal parts)	Endurantism and perdurantism	Endurantism and perdurantism	Endurantism and perdurantism	Endurantism and perdurantism	Endurantism and perdurantism	Endurantism and perdurantism
Actualism vs. Possibilism (everything that exists in the ontology is real vs. objects are allowed independent of their actual existence)	Possibilism	Actualism	Unclear	Unclear	Actualism	Possibilism
Eternalist stance (the past, present, future exist)	Eternalist	Eternalist	Eternalist	Eternalist	Non-eternalist	Eternalist
Concrete & Abstract entities (Concrete: entities that exist in space and time; Abstract: entities that exist neither in space nor time)	Concrete, abstract	Concrete	Concrete, abstract	Concrete, abstract	Concrete, abstract	Concrete, abstract
Mereology (theory of parts)	GEM	Own mereology	Own mereology	Own mereology	Own mereology	Own mereology
Temporal aspects (e.g., time-indexed axioms)	Provided	Not provided	Provided	Provided	Provided	Provided
Granularity (different levels of detail contained in an ontology)	High level	Sensitive to granularity	Unclear	Unclear	High level	High level
Properties and values ('attribute'; e.g., the colour of an apple)	Included	Not included	Included	Included	Included	Included
Model for space and time (Consists of time and space regions and boundaries)	Not included	Not included	Included	Not included	Not included	Not included
One-layered vs. Three-layered architecture (a basic level only; an abstract top level, abstract core level and basic level)	One-layered	One-layered	Three-layered architecture	One-layered	One-layered	One-layered
Situations and situoids (Situation: an aggregate of facts that can be comprehended as a whole and satisfies certain conditions of unity; Situoid: is a part of the world that is a comprehensible whole and can exist independently)	Not included	Not included	Included	Not included	Not included	Not included

Table 2.2.: Comparison of representation languages and software engineering properties for each foundational ontology.

Term	DOLCE	BFO	GFO	SUMO	YAMATO	GIST
Representation Languages						
OWL DL	Yes	Yes	Yes	Yes	Yes	Yes
OWL 2 DL	Yes	Yes	Yes	-	-	Yes
OWL 2 EL	Yes	Yes	Yes	-	-	-
OWL 2 QL	Yes	Yes	Yes	-	-	-
OWL 2 RL	Yes	Yes	Yes	-	-	-
FOL	Yes	Yes	-	-	-	-
DAML	Yes	-	-	-	-	-
KIF	Yes	-	-	-	-	-
SUO-KIF	-	-	-	Yes	-	-
OBO	-	Yes	-	-	-	-
HOZO	-	-	-	-	Yes	-
Software engineering properties						
Number of classes in ontology	37	39	78	630	2311	105
Number of object properties in ontology	70	0	67	217	829	128
Number of axioms in ontology	349	95	323	1894	10790	997
Modular: Lighter/more-detailed versions	Yes	No	Yes	Yes	No	Yes (for clients, not freely available)
Modular: Separate branches for 3D and 4D entities	Yes	Yes	No	Yes	Yes	No
Modular: Functions and roles	No	No	Yes	No	Yes	No
Modular: Built-in domain support	Yes	No	No	Yes	Yes	No
Modular: OWL 2 profiles	Yes	Yes	Yes	No	No	No
Registerable on the OBO Foundry	No	Yes	No	No	No	No
Freely available	Yes	Yes	Yes	Yes	Yes	Yes
Actively maintained	Yes	Yes	Yes	Yes	Yes	Yes

2.4. Existing ontology repositories

Ontology repositories are systems in which ontologies are publicly hosted, along with some other functionality. Ontology usage and sharing is promoted with repositories. There are many types of ontology repositories, of which a few are discussed below.

The TONES ontology repository [2] aims to be a central location for various ontologies. It simply allows one to browse and download each ontology. Additionally, the repository allows one to select metrics to be displayed for each ontology. TONES has 35 different types of ontology metrics, which include: DL Expressivity, LogicalAxioms, Classes, Object properties, Data properties, Individuals, SubClassOf, EquivalentClasses, and others.

The Open Ontology Repository (OOR) Initiative is an effort to create support for storing, managing and integration of ontologies. Some use cases of the OOR are to find relationships between entities in different ontologies and to find mappings. With this in mind, OOR compliant repositories may be used to meet a number of research objectives of the project, discussed in Section 1.3. In this section we use the term “instance” when referring to an OOR compliant repositories.

The OOR instance allows a user to add a project to it. Thereafter, a user is able to perform a number of tasks: finding and creating relations between entities in different ontologies, creating annotations, submitting ontologies, and adding ontology mappings to the repository. One is able to browse existing projects and ontologies of the repository for mappings and relations. Presently, five ontologies are implemented in the repository. Mappings do not exist, and the ontologies of the repository are not relevant to the project at hand. This is because no foundational ontologies exist in this OOR instance and consequently no foundational ontology mappings are found. Metadata of the submitted ontologies consisting of details, metrics, reviews, versions, views, and project involved are displayed, when available. A user is also able to contribute to this. OOR offers many features, useful in the creation of an ontology repository. However, at present we are not possible to make use of it because there are only a few submitted ontologies and mappings, which is insufficient to contribute to foundational ontology interchangeability.

The SOCoP OOR instance [17] offers the same functionality and has the same interface as the OOR instance. However, it seems to be more populated than the OOR instance. Presently, there are 27 submitted ontologies and some mappings between the ontologies. Foundational ontologies including BFO, DOLCE+DnS Ultralite and SUMO are found in the repository. Some useful metadata for each of these ontologies are found. Fig. 2.8 shows a screenshot of the summary of BFO metadata found in the repository. The proposed foundational ontology repository, ROMULUS, will have metadata for all modules, in a similar format to this. A few mappings between DOLCE+DnS Ultralite and the Semantic Sensor Net ontology [35] are available. However, these are irrelevant to the project at hand because we require mappings between foundational ontologies, and not those between domain and foundational ontologies. While SOCoP does include a number of ontologies and a few mappings that contribute to ontology interchangeability, this is insufficient to contribute to foundational ontology interchangeability at present.

Basic Formal Ontology

Summary

Details

ONTOLOGY ID:	1012
STATUS:	Production
FORMAT:	OWL
CATEGORIES:	Upper Ontologies
GROUPS:	
CONTACT:	Holger Stenzhorn, holger.stenzhorn@uks.eu
HOME PAGE:	http://www.ifomis.org/bfo
PUBLICATIONS PAGE:	http://www.ifomis.org/bfo/publications
DOCUMENTATION PAGE:	http://www.ifomis.org/bfo/manual
DESCRIPTION:	A genuine upper ontology which can be used in support of domain ontologies developed for scientific research. Thus BFO does not contain physical, chemical, biological or other terms which would properly fall within the special sciences domains.

Metrics

NUMBER OF CLASSES:	39
NUMBER OF INDIVIDUALS:	0
NUMBER OF PROPERTIES:	0
MAXIMUM DEPTH:	7
MAXIMUM NUMBER OF SIBLINGS:	5
AVERAGE NUMBER OF SIBLINGS:	4
CLASSES WITH A SINGLE SUBCLASS:	1
CLASSES WITH MORE THAN 25 SUBCLASSES:	0
CLASSES WITH NO DEFINITION:	39

Versions

VERSION	RELEASE DATE	DOWNLOADS
1.1.1	06/28/2009	Ontology
1.0 RC	12/06/2006	Ontology

Figure 2.8.: A summary of BFO metadata from SOCoP.

The next OOR instance is Ontohub [86]. It is an ontology repository engine which specialises in managing distributed ontologies. It consists of 115 ontologies at present. Each ontology has some metadata displayed as an overview. Furthermore, one can search through the entire repository for entities. Users are able to add an ontology to the repository, and if successful, information about the logic, entities and axioms of the ontology are displayed. We initially considered using Ontohub's open source code to provide some of the features of ROMULUS. However, Ontohub is a relatively new repository and since its initial development stage, it has drastically changed, making it a rather unstable choice at present.

The Common Logic Ontology Repository (COLORE) [26] was created to support the integration of ontologies and reuse of ontologies for manufacturing standards. The ontologies in COLORE are divided into three levels: foundational ontologies, generic ontologies and ontologies for standards. Fig. 2.9 shows the layered architecture of COLORE. The bottom layer of the figure represents foundational ontologies, the middle layer represents generic ontologies and the top layer represents ontologies for manufacturing standards. COLORE ontologies, are, however, represented in Common Logic only, and not OWL. Therefore they are not easily usable in the Semantic Web. When the COLORE ontologies are incorporated into the OOR architecture, they will serve as a testbed for ontology integration and evaluation techniques. This is useful as it can aid in the development of the envisioned foundational ontology repository by performing foundational ontology integration.

The existing OOR instances implement a small subset of the requirements of the OOR initiative. Some of the repositories support a few of the functional requirements of ROMULUS that will be described in Chapter 5, but there is currently no infrastructure that provides all of ROMULUS's functional requirements. Therefore we are required to build a new repository.

¹<http://socop.oor.net/ontologies/1012/?p=summary>

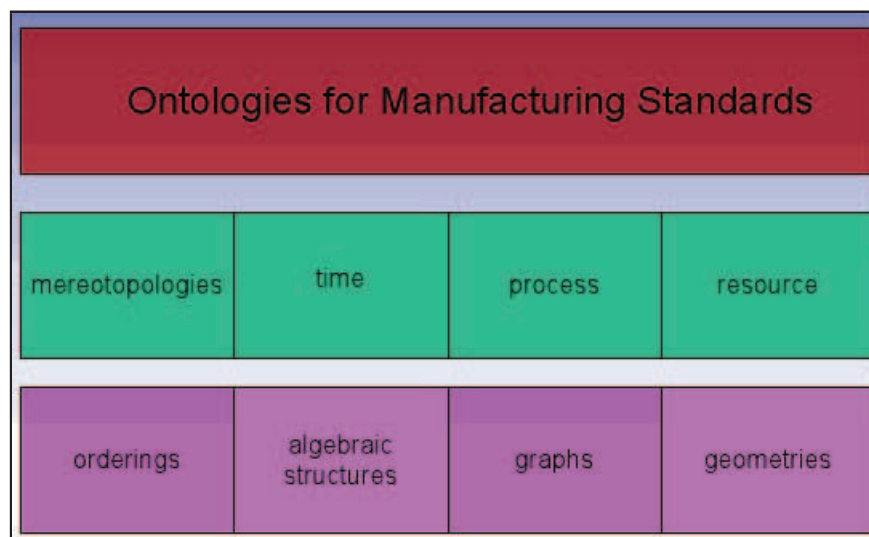


Figure 2.9.: The layered architecture of COLORE. Source: [27].

2.5. Ontology modularisation

Ontology modularisation deals with creating or altering an ontology to be broken down into modules for specific functions. The idea behind it is to hide unnecessary detail when not required. Modularity is important in that it aids in ontology maintenance, publication, validation, and processing. Elsewhere [15], these factors are discussed and modularity is evaluated by introducing modularisation evaluation techniques.

The purpose of modularity is analysed and discussed [7] by classifying modules into broad types: ‘modules for a single ontology’, ‘modules for several ontologies’ and ‘modules for everything’. In ‘modules for single ontologies’, a monolithic approach is taken whereby modularity is used solely to manage domain coverage in ontologies. In this sense, ontologies are divided according to structure rather than function e.g., the myExperiment ontology [62], which is a collaborative environment where scientists publish and share their work flows and experiment plans is modularised into 10 structural modules, each which can be used without affecting the whole ontology. A module may be added or removed, as required, without altering the overall system itself. In ‘modules for several ontologies’, a functional approach is taken. Here, emphasis is placed on techniques and relations used to properly build foundational ontologies with higher expressivity. One such way to achieve this is for the ontology to include a mereology theory. Lastly, in ‘modules for everything’, a module might be the result of many things such as isolating branches of a taxonomy, collecting categories according to a domain, separating (sub)systems to improve ontology matching, and more. By summarising the types of modules introduced in this work [7], we find that a module may be any of the following types:

- Modules to organise and manage domain coverage.
- Modules to add functionality.

- Modules by isolating/developing branches of a taxonomy.
- Modules for a particular subject domain (biomedical, engineering, military etc.)
- Modules by isolating (sub)theories to identify a context.
- Modules by isolating primitives and their axiomatisations.
- Modules by isolating patterns.
- Modules as a result of isolating (sub)systems by minimizing the number of cross-relationship.
- Modules by dividing/developing a large system to assist with overall reasoning.
- Modules by separating (sub)systems suitable for compatible reasoning engines.
- Modules by separating (sub)systems to aid with ontology matching.
- Modules as a result of simplifying the ontology by removing all relational properties.

The concept of modularity in terms of ‘modules for everything’ will be further investigated and applied to the project at hand. This is because, for the proposed repository, branches of the foundational ontologies may be isolated, sub systems may be separated to improve ontology matching and modules may be identified for some purpose.

Important principles such as inconsistency and subsumption, and formal properties such as robustness that are encountered when modularising ontologies are identified and discussed [47]. These are explored from a logical point of view using description logic (DL) and classical predicate logic. While, at present, we will focus only on equivalence relations between entities, at a later stage we will include subsumption and other relations whereby such published works will be referred to.

Automated tools for ontology modularisation are available. OWL Module extractor, which is based on logic-based module extraction [12], is one such tool. It allows one to paste in the text of an ontology file and a set of entities to be extracted. Based on these inputs, it extracts a module. By pasting DOLCE-Lite ontology, and selecting ‘perdurant’ as the entity to be extracted, the tool may be used to create a module of perdurants of the DOLCE-Lite ontology.

Swoop [39] is an OWL ontology browser and editor tool. It has a modularisation plugin which offers support for extracting modules from ontologies. Swoop allows for two kinds of module extraction: locality and dual locality modules. Locality modules preserve the meaning of the selected entity in terms of its super-entities while dual locality modules preserve the meaning of the selected entities in terms of its sub-entities. Therefore locality modules can be used to create less-detailed modules, and dual locality modules can be used to create more-detailed modules. One can also manually add new entities to modules if required.

Protégé v4.2 [1] has built-in functionality for modularity. It allows one to select axioms of an ontology by choosing one of three methods: axioms by profile, axioms by reference or axioms by type. Axioms by profile allows one to select axioms in a sublanguage of OWL, axioms by reference allows one to select specific entities from an ontology and axioms by type allows one to select subclass axioms and annotation axioms. Thereafter, the user chooses the specific entities, axioms or subclass and annotation axioms, resulting in a new module being created. A screenshot of the axioms by reference method for modularisation is displayed in Fig. 2.10

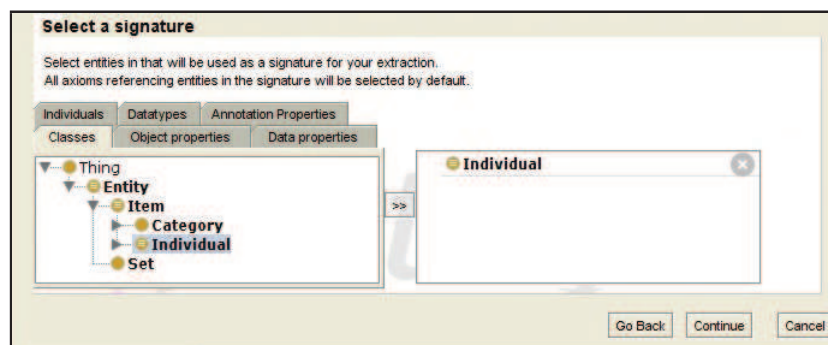


Figure 2.10.: Modularisation by axioms in Protégé.

2.5.1. Evaluating modularity tools

A starting point for evaluating modularity tools was a study conducted on evaluating ontology modularisations techniques by widely-used evaluation criteria [15]. The following modularity tools were used in the study [15]: Swoop [39], Pato [81], KMi [14] and PROMPT [66]. It was found that Swoop's resulting modules were large and almost the same size as the original ontology. PROMPT's modules were almost as large as Swoop's. KMi and Pato generated smaller modules. However, KMi included most of the ontologies properties. In terms of performance, Swoop, Pato and KMi performed modularity quickly in a matter of seconds. However, the PROMPT system crashed and consequently cannot be measured for performance.

Logically, Swoop is the only tool that guarantees local completeness of the module. Local completeness of a modules states that every axiom that contains elements of a module local signature M in an ontology O , must be preserved in the module. The logical criteria that Swoop enforces is the reason why its modules are so large. In the case of modularising DOLCE-Endurants (an ontology containing classes that are subsumed by DOLCE's endurant class only) with Swoop, it preserves perdurants in the resulting module because DOLCE's perdurant entity has axioms containing endurant e.g., perdurant subclass of participant some endurant in order to ensure that the local completeness of the module is preserved.

We evaluate OWL Module Extractor, Swoop and Protégé for modularising foundational ontologies in terms of the following two criteria: size, performance and local completeness of the module. For modularising the BFO ontology into a module of occurrent entities only, both OWL module extractor and Swoop generated modules that were 92% the size of the original ontology. Protégé's module was 97% of the size of the original ontology. All of the modularity tools performed poorly in terms of the size criteria, generating modules close to the size of the original ontology. In terms of performance, all three tools performed modularity quickly within seconds. All three modularity tools guarantee the local completeness of each module which is the reason for them generating large modules.

2.6. Ontology mediation

Ontology mediation is a term used to describe determining and overcoming differences between ontologies in order to allow for ontology reuse. Ontology mediation [16] is divided into three operations: ontology mapping, alignment and merging. Ontology alignment is the process of specifying correspondences between entities, by using a relation. To perform this, similarities and differences between ontological entities must be identified. Ontology mapping deals with creating correspondences between ontologies based on the alignments. In ontology merging, a new merged ontology is created from the original ontologies. Elsewhere [16], an overview of approaches, frameworks, and technology used to perform ontology mapping, alignment and merging is given. A number of tasks for the problem at hand are based on ontology mediation.

Existing work on mediation [61] includes a hybrid approach, based on both syntactic and semantic matching measures. Syntactic similarity is calculated by comparing substring matches of classes and terms found in the ontologies. Semantic similarity is calculated by comparing the meanings of classes and terms by using reliable algorithms. The results show that by including semantic measures, erroneous data is filtered and the ontology becomes considerably smaller in size. In [20] the challenges faced in ontology matching as well as recent advances in the fields are discussed with the hope of accelerating the progress of ontology matching. Various ontology matching applications are also compared here.

In order to perform ontology mediation, a number of tools and methods were considered. Among others, we have considered Ontobuilder [70] and S-Match [23] which no longer worked. The working tools are explored below.

LogMap [36] automatically generates mappings between ontologies using logic-based semantics of the input ontologies. It offers an improvement to other mapping tools in that it addresses scalability and logical inconsistencies. There is both a stand-alone and web-based application for this. The web-based application simply asks the user for some details (name, email address) and to upload each ontology. Thereafter, within minutes a link is emailed to the user containing a mapping ontology and a merged ontology based on the input files. LogMap allows a user to upload ontologies in a number of formats e.g., OWL, OBO and Turtle, and implements existing reasoners to check the satisfiability of the ontologies.

YAM++ [63] aligns entities by information retrieval or machine learning if training data is available. Three matchers are implemented in YAM++: an element level matcher, a structural matcher and a semantic matcher. The element level and structural mapper discover alignments while the semantic matcher revises these alignments to remove inconsistencies and ensure logical mappings.

HotMatch [13] is a tool based on a combination of many matching algorithms. The two types of algorithms are element level and structural matching. However, there is more than one of each implemented. There are also filters in HotMatch, used to remove duplicate mappings found by the matchers. Upon input of a source and target ontology, HotMatch deploys its matchers and filters sequentially resulting in mappings between the two.

Hertuda [31] is an entity matcher that applies element level matching with a string comparison. The alignments generated by Hertuda are only satisfiable in OWL Lite/DL. As a result, object properties in the ontologies are handled separately. This may cause some difficulties in aligning object properties in the foundational ontologies because their domains and ranges affect the alignments.

Optima [46] is an automatic tool which iteratively improves alignments. It is aimed at aligning large ontologies but may also be used for smaller ontologies. Its similarity measure is based on both syntactic and semantic similarity.

H-Match [9] is an algorithm for matching ontologies at different levels of depth, with different accuracies, based on user preferences. The H-Match algorithm takes into account both linguistic and semantic features of ontologies to perform matching based on user preferences. H-Match uses one of four matching models: surface, shallow, deep or intensive. Once the user selects one of these, H-Match computes comparisons between entities of two ontologies, with a corresponding matching value for each pair of entities. When the surface model is used, only linguistic affinity between entity names is used to measure similarity. In shallow, deep and intensive models, context is also considered to determine entity similarity. We will use H-Match's intensive model to perform foundational ontology alignment.

PROMPT [66] is a plug-in for Protégé that allows for comparison, mappings, and merging between ontologies. It is a semi-automatic method that invokes algorithms based on a combination of concept-representation structure, the relations between entities and user's actions. PROMPT offers the user four different algorithms to use for initial comparison: lexical matching, FOAM plugin, lexical matching with synonyms and using UMLS [50] concept identifiers for matching. The flow of PROMPT is illustrated in Fig. 2.11. PROMPT is only supported in older version of Protégé, which makes it quite unstable. It will, however, be used to assist in mapping and merging the foundational ontologies. When PROMPT's algorithm is executed, it generates a list of suggestions which the user must accept or ignore. As the user accepts suggestions, more suggestions are generated. Thereafter, it performs final mapping and merging.

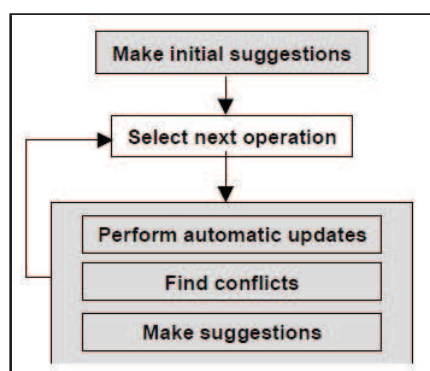


Figure 2.11.: The flow of the PROMPT algorithm. Source: [66].

2.7. Ontology metadata

Ontology metadata is additional data used to annotate an ontology in order to enable ontology reuse. It is used to help ontology developers and artefacts to detect change in an ontology, because the change might affect its systems and applications. For ROMULUS, changes in an ontology might affect its modules, ontological alignments, mapping, and merged ontologies. There are a number of existing ontology metadata models used in existing repositories and applications. In this section, we explore a number of models in order to select an appropriate one to be applied to ROMULUS's foundational ontologies.

The Dublin Core Metadata [88] contains a general vocabulary list of properties for use in resource description. The terms are not limited for usage in ontologies but also used in a variety of resources such as music and videos. Dublin Core Metadata Initiative consists of fifteen terms of the Dublin Core Metadata Element Set as well as additional elements that can be used for a number of applications. The Dublin Core Metadata Initiative is multidimensional in that it provides terms, elements, vocabulary encoding schemes, syntax encoding schemes, classes, and types, each with a number of criteria. While the Dublin Core does provide a variety, there is not sufficient support for describing an ontology in detail, particularly its metrics.

The OMV [29] provides a vocabulary for ontology metadata with the aim to provide ontology reuse. The OMV is formalised in OWL, which facilitates interoperability among machines. It consists of a number of classes, properties and relationships. By including the OMV in our repository, it may assist in ontology module management and reuse. An overview of the OMV model is displayed in Fig. 2.12.

The OM²R metadata model [84] aims to promote ontology mapping reuse. OM²R is formalised as an OWL ontology. It provides common criteria to document mappings, but is separate from the mappings themselves. Using such a metadata model in ROMULUS is useful for managing the mappings between foundational ontologies. Table 2.13 displays the metadata fields from the OM²R model compared to those of the Ontology Alignment Evaluation Initiative (OAEI). The OM²R metadata model has many metadata fields which are important in general alignment ontologies but do not exist in the OAEI.

2.8. Ontology browsing tools

It is useful to have a feature to allow online ontology browsing in ROMULUS. By having this feature, users can simply browse through ontologies of interest without having to download or install additional software. In particular we require the browsing of OWL ontologies with respect to navigating through its class and object property hierarchies. There is not a wide variety of existing online ontology browsing tools. To the best of our knowledge, jOWL [18] and WebProtégé are the only working tools.

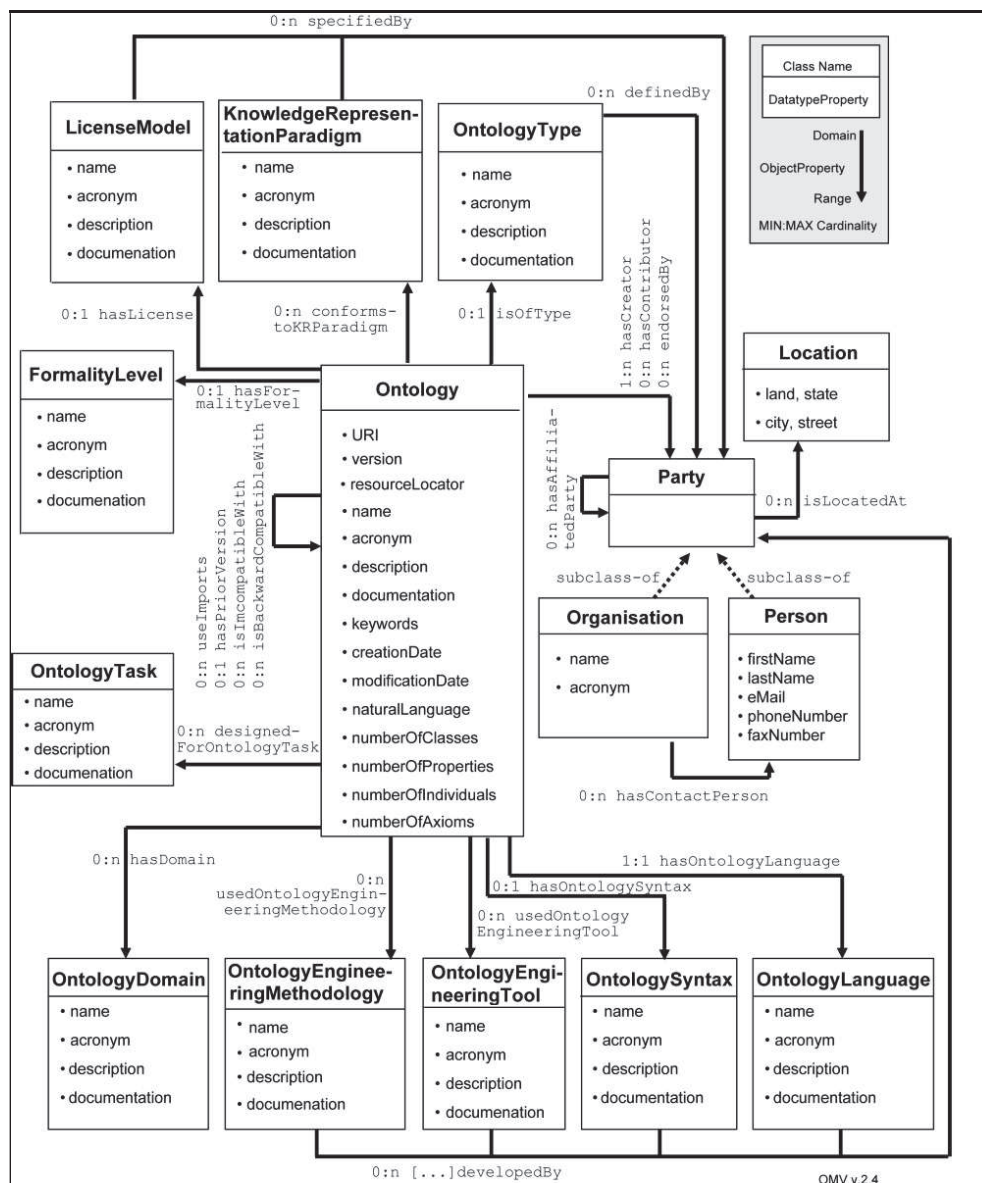


Figure 2.12.: An overview of the OMV model. Source :[29].

jOWL is a tool aimed at navigating and visualising OWL-RDFS documents. Usage is simple, one has to simply edit the well-documented HTML file to load an OWL file using a command and choose which content (classes, properties, individuals, and SPARQL queries) to include in the output. Thereafter, when the file is run in a browser, one may view and navigate through the ontology. jOWL provides two views for browsing, tree view and navigation bar. BFO uses jOWL for online browsing. Fig. 2.14 displays this.

Meta-Data field	OAEI Fields	OM ² R - Meta-Data Fields
Name of ontologies	Text (A) Field (S)	SourceOntology :Om2r:human_readable_name: "Biology Top Level Ontology"
Description of ontologies	Text (S) Field (S)	Om2r:description
Location of ontology	Text (A)	Om2r:hasLocation (type url)
Creation date of ontologies	Field (S)	Om2r:hasCreationDate (type date)
Unique identifier for ontologies	Field (A)	Om2r:hasIdentifier
Ontology Version	Missing	Om2r:hasVersion (URI)
Complexity of the ontology	Text (S)	Om2r:hasClassCount 73, hasInstanceCount 3 hasPropertyClass 3
Design of the ontologies	Text (S)	Om2r:hasDesign om2r:deep_hierarchy.
Notation of Ontologies	Text (S)	Om2r:hasNotation RDF/XML
Formal Language of Ontologies	Text (S)	Om2r:hasFormalLangauge OWL
Matching Location	Text (A)	Matching Om2r:hasLocation: www (URL)
Formal Language of the Matching	Text (S)	Om2r:hasformalMatchingLanguage: EDOAL
Notation of the Matching	Missing	Om2: hasNotation: RDF/XML.
Matching Method	Missing	Om2r:hasMethod (manual, automatic, mixed)
Matching Tool	Missing	Om2r:isTool AlignmentServer
Matching Algorithm	Missing	Algorithm :encodedIn: Java, Algorithm :hasJavaClass: org.stringComp, Algorithm :hasSource: freecode.org/a.zip
Algorithm is based on	Missing	Om2r:isBasedOn rdfs:label, rdfs:class
Applied Threshold	Missing	Om2r:has_Applied_Threshold
Matching Scope	Missing	Om2r:hasScope (complete or partial)
Matching Requirements	Missing	Om2r:hasMatchRequirements (text)

Figure 2.13.: The OM²R metadata model fields compared to the OAEI fields. Source: [84].

OntologyOnline

Basic Formal Ontology (BFO) Browser

Treeview: ☒ Navigation Bar: ☐

Treeview

- entity
 - occurent
 - processual_entity
 - spatiotemporal_region
 - temporal_region

Description of occurrent

Definition: An entity [bfo:Entity] that has temporal parts and that happens, unfolds or develops through time. Sometimes also called perdurants.

Examples: the life of an organism, a surgical operation as processual context for a nosocomical infection, the spatiotemporal context occupied by a process of cellular meiosis, the most interesting part of Van Gogh's life, the spatiotemporal region occupied by the development of a cancer tumor

Synonyms: perdurant

Disjoint With:
[continuant](#)

Figure 2.14.: BFO in jOWL. Source: The jOWL website¹.

WebProtégé is a lightweight ontology editor, aimed at providing assistance in distributed ontology development. Users are able to read and write to an ontology online, without having to

¹<http://jowl.ontologyonline.org/bfo.html>

install additional software. The interface of WebProtégé is easy to use and understand, and users are able to configure it to their liking.

WebProtégé, in addition to allow online ontology browsing and editing, is a collaborative tool, allowing customized ontology views for different users. The predefined tabs in WebProtégé include Classes, Properties and Individuals tabs. Within tabs, there are portlets that provide functionality for a particular tab. e.g., class tree view, properties, asserted conditions, notes etc. These portlets support creating, deleting and searching entities (classes, properties and individuals).

2.8.1. Evaluating ontology browsing tools

We evaluate jOWL and WebProtégé for online ontology browsing in terms of the following criteria:

- **Ontology visualisation:** The structural view that the ontology visualised in e.g., tree view.
- **Ontology building blocks:** The ontology entities that are represented in the tool e.g., classes, properties, individuals.
- **Import statements:** Whether ontology import statements are processed.
- **Queries:** Ontology queries that are supported by the tool e.g., DL, SPARQL queries.
- **Operations:** The operations that a user may perform to the ontology e.g., view, edit.
- **Ontology metadata:** Whether administrators may specify metadata for the ontology.
- **Multiple tabs:** Whether multiple ontologies can be opened simultaneously.
- **Customisable:** Whether the interface is customisable for users.

The results for the evaluation based on this criteria is shown in Table 2.3.

Table 2.3.: An evaluation of the ontology browsing tools

	jOWL	WebProtégé
Ontology visualisation	Tree view and navigation bar	Tree view
Ontology building blocks	Classes, properties and individuals	Classes, properties and individuals
Import statements	-	Yes
Queries	SPARQL queries	-
Operations	View	View and edit
Ontology metadata	-	Yes
Multiple tabs	-	Yes
Customisable	-	Yes

From this comparison of ontology browsing tools, we observe that WebProtégé provides more functionality for most of the criteria used in this evaluation. Particularly, import statements, which will be used in the ontologies of the repositories are supported by WebProtégé and not by jOWL. Therefore we have selected WebProtégé to be used for ontology browsing.

2.9. Ontology view tools

To assist the user with understanding foundational ontologies, it is useful to provide the axioms of each ontology in a human-readable format. We require some infrastructure to generate different views for each ontology. We explore existing tools that provide this.

OWL verbalizer, based on existing work on Attempto Controlled English (ACE) [38], accepts input as OWL/XML and converts it into Attempto Controlled English (ACE). ACE is a subset of English that follows some rules and has a limited syntax. Verbalisation is performed by simplifying ontology axioms to basic expressions and thereafter mapping them to defined ACE constructs. Fig. 2.15 displays some of the mappings between OWL and ACE that are used to convert between the two. A user has the choice of ACE output in text, in a HTML table aligning original axioms to ACE concepts or in a CSV file tokenized and without lexicon lookup.

OWL	ACE
<i>Named property</i>	<i>Transitive verb, e.g. own</i>
<code>ObjectInverseOf(R)</code>	<i>Passive verb, e.g. is owned by</i>
<i>Named class</i>	<i>Common noun, e.g. person</i>
<code>owl:Thing</code>	something, thing, X, Y, ...
<code>ObjectComplementOf(C)</code>	something that is not a person; something that does not own a car
<code>ObjectIntersectionOf(C1 ... Cn)</code>	something that is a person and that owns a car
<code>ObjectUnionOf(C1 ... Cn)</code>	something that is a wild-animal or that is a zoo-animal
<code>ObjectOneOf(a)</code>	<i>Proper name, e.g. John</i>
<code>ObjectSomeValuesFrom(R C)</code>	something that owns a car
<code>ObjectHasSelf(R)</code>	something that likes itself
<code>ObjectMinCardinality(n R C)</code>	something that borders at least 2 countries
<code>SubClassOf(C D)</code>	Every country that borders no bodies-of-water is a landlocked-country.
<code>SubObjectPropertyOf(ObjectPropertyChain(R1 ... Rn) S)</code>	If X owns something that is-part-of Y then X owns Y.
<code>DisjointObjectProperties(R S)</code>	If X is-child-of Y then it is false that X is-spouse-of Y.

Figure 2.15.: Mappings between OWL and ACE constructs from OWL verbalizer. Source: The OWL verbalizer website¹.

SWAT natural language tools [83] accepts input in the form OWL/RDF or OWL/XML and is able to convert it to a number of formats. The output formats are: Alphabetical English Glossary, English Sentences, Prolog Terms, Lexicon, Axiom Patterns, and OWL/XML, with verbalisations as SWATDescription annotations.

SWAT natural language tools provided many views for the ontology axioms. For ROMULUS, the Alphabetical English Glossary will be used. The Alphabetical English Glossary view contains the classes, individuals and object properties of the OWL ontology, provided in alphabetical order; each coupled with its typology, description and distinctions. This view is neat and human-readable as it allows for easy ontology understanding.

Protégé v4.1 has a feature to convert ontologies from its ontology language format to that of description logic. This is performed by saving the OWL file as a latex file, and thereafter

¹http://attempto.ifi.uzh.ch/site/docs/owl_to_ace.html

generating a file of the description logic axioms in PDF format. In this way, we are able to add the PDF file of description logic axioms to ROMULUS. This may be useful in cases where users wish to view ontology axioms in description logic format.

2.9.1. Evaluation of ontology view tools

The verbaliser in SWAT natural language tools has been evaluated with the Experimental Factor Ontology (EFO) [51]. It was found that SWAT natural languages tool satisfactorily conveyed the intended meaning of the ontology. It was also found that human-readable verbalisation format was preferred over other ontology verbalisation.

OWL verbaliser has been evaluated with Hydrology ¹ and GALEN [69] ontologies. The hydrology is a complex, large ontology containing 1815 axioms while the GALEN ontology is a complex, enormous medical ontology containing 37 696 axioms. In both these ontologies, less than 1% of axioms could not be verbalised. Performance is good; the Hydrology ontology was verbalised in less than 1 second and the GALEN ontology was verbalised in 30 seconds.

The theoretical and practical aspects explored in this chapter addresses the research objectives introduced in Section 1.3. Applying these fields to create a repository of aligned foundational ontologies will enable us to investigate and overcome the issues posed in foundational ontology interchangeability, and improve semantic interoperability with regard to foundational ontologies.

Materials and Methods

The proposed library, ROMULUS, will be developed as a web-based application using web markup and scripting languages for display and operations. The web-based application, will be created with the following languages and tools: HTML, PHP, MySQL, SWAT natural language tools [83], ONSET [45], Tomcat, and WebProtégé [85].

3.1. Methodology

The traditional waterfall method is used as an approach for the software development process of the proposed repository. The tasks to be performed are outlined here:

1. Perform a literature review on topics including: The proposed WFOL, official foundational ontology publications, comparative studies of popular foundational ontologies, existing ontology repositories, ontology modularisation, ontology mediation, ontology metadata, ontology browsing tools, and ontology verbalisation and view.
2. Identify functional and non-functional requirements of the repository.
3. Select widely used foundational ontologies to implement in the repository. Motivate for these choices.
4. Perform a content comparison of the selected foundational ontologies. This involves identifying similarities and differences between foundational ontologies.
5. Perform modularity of the foundational ontologies using modularity tools and manually:
 - Separate 3D entities from 4D entities in the ontologies for modules.
 - Create OWL 2 profiles modules.
 - Create more/less-detailed modules.
6. Acquire different views of the foundational ontologies by using existing tools.

7. Perform ontology mediation of the foundational ontologies:

- Perform ontology alignment. Identify approximate alignments as follows:
 - Use existing ontology mediation tools to automatically and semi-automatically identify alignments.
 - Use documentation to find existing ontological alignments between foundational ontologies.
 - Perform alignment manually. Use content comparison performed earlier to specify relations for similar entities and relational properties.
- Perform ontology mapping: Specify mappings between foundational ontologies based on the alignments from the previous step. Identify ontological inconsistencies that may arise as follows:
 - For candidate class mappings:
 - * Run a reasoner.
 - * Check if there are any unsatisfiable classes.
 - * If there are unsatisfiable classes, use the reasoner explanation feature to generate an explanation.
 - * Analyse explanations.
 - * Remove inconsistent mappings.
 - For candidate object property mappings, since object property inconsistencies and flaws are not properly recognised by reasoners [43], perform the following tasks:
 - * Check if the domain and range restriction in an object property alignment conflicts by using the above method for class alignments.
 - * If an object property does not have domain and range restrictions, check if it is the subproperty of another object property by looking at the object property hierarchy. If it is a subproperty of another, check if its superproperty has domain and range restrictions. The domain and range restrictions of an object property's superproperty is inherited by that object property.
 - * Check if its domain and range restrictions conflicts with its aligning object property domain and range restrictions by using the above method for class alignments.

Attempt to provide a solution for inconsistencies by changing an equivalence relation to a subsumption relation.

- Perform ontology merging: Create merged ontologies from the foundational ontologies.

- Create a high-level foundational ontology based on high-level common entities from the foundational ontologies.
 - Create a method to be used together with other mediation outputs to assist the user with foundational ontology interchangeability.
8. Compile a general metadata list and gather values for each foundational ontology module.
 9. Design a software infrastructure meeting research objectives in order to assist the user with foundational ontology interoperability and linking.
 10. Select and implement an ontology browsing tool to facilitate online browsing.
 11. Design and perform an evaluation of the software. Refer to Section 3.2.
 12. Perform modification of the developed software and documentation if deemed necessary based on the results of the evaluation.

A flow of these tasks is provided in Fig. 3.1.

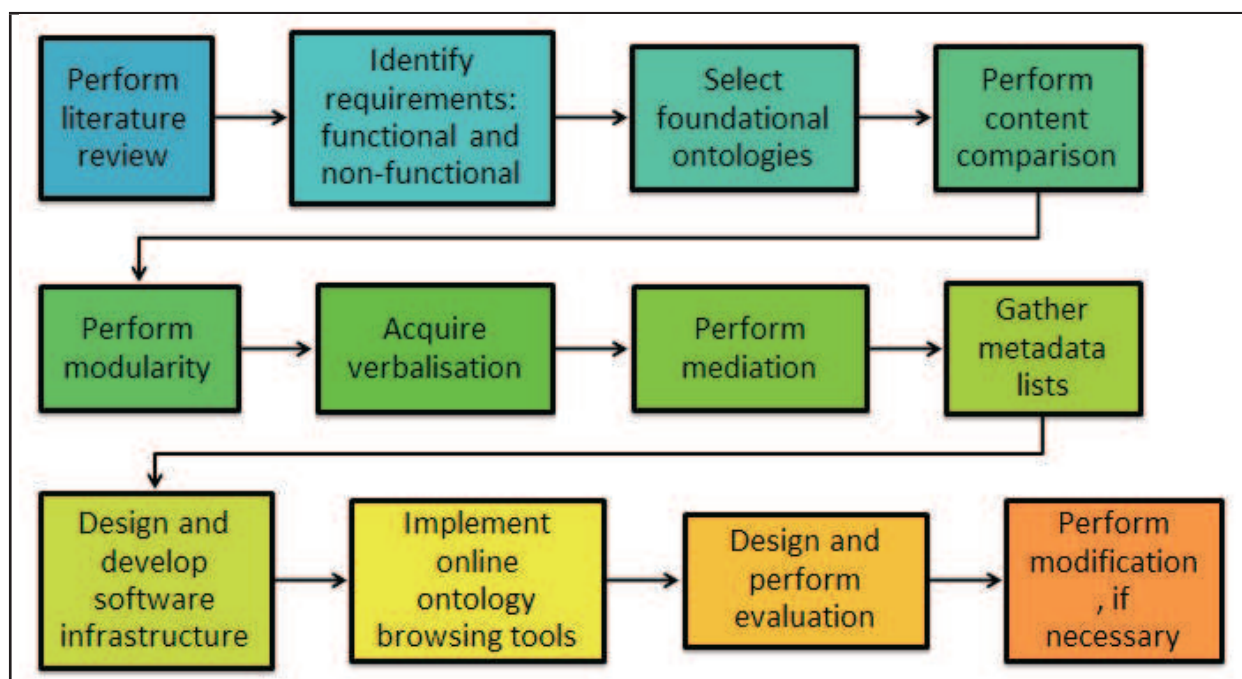


Figure 3.1.: Flow of the materials and methods.

3.2. Evaluation technique

Evaluation techniques have been designed to assess the functional requirements and the ontological alignment sets of ROMULUS. It is to be evaluated in three ways:

1. **Evaluate foundational ontology interchangeability:** Using the mappings, merged ontologies and foundational ontology interchangeability method, convert a domain ontology linked to a particular foundational ontology to another foundational ontology.
2. **Evaluate ontological alignments by users:** The alignments between foundational ontologies that were created in the ontology mediation process will be provided to participants to assess. Participants to be are provided with, an annotation of each entity and a number of options, for each alignment. The options, with their meanings in terms of an example alignment between classes Chips and Crisps are provided below:
 - **Agree:** I agree that Chips and Crisps are exactly the same thing.
 - **Partially agree:** Chips and Crisps are sometimes the same thing. However, if the chips we are talking about refers to french fries, this is different.
 - **Disagree:** Chips and Crisps are two totally different things.
 - **Unsure:** After thinking about this alignment, I still do not know.
 - **Skip:** I do not wish to answer this.
3. **Evaluate functionality by comparison with other ontology repositories:** ROMULUS's functions will be compared against those of other ontology repositories with respect to the following functions: browse, mediation, search, metadata, ontology selection, ontology views, ontology comparison, and ontology access.

3.3. Foundational ontologies for the repository

DOLCE, GFO and BFO have been selected to be implemented in the repository. We also select related modules of these foundational ontologies. For DOLCE, we include the FunctionalParticipation, SpatialRelations and TemporalRelations modules. For BFO, we include the BFO with RO ontology. For GFO, we include the GFO-Basic module. DOLCE, GFO and BFO are popular and up-to-date ontologies. The size and dimensions of these ontologies appear suitable to enable a thorough understanding of them and identify alignments between them. The ontological criteria of each foundational ontology such as philosophical choice, representation languages etc. differ; therefore the library will be able to cover different commitments, philosophies and purposes.

At present, the OWL formalisation of each foundational ontology will be used only. In order to enhance semantic interoperability with foundational ontologies, we need a language that is usable on the Semantic Web, therefore we use OWL ontology language which is easily interpretable by machines and usable by Semantic Web applications. While the FOL versions of the ontologies are more expressive and accurate, these versions cannot be practically used for Semantic Web applications. Furthermore, ontology developers more commonly use the OWL versions of the foundational ontologies and not the FOL versions. We would like to use DOL ontology language for representation and to assist with achieving semantic interoperability and integration, but it is

currently under development and still being standardised. It will only be an international standard as of August 2015.

In future, other ontology language representations of the foundational ontologies will be included. Also, other foundational ontologies will be included in the proposed repository.

Ontology Mediation

Ontology mediation [16], in general, is concerned with determining and overcoming differences between ontologies. It is a key task in achieving semantic interoperability. It is made up of three processes: alignment, mapping and merging. Alignment deals with identifying correspondences between entities, independent of the ontology. To assist with alignment, we perform a content comparison of the foundational ontologies. Ontology mapping deals with creating correspondences between ontologies based on alignments such that the resultant ontology is still consistent and does not have unsatisfiable classes or relations. In merging, a new merged ontology is created from the original ontologies.

We provide several definitions on ontology mediation in this section, which are taken from [20], Chapter 3. Firstly, in order to determine the alignments, there is the matching process:

Definition 1 (Matching Process) *The matching process can be seen as a function f which, from a pair of ontologies to match o and o' , an input alignment A , a set of parameters p and a set of oracles and resources r , returns an alignment A' between these ontologies: $A' = f(o, o', A, p, r)$.*

In order to define an alignment or mapping, the term “entity language” has to be introduced, which is used to express precisely those entities that will be matched.

Definition 2 (Entity language) *Given an ontology language L , an entity language Q_L is a function from any ontology $o \subseteq L$ which defines the matchable entities of ontology o .*

A correspondence consists of a relation between two entities in different ontologies, which is uniquely identified and has some confidence value assigned to it and has to be found by matching algorithms. An example of a correspondence is the relation of automobile being equivalent to car with a confidence value of 1.0.

Definition 3 (Correspondence) *Given two ontologies o and o' with associated entity languages Q_L and $Q_{L'}$, a set of alignment relations θ and a confidence structure over Ξ (a degree of confidence), a correspondence is a 5-tuple: $\langle id, e, e', r, n \rangle$, such that id is a unique identifier of the given correspondence, $e \subseteq Q_L(o)$ and $e' \subseteq Q_{L'}(o')$, $r \subseteq \theta$, and $n \subseteq \Xi$.*

From these definitions, alignment is described as the process of specifying correspondences between entities, by using a particular alignment relation, such as equivalence, subsumption, or a predefined similarity relation. An alignment is therefore a set of correspondences between two ontologies. An example alignment contains the following relations: candy is equivalent to sweet with a confidence value of 1.0, biscuit is equivalent to cookie with a confidence value of 1.0, feast is equivalent to banquet with a confidence value of 0.8, spoonful is equivalent to dollop with a confidence value of 0.5, and pastry is equivalent to dough with a confidence value of 1.0.

Definition 4 (Alignment) *Given two ontologies o and o' , an alignment is made up of a set of correspondences between pairs of entities belonging to $Q_L(o)$ and $Q_{L'}(o')$ respectively.*

In ontology mapping, the correspondences that were earlier identified are now declared in the ontology file without inconsistencies, and in merging a new ontology is created with the mappings and original ontology. Euzenat and Shvaiko [20] describe mappings with respect to models of the ontologies and a number of other definitions. Therefore, we capture the gist of these terms in the following definitions, using Euzenat and Shvaiko's notational conventions.

Definition 5 (Mapping) *Given two ontologies o and o' , a mapping is made up of a set of correspondences between pairs of entities belonging to $Q_L(o)$ and $Q_{L'}(o')$, respectively, and this mapping is satisfiable and does not lead to an unsatisfiable entity in either o or o' .*

Definition 6 (Merging) *Given two ontologies o and o' , a merging is the creation of a new ontology o'' containing o and o' and all mappings between entities belonging to $Q_L(o)$ and $Q_{L'}(o')$ such that o'' does not have unsatisfiable entities and is consistent.*

4.1. Foundational ontology content comparison

In order to proceed forward with alignment, it is necessary to compare the content of each foundational ontology. Here we perform an informal content comparison of the foundational ontologies by identifying differences and similarities between the ontologies. The ontologies' philosophies and ontological commitments have been discussed and compared previously in Section 2.3.

4.1.1. Similarities and differences between DOLCE and BFO

DOLCE and BFO use different entity names for describing 3D and 4D entities. DOLCE names these entities *endurant* and *perdurant* while BFO names them *Continuant* and *Occurrent*. Some synonyms exist between DOLCE and BFO e.g., DOLCE's *space-region* vs. BFO's *SpatialRegion*, DOLCE's *physical-endurant* vs. BFO's *MaterialEntity*. Entities that share the same meaning and name in both ontologies are *quality* and *process*. While DOLCE and BFO do have similar structure at a high-level in that both have separating branches of 3D and 4D entities, a number of

other aspects of structure are different. DOLCE's separate *endurant* and *perdurant* branches are linked by participation relations while BFO's branches are completely independent to each other. DOLCE's *quality* branch is disjoint to its *endurant* and *perdurant* branches. BFO, on the other hand, subsumes *Quality* entities under its *Continuant* branch. BFO's temporal entities, including temporal regions and are subsumed by *Occurrent*, while DOLCE's temporal entities are split up into three parts, temporal regions which are subsumed by abstract entities, temporal qualities which are subsumed by quality entities and subclasses of *perdurants*. DOLCE has abstract objects while BFO does not. Entities in DOLCE are of type *particular* while entities in BFO are of type *Universal*. Included in DOLCE, are relational properties. BFO does not have relational properties included in the ontology, but rather as a separate ontology, the *Relational Ontology* (RO) [80]. BFO 2.0 is currently being developed, whereby BFO is integrated with RO. For mereology, DOLCE adopts the axioms of GEM. This includes *parthood*, *proper part*, *overlap*, *strong supplementation*, and *unrestricted fusion*. BFO core is a first-order logic based representation of its mereology. It contains collections, sums and universal axioms.

4.1.2. Similarities and differences between BFO and GFO

BFO and GFO use the same entity names for describing 3D and 4D entities. Both BFO and GFO use the terms *Continuant* and *Occurrent*. GFO additionally uses the terms *Presential* and *Persistent* for describing such entities. Other entities that share the same name and meaning include: *Entity*, *Role*, *Function*, and *Process*. To describe entity properties, BFO uses the term *Quality* while GFO uses the term *Property*. Some synonyms exist between the two ontologies e.g., BFO's *SpatialRegion* vs. GFO's *Spatial_region*, BFO's *MaterialEntity* vs. GFO's *Material_Persistent*. GFO contains both entities of type *Individuals* and *Universal* while BFO contains only those of type *Universal*. The organisation of entities within BFO and GFO differ greatly. GFO's spatial and temporal entities are subsumed by its *Space-time* entity. BFO's spatial entities are subsumed by its *Continuant* while its temporal entities are subsumed by its *Occurrent*. GFO has abstract entities while BFO does not. Included in GFO, are relational properties. As mentioned above, BFO, on the other hand, does not have relational properties included in the ontology, but rather as a separate ontology of relations. Both BFO and GFO use their own mereology. BFO's mereology is discussed in Section 4.1.2. GFO's mereology contains the following axioms: *antisymmetry*, *transitivity*, *set inclusion*, *proper parthood*, and other GFO-specific axioms based on these.

4.1.3. Similarities and differences between GFO and DOLCE

GFO and DOLCE use different entity names for describing 3D and 4D entities. GFO names these entities *Continuant*, *Occurrent*, *Presential*, and *Persistent* while DOLCE names them *endurant* and *perdurant*. To describe entity properties and their values, DOLCE uses the terms *quality*, *quale* and *quality-space* while GFO uses *Property*, *Property_value* and *Value_space*. DOLCE's

amount-of-matter and GFO's Amount_of_substrate refer to the same type of entity. Some synonyms exist between the two ontologies e.g., GFO's Temporal_region vs. DOLCE's temporal-region, GFO's Spatial_region vs. DOLCE's space-region. Entities that share the same name and meaning in both ontologies are process, state, abstract, and set. GFO contains entities of both type Individual and of type Universal while DOLCE contains only entities of type particular. The organisation of entities within GFO and DOLCE differ greatly. DOLCE's spatial and temporal entities are subsumed by its abstract entity while GFO's spatial and temporal entities are subsumed by its space-time entity which is completely disjoint to its abstract entity. Both DOLCE and GFO have relational properties. DOLCE's relational properties are all based on either of its six primitive relations: parthood, temporary parthood, constitution, participation, quality, and quale. For mereology, DOLCE uses GEM while GFO uses its own mereology. DOLCE's mereology is discussed in Section 4.1.1 while GFO's mereology is discussed in Section 4.1.2.

The content comparison between each pair performed here will aid in creating ontological alignments. DOLCE, BFO and GFO have different taxonomic structures. In some cases, entities that seem similar fall in contradicting or disjoint classes. These differences in structure and organisation may cause inconsistencies when performing mapping.

4.2. Alignment

Ontology alignment is the process of identifying similarities between ontologies. We have decided to ignore the underlying philosophies of each foundational ontology or it would result in few or no alignments; e.g., DOLCE is descriptive and an ontology of particulars, while BFO is realist and an ontology of universals. If we had taken this into consideration for alignment, there would be no alignments between the two ontologies. At present, we focus only on aligning classes and relational properties with equivalence relations. Subsumption relations will be used at a later stage to resolve mapping inconsistencies.

A list of ontology pairs for which we identify alignments follows. In total 20 pairs of alignments were created, each consisting of DOLCE-Lite, BFO, GFO, or related ontology modules. By BFORO, we mean the merged ontology of BFO with the RO mentioned in Section 4.1.1. It must be noted that GFO-Basic has some entities that do not exist in GFO and therefore is not a subset of GFO. BFO, however, is a subset of BFORO.

- BFO $\leftrightarrow$ GFO-Basic
- BFO $\leftrightarrow$ GFO
- BFORO $\leftrightarrow$ DOLCE-Lite
- BFORO $\leftrightarrow$ FunctionalParticipation
- BFORO $\leftrightarrow$ GFO-Basic
- BFORO $\leftrightarrow$ GFO
- BFORO $\leftrightarrow$ SpatialRelations
- BFORO $\leftrightarrow$ TemporalRelations
- DOLCE-Lite $\leftrightarrow$ BFO

- DOLCE-Lite $\leftrightarrow$ GFO-Basic
- DOLCE-Lite $\leftrightarrow$ GFO
- FunctionalParticipation $\leftrightarrow$ BFO
- FunctionalParticipation $\leftrightarrow$ GFO-Basic
- FunctionalParticipation $\leftrightarrow$ GFO
- SpatialRelations $\leftrightarrow$ BFO
- SpatialRelations $\leftrightarrow$ GFO-Basic
- SpatialRelations $\leftrightarrow$ GFO
- TemporalRelations $\leftrightarrow$ BFO
- TemporalRelations $\leftrightarrow$ GFO
- TemporalRelations $\leftrightarrow$ GFO-Basic

4.2.1. Accurate alignments

4.2.1.1. Identifying accurate alignments

In order to identify accurate alignments, we use the method from Section 3.1. This involves:

- Using existing tools to identify accurate alignments. LogMap, YAM++, HotMatch, Her-tuda and Optima have been evaluated with positive results by the Ontology Alignment Evaluation Initiative (OAEI) in terms of their precision, recall and other performance measures.
- Use documentation to find existing ontological alignments between foundational ontologies. We look at the official GFO publication [30] which contains some rough mappings between DOLCE and GFO, Temal et al.'s publication [82] which contains some alignment between DOLCE and BFO and Seyed's publication [75] which compares DOLCE and BFO relations.
- Perform alignment manually. Use content comparison performed earlier to identify ontological alignments.

For each resource (tool, documentation or manual alignment), we measure its *accuracy* by firstly examining each of its output alignments to determine whether or not the equivalence relation is correct. Thereafter we define accuracy as the number of 'correct' alignments over the total alignments given by the resource (Eq. 4.1), where 'correct' denotes the alignment is also in the set of alignments found manually, i.e., what is typically considered as the 'gold standard'. We define the *found* measure of the resources as the number of correct alignments over the total possible correct alignments, after manual intervention (Eq. 4.2).

$$Accuracy = \frac{|correct\ alignments|}{|total\ alignments_{resource}|} \times 100 \quad (4.1)$$

$$Found = \frac{|correct\ alignments|}{|total\ alignments_{gold}|} \times 100 \quad (4.2)$$

4.2.1.2. Results and discussion of the automated alignments

Table 4.1 lists the numbers of the alignments from the documentation and manually. The manual alignments were aided by the GFO documentation [30] and checked against the alignments proposed by other literature [82, 75]. The GFO documentation [30] contains a list of similarities between GFO and DOLCE which helped with the alignment process. Some of the alignments could not be used, however, due to changes in the two foundational ontologies. We were able to use 42% of the alignments from the documentation. Seyed [75] examined only three relations—dependency, quality, and constitution—and found that they are different in DOLCE and BFO. Temal et al. [82] aligned the FOL versions of DOLCE and BFO and subsequently we were only able to consider four relations that exist in the current OWL versions of the ontologies. We discuss these four equivalence alignments here.

Table 4.1.: Comparison of manually performed alignment accuracies of the GFO documentation, related works, and ours, and aggregates for mappings.

	Seyed	Herre	Temal et al.	Ours
<i>Class alignments</i>				
DOLCE-Lite ↔ BFO	-	-	2/7	9/9
BFO ↔ GFO	-	-	-	13/13
GFO ↔ DOLCE-Lite	-	13/31	-	16/16
<i>Object property alignments</i>				
DOLCE-Lite ↔ BFO	0	-	-	8/8
BFO ↔ GFO	-	-	-	10/10
GFO ↔ DOLCE-Lite	-	0	-	19/19
Overall alignments				
Total	0/0	13/31	2/7	75/75
Accuracy	0%	42%	29%	100%
Found	0%	37%	12%	100%
Overall mappings				
Total	0/0	8/31	1/7	40/40
Accuracy	0%	26%	14%	100%
Found	0%	61%	9%	100%

- **BFO:ProcessualEntities ↔ DOLCE:perdurant:** We changed the alignment to BFO:Occurrent ↔ DOLCE:perdurant, because by definition occurrents and perdurants both represent entities that have temporal parts and unfold in time.
- **BFO:Quality ↔ DOLCE:physical-quality:** This alignment is more precise than ours, because, we chose to ignore some philosophies (the realist debate) with the hope of achieving a higher number of alignments. That is, our mapping has BFO:Quality ↔ DOLCE:quality, thereby ignoring the fact that BFO does not consider abstract entities.
- **BFO:SpatialRegion ↔ DOLCE:space-region:** We agree with this alignment and use this relation.
- **BFO:TemporalRegion ↔ DOLCE:temporal-region:** We agree with this alignment and use this relation.

Table 4.2.: Comparison of alignment accuracies of the matching tools and aggregates for mappings.

	H-Match	PROMPT	LogMap	YAM++	Hot Match	Hertuda	Optima
<i>Class alignments</i>							
DOLCE-Lite ↔ BFO	4/16	3/8	2/2	4/4	3/3	3/3	4/12
BFO ↔ GFO	5/31	7/12	7/8	6/7	7/7	7/7	8/14
GFO ↔ DOLCE-Lite	4/25	4/8	3/3	8/11	5/5	5/5	5/16
<i>Object property alignments</i>							
DOLCE-Lite ↔ BFO	0	0	0	0	0	0	0/1
BFO ↔ GFO	0	0	4/4	0	0	0	1/3
GFO ↔ DOLCE-Lite	0	4/4	0	5/14	5/7	6/8	2/23
Overall alignments							
Total	13/72	18/32	16/17	23/36	20/22	21/23	20/69
Accuracy	18%	56%	94%	64%	91%	91%	29%
Found	17%	24%	21%	31%	27%	28%	27%
Overall mappings							
Total	10/72	11/32	16/17	15/36	11/22	12/23	13/69
Accuracy	14%	34%	94%	42%	50%	52%	19%
Found	25%	28%	40%	38%	28%	30%	33%

Table 4.2 lists the numbers of alignments found by the selected tools. We describe some further data in the remainder of this section. H-Match generated many alignments, but most of the output was not accurate. Many entity pairs that were matched using H-Match were found to be incorrectly aligned; e.g., DOLCE-Lite:quale ↔ BFO:Role. This resulted in us being able to use only 18% of these alignments, with the rest being false positives. PROMPT was generally unstable resulting in force closure of the application. It generated suggestions of which we were able to use 56%, with the rest being false positives; e.g., BFO:Site ↔ GFO:Situoid that indicates a string matching limitation.

While LogMap provided few alignments between the foundational ontologies (less than ten in all cases), the alignments were accurate and thus we were able to use almost all of the alignments. The one false positive in LogMap was the alignment of BFO:IndependentContinuant ↔ GFO:Independent. YAM++ generated many alignments. However, while most of the alignments for DOLCE ↔ BFO and BFO ↔ GFO were accurate, only about half were accurate for GFO ↔ DOLCE. Overall we were able to use almost 64% of its alignments. Like LogMap, YAM++ also incorrectly aligned BFO:IndependentContinuant ↔ GFO:Independent. HotMatch generated a fair amount of alignments between the ontologies. Overall, we were able to use 91% of HotMatch’s alignments, with just 2 alignments out of all 22 being false positives. Hertuda’s output was surprisingly similar to HotMatch’s output, with just one more alignment than HotMatch. We were able to use 91% of HotMatch’s alignments, with just 2 alignments out of all 23 being false positives. A common false positive in YAM++, Hertuda and HotMatch was the alignment between DOLCE-Lite:part ↔ GFO:has part, which is discussed in Section 4.2.1.4. Optima generated many alignments for each pair. However, there were many false positives. This may be

because Optima is aimed at aligning large ontologies and the foundational ontologies in question are of reasonable size.

Concerning feasibility to carry out automated alignments, in most cases, the tools evaluated with the OAEI performed better than the others, with the exception of Optima. LogMap had the highest accuracy, because it also considers the logic-based semantics of the ontologies and uses reasoning-based techniques throughout the process, therewith eliminating those false positives that would have led to a logical inconsistency. However, LogMap generated very few alignments compared to other accurate tools (YAM++, Hertuda and HotMatch), indicating that the additional heuristics implemented are too strict at least for foundational ontology alignment.

Most false positive alignments generated by the tools, such as `bfo:IndependentContinuant` $\leftrightarrow$ `gfo:Independent`, indicate that the algorithms implement syntactic matching, which, based on the results we obtained, is not sufficient or suitable for foundational ontology matching because many entities have a common syntax e.g., `dolce:quality-space` $\leftrightarrow$ `gfo:Space` both have the string ‘space’ in common but are entirely different entities. Table 4.3 includes a selection of such false positives that are caused by syntactic matching in the tools when aligning the three foundational ontologies. The tools failed to recognise simple alignments such as `dolce:perdurant` $\leftrightarrow$ `gfo:Occurrent`, `bfo:Quality` $\leftrightarrow$ `gfo:Property`. In this sense, semantic matching is not considered, or if it is, it fails to recognise synonyms of the philosophical scope on which foundational ontologies are built upon. Structural matching is not an effective method either, due to the fact that the hierarchies and structures of the foundational ontologies differ greatly which causes the root distances of mappable entities to differ. For aligning foundational ontologies, it will be useful if existing semantic matchers would include something alike a ‘philosophy WordNet’ that specialises in philosophical terms, synonyms, and definitions used in foundational ontologies. Refer to Appendix B to view the complete alignment output of the tools and documentation.

Table 4.3.: False positives caused by syntactic matching generated by the alignment tools; the terms in *italics* represent the strings that are common between aligned entities.

DOLCE-Lite	BFO
<i>physical-region</i>	<i>ConnectedSpatioTemporalRegion</i>
<i>non-physical-object</i>	<i>Object</i>
<i>region</i>	<i>SpatioTemporalRegion</i>
BFO	GFO
<i>IndependentContinuant</i>	<i>Independent</i>
<i>Site</i>	<i>Situoid</i>
<i>Continuant</i>	<i>Continuous</i>
GFO	DOLCE-Lite
<i>has_sequence_constituent</i>	<i>generic-constituent</i>
<i>has-part</i>	<i>part</i>
<i>Space</i>	<i>quality-space</i>

The results of the tool analysis is a good indication of which tools to experiment with for foundational ontology alignment in general. However, they found less than a third of the actual alignments at this stage, and therefore it is still vital to perform manual alignment for foundational ontologies. The tools also did not generate subsumption relations for any of the alignments, but this could perhaps be an extension to the basic idea of LogMap by means of another call to the reasoner. One could investigate whether Optima may be used to identify accurate alignments among the larger foundational ontologies SUMO [64] and YAMATO [56].

4.2.1.3. Results and discussion of the manual alignments

Since the tools and documentation did not assist completely with a full set of alignments for any given pair, it was necessary to manually match similar entities from ontologies by looking at the annotations, axioms and names of entities. The yield of the manual alignments between the main foundational ontologies (DOLCE-Lite, BFO and GFO) resulted in 35 alignments for GFO $\leftrightarrow$ DOLCE-Lite, 23 alignments for BFO $\leftrightarrow$ GFO, and 17 alignments for DOLCE-Lite $\leftrightarrow$ BFO, hence, 75 in total, displayed in Tables 4.7, 4.5, 4.6 below. There are 14 alignments common between these ontologies, which is displayed in Table 4.4. In these tables, the alignments numbered in bold font are those that can be mapped, which is further discussed in Section 4.3. In total, from the 20 alignment pairs, there are 85 distinct alignments. Naturally, there are many more than 85 alignments if we consider identical alignments that occur among the same entities in related modules; e.g., DOLCE-Lite:process $\leftrightarrow$ GFO:Process and DOLCE-Lite:process $\leftrightarrow$ GFOBasic:Process. Alignments between other pairs are available in Appendix A.

Table 4.4.: Common alignments between DOLCE-Lite, BFO and GFO.

	DOLCE-Lite	BFORO	GFO
Class			
1.	endurant	Independent Continuant	Presential
2.	physical-object	Object	Material_object
3.	perdurant	Occurrent	Occurrent
4.	process	Process	Process
5.	quality	Quality	Property
6.	space-region	SpatialRegion	Spatial_region
7.	temporal-region	Temporal-Region	Temporal_region
Relational property			
1.	proper-part	has_proper_part	has_proper_part
2.	proper-part-of	proper_part_of	proper_part_of
3.	participant	has_participant	has_participant
4.	participant-in	participates_in	participates_in
5.	generic-location	located_in	occupies
6.	generic-location-of	location_of	occupied_by

Table 4.5.: Equivalence alignments between DOLCE-Lite and GFO ontologies.

	DOLCE-Lite	GFO
Class		
1.	particular	Individual
2.	endurant	Presential
3.	physical-endurant	Discrete_presential
4.	physical-object	Material_object
5.	amount-of-matter	Amount_of_substrate
6.	perdurant	Occurrent
7.	process	Process
8.	state	State
9.	abstract	Abstract
10.	set	Set
11.	quality	Property
12.	quale	Property_value
13.	quality-space	Value_space
14.	time-interval	Chronoid
15.	space-region	Spatial_Region
16.	temporal-region	Temporal_Region
Relational property		
1.	generic-constituent	has_constituent_part
2.	generic-constituent-of	constituent_part_of
3.	generically-dependant-on	depends_on
4.	generic-dependant	necessary_for
5.	has-quale	has_value
6.	quale-of	value_of
7.	boundary	has_boundary
8.	boundary-of	boundary_of
9.	q-present-at	exists_at
10.	temporary-participant-in	agent_in
11.	temporary-participant	has_agent
12.	generic-location	occupies
13.	generic-location-of	occupied_by
14.	part	abstract_has_part
15.	part-of	abstract_part_of
16.	proper-part	has_proper_part
17.	proper-part-of	proper_part_of
18.	participant	has_participant
19.	participant-in	participates_in

Table 4.6.: Equivalence alignments between BFORO and GFO ontologies.

	BFORO	GFO
Class		
1.	Entity	Entity
2.	IndependentContinuant	Presential
3.	DependentContinuant	Dependent
4.	MaterialEntity	Material_persistent
5.	Object	Material_object
6.	ObjectBoundary	Material_boundary
7.	Function	Function
8.	Role	Role
9.	Occurrent	Occurrent
10.	Process	Process
11.	Quality	Property
12.	SpatialRegion	Spatial_region
13.	TemporalRegion	Temporal_region
Relational property		
1.	has_part	has_part
2.	part_of	part_of
3.	has_proper-part	has_proper_part
4.	proper_part_of	proper_part_of
5.	has_participant	has_participant
6.	participant_in	participates
7.	located_in	occupies
8.	location_of	occupied_by
9.	has_agent	has_agent
10.	agent_in	agent_in

Table 4.7.: Equivalence alignments between DOLCE-Lite and BFORO ontologies.

	DOLCE-Lite	BFORO
Class		
1.	endurant	IndependentContinuant
2.	physical-endurant	MaterialEntity
3.	physical-object	Object
4.	perdurant	Occurrent
5.	process	Process
6.	quality	Quality
7.	spatio-temporal-region	SpatioTemporalRegion
8.	temporal-region	TemporalRegion
9.	space-region	SpatialRegion
Relational property		
1.	generic-location	located_in
2.	generic-location-of	location_of
3.	part	has_part
4.	part-of	part_of
5.	proper-part	has_proper_part
6.	proper-part-of	proper_part_of
7.	participant	has_participant
8.	participant-in	participates_in

4.2.1.4. Transitivity in alignments

In most cases, the alignments are transitive. By this we mean, if the equivalence relation holds between concepts from the first and second ontologies and it holds between concepts from the second and third ontologies; it necessarily holds between concepts from the first and third ontologies. In most cases, the foundational ontology alignments are transitive. There are two types of exceptions, being the absence of an entity and what can be termed as consequences of conflicting philosophies.

Absence of an entity. An alignment cannot be a candidate for transitivity if there is an equivalence between only two out of the three ontologies. From the three main ontology alignments, the following ones were not transitive due to the absence of an entity:

- **Absence of a DOLCE entity:**
 1. BFO:Entity $\leftrightarrow$ GFO:Entity
 2. BFO:DependentContinuant $\leftrightarrow$ GFO:Dependent
 3. BFO:ObjectBoundary $\leftrightarrow$ GFO:MaterialBoundary
 4. BFO:Function $\leftrightarrow$ GFO:Function

5. BFO:Role $\leftrightarrow$ GFO:Role
6. BFO:has_agent $\leftrightarrow$ GFO:has_agent
7. BFO:agent_in $\leftrightarrow$ GFO:agent_in

- **Absence of a GFO entity:**

1. DOLCE-Lite:spatio-temporal-region $\leftrightarrow$ BFO:SpatioTemporalRegion

- **Absence of a BFO entity:**

1. GFO:Individual $\leftrightarrow$ DOLCE-Lite:particular
2. GFO:Amount_of_substrate $\leftrightarrow$ DOLCE-Lite:amount-of-matter
3. GFO:State $\leftrightarrow$ DOLCE-Lite:state
4. GFO:Abstract $\leftrightarrow$ DOLCE-Lite:abstract
5. GFO:Set $\leftrightarrow$ DOLCE-Lite:set
6. GFO:Property_value $\leftrightarrow$ DOLCE-Lite:quale
7. GFO:Value_space $\leftrightarrow$ DOLCE-Lite:quality-space
8. GFO:Chronoid $\leftrightarrow$ DOLCE-Lite:time_interval
9. GFO:has_constituant_part $\leftrightarrow$ DOLCE-Lite:generic-consitituant
10. GFO:constituant_part_of $\leftrightarrow$ DOLCE-Lite:generic-constituant-of
11. GFO:necessary_for $\leftrightarrow$ DOLCE-Lite:generic-dependent
12. GFO:depends_on $\leftrightarrow$ DOLCE-Lite:generically-dependent-on
13. GFO:has_value $\leftrightarrow$ DOLCE-Lite:has-quale
14. GFO:value_of $\leftrightarrow$ DOLCE-Lite:quale-of
15. GFO:has_boundary $\leftrightarrow$ DOLCE-Lite:boundary
16. GFO:boundary_of $\leftrightarrow$ DOLCE-Lite:boundary-of
17. GFO:exists_at $\leftrightarrow$ DOLCE-Lite:q-present-at

From this type of transitivity issue, we see that for the three main ontology alignments, in most cases BFO entities are absent. There are a few cases of absent DOLCE entities and one case of an absent GFO entity.

Conflicting philosophies. The philosophies of foundational ontologies affect their entities to a certain extent. In some cases, two entities that are aligned to each other may not be aligned to the same entity of a third ontology.

- **DOLCE:physical-endurant $\leftrightarrow$ BFO:MaterialEntity, DOLCE:physical-endurant $\leftrightarrow$ GFO:Discrete_presential and BFO:MaterialEntity $\leftrightarrow$ GFO:Material_persistent:** We align BFO:MaterialEntity $\leftrightarrow$ DOLCE:physical-endurant, and ignore their underlying philosophies (i.e., that BFO is an ontology of universals and DOLCE of particulars). However, in GFO, there are two entities for representing this type of entity, based on distinct philosophical notions: GFO:Discrete_presential, being subsumed GFO:Individual, is suited for DOLCE:physical-endurant since DOLCE is an ontology of particulars while GFO:Material_persistent, being subsumed by GFO:Universal, is suited for BFO:MaterialEntity since BFO is an ontology of universals.

- **DOLCE:part , BFORO:has_part, GFO:has_part and GFO:has_abstract_part:** In DOLCE, both the domain and range of part is particular. In BFORO, there is no domain and range for has_part. In GFO, both the domain and range of abstract_has_part is Item, while both the domain and range for has_part is Concrete. The former relational property is better suited for DOLCE as it includes abstract entities. The latter is better suited for BFORO as it is restricted to concrete entities, and BFORO only includes concrete entities.
- **DOLCE:part-of , BFORO:part_of, GFO:part_of and GFO:abstract_part_of:** In DOLCE, both the domain and range of part-of is particular. In BFORO, there is no domain and range for part_of. In GFO, both the domain and range of abstract_part_of is Item, while both the domain and range for part_of is Concrete. The former relational property is better suited for DOLCE as it includes abstract entities. The latter is better suited for BFORO as it is restricted to concrete entities, and BFORO only includes concrete entities.

Table 4.8 displays the transitivity of the alignments in the direction of DOLCE-Lite $\leftrightarrow$ BFORO $\leftrightarrow$ GFO $\leftrightarrow$ DOLCE-Lite. In the table, dash (-) values in rows are alignments that are not candidates for transitivity simply because of an absence of an entity in one of the ontologies. The alignments that are numbered in bold font represent those that are not transitive due to conflicting philosophies, explained further below.

4.2.2. Approximate alignments

We have identified a number of pairs between foundational ontologies that are approximate. By this we mean that they are not equivalent to each other or subsumed by one another, but share some common characteristics. By identifying and providing these approximate relations between these entities, foundational ontology developers could possibly, in the future, include them as sibling classes by grouping them both under a common superclass.

4.2.2.1. Approximate alignments between DOLCE-Lite and BFO

- **DOLCE-Lite:arbitrary-sum and BFO:ObjectAggregate:** Both these entities are a collection of something. DOLCE's arbitrary-sum, however, has no unity criterion e.g., A pencil and laundry basket are together an arbitrary sum. It can contain both physical-endurant and non-physical-endurant entities. DOLCE's physical-endurant is not restricted just to instances of physical-object but can possibly include feature and amount-of-matter. BFO's ObjectAggregate, on the other hand, has overall unity and can be considered as a whole. It is restricted to BFO's Object only and in the case of BFO all objects are physical.
- **DOLCE-Lite:state and BFO:SpatioTemporalInstant:** DOLCE provides an example of its state by using an example of a rock erosion describing state as a time interval of the erosion is collapsed into a time point. Similarly BFO defines SpatioTemporalInstant as a "connected spatiotemporal region at a specific moment". The difference between the two lies in the fact that DOLCE's state is homeomeric while BFO's SpatioTemporalInstant is not.

Table 4.8.: Transitivity in the alignments between the three foundational ontologies.

	DOLCE-Lite	BFORO	GFO	DOLCE-Lite
Class				
1.	–	Entity	Entity	–
2.	endurant	IndependentContinuant	Presential	endurant
3.	–	DependentContinuant	Dependent	–
4.	physical-endurant	MaterialEntity	Material_persistent	
5.	physical-object	Object	Material_object	physical-object
6.	–	ObjectBoundary	Material_boundary	–
7.	–	Function	Function	–
8.	–	Role	Role	–
9.	perdurant	Occurrent	Occurrent	perdurant
10.	process	Process	Process	process
11.	quality	Quality	Property	quality
12.	space-region	SpatialRegion	Spatial_region	space-region
13.	temporal-region	TemporalRegion	Temporal_region	temporal-region
Relational property				
1.	part	has_part	has_part	
2.	part-of	part_of	part_of	
3.	proper-part	has_proper-part	has_proper_part	proper-part
4.	proper-part-of	proper_part_of	proper_part_of	proper-part-of
5.	has-participant	has_participant	has_participant	has-participant
6.	participant-in	participates_in	participates_in	participant-in
7.	generic-location	located_in	occupies	generic-location
8.	generic-location-of	location_of	occupied_by	generic-location-of
9.	–	has_agent	has_agent	–
10.	–	agent_in	agent_in	–

- **DOLCE-Lite:relevant-part and BFO:FiatObjectPart:** DOLCE describes `dolce:relevant-part` as a feature that is a relevant part of their host; e.g., the edge of a cube. BFO defines `FiatObjectPart` as a material entity that is part of an object but not demarcated by physical discontinuities; e.g., the lower portion of the leg. In this sense they are both part objects that are physical entities. However, it is unclear whether DOLCE's `relevant-part` is demarcated by physical discontinuities or not and whether BFO's `FiatObjectPart` are 'relevant' somehow. This requires further investigation.

4.2.2.2. Approximate alignments between BFO and GFO

- **BFO:ObjectAggregate and GFO:Configuration:** Both these entities are a collection of something. BFO's `ObjectAggregate` has overall unity and is restricted to `Object` only. In the case of BFO all objects are physical. GFO's `Configuration` is simply a collection of GFO's `Presential` facts. GFO's `Presentials` are not restricted to whole physical objects and can include other `Presential` entities.

4.2.2.3. Approximate alignments between GFO and DOLCE-Lite

- **GFO:Configuration and DOLCE-Lite:arbitrary-sum:** Both these entities are a collection of something. GFO's `Configuration` is a collection of `presential` facts but holds a condition that states that it must contain at least one material entity. DOLCE's `arbitrary-sum` is simply a sum of `endurants`. The `endurants` could be physical, non-physical or both.

4.3. Mapping and Merging

Ontology mapping and merging was performed by equating classes and object properties with Protégé. Mapping and merged ontologies are available for users to browse through and download in ROMULUS. Entities were mapped in the order of their level in the hierarchy, from higher to lower level. The reason this is that foundational ontologies by definition are general high-level ontologies. Since ontology mapping sometimes results in ontological inconsistencies, it is important to ensure that firstly, higher-level entities exist. Reconsider Tables 4.7, 4.5, and 4.6: there are 11 successful mappings for DOLCE-lite $\leftrightarrow$ BFORO, 13 for DOLCE-Lite $\leftrightarrow$ GFO, and 16 for BFORO $\leftrightarrow$ GFO.

The unsuccessful mappings are ontology alignments that could not be mapped as they resulted in ontology inconsistencies. The inconsistencies were identified by using the method described earlier in Chapter 3. It is important to note that if the entities were mapped in the opposite order, from lower to higher level, this would result in different inconsistencies. Many of the existing higher-level successful entity mappings would be inconsistent, and the existing inconsistent lower-level mappings would in some cases be consistent. In the following sections, we discuss the inconsistencies and provide possible solutions for some of them.

4.3.1. Logical inconsistencies

Each inconsistent alignment as well as a short description of the inconsistency is provided below. The explanations for the inconsistencies were generated with the Protégé explanation feature. Thereafter we manually translated these explanations to natural language, reordered the sentences, analysed them and identified root causes for each inconsistency.

4.3.1.1. Logical inconsistencies between DOLCE-Lite and BFO modules

1. **DOLCE-Lite:spatio-temporal-region - BFO:SpatioTemporalRegion:** DOLCE's spatio-temporal-region is a subclass of DOLCE's abstract. DOLCE's abstract is disjoint to DOLCE's perdurant. DOLCE's perdurant is equivalent to BFO's Occurrent. BFO's SpatioTemporalRegion is a subclass of BFO's Occurrent. This inconsistency is a result of the OWL DisjointClasses class axiom. In this equivalence relation, both DOLCE's spatio-temporal-region and BFO's SpatioTemporalRegion are subclasses of two classes that are disjoint, hence the two classes cannot be equivalent.
2. **DOLCE-Lite:temporal-region - BFO:TemporalRegion:** This inconsistency is similar to the above inconsistency, having the same root cause. DOLCE's temporal-region is a subclass of DOLCE's abstract. DOLCE's abstract is disjoint to DOLCE's perdurant. DOLCE's perdurant is equivalent to BFO's Occurrent. BFO's TemporalRegion is a subclass of BFO's Occurrent. This inconsistency is a result of the OWL DisjointClasses class axiom. In this equivalence relation, both DOLCE's temporal-region and BFO's TemporalRegion are subclasses of two classes that are disjoint, hence the two classes cannot be equivalent.
3. **DOLCE-Lite:participant - BFORO:has_participant:** The range of DOLCE's participant is endurant. The range of BFO's has_participant is Continuant. DOLCE's endurant is disjoint to its quality. DOLCE's quality is equivalent to BFO's Quality. BFO's Quality is a subclass of its Continuant. This inconsistency is a result of the OWL DisjointClasses class axiom. In this equivalence relation, DOLCE's endurant is disjoint to a subclass of BFO's Continuant, causing the range restrictions of DOLCE and BFO to conflict for this relation.
4. **DOLCE-Lite:participant-in - BFORO:participates_in:** This inconsistency is similar to the above inconsistency, having the same root cause. The domain of DOLCE's participant-in is endurant. The domain of BFO's participates_in is Continuant. DOLCE's endurant is disjoint to its quality. DOLCE's quality is equivalent to BFORO's Quality. BFO's Quality is a subclass of its Continuant. This inconsistency is a result of the OWL DisjointClasses class axiom. In this equivalence relation, DOLCE's endurant is disjoint to a subclass of BFO's Continuant, causing the domain restrictions of DOLCE and BFO to conflict for this relation.
5. **DOLCE-Lite:generic-location - BFORO:located_in:** The range of DOLCE's generic-location is particular. The range of BFO's located_in is Continuant. BFO's Continuant is disjoint to its Occurrent. BFO's Occurrent is equivalent to DOLCE's perdurant. DOLCE's

perdurant is a subclass of has-Quality some temporal-location-q. The domain of DOLCE's has-quality is particular. This inconsistency is a result of the OWL DisjointClasses class axiom. In this equivalence relation, BFO's Continuant is disjoint to a subclass of DOLCE's has-Quality some temporal-location-q having a domain particular, causing the range restrictions of DOLCE and BFO to conflict for this relation.

6. **DOLCE-Lite:generic-location-of - BFORO:located_of:** This inconsistency is similar to the above inconsistency, having the same root cause. The range of DOLCE's generic-location-of is particular. The range of BFO's located_of is Continuant. BFO's Continuant is disjoint to its Occurrent. BFO's Occurrent is equivalent to DOLCE's S perdurant. DOLCE's perdurant is a subclass of has-Quality some temporal-location-q. The domain of DOLCE's has-quality is particular. This inconsistency is a result of the OWL DisjointClasses class axiom. In this equivalence relation, BFO's Continuant is disjoint to a subclass of DOLCE's has-Quality some temporal-location-q having a domain particular, causing the range restrictions of DOLCE and BFO to conflict for this relation.

4.3.1.2. Logical inconsistencies between DOLCE-Lite and GFO modules

1. **DOLCE-Lite:set - GFO:Set:** DOLCE's set is a subclass of DOLCE's abstract. DOLCE's abstract is equivalent to GFO's Abstract. GFO's Abstract is a subclass of GFO's Item. GFO's Set is disjoint to GFO's Item. This inconsistency is a result of the OWL DisjointClasses class axiom. In this equivalence relation, GFO's Set is disjoint to a superclass of DOLCE's set, hence the two classes cannot be equivalent.
2. **DOLCE-Lite:quale - GFO:Property_value:** DOLCE's quale is a subclass of DOLCE's abstract. DOLCE's abstract is equivalent to GFO's Abstract. GFO's Property_value is a subclass of GFO's Concrete. GFO's Concrete is disjoint to GFO's Abstract. This inconsistency is a result of the OWL DisjointClasses class axiom. In this equivalence relation, both DOLCE's quale and GFO's Property_value are subclasses of two classes that are disjoint, hence the two classes cannot be equivalent.
3. **DOLCE-Lite:quality-space - GFO:Value_space:** DOLCE's quality-space is a subclass of DOLCE's particular. DOLCE's particular is equivalent to GFO's Individual. GFO's Value_space is a subclass of GFO's Category. GFO's Category is disjoint to GFO's Individual. This inconsistency is a result of the OWL DisjointClasses class axiom. In this equivalence relation, both DOLCE's quality-space and GFO's Value_space are subclasses of two classes that are disjoint, hence the two classes cannot be equivalent.
4. **DOLCE-Lite:time-interval - GFO:Chronoid:** DOLCE's time-interval is a subclass of DOLCE's abstract. DOLCE's abstract is equivalent to GFO's Abstract. GFO's Chronoid is a subclass of GFO's Space_Time. GFO's Space_Time is disjoint to GFO's Abstract. This inconsistency is a result of the OWL DisjointClasses class axiom. In this equivalence relation, both DOLCE's time-interval and GFO's Chronoid are subclasses of two classes that are disjoint, hence the two classes cannot be equivalent.

5. **DOLCE-Lite:space-region - GFO:Spatial_region:** This inconsistency is similar to the above inconsistency, having the same root cause. DOLCE's space-region is a subclass of DOLCE's abstract. DOLCE's abstract is equivalent to GFO's Abstract. GFO's Spatial_region is a subclass of GFO's Space_Time. GFO's Space_Time is disjoint to GFO's Abstract. This inconsistency is a result of the OWL DisjointClasses class axiom. In this equivalence relation, both DOLCE's space-region and GFO's Spatial_region are subclasses of two classes that are disjoint, hence the two classes cannot be equivalent.
6. **DOLCE-Lite:temporal-region - GFO:Temporal_region:** This inconsistency is similar to the two above inconsistencies, having the same root cause. DOLCE's temporal-region is a subclass of DOLCE's abstract. DOLCE's abstract is equivalent to GFO's Abstract. GFO's Temporal_region is a subclass of GFO's Space_Time. GFO's Space_Time is disjoint to GFO's Abstract. This inconsistency is a result of the OWL DisjointClasses class axiom. In this equivalence relation, both DOLCE's temporal-region and GFO's Temporal_region are subclasses of two classes that are disjoint, hence the two classes cannot be equivalent. Refer to Fig. 4.1 for a graphical explanation of the inconsistencies between temporal region entities of DOLCE-Lite, BFO and GFO ontologies.

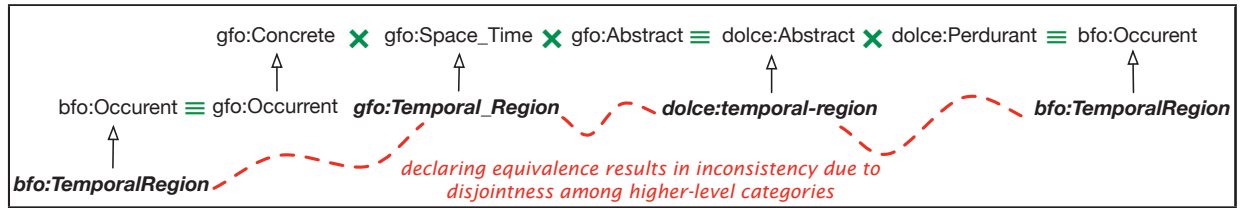


Figure 4.1.: Due to the OWL DisjointClasses class axiom, DOLCE:temporal-region, BFO:TemporalRegion and GFO:Temporal_region cannot be mapped in any way without causing an inconsistency; $\equiv$: aligned entities, $\times$: disjoint entities.

7. **DOLCE-Lite:amount-of-matter - GFO:Amount_of_substrate:** DOLCE's amount-of-matter is a subclass of its physical-endurant. DOLCE's physical-endurant is equivalent to GFO's Discrete_presential. GFO's Discrete_presential is a subclass of its Discrete entity. GFO's Continuous is equivalent to its Individual and (not (Discrete)). GFO's Amount_of_substrate is a subclass of its Continuous entity. This inconsistency is a result of the OWL Complement class constructor. In this equivalence relation, GFO's Amount_of_substrate is a subclass of its Continuous while DOLCE's amount-of-matter is a subclass of the complement of that class, hence the two classes cannot be equivalent.
8. **DOLCE-Lite:state - GFOBasic:State:** DOLCE's state is a subclass of its perdurant. DOLCE's state is equivalent to GFO's State. GFO's State is a subclass of its Process. GFO's Process is disjoint to its Occurrent. GFO's Occurrent is equivalent to DOLCE's perdurant. This inconsistency is a result of the OWL DisjointClasses class axiom. In this equivalence relation, both DOLCE's state and GFO's State are subclasses of two classes that are disjoint, hence the two classes cannot be equivalent.

9. **DOLCE-Lite:process - GFOBasic:Process:** This inconsistency is similar to the above inconsistency, having the same root cause. DOLCE's process is a subclass of its perdurant. GFO's Process is disjoint to its Occurrent. GFO's Occurrent is equivalent to DOLCE's perdurant. This inconsistency is a result of the OWL DisjointClasses class axiom. In this equivalence relation, GFO's Process is disjoint to a superclass of DOLCE's process, hence the two classes cannot be equivalent.
10. **FunctionalParticipation:concept - GFO:Concept:** GFO's Concept is a subclass of GFO's Category. DOLCE's concept is a subclass of DOLCE's endurant. DOLCE's endurant is equivalent to GFO's Presential. GFO's Presential is a subclass of GFO's Individual. GFO's Category is disjoint to GFO's Individual. This inconsistency is a result of the OWL DisjointClasses class axiom. In this equivalence relation, both DOLCE's concept and GFO's Concept are subclasses of two classes that are disjoint, hence the two classes cannot be equivalent.
11. **FunctionalParticipation:role - GFO:Role:** GFO's Processual_role is a subclass of GFO's Occurrent. GFO's Processual_role is a subclass of GFO's Role. DOLCE's role is a subclass of DOLCE's endurant. DOLCE's endurant is equivalent to GFO's Presential. GFO's Occurrent is disjoint to GFO's Presential. This inconsistency is a result of the OWL DisjointClasses class axiom. In this equivalence relation, DOLCE's role is a subclass of GFO's Presential, and GFO's Role is a superclass of Occurrent's subclass, with Presential and Occurrent being disjoint, hence the two classes cannot be equivalent.
12. **DOLCE-Lite:part - GFO:abstract_has_part:** The domain and range for DOLCE's part is particular. The domain and range for GFO's abstract_has_part is Item. GFO's Category is disjoint to Individual. GFO's Item is equivalent to Category. DOLCE's particular is equivalent to Individual. This inconsistency is a result of the OWL DisjointClasses class axiom. In this equivalence relation, DOLCE's particular is disjoint to GFO's Item, due to other equivalence relations in the ontologies, causing the domain and range restrictions of DOLCE and GFO to conflict for this relation.
13. **DOLCE-Lite:part-of - GFO:abstract_part_of:** This inconsistency is similar to the above inconsistency, having the same root cause. The domain and range for DOLCE's part-of is particular. The domain and range for GFO's abstract_part_of is Item. GFO's Category is disjoint to Individual. GFO's Category is a subclass of its Item. DOLCE's particular is equivalent to Individual. This inconsistency is a result of the OWL DisjointClasses class axiom. In this equivalence relation, DOLCE's particular is disjoint to the subclass of GFO's Item, due to other equivalence relations in the ontologies, causing the domain and range restrictions of DOLCE and GFO to conflict for this relation.
14. **DOLCE-Lite:generic-dependent - GFO:necessary_for:** This inconsistency is similar to the above inconsistency, having the same root cause. The domain and range for DOLCE's generic-dependent is particular. The domain and range for GFO's necessary_for is Item. GFO's Category is disjoint to Individual. GFO's Category is a subclass of its Item. DOLCE's particular is equivalent to Individual. This inconsistency is a result of the OWL DisjointClasses class axiom. In this equivalence relation, DOLCE's particular is disjoint to the

subclass of GFO's *Item*, due to other equivalence relations in the ontologies, causing the domain and range restrictions of DOLCE and GFO to conflict for this relation.

15. **DOLCE-Lite:generically-dependent-on - GFO:depends_on:** This inconsistency is similar to the above inconsistency, having the same root cause. The domain and range for DOLCE's *generically-dependent-on* is *particular*. The domain and range for GFO's *depends_on* is *Item*. GFO's *Category* is disjoint to *Individual*. GFO's *Category* is a subclass of its *Item*. DOLCE's *particular* is equivalent to *Individual*. This inconsistency is a result of the OWL *DisjointClasses* class axiom. In this equivalence relation, DOLCE's *particular* is disjoint to the subclass of GFO's *Item*, due to other equivalence relations in the ontologies, causing the domain and range restrictions of DOLCE and GFO to conflict for this relation.
16. **DOLCE-Lite:proper-part - GFO:has_proper_part:** The domain and range for DOLCE's *proper-part* is *particular*. GFO's *proper-part* is a subproperty of its *has_part*. The domain and range for GFO's *has_part* is *Concrete*, therefore the domain and range of GFO's *has_proper_part* is *Concrete*. DOLCE's *particular* is equivalent to GFO's *Individual*. GFO's *Abstract* is a subclass of its *Individual*. GFO's *Abstract* is disjoint to *Concrete*. This inconsistency is a result of the OWL *DisjointClasses* class axiom. In this equivalence relation, GFO's *Concrete* is disjoint to a subclass of DOLCE's *particular*, causing the domain and range restrictions of DOLCE and GFO to conflict for this relation.
17. **DOLCE-Lite:proper-part-of - GFO:proper_part_of:** This inconsistency is similar to the above inconsistency, having the same root cause. The domain and range for DOLCE's *proper-part-of* is *particular*. GFO's *proper-part-of* is a subproperty of its *part_of*. The domain and range for GFO's *has_part* is *Concrete*, therefore the domain and range of GFO's *has_proper_part* is *Concrete*. DOLCE's *particular* is equivalent to GFO's *Individual*. GFO's *Abstract* is a subclass of its *Individual*. GFO's *Abstract* is disjoint to *Concrete*. This inconsistency is a result of the OWL *DisjointClasses* class axiom. In this equivalence relation, GFO's *Concrete* is disjoint to a subclass of DOLCE's *particular*, causing the domain and range restrictions of DOLCE and GFO to conflict for this relation.
18. **DOLCE-Lite:generic-constituent-of - GFO:constituent_part_of:** This inconsistency is similar to the above inconsistency, having the same root cause. The domain and range for DOLCE's *generic-constituent-of* is *particular*. GFO's *constituent_part_of* is a subproperty of its *has_part*. The domain and range for GFO's *has_part* is *Concrete*, therefore the domain and range of GFO's *constituent_part_of* is *Concrete*. DOLCE's *particular* is equivalent to GFO's *Individual*. GFO's *Abstract* is a subclass of its *Individual*. GFO's *Abstract* is disjoint to *Concrete*. This inconsistency is a result of the OWL *DisjointClasses* class axiom. In this equivalence relation, GFO's *Concrete* is disjoint to a subclass of DOLCE's *particular*, causing the domain and range restrictions of DOLCE and GFO to conflict for this relation.
19. **DOLCE-Lite:generic-constituent - GFO:has_constituent_part:** The domain for DOLCE's *generic-constituent* is *particular*. The domain for GFO's *has_constituent_part* is *Configuration*. GFO's *Concrete* is a subclass of *Individual*. GFO's *Individual* is equivalent to DOLCE's *particular*. GFO's *Configuration* is a subclass of its *Presential*. GFO's *Presential* is disjoint to *Occurrent*. GFO's *Occurrent* is a subclass of *Concrete*. This inconsistency is

a result of the OWL `DisjointClasses` class axiom. In this equivalence relation, GFO's `Configuration`'s superclass is disjoint to a subclass of DOLCE's `particular`, causing the domain restrictions of DOLCE and GFO to conflict for this relation.

20. **DOLCE-Lite:generic-location - GFO:occupies:** The range for DOLCE's `generic-location` is `particular`. The range for GFO's `occupies` is `Space`. GFO's `Space` is a subclass of `Space_time`. GFO's `Space_time` is disjoint to its `Abstract`. GFO's `Abstract` is a subclass of `Individual`. GFO's `Individual` is equivalent to `particular`. This inconsistency is a result of the OWL `DisjointClasses` class axiom. In this equivalence relation, GFO's `Space`'s superclass is disjoint to a subclass of DOLCE's `particular`, causing the range restrictions of DOLCE and GFO to conflict for this relation.
21. **DOLCE-Lite:generic-location-of - GFO:occupied_by:** This inconsistency is similar to the above inconsistency, having the same root cause. The domain for DOLCE's `generic-location-of` is `particular`. The domain for GFO's `occupied_by` is `Space`. GFO's `Space` is a subclass of `Space_time`. GFO's `Space_time` is disjoint to its `Abstract`. GFO's `Abstract` is a subclass of `Individual`. GFO's `Individual` is equivalent to `particular`. This inconsistency is a result of the OWL `DisjointClasses` class axiom. In this equivalence relation, GFO's `Space`'s superclass is disjoint to a subclass of DOLCE's `particular`, causing the domain restrictions of DOLCE and GFO to conflict for this relation.
22. **DOLCE-Lite:has-quale - GFO:has_value:** The range for DOLCE's `has-quale` is `quale`. The range for GFO's `has_value` is `property_value`. DOLCE's `quale` is a subclass of DOLCE's `abstract`. DOLCE's `abstract` is equivalent to GFO's `Abstract`. GFO's `Property_value` is a subclass of GFO's `Concrete`. GFO's `Concrete` is disjoint to GFO's `Abstract`. This inconsistency is a result of the OWL `DisjointClasses` class axiom. In this equivalence relation, both DOLCE's `quale` and GFO's `Property_value` are subclasses of two classes that are disjoint, causing the range restrictions of DOLCE and GFO to conflict for this relation.
23. **DOLCE-Lite:quale-of - GFO:value_of:** This inconsistency is similar to the above inconsistency, having the same root cause. The domain for DOLCE's `quale-of` is `quale`. The domain for GFO's `value_of` is `property_value`. DOLCE's `quale` is a subclass of DOLCE's `abstract`. DOLCE's `abstract` is equivalent to GFO's `Abstract`. GFO's `Property_value` is a subclass of GFO's `Concrete`. GFO's `Concrete` is disjoint to GFO's `Abstract`. This inconsistency is a result of the OWL `DisjointClasses` class axiom. In this equivalence relation, both DOLCE's `quale` and GFO's `Property_value` are subclasses of two classes that are disjoint, causing the domain restrictions of DOLCE and GFO to conflict for this relation.
24. **DOLCE-Lite:q-present-at - GFO:exists_at:** The domain for DOLCE's `q-present-at` is `physical-quality`. The domain for GFO's `exists_at` is `Presential`. DOLCE's `physical-quality` is a subclass of DOLCE's `quality`. DOLCE's `quality` is disjoint to its `endurant`. DOLCE's `endurant` is equivalent to GFO's `Presential`. This inconsistency is a result of the OWL `DisjointClasses` class axiom. In this equivalence relation, the superclass of DOLCE's `physical-quality` is disjoint to GFO's `Presential`, causing the domain restrictions of DOLCE and GFO to conflict for this relation.

25. **DOLCE-Lite:participant - GFOBasic:has_participant:** The domain for DOLCE's participant is perdurant. The domain for GFO's has_participant is Processual_structure. GFO's Occurrent is disjoint to its Process. GFO's Process is a subclass of its Processual_structure. DOLCE's perdurant is equivalent to GFO's Occurrent. This inconsistency is a result of the OWL DisjointClasses class axiom. In this equivalence relation, DOLCE's perdurant is disjoint to a subclass of GFO's Processual_structure, causing the domain restrictions of DOLCE and GFO to conflict for this relation.
26. **DOLCE-Lite:participant-in - GFOBasic:participates_in:** This inconsistency is similar to the above inconsistency, having the same root cause. The range for DOLCE's participant-in is perdurant. The range for GFO's participates_in is Processual_structure. GFO's Occurrent is disjoint to its Process. GFO's Process is a subclass of its Processual_structure. DOLCE's perdurant is equivalent to GFO's Occurrent. This inconsistency is a result of the OWL DisjointClasses class axiom. In this equivalence relation, DOLCE's perdurant is disjoint to a subclass of GFO's Processual_structure, causing the range restrictions of DOLCE and GFO to conflict for this relation.

4.3.1.3. Logical inconsistencies between BFO and GFO modules

1. **BFO:Role - GFO:Role:** GFO's Processual_role is a subclass of Role and Process. GFO's Process is a subclass of its Occurrent. GFO's Occurrent is equivalent to BFO's Occurrent. BFO's role is a subclass of its Continuant. BFO's Continuant is disjoint to its Occurrent. This inconsistency is a result of the OWL DisjointClasses class axiom. In this equivalence relation, GFO's Role is a superclass of Processual_role, which is a subclass of Occurrent, and BFO's Role is a subclass of Continuant with Occurrent and Continuant being disjoint, hence the two classes cannot be equivalent.
2. **BFO:TemporalRegion - GFO:Temporal_region:** BFO's TemporalRegion is a subclass of BFO's Occurrent. BFO's Occurrent is equivalent to GFO's Occurrent. GFO's Occurrent is a subclass of GFO's Concrete. GFO's Space_Time is disjoint with GFO's Concrete. GFO's Temporal_Region is a subclass of GFO's Space_Time. This inconsistency is a result of the OWL DisjointClasses class axiom. In this equivalence relation, both BFO's TemporalRegion and GFO's Temporal_region are subclasses of two classes that are disjoint, hence the two classes cannot be equivalent.
3. **BFO:MaterialEntity - GFO:Material_persistent:** GFO's Material_persistent is a subclass of its Universal. GFO's Universal is a subclass of instantiated_by some GFO's Item. BFO's Material_entity is a subclass of its IndependentContinuant. BFO's IndependentContinuant is equivalent to GFO's Presential. GFO's Presential is a subclass of its Individual. GFO's Individual is a subclass of the complement of instantiated_by some GFO's Item. This inconsistency is a result of the OWL Complement class constructor. In this equivalence relation, GFO's Material_persistent is a subclass of its instantiated_by some GFO's Item while BFO's MaterialEntity is a subclass of the complement of that class, hence the two classes cannot be equivalent.

4. **BFO:DependentContinuant - GFO:Dependent:** GFO's Entity is equivalent to its Item or Set. GFO's Category is a subclass of Item. GFO's Category is disjoint to its Individual. GFO's Dependent is a subclass of Individual. GFO's Entity is equivalent to BFO's Entity. BFO's Entity is equivalent to its Occurrent or Continuant. BFO's Continuant is equivalent to its IndependentContinuant or DependentContinuant or SpatialRegion. This inconsistency is a result of the OWL DisjointClasses class axiom. In this equivalence relation, BFO's DependentContinuant is a superclass of GFO's Category, and GFO's Dependent is a subclass of GFO's Individual with Category and Individual being disjoint, hence the two classes cannot be equivalent.
5. **BFO:Process - GFOBasic:Process:** BFO's Process is a subclass of BFO's Occurrent. BFO's Occurrent is equivalent to GFO's Occurrent. GFO's Occurrent is disjoint to GFO's Process. This inconsistency is a result of the OWL DisjointClasses class axiom. In this equivalence relation, GFO's Process is disjoint to a superclass of BFO's process, hence the two classes cannot be equivalent.
6. **BFORO:located_in - GFO:occupies:** The range of BFO's located_in is Continuant. The range of GFO's occupies is Space. GFO's Presential is a subclass of Concrete. GFO's Concrete is disjoint with its Space_time. GFO's Presential is equivalent to BFO's IndependentContinuant. BFO's IndependentContinuant is a subclass of its Continuant. GFO's Space is a subclass of its Space_time. This inconsistency is a result of the OWL DisjointClasses class axiom. In this equivalence relation, GFO's Space is a subclass of GFO's Space_time, and, by equivalence BFO's Continuant is a superclass of Presential, which is Concrete's subclass, with Space_time and Concrete being disjoint, causing the range restrictions of BFO and GFO to conflict for this relation.
7. **BFORO:location_of and GFO:occupied_by:** This inconsistency is similar to the above inconsistency, having the same root cause. The domain of BFO's location_of is Continuant. The domain of GFO's occupied_by is Space. GFO's Presential is a subclass of Concrete. GFO's Concrete is disjoint with its Space_time. GFO's Presential is equivalent to BFO's IndependentContinuant. BFO's IndependentContinuant is a subclass of its Continuant. GFO's Space is a subclass of its Space_time. This inconsistency is a result of the OWL DisjointClasses class axiom. In this equivalence relation, GFO's Space is a subclass of GFO's Space_time, and, by equivalence BFO's Continuant is a superclass of Presential, which is Concrete's subclass, with Space_time and Concrete being disjoint, causing the domain restrictions of BFO and GFO to conflict for this relation.
8. **BFORO:has_participant - GFO:has_participant:** The range for BFO's has_participant is Continuant. The range for GFO's has_participant is Presential. BFO's SpatialRegion is a subclass of Continuant. BFO's SpatialRegion is disjoint to its IndependentContinuant. BFO's IndependentContinuant is equivalent to GFO's Presential. This inconsistency is a result of the OWL DisjointClasses class axiom. In this equivalence relation, GFO's Presential is disjoint to a subclass of BFO's Continuant, causing the range restrictions of BFO and GFO to conflict for this relation.

9. **BFORO:has_participant - GFOBasic:has_participant:** The domain for BFO's `has_participant` is `Occurrent`. The domain for GFO's `has_participant` is `Processual_structure`. GFO's `Occurrent` is disjoint to its `Process`. GFO's `Process` is a subclass of its `Processual_structure`. GFO's `Occurrent` is equivalent to BFO's `Occurrent`. This inconsistency is a result of the OWL `DisjointClasses` class axiom. In this equivalence relation, BFO's `Occurrent` is disjoint to a subclass of GFO's `Processual_structure`, causing the domain restrictions of BFO and GFO to conflict for this relation.
10. **BFORO:participates_in - GFOBasic:participates_in:** This inconsistency is similar to the above inconsistency, having the same root cause. The range for BFO's `participates_in` is `Occurrent`. The range for GFO's `participates_in` is `Processual_structure`. GFO's `Occurrent` is disjoint to its `Process`. GFO's `Process` is a subclass of its `Processual_structure`. GFO's `Occurrent` is equivalent to BFO's `Occurrent`. This inconsistency is a result of the OWL `DisjointClasses` class axiom. In this equivalence relation, BFO's `Occurrent` is disjoint to a subclass of GFO's `Processual_structure`, causing the range restrictions of BFO and GFO to conflict for this relation.

4.3.1.4. Dealing with ontological inconsistencies

We were able to deal with a few inconsistencies by altering the equivalence relation. There were two types of changes made to relations. For the first type of change, we checked if there was an alternate entity in each ontology that would logically hold in the relation. Secondly, we checked whether each entity in the relation would be able to logically subsume its partner. A list of each inconsistency, with its altered relation follows.

- **DOLCE-Lite:participant - BFORO:has_participant:** Section 4.3.1.1, inconsistency 3: The relation was changed to: `BFORO:has_participant` subsumes `DOLCE-Lite:participant`.
- **DOLCE-Lite:participant-in - BFORO:participates_in:** Section 4.3.1.1, inconsistency 4: The relation was changed to: `BFORO:participates_in` subsumes `DOLCE-Lite:participant-in`.
- **FunctionalParticipation:role - GFO:Role:** Section 4.3.1.2, inconsistency 11: The relation was changed to: `GFO:Role` subsumes `BFO:Role`.
- **DOLCE-Lite:part - GFO:abstract_has_part:** Section 4.3.1.2, inconsistency 12: The relation was changed to `GFO:abstract_has_part` subsumes `DOLCE-Lite:part`.
- **DOLCE-Lite:part-of - GFO:abstract_part_of:** Section 4.3.1.2, inconsistency 13: The relation was changed to `GFO:abstract_part_of` subsumes `DOLCE-Lite:part-of`.
- **DOLCE-Lite:generic-dependent - GFO:necessary_for:** Section 4.3.1.2, inconsistency 14: The relation was changed to `GFO:necessary_for` subsumes `DOLCE-Lite:generic-dependent`.
- **DOLCE-Lite:generically-dependent-on - GFO:depends_on:** Section 4.3.1.2, inconsistency 15: The relation was changed to `GFO:depends_on` subsumes `DOLCE-Lite:generically-dependent-on`.

- **DOLCE-Lite:proper-part - GFO:has_proper_part:** Section 4.3.1.2, inconsistency 16: The relation was changed to DOLCE-Lite:proper-part subsumes GFO:has_proper_part.
- **DOLCE-Lite:proper-part-of - GFO:proper_part_of:** Section 4.3.1.2, inconsistency 17: The relation was changed to DOLCE-Lite:proper-part-of subsumes GFO:proper_part_of.
- **DOLCE-Lite:generic-constituent-of - GFO:constituent_part_of:** Section 4.3.1.2, inconsistency 18: The relation was changed to DOLCE-Lite:generic-constituent-of subsumes GFO:constituent_part_of.
- **DOLCE-Lite:generic-constituent - GFO:has_constituent_part:** Section 4.3.1.2, inconsistency 19: The relation was changed to DOLCE-Lite:generic-constituent subsumes GFO:has_constituent_part.
- **DOLCE-Lite:generic-location - GFO:occupies:** Section 4.3.1.2, inconsistency 20: The relation was changed to DOLCE-Lite:generic-location subsumes GFO:occupies.
- **DOLCE-Lite:generic-location-of - GFO:occupied_by:** Section 4.3.1.2, inconsistency 21: The relation was changed to DOLCE-Lite:generic-location-of subsumes GFO:occupied_by.
- **DOLCE-Lite:participant - GFOBasic:has_participant:** Section 4.3.1.2, inconsistency 25: The relation was changed to GFO:has_participant subsumes DOLCE-Lite:participant.
- **DOLCE-Lite:participant-in - GFOBasic:participates_in:** Section 4.3.1.2, inconsistency 26: The relation was changed to GFO:participates_in subsumes DOLCE-Lite:participant-in.
- **BFO:Role - GFO:Role:** Section 4.3.1.3, inconsistency 1: The relation was changed to: GFO:Role subsumes BFO:Role.
- **BFO:MaterialEntity - GFO:Material_persistent:** Section 4.3.1.3, inconsistency 3: The relation was changed to: BFO:MaterialEntity - GFO:Discrete_presential, which avoids the complement issue but it is not free of argument because GFO:Material_persistent is better suited for BFO:MaterialEntity due to the philosophy of BFO (see Section 4.2.1.4).
- **BFORO:located_in - GFO:occupies:** Section 4.3.1.3, inconsistency 6: The relation was changed to: BFORO:located_in subsumes GFO:occupies.
- **BFORO:location_of - GFO:occupied_by:** Section 4.3.1.3, inconsistency 7: The relation was changed to: BFORO:location_of subsumes GFO:occupied_by.
- **BFORO:has_participant - GFO:has_participant:** Section 4.3.1.3, inconsistency 8: The relation was changed to: BFORO:has_participant subsumes GFO:has_participant.

4.3.2. A higher-level foundational ontology

One of the main goals of the envisioned WFOL was for it to act as a starting point in ontology development, by adopting a high-level view of the most basic and common classes of ontologies within the WonderWeb library. This is achieved, in ROMULUS, by providing a higher-level foundational ontology, which we have called Foundational Foundational Ontology (FFO). FFO

may assist in foundational ontology interoperability because it is a single foundational ontology that consists of common entities of the three foundational ontologies: DOLCE-Lite, BFO and GFO.

The FFO is worthwhile to have in ROMULUS in that it assists the ontology developer with initial ontology entity representation. e.g., Based on the annotations of FFO:3D, a user could initially model a class *Armchair* as a FFO:3D entity. Thereafter, based on which entity branches of the FFO were used for development, a user can select an appropriate foundational ontology for development. If the FFO:3D branch is the only entity where the user creates classes, the user can select a module of ROMULUS with only 3D entities such as DOLCE-Endurants, BFO-Continuants etc. The FFO also gives us a good idea of the level of commonality that currently exists between the three foundational ontologies. Furthermore, any ontology based on FFO can be converted to either DOLCE, BFO or GFO.

It must be noted that GFO was envisioned to be a foundational ontology that is expressive enough to include other foundational ontologies. This may be true to a certain extent, but we have noticed that many entities from DOLCE-Lite and BFO are not found in GFO. Conversely, GFO has entities that are not found in DOLCE-Lite and BFO. Therefore, we thought it was best to rather create a higher-level foundational ontology by integrating only the most common mappings of DOLCE-Lite, BFO and GFO ontologies.

The FFO is represented in OWL DL and OWL 2 DL. Since FFO only contains entities that are common between DOLCE-Lite, BFO and GFO ontologies, it has only 7 classes and no object properties. Each entity in FFO is annotated with a clear definition and its equivalent class in DOLCE, BFO and GFO ontologies in order to enable usage of FFO. An example of an annotation in FFO is displayed in Fig. 4.2. FFO contains both 3D and 4D entities. Temporal entities in FFO mean that it takes on an eternalist stance. FFO has some support for entity attributes. FFO is modular in that its 3D and 4D entities are found in separate branches. It is freely available and actively maintained. To promote usage of FFO, its metadata can be found online at ROMULUS's metadata page¹. FFO is available for users to browse through and download in ROMULUS. The class taxonomy of FFO is displayed in Fig. 4.3.

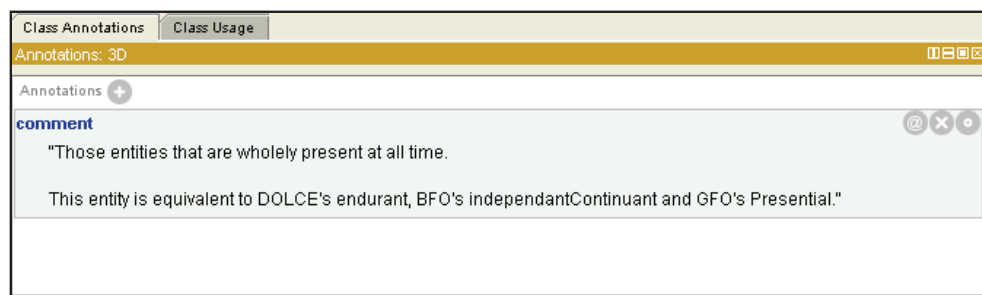


Figure 4.2.: Annotating the 3D entity in FFO.

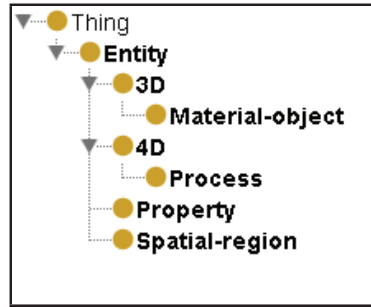


Figure 4.3.: The FFO class taxonomy.

4.3.3. FFO Mappings

In this section, we present a mapping between each FFO entity and its equivalent entity in the three foundational ontologies.

- FFO:3D is equivalent to DOLCE:endurant, BFO:IndependentContinuant and to GFO:Presential;
- FFO:Material-object is equivalent to DOLCE:physical-object, BFO:Object, and to GFO:Material-object;
- FFO:4D is equivalent to DOLCE:perdurant, BFO:Occurrent, and to GFO:Occurrent;
- FFO:Process is equivalent to DOLCE:process, BFO:Process, and to GFO:Process;
- FFO:Property is equivalent to DOLCE:quality, BFO:Quality, and to GFO:Property;
- FFO:Spatial-region is equivalent to DOLCE:space-region, BFO:SpatialRegion, and to GFO:Spatial-region;

4.4. Foundational ontology interchangeability method

In this section, we propose a manual method to perform foundational ontology interchangeability between domain ontologies. For this, we use the merged ontologies from ROMULUS, depending on which ontologies we wish to convert between. If we wish to convert a domain ontology linked to a source foundational ontology *S* to a target foundational ontology *T*, we use one of the merged ontologies that include ontologies *S* and *T*.

Notation: In the following method, we shall adopt the following conventions:

FOS A source foundational ontology.

FOT A target foundational ontology.

Entity_S The highest-level class in the taxonomy of FOS with equivalence relations to FOT.

EntityT The highest-level class in the taxonomy of FOT with equivalence relations to FOS.

EntityDS A main entity containing domain entities linked to FOS.

EntityDomain A domain entity.

EntitySuper A superclass of an EntityDomain.

EntitySuperEqual A class that is equivalent to EntitySuper.

Steps:

1. **Create new ontology:** Create a new ontology in Protégé.
2. **Import ontology files:** Directly import the following files into the ontology: the domain ontology file and the merged foundational ontology file from ROMULUS.
3. **Group ontology entities into highest-level taxonomy classes:** To simplify the process, we group the entities into highest-level taxonomy classes. In some cases there are already highest-level taxonomy classes e.g., DOLCE's highest-level class in the taxonomy is particular. Usually there should be two highest-level taxonomy classes: EntityS and EntityT. The domain entities are subsumed by EntityS. However, in some cases where the domain ontology uses a different version of FOS, we have a third highest-level taxonomy classes, EntityDS. If there are two highest-level taxonomy classes in the ontology, we skip the next step of the method, step 4. If there are three highest-level taxonomy classes in the ontology, we proceed with the next step of the method, step 4.
4. **Move domain entities from EntityDS to EntityS:** EntityDS and EntityS contain the same top-level classes, since they are of the same FOS. EntityDS, is however, populated with entities from a domain. Move an EntityDomain from EntityDS to EntityS as follows:
 - Select any EntityDomain from EntityDS to be moved to EntityS.
 - Identify EntityDomain's EntitySuper from EntityDS. If EntityDomain's EntitySuper is EntityDS, move EntityDomain to be a subclass of Thing, and restart the method with the next EntityDomain. In this case, EntityDomain is not contained in FOS and kept separate.
 - Identify a class with the same name as EntitySuper in EntityS.
 - Add EntityS's EntitySuper as a superclass of EntityDomain.
 - Remove EntityDS's EntitySuper as a superclass of EntityDomain.Continue moving EntityDomain entities from EntityDS to EntityS until there are no longer any EntityDomain entities in EntityDS.
5. **Move domain entities from EntityS to EntityT:** EntityS and EntityT contain classes from FOS and FOT respectively, linked by equivalence relations. However, EntityS contains EntityDomain entities. Move an EntityDomain from EntityS to EntityT as follows:
 - Select any EntityDomain from EntityS to be moved to EntityT.
 - Identify EntityDomain's superclass EntitySuper from EntityS.

- Identify a EntitySuperEqual corresponding to EntitySuper in EntityT. If there is no EntitySuperEqual in EntityT, we treat the identified EntitySuper in EntityS as an EntityDomain, until we find an EntitySuper that has a corresponding EntitySuperEqual in EntityT. If there is no corresponding EntitySuperEqual thereafter, we move the EntityDomain to be a subclass of Thing, and restart the method with the next EntityDomain. In this case, EntityDomain is not contained in FOT and kept separate.
- Add EntityT's EntitySuper as a superclass of EntityDomain.
- Remove EntityS's EntitySuper as a superclass of EntityDomain.

Continue moving EntityDomain entities from EntityS to EntityT until there are no longer any EntityDomain entities in EntityS.

6. **Repeat:** Repeat steps 3 - 5 for object properties in the ontology.
7. **Delete EntityS and EntityDS :** Once all EntityDomain entities have been moved from EntityS and EntityDS; delete EntityS, EntityDS and their subclasses from the ontology. Now EntityT exists as the main entity. EntityT contains the EntityDomain entities linked to FOT, as desired.

The mediation of foundational ontologies has proven to be a promising method to assist with foundational ontology interchangeability. Naturally there are issues that arise when aligning foundational ontologies i.e., approximate alignments and logical inconsistencies when mapping. We have solved some of these issues, however, there are still a number of unsolved issues that seem to be beyond our scope due to foundational ontology structure and hierarchy. Identifying these issues brings us a step forward to solving them and improving interoperability. In the following chapter we will discuss the design and implementation of the web-based repository, ROMULUS, and how it assists a user with foundational ontology usage and mediation.

Web-based ROMULUS

In line with our goal of creating a foundational ontology repository, we have designed and implemented a web-based software system ROMULUS in order to allow individuals to publicly access and benefit from all the functionality of the repository.

5.1. Requirements

A number of functional and non-functional requirements must be met to ensure the success of the repository.

5.1.1. Functional requirements

The proposed repository must meet a number of functional requirements. The first three functional requirements are adapted from WFOL [53].

- **Minimal:** The repository is to be as general as possible, including only the most reusable and widely applicable upper-level entities.
- **Rigorous:** The ontologies in the libraries will be characterized by means of rich axiomatisations.
- **Extensively researched:** Each module in the library will be added only after careful consideration and thorough research.
- **Metadata:** Metadata of each ontology must be available to the user.
- **Foundational ontology comparison:** It must provide a comparison of implemented foundational ontologies.
- **Ontology views:** Human-readable views of each ontology must be provided to the user.

- **Modularity:** Foundational ontologies in the repository must be modularised.
- **Mediation:** Foundational ontology mediation must be performed. This includes alignment, mapping and merging of foundational ontologies.
- **Online ontology browsing:** It must allow for easy and effective online browsing of all the ontologies in the repository.
- **Access to download resources:** It must allow the user to download all ontologies and resources. This includes foundational ontologies, modules, mappings, merged ontologies, ROMULUS's documentation, ONSET and ONSET's documentation.

5.1.2. Non-functional requirements

Non-functional requirements are essential for the overall quality of the software. These include:

- **Maintainability:** The proposed library must be designed to ensure that it may be extended easily.
- **Usability:** Users must feel comfortable and at ease using the tool.
- **Modular design:** The tool must be designed in a modular way making it simple to perform specific operations.
- **Response time:** The time taken in performing operations must be minimal.
- **Accessibility:** The repository must be available to the general public for usage.
- **Documentation:** Documentation to enhance usage must be easily accessible for users.

5.2. Design

In this section we discuss the design of ROMULUS. ROMULUS is modular in design because it has many functions and it is not feasible to group all features together. The different functions of ROMULUS are provided in different tabs of the repository page. We provide a top-level conceptualisation of ROMULUS's front-end in Fig. 5.1, in which we see that each function is conceptualised separately. The three main components of ROMULUS are the web server, Tomcat server and database. The PHP-based web server is used to execute HTML pages and PHP scripts, the Tomcat server is used to execute JSP pages and the MySQL database is used to store ontology alignments, ontology metadata and users' ontology selection results and to assist with search functions. The interaction of these components is provided in Fig. 5.2.

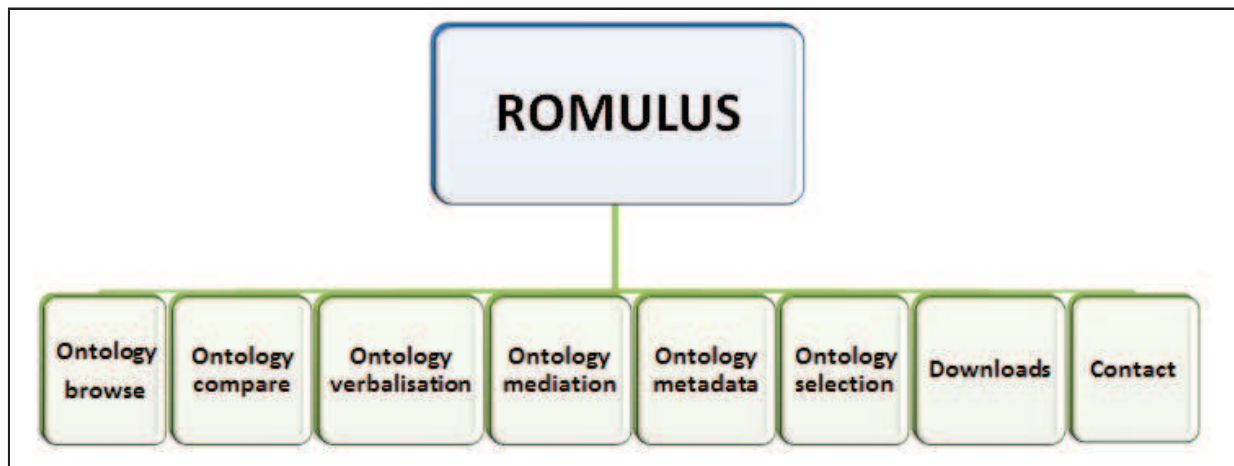


Figure 5.1.: A conceptualisation of ROMULUS's front-end system.

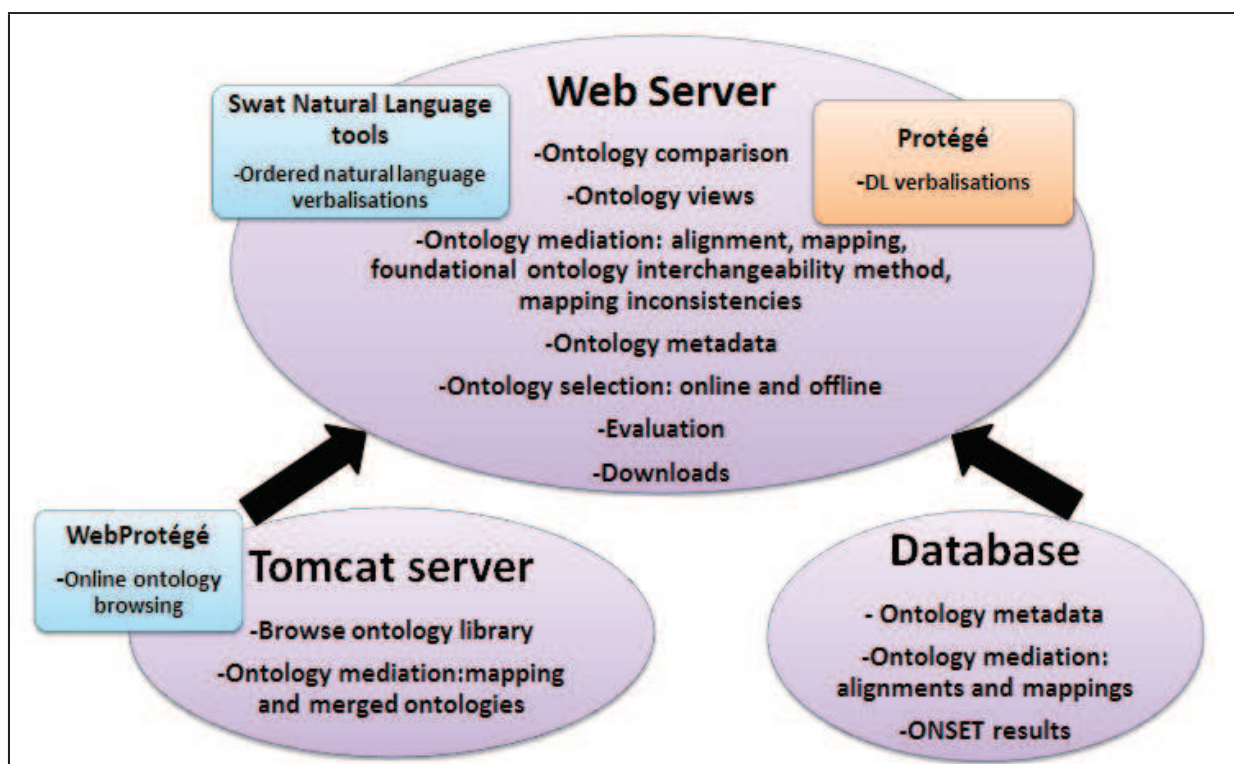


Figure 5.2.: The interaction of ROMULUS's components.

In order to use the FFO, there are three layers of ontologies to be used. The FFO forms the top-layer of the architecture, and is aimed at providing the user with a starting point in ontology development. This layer assists the user with representing their high-level entities and based on which entities of the FFO are used, the user can select an appropriate ontology module to be used

for development (refer to Section 4.3.2). The middle layer is the layer of foundational ontologies. These ontologies are used to improve the quality of the ontology to be developed and speed up the development process. Lastly, the bottom layer has domain ontologies. This is displayed in Fig. 5.3.

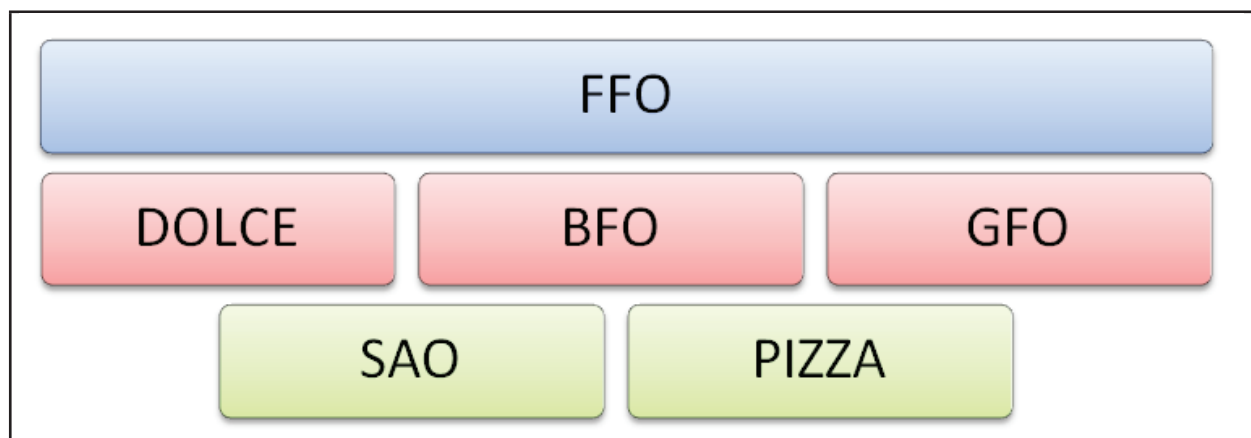


Figure 5.3.: The ontology layers upon which FFO may be used.

ROMULUS has a centralised database, which is used to store ontology selection results, ontology mediation data and ontology metadata. A conceptual data model for the ontology selection results that are stored in the database results is displayed in Fig. 5.4. Fig. 5.5 depicts a conceptual data model of ROMULUS's metadata and mediation in the database.

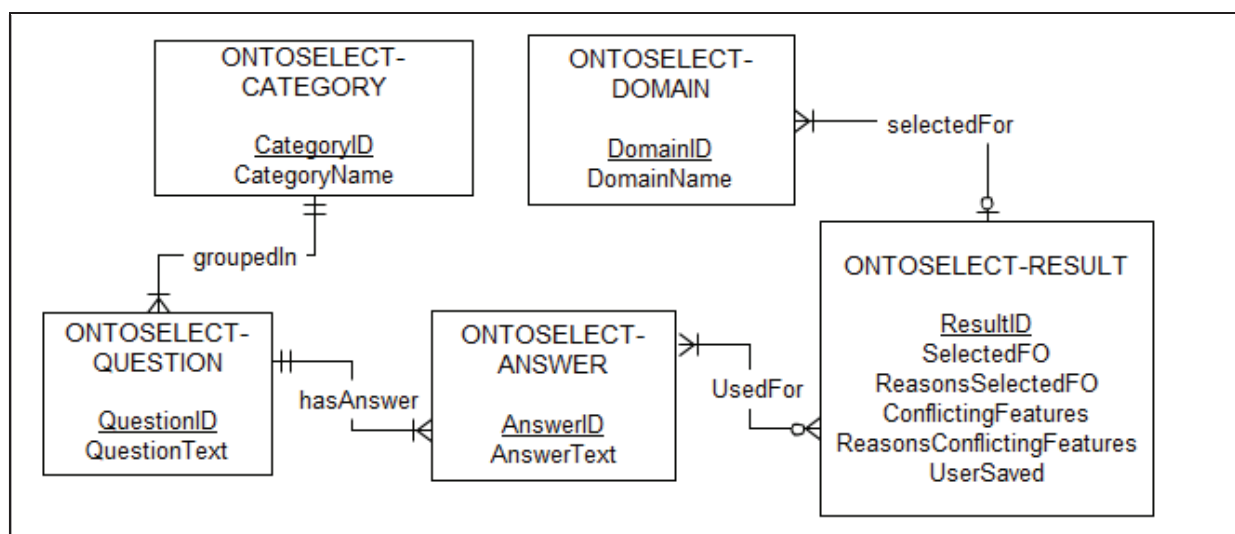


Figure 5.4.: ER diagram of ROMULUS's database regarding ONSET's saved data.

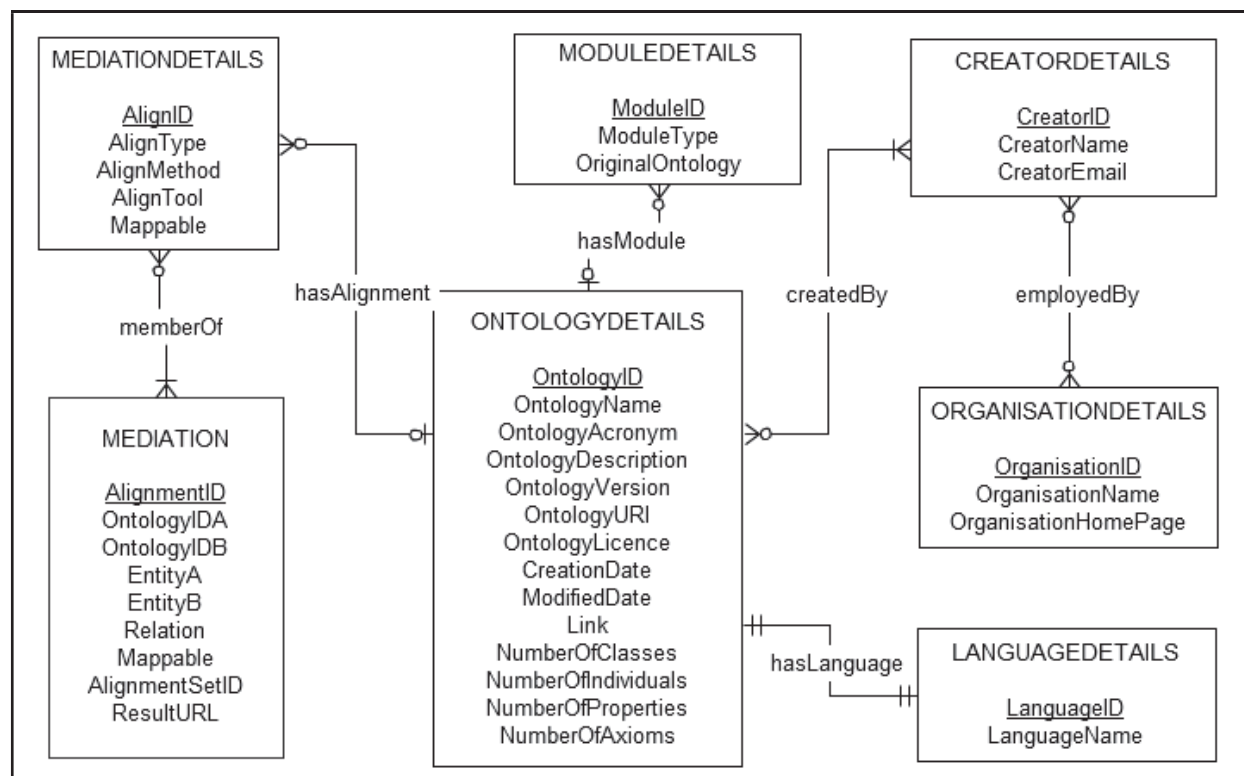


Figure 5.5.: ER diagram of ROMULUS's database regarding ontology metadata and mediation.

5.2.1. Meeting functional requirements

ROMULUS meets all its functional requirements as follows:

- **Minimal:** The FFO in ROMULUS is a general high-level ontology which contains the most reusable entities from the three foundational ontologies.
- **Rigorous:** ROMULUS contains the OWL versions of all the foundational ontologies and related modules.
- **Extensively researched:** Each ontology in ROMULUS has been added after an extensive review of literature and multi-dimensional comparison were performed.
- **Metadata:** There is comprehensive metadata for each ontology in ROMULUS. This facilitates foundational ontology reuse, and there is search functionality to easily locate specific metadata.
- **Foundational ontology comparison:** ROMULUS provides a multi-dimensional comparison of the foundational ontologies.
- **Ontology views:** There are human-readable views in ordered natural language and DL format of the ontologies.

- **Modularity:** The foundational ontologies in ROMULUS have been modularised in a number of ways e.g., separate 3D and 4D entity modules, OWL 2 profiles etc. (Refer to Section 5.4.1.2 for the module details).
- **Mediation:** The foundational ontologies have been aligned, mapped and merged. Furthermore there is a method to assist the user with performing foundational ontology interchangeability.
- **Online ontology browsing:** ROMULUS offers the user with online browsing pages for the ontologies.
- **Access to download resources** ROMULUS's download page allows the user to download all resources including foundational ontologies, mapped and merged ontologies, modules, ONSET, ONSET's documentation and ROMULUS's documentation.

5.3. Implementation

ROMULUS is built upon many existing technologies, tools and tool outputs. In this section, we discuss the roles and functions of the employed tools and technologies in ROMULUS.

ROMULUS is hosted on Openshift Online, Red Hat's Platform as a Service (PaaS) offering. It enables users to build, execute and host applications online, and supports a variety of widely-used technologies, such as Java, PHP and MySQL. From the evaluation of browsing tools performed earlier in Section 2.8.1, WebProtégé was selected to provide online ontology browsing for a number of reasons. Firstly, it is one of the very few existing online ontology browsing tools. Its interface is simple to use and understand yet contains important ontology building blocks such as class and object property hierarchies, relations and annotations. WebProtégé's design is easily customizable. Similar to Protégé, WebProtégé is a java application, but since it is web-based it is made up of JavaServer Pages (JSP). In order to execute these JSP pages, a JSP enabled web-server such as Tomcat is required. For this, all of WebProtégé's pages and resources to be used in ROMULUS, are to be executed on the JSP-enabled Tomcat web-server. ROMULUS's Tomcat server presently only contains WebProtégé-related resources. We provide ontology browsing pages for the foundational ontologies and related modules, the mapping ontologies and the merged ontologies.

HTML tables and lists are used to illustrate a comparison of foundational ontologies for the different categories of criteria. Protégé was used to generate the axioms of each ontology module in DL. This was saved into PDF files, which are embedded into the ROMULUS pages. SWAT natural language tools was used to generate HTML pages of the views of each ontology in an ordered natural language format. ROMULUS's ontology alignments and metadata for each ontology are stored in the centralised database and rendered as HTML tables. Users are able to perform filtered searches through both the alignments and the metadata. ONSET, the foundational ontology selection tool is integrated into ROMULUS where it may be executed online and downloaded for offline usage.

5.4. ROMULUS's features

In this section, we describe each feature provided by ROMULUS, and how it may be accessed and used. Refer to Appendix C for the ROMULUS documentation.

5.4.1. Browse ontologies

ROMULUS uses WebProtégé to provide ontology browsing. Similarly to Protégé, there are different tabs with classes, object properties and individuals of an ontology. Users are able to traverse through the class and object property hierarchies. One can also view the annotations and asserted conditions of an entity. A screenshot in Fig. 5.6 displays the ontology browsing feature.

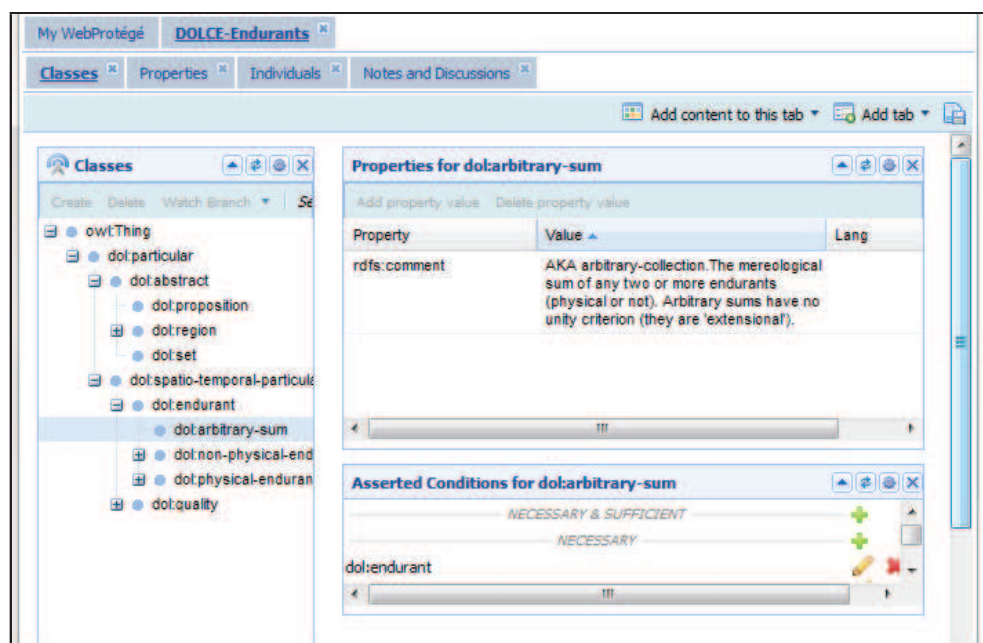


Figure 5.6.: The ontology browsing feature in ROMULUS.

The ontology browsing library contains foundational ontologies and related modules. Mappings and merged ontologies exist in ROMULUS's mediation pages, introduced in Section 5.4.5, in separate libraries of the same type.

5.4.1.1. Foundational ontologies

The following foundational ontologies exist in ROMULUS:

- **DOLCE2.0-Lite-v3:** This version of DOLCE does not contain modality, temporal indexing, relation composition.

- **BFO-1.1:** The BFO ontology.
- **BFORO:** The BFO ontology combined with relational ontology.
- **GFO:** The full version of GFO.

5.4.1.2. Modular ontologies

We have the following types of modules in ROMULUS:

- **Separate branches of 3D and 4D entities in the ontologies:** This is used when one wants to keep the entities that exist as a whole at all times (3D entities) separate from entities with temporal parts that unfold over time (4D entities).
- **Isolated branches of taxonomies of the ontologies for available subject domains support:** Having modules that can be used for specific subject domains e.g., biomedical, business.
- **More/less-detailed versions of the ontologies:** Having different variations of ontologies with fewer/more entities, relational properties and axioms e.g., gfo-basic.
- **OWL 2 profiles:** Having modules in different fragments of OWL 2. e.g., OWL 2 QL

The ontology modularisation tools, OWL Module extractor, Swoop and Protégé, introduced in Section 2.5 were considered for modularising the foundational ontologies. A list of existing and newly created modules, in OWL, are presented below.

5.4.1.3. DOLCE modules

The following modules which we will use are already available:

- **FunctionalParticipation:** This module contains functional participation relations, based on traditional literature on thematic relational properties.
- **SpatialRelations:** This module contains spatial relations which extend the local relations from DOLCE.
- **TemporalRelations:** This module contains temporal relations.

Additionally we have created the following modules for DOLCE:

- **DOLCEEndurants:** This module contains no perdurant entities.
- **DOLCEPerdurants:** This module contains no endurant entities.
- **DOLCENoQualityAndQualia:** This module contains no quality and qualia entities.
- **DOLCEEL:** This module is an OWL 2 EL profile of DOLCE.
- **DOLCEQL:** This module is an OWL 2 QL profile of DOLCE.

5.4.1.4. BFO modules

No modules exist for the BFO ontology.

We have created the following modules for BFO:

- **BFOContinuants:** This module contains only Continuant entities.
- **BFOOccurrents:** This module contains only Occurrent entities.

5.4.1.5. GFO modules

The following modules which we will use are already available:

- **gfo-basic:** This is a stable core of GFO.

Additionally we have created the following modules for GFO:

- **GFOATO:** This module is based on the Abstract Top Level layer which contains mainly two meta-categories: Set and Item.
- **GFOACO:** This module is based on the Abstract Core Level which contains meta-categories over the basic level: Category and Individual.
- **GFONoOccurrents:** This module contains no Occurrent entities.
- **GFONoPersistantsAndPresentials:** This module contains no Persistent nor Presential entities.
- **GFOBasicEL:** This module is an OWL 2 EL profile of GFO.
- **GFOBasicQL:** This module is an OWL 2 QL profile of GFO.

5.4.1.6. Difficulties in performing modularisation

OWL Module extractor, Protégé and Swoop use a logic based analysis of the axioms only and this resulted in large modules similar to the original ontologies. For this reason, we only applied these tools as a starting point in module creation, and in most cases we manually modularised the foundational ontologies after using the tools.

In DOLCE, *endurant* and *perdurant* are linked by a participation relation, making it difficult to separate them into separate hierarchies. In order to create modules of these types, it was necessary to manually remove some of the axioms relating the two entities. We encountered a similar problem when modularising DOLCE to be a module without quality and qualia.

5.4.1.7. Mapping and merged ontologies

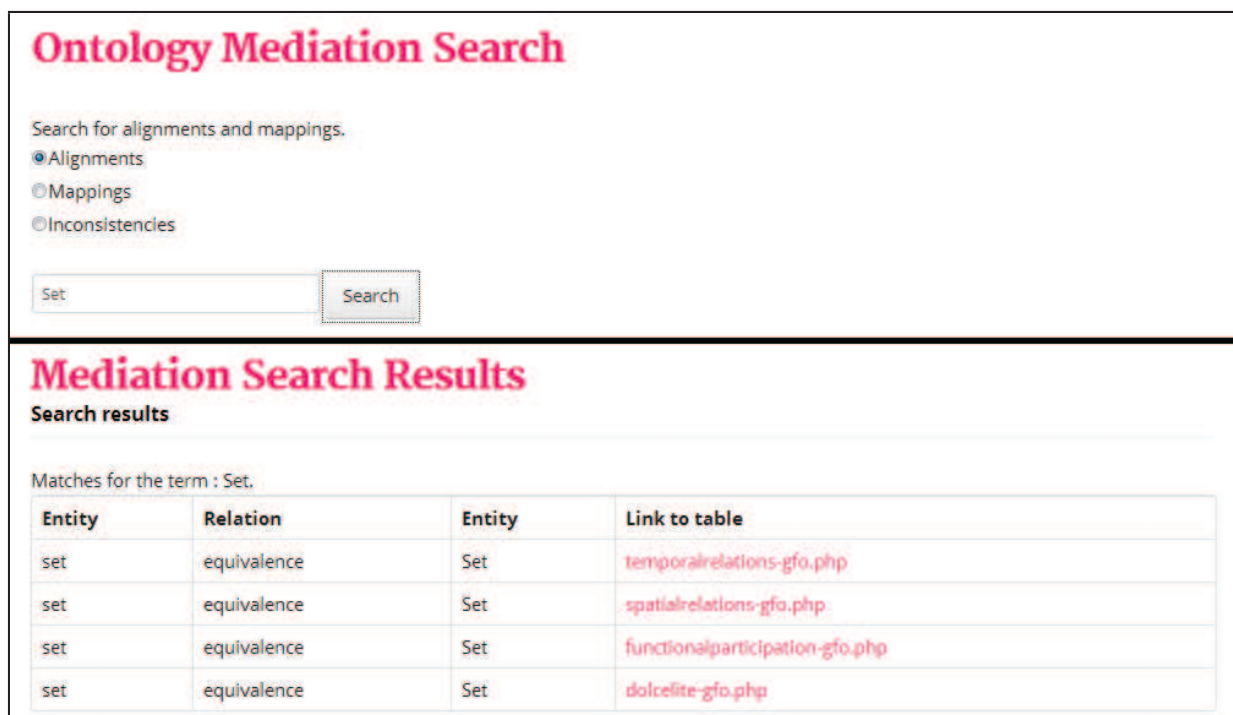
The mapping and merged ontologies are based on the ontology alignment pairs presented in Section 4.2.

5.4.2. Ontology comparison

ROMULUS has a multi-categorical criteria comparison of foundational ontologies. Within the comparison function, there is a page for each of the following criteria: ontological commitments, representation languages, software engineering properties, subject domains, and applications

5.4.3. Search

ROMULUS's metadata and mediation data are stored in its centralised database and rendered as HTML tables. Therefore users are able to search through the metadata and mediation pages using keywords. For the metadata search, a user is able to filter a search by the following criteria: ontology name, ontology acronym, ontology language, organisation name and ontology developer. This is shown in Fig. 5.7. For the mediation search a user is able to filter a search by the following criteria: alignments, mappings and inconsistencies. This is shown in Fig. 5.8.



Ontology Mediation Search

Search for alignments and mappings.

☒ Alignments
☐ Mappings
☐ Inconsistencies

Mediation Search Results

Search results

Matches for the term : Set.

Entity	Relation	Entity	Link to table
set	equivalence	Set	temporalrelations-gfo.php
set	equivalence	Set	spatialrelations-gfo.php
set	equivalence	Set	functionalparticipation-gfo.php
set	equivalence	Set	dolcelite-gfo.php

Figure 5.7.: Ontology alignment search.

5.4.4. Ontology views

A view of the axioms in each ontology is provided in ROMULUS. We have two different views available:

Ontology Metadata Search

To search through the metadata, please select an attribute type and type in a keyword below.

☐ Ontology Name/ Acronym
☐ Ontology Language
☐ Organisation Name
☒ Ontology Developer

Loebe Search

Metadata Search Results

Search results

ontologydeveloperMatches for the term : **Loebe**.

Ontology Developer Name	Link to metadata
Frank Loebe	gfo
Frank Loebe	gfo-basic

Figure 5.8.: Ontology metadata search.

1. **DL view:** The axioms are expressed in the DL language. To generate this view, we use Protégé version 4.2, which renders some text. Thereafter, we save the text in a file in $\text{\LaTeX}$ format and execute it, which provides a PDF file of the axioms in DL. A screenshot of this view is shown in Fig. 5.9.

Description logic view: GFO	
Classes	
Abstract	
Abstract	$\sqsubseteq$ Individual
Abstract	$\sqsubseteq \neg$ Space_time
Abstract	$\sqsubseteq \neg$ Concrete
Action	
Action	$\sqsubseteq$ Occurrent
Action	$\sqsubseteq \exists$ has_agent Presential

Figure 5.9.: Ontology view: DL view.

2. **Natural language view:** The axioms are formalised in natural language. To generate this view, we use the SWAT Natural language tools, which creates a HTML page of the ontology view. A screenshot of this view is shown in Fig. 5.10.

CONNECTED SPATIOTEMPORAL REGION (class)	
Definition	A <u>connected spatiotemporal region</u> is defined as something that is a <u>spatiotemporal instant</u> , or is a <u>spatiotemporal interval</u> .
Typology	A <u>connected spatiotemporal region</u> is a <u>spatiotemporal region</u> .
Examples	<u>Spatiotemporal intervals</u> , and <u>spatiotemporal instants</u> are <u>connected spatiotemporal regions</u> .
Distinctions	No <u>connected spatiotemporal region</u> is a <u>scattered spatiotemporal region</u> .
CONNECTED TEMPORAL REGION (class)	
Definition	A <u>connected temporal region</u> is defined as something that is a <u>temporal instant</u> , or is a <u>temporal interval</u> .
Typology	A <u>connected temporal region</u> is a <u>temporal region</u> .
Examples	<u>Temporal intervals</u> , and <u>temporal instants</u> are <u>connected temporal regions</u> .
Distinctions	No <u>connected temporal region</u> is a <u>scattered temporal region</u> .

Figure 5.10.: Ontology view: Natural language view.

5.4.5. Ontology mediation

The output of ontology mediation, previously described in Chapter 4, is available in ROMULUS. Users are provided with ontological alignments in the form of tables, and mapping and merged ontologies to be browsed online, as visualised in Fig. 5.6. Since the alignments are stored in ROMULUS's database, users are also able to search through them. The method for performing foundational ontology interchangeability from Section 4.4 and the logical inconsistencies between aligned entities from Section 4.3.1 is also available in the mediation pages of ROMULUS.

5.4.6. Ontology selection

Since ROMULUS is a repository of foundational ontologies, we believe it is important to have a foundational ontology selector in it. ONSET is such a—and, to date, the only—foundational ontology selection and explanation tool that was designed to aid the user with the task of selecting an appropriate foundational ontology for domain ontology development. It assists developers by informing them about the criteria of a particular foundational ontology and the way in which it relates to the ontology to be developed. If the user has requirements relating to more than one foundational ontology, conflicting results are compared and displayed. Conflicting results is the term we use to describe what is provided by the selected foundational ontology compared to what the user wants. For instance, if DOLCE is the selected ontology but the user required a realist ontology, this is a conflicting result because realism is offered by other foundational ontologies (BFO, GFO etc.) and not by DOLCE. ONSET also generates, where possible, a list of existing projects related to the user's selected domain, in the form of references. ONSET may be used at different stages of ontology development whether it is at the start of development or during improvement of existing domain ontology. In this section, we summarise the evolution and evaluations of ONSET, since its initial development as a BSc Honours project, and the subsequent publication at The 18th International Conference on Knowledge Engineering and Knowledge Management [45].

Since the initial development of ONSET, where it had DOLCE and BFO ontologies implemented, it has been extensively modified. Significant changes and improvements have been made to ONSET, with an aim to integrate it in ROMULUS. Instead of a sole stand-alone application, there is also now a web-based version has the following features:

- It has access to ROMULUS's centralised database.
- This web-based version provides the user with links to features that are met by ROMULUS, such as metadata and modules for the selected foundational ontology.
- There is also a new feature where users can save their ontology selection results both locally in a uniformed CSV file format, and to let ONSET store a copy of it in ROMULUS's database, which can be used for further analysis and investigation with regard to foundational ontology usage and selection. The following data is saved from ONSET to the CSV file or database: Each answer that the user provided together with its respective question and category, the selected foundational ontology and reasons and conflicting results and reasons (if any).
- We have added GFO, SUMO, YAMATO and GIST foundational ontologies to the new version after receiving feedback from the respective ontology developers, therewith providing the user with more possible foundational ontology choices. Thanks to the GFO ontology developers, we received feedback for the GFO criteria list that we created thereby improving our criteria list. Thanks to the YAMATO and GIST ontology developers, we received criteria for the respective ontologies and we were able to easily include their ontologies in ONSET. Furthermore we have received feedback on the general ontological criteria used in ONSET from them.
- By adding GFO, SUMO, YAMATO and GIST to ONSET, there has been some new criteria added for each category e.g., situations and situoids under the ontological commitments category and ontologies for learning and instructional theories under the applications category.

The stand-alone version of ONSET has all of the above new features except for saving the data in ROMULUS's database. Fig. 5.11 is a screenshot of ONSET's output.

ONSET's functionality has been evaluated by ontological use-cases and experimental evaluation. In the ontological use-cases from existing applications it was found that ONSET's choice of a foundational ontology corresponds to the choices and selection of the use-cases. From the simulated ontological use-cases we see that when the scaling functionality in ONSET is used, it makes a difference to its selection. For ONSET's experimental evaluation, novice ontology developers from the UKZN honours class were to perform ontology selection for five scenarios. The class was divided into two groups, A and B. Group A was to complete the task of five scenarios using their lecture notes and resources from the internet. Group B was to complete the same tasks using the same resources as Group A and additionally, ONSET. The results of the experiment indicated that Group B performed ontology selection twice as accurately as Group A and had also completed their tasks in a shorter time period. Table 5.1 shows a comparison of the

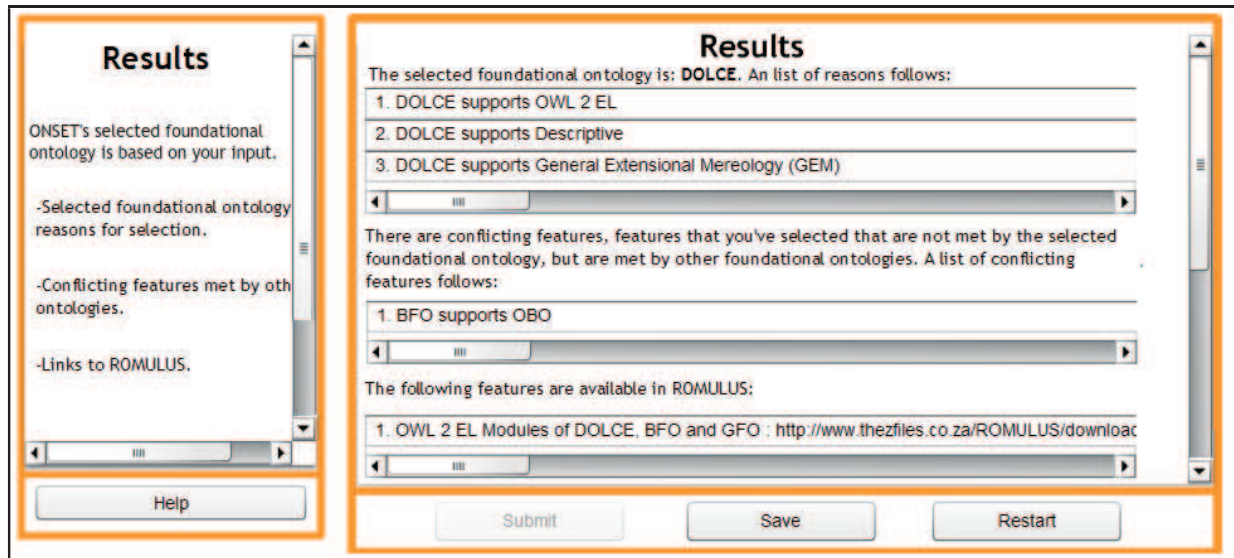


Figure 5.11.: Output from the new version of ONSET (version 2.0).

Table 5.1.: A comparison of the accuracy of ontology selection by Group A and Group B, for each of the five scenarios of the experiment. Source:[45].

Scenario	Group A Average	Group B Average
1.Ontology of heart diseases	22%	52%
2.Ontology for the integration of databases of a manufacturing factory	16%	43%
3.Ontology of economic systems	20%	48%
4.Ontology of banks	16%	37%
5.Ontology for conceptual data models	8%	51%
<i>All Scenarios</i>	<i>16%</i>	<i>46%</i>

accuracy rates of both groups. From the experimental evaluation, we conclude that ONSET aids in ontology selection with respect to accuracy and time-taken.

5.4.7. Ontology metadata

Metadata values for each original, modularised, mapped, and merged ontology are provided in ROMULUS. Metadata is important to have in ROMULUS in that provides the ontology developer with additional data pertaining to a foundational ontology to assist the ontology developer with reusing that ontology effectively. For the proposed library, we aim to create an extensive and descriptive list of metadata for each ontology. We use criteria from the Ontology Metadata Vocabulary (OMV), the OM²R metadata model and introduce a new metadata criteria. The metadata is stored in ROMULUS's database, and rendered as HTML tables, through which users

can conduct filtered searches. At this stage, we do not include the OWL formalisation of OMV and OM²R in the ontologies themselves, but standalone metadata for each ontology. In the near future, we intend to integrate the OWL formalisation of OMV and OM²R to each ontology in order to facilitate interoperability between machines.

5.4.7.1. OMV metadata criteria

We used the following metadata from the OMV model for ROMULUS ontologies.

Ontology details

- Ontology name
- Ontology acronym
- Ontology ID
- Ontology description
- Ontology creation date
- Ontology latest modified date
- Ontology version
- Ontology URI
- Ontology languages
- Ontology licence

Organisation details

- Ontology documentation page
- Ontology creators contact details
- Organisation name
- Organisation homepage

Metrics

- Number of classes in the ontology
- Number of individuals in the ontology
- Number of properties in the ontology
- Number of axioms in the ontology

5.4.7.2. OM²R metadata criteria

We used the following metadata from the OM²R model for ROMULUS ontologies.

Matching

- Matching method
- Matching tool

5.4.7.3. ROMULUS metadata criteria

We created the following new metadata for ROMULUS ontologies.

Modularity

- Module type
- Original ontology

Mediation

- Original ontologies
- Alignment type

5.4.7.4. Metadata lists

We provide the metadata values for the following ontologies: DOLCE, BFO and GFO, being the three main ontologies in ROMULUS; DOLCE-Lite-EL, being an ontology module thus having modularity criteria (in bold font); and BFOGFOMappings, being a mediated ontology thus having mediation criteria (in bold font). Refer to Tables 5.2, 5.3, 5.4, 5.5, and 5.6 to view these metadata values. Metadata values for all the ontologies in ROMULUS can be accessed at ROMULUS's metadata page¹.

5.4.8. Downloads

This page contains links to each foundational ontology (original, modularised, mapping, and merged) within ROMULUS, ROMULUS's documentation, ONSET and ONSET's documentation, and supplementary materials.

The web-based repository of foundational ontologies, ROMULUS, provides capability to assist the user with foundational ontology usage and interoperability by providing features such as foundational ontology browsing, verbalisation and views, metadata, comparison, mediation and ontology selection. In order to facilitate this, a number of ontology engineering tools and techniques are implemented and share interaction within ROMULUS. Every functional and non-functional requirement identified has been successfully met by ROMULUS. In the next chapter, we evaluate ROMULUS, both theoretically, by assessing the quality of its alignments, and practically, by comparing its functions to those of other ontology repositories.

¹<http://www.thezfiles.co.za/ROMULUS/ontologyMetadata.html>

Table 5.2.: List of metadata for DOLCE-Lite.

Entity	Value
Ontology details	
Ontology name	A Descriptive Ontology for Linguistic and Cognitive Engineering (Lite)
Ontology acronym	DOLCE-Lite
Ontology ID	1
Ontology description	DOLCE is the first module of a Library of Foundational Ontologies in WonderWeb project. DOLCE's categories are based on common-sense principals and natural language. DOLCE-Lite is a simplified version of the full ontology that does not consider: modality, temporal indexing, relation composition.
Ontology creation date	10 December 2002
Ontology latest modified date	28 June 2006
Ontology version	397
Ontology URI	http://www.loa-cnr.it/ontologies/DOLCE-Lite.owl
Ontology languages	DAML, KIF, FOL, OWL 2, OWL 2 DL
Ontology licence	Free
Organisation details	
Ontology documentation page	http://www.loa.istc.cnr.it/Papers/dolce_docs.zip
Ontology creators contact details	Claudio Masolo email: masolo@loa-cnr.it , Stefano Borgo email: borgo@loa-cnr.it , Aldo Gangemi email: aldo.gangemi@cnr.it , Nicola Guarino email: guarino@loa-cnr.it , Alessandro Oltramari email: aoltrama@andrew.cmu.edu
Organisation name	The Laboratory for Applied Ontology (LOA). Institute of Cognitive Science and Technology Italian National Research Council
Organisation homepage	http://www.loa.istc.cnr.it/
Metrics	
Number of classes in the ontology	37
Number of individuals in the ontology	0
Number of properties in the ontology	70
Number of axioms in the ontology	349

Table 5.3.: List of metadata for BFO.

Entity	Value
Ontology details	
Ontology name	Basic Formal Ontology (1.1)
Ontology acronym	BFO1.1
Ontology ID	11
Ontology description	BFO is a foundational ontology which focusses on support of domain ontologies developed for scientific research.
Ontology creation date	2003
Ontology latest modified date	Unknown
Ontology version	1.1.1
Ontology URI	http://www.ifomis.org/bfo/1.1
Ontology languages	All OWL species, OBO.
Ontology licence	Free
Organisation details	
Ontology documentation page	http://www.ifomis.org/bfo/home
Ontology creators contact details	Barry Smith email: phismith@buffalo.edu , Pierre Grenon email: pgrenon@ebi.ac.uk , Holger Stenzhorn email: holger.stenzhorn@uks.eu and others
Organisation name	Institute for Formal Ontology and Medical Information Science (IFOMIS)
Organisation homepage	http://www.ifomis.org
Metrics	
Number of classes in the ontology	39
Number of individuals in the ontology	0
Number of properties in the ontology	0
Number of axioms in the ontology	95

Table 5.4.: List of metadata for GFO.

Entity	Value
Ontology details	
Ontology name	General Formal Ontology
Ontology acronym	GFO
Ontology ID	15
Ontology description	GFO is a foundational ontology for conceptual modelling. GFO exhibits a three layered metaontological architecture consisting of an abstract top level, an abstract core level, and a basic level.
Ontology creation date	28 August 2006
Ontology latest modified date	28 August 2006
Ontology version	1.0 build 9
Ontology URI	http://www.onto-med.de/ontologies/gfo.owl
Ontology languages	OWL 2, OWL 2 DL
Ontology licence	Free
Organisation details	
Ontology documentation page	http://www.onto-med.de/publications/index.jsp
Ontology creators contact details	Heinrich Herre email: herre@informatik.uni-leipzig.de , Barbara Heller email: barbara.heller@ontomed.de , Patryk Burek email: burek@infomatik.uni-leipzig.de , Robert Hoehndorf email: rh497@cam.ac.uk , Frank Loebe email: frank.loebe@informatik.uni-leipzig.de and Hannes Michalek email: hannes@michalek.de .
Organisation name	Ontologies in Medicine and Life Sciences Foundations, Development and Applications (Onto-Med)
Organisation homepage	http://www.onto-med.de/
Metrics	
Number of classes in the ontology	78
Number of individuals in the ontology	1
Number of properties in the ontology	67
Number of axioms in the ontology	323

Table 5.5.: List of metadata for DOLCE-Lite-EL module; the criteria in boldfont is ROMULUS's new metadata criteria.

Entity	Value
Ontology details	
Ontology name	A Descriptive Ontology for Linguistic and Cognitive Engineering (Lite-EL)
Ontology acronym	DOLCE-Lite-EL
Ontology ID	9
Ontology description	DOLCE modularised in an OWL 2 EL fragment.
Ontology creation date	24 August 2012
Ontology latest modified date	24 August 2012
Ontology version	1
Ontology URI	http://www.thezfiles.co.za/ontologies/DOLCE-EL.owl
Ontology languages	OWL 2, OWL 2 DL, OWL 2 EL
Ontology licence	Free
Organisation details	
Ontology documentation page	
Ontology creators contact details	Zubeida Casmod Dawood email: zkhan@csir.co.za
Organisation name	University of KwaZulu-Natal (UKZN) and UKZN/CSIR Centre for Artificial Intelligence Research (CAIR), South Africa
Organisation homepage	http://www.ukzn.ac.za
Metrics	
Number of classes in the ontology	37
Number of individuals in the ontology	0
Number of properties in the ontology	70
Number of axioms in the ontology	280
Modularity	
Module type	OWL 2 profile
Original ontology	DOLCE-Lite

Table 5.6.: List of metadata for BFOGFOMappings; the criteria in boldfont is ROMULUS's new mediation criteria.

Entity	Value
Ontology details	
Ontology name	Basic Formal Ontology - General Formal Ontology Mappings
Ontology acronym	BFOGFOMappings
Ontology ID	44
Ontology description	The mappings between BFO and GFO ontologies.
Ontology creation date	19 November 2012
Ontology latest modified date	19 November 2012
Ontology version	1
Ontology URI	http://www.thezfiles.co.za/ontologies/BFOGFOMappings.owl
Ontology languages	OWL 2
Ontology licence	Free
Organisation details	
Ontology documentation page	
Ontology creators contact details	Zubeida Casmod Dawood email: zkhan@csir.co.za
Organisation name	University of KwaZulu-Natal (UKZN) and UKZN/CSIR Centre for Artificial Intelligence Research (CAIR), South Africa
Organisation homepage	http://www.ukzn.ac.za
Metrics	
Number of classes in the ontology	12
Number of individuals in the ontology	0
Number of properties in the ontology	0
Number of axioms in the ontology	6
Mediation	
Matching method	Mixed: Manual matching with some tool output
Matching tool/s	H-Match, PROMPT, LogMap, YAM++, HotMatch, Hertuda and Optima
Original ontologies	BFO, GFO
Alignment type	Foundational ontology to foundational ontology

Evaluation and discussion

In order to assess ROMULUS’s functionality, usability and accuracy of alignments, it has been evaluated in three ways. We discuss and summarise each evaluation and its results in the following sections. Thereafter we discuss ROMULUS’s goals and functional requirements relative to those of the envisioned WFOL.

6.1. Evaluating foundational ontology interchangeability

We evaluate the foundational interchangeability in ROMULUS by converting the Subcellular Anatomy Ontology(SAO) [49] from BFO to DOLCE-Lite. We use the method proposed in Section 4.4 to perform foundational ontology interchangeability.

1. **Create new ontology:** We use Protégé v4.1.
2. **Import ontology files:** We have imported the following files:
 - SAO.owl: The SAO ontology in BFO
 - BFORODOLCE-LiteMerged.owl: A merged ontology of BFO with RO and DOLCE-Lite.
3. **Group ontology entities into main entities:** After importing the two files, we notice that there are three main super entities in the ontology: entity, entity and particular. Fig. 6.1 displays this. There are two main BFO entities: entity and entity. The first entity contains entities from SAO ontology axiomatised in BFO, while the second entity is the top-level entity of BFO from the merged ontology. This is because SAO is linked to an older version of BFO, therefore SAO’s domain entities are not automatically linked to the BFO entities from the merged ontology. Lastly, particular is the top-level entity of DOLCE-Lite from the merged ontology.

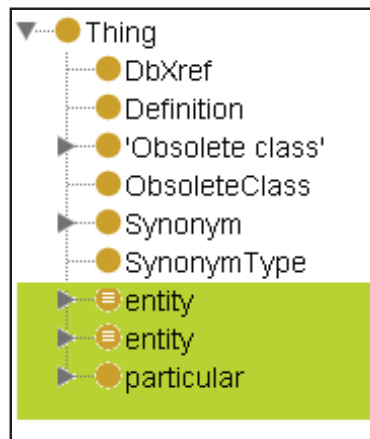


Figure 6.1.: The three main entities in the new ontology.

4. **Move entities from entity to entity:** Move each domain entity from SAO's entity to BFO's entity according to the method's step 4. Continue until all the domain entities are moved to BFO's entity, ensuring that no domain entities are subsumed by SAO's entity.
5. **Move entities from entity to particular:** Move each domain entity from BFO's entity to DOLCE's particular according to the method's step 5. Continue until all the domain entities are moved to DOLCE's particular, ensuring that no domain entities are subsumed by BFO's entity.
6. **Repeat:** We repeat steps 3 to 5 for object properties in the ontology. None of SAO's object properties were able to be contained in DOLCE.
7. **Delete SAO's entity and BFO's entity:** Delete SAO's entity and sub-entities and BFO's entity and sub-entities. We now have two main entities left: DOLCE's particular, which contains DOLCE's classes and SAO's domain classes, and a main entity for DOLCE's object properties.

Using the method introduced in Section 4.4 and merged ontologies found in ROMULUS, we have successfully converted the SAO ontology from BFO to DOLCE-Lite. The resulting ontology, SAO-DOLCE-Lite.owl can be accessed at ROMULUS's download page¹. In this case of foundational ontology interchangeability, all the domain entities from SAO had been successfully interchanged from the BFO to the DOLCE-Lite ontology. There are 788 domain entities in SAO-DOLCE-Lite.owl, in total. However, none of the object properties were equivalent to those in DOLCE-Lite, therefore none of the 36 object properties were able to be represented in DOLCE-Lite. While BFO and DOLCE-Lite have different philosophies, the proposed method allowed a conversion between the two, which provides the user with greater freedom in the use of foundational ontologies. While there are entities in the ontologies that are not related by equivalence, the method allows the user to relate the entities based on subsumption. This will be further reinforced in the future where subsumption relations will be explicitly included in the ontology mappings.

6.2. Evaluating ontological alignments by users

In this evaluation, participants were given class alignments between the three foundational ontologies to assess the accuracy of them. A screenshot of a single evaluation alignment is shown in Fig. 6.2. In each alignment survey, there is an open-ended question asking for general comments and suggestions. Every alignment in each survey has the following answering options: Agree, Partially Agree, Disagree, Unsure, and Skip, and an annotation from the entity's respective foundational ontology. Section 3.2 provides more detail about the meaning of each option. The participants for this evaluation were members of the Digital Enterprise Research Institute (DERI). DERI is leading research institute aimed at enabling networked knowledge using Semantic Web technologies. The participants were given access to the ontological alignments in the form of a web-based survey, with a time-limit of two weeks to complete the evaluation.

Alignment 2	Persistent	endurant
Annotation and example of concept	Persistants are GFO's way to capture identity over time. GFO pursues an approach which accounts for persistence by means of a suitable universal whose instances are presentials. Such universals are called persistants. These do not change and they can be used to explain how presentials which have different properties at different times can nevertheless be the same.	The main characteristic of endurants is that all of them are independent essential wholes. Endurants can 'genuinely' change in time, in the sense that the very same endurant as a whole can have incompatible properties at different times. To see this, suppose that an endurant - say 'this paper' - has a property at a time t 'it's white', and a different, incompatible property at time t' 'it's yellow': in both cases we refer to the whole object, without picking up any particular part of it.
Your choice	☐ Agree ☐ Partially Agree ☐ Disagree ☐ I don't know ☒ Skip	

Figure 6.2.: Evaluating an ontological alignment.

Each ontological alignment set received a different number of responses: DOLCE and BFO had 18 responses, BFO and GFO had 10 responses and GFO and DOLCE had 13 responses. For each alignment set, most responses were for the Agree option. On average, 44.9% of all responses were for the Agree option. A large portion of responses were for the Partially agree option. Thereafter, 11.0% and 17.7% of responses were for the Unsure and Skip options respectively. The smallest portion of responses, 7.1% were from the Disagree option. Table 6.1 provides a summary of the responses received for each option. As indicated in the table provided, the highest percentage of responses was for the Agree option. For what they agreed upon, in most cases participants agreed on the same alignments. An alignment that many participants agreed on is the equivalence of DOLCE:spatio-temporal-region and BFO:SpatioTemporalRegion. In most cases, the Agree option received few or no responses when ontology entity annotations were not clearly defined. Fig. 6.3 displays two alignments that received less than 2 responses for Agree due to the fact that some entity annotations were not clearly defined. The few Disagree options were for different alignments. An alignment that received some Disagree responses is the equivalence of DOLCE:perdurant and GFO:Occurrent. Likewise, participants were not united in their Unsure and Skip responses. Most of the general comments received indicated that the annotations from the foundational ontologies were difficult to understand, not properly defined and missing in some cases. Perhaps if the annotations were better defined, the number of Agree options would have increased.

Table 6.1.: A comparison of alignment evaluation responses.

	DOLCE and BFO	BFO and GFO	GFO and DOLCE	Average
No. of responses	18	10	13	13.7
Agree	49.4%	47.1%	38.1%	44.9%
Partially agree	21.7%	20.0%	16.3%	19.3%
Disagree	7.8%	6.4%	7.1%	7.1%
Unsure	8.3%	9.3%	15.4%	11.0%
Skip	12.8%	17.1%	23.1%	17.7%

	Entity from BFO	Entity from GFO
Alignment 3	IndependentContinuant	Presential
Annotation and example of concept	A continuant is an entity [bfo:Entity] that exists in full at any time in which it exists at all, persists through time while maintaining its identity and has no temporal parts. An independent continuant is a continuant [snap:Continuant] that is a bearer of quality [snap:Quality] and realizable entity [snap:RealizableEntity] entities, in which other entities inhere and which itself cannot inhere in anything. Examples: an organism, a heart, a leg, a person, a symphony orchestra, a chair, the bottom right portion of a human torso, the lawn and atmosphere in front of our building.	A presential exists wholly at exactly one time boundary.
Your choice	☐ Agree ☐ Partially Agree ☐ Disagree ☐ I don't know ☒ Skip	
Alignment 4	DependentContinuant	Dependent
Annotation and example of concept	Definition: A continuant [snap:Continuant] that is either dependent on one or other independent continuant [snap:IndependentContinuant] bearers or inheres in or is borne by other entities.	Dependent entities.
Your choice	☐ Agree ☐ Partially Agree ☐ Disagree ☐ I don't know ☒ Skip	

Figure 6.3.: Entity annotations that are not clearly defined.

6.3. Evaluating functionality by comparison with other ontology repositories

For the third evaluation of ROMULUS, we performed a side-by-side comparison with other existing ontology repositories. We have selected OOR [5], BioPortal [89], TONES [2], COLORE [26] and Ontohub [86] repositories to compare it with, seeing that they share some common functionality. We will compare the repositories in terms of the following functionality criteria: browse, mediation, search, metadata, ontology selection, ontology view, ontology comparison, and ontology access.

In terms of repository vision, ROMULUS is a repository of foundational ontologies. Users are not able to upload their own ontologies or data on ROMULUS, but they are encouraged to

download the ontologies and data in the repository. BioPortal, OOR and Ontohub are an open repositories where users are encouraged to upload their ontology projects, contributions, and download resources. BioPortal is specifically a repository of biomedical ontologies. TONES is aimed at being a central location for ontologies that will be helpful for application developers for testing purposes. It is closed, users are only allowed to download the ontologies and view some metadata. COLORE aims to be an open repository of first-order ontologies to aid in ontology evaluation and integration techniques, and to support the design, evaluation, and application of ontologies in first-order logic.

Table 6.2 displays a comparison of their functionality. From this comparison of functionality, we observe that ROMULUS provides advanced functionality in most of the criteria used in this evaluation. Instances where ROMULUS was found lacking such as an advanced search function will be considered for future works.

6.4. Summary of evaluation

The three different evaluations conducted to assess ROMULUS's functionality, achieved the following results:

- **Evaluating foundational ontology interchangeability:** The successful use of the method proposed in Section 4.4 and applied in Section 6.1 to convert a domain ontology from BFO to DOLCE demonstrates that the use of the method is both feasible and simplifies the process of foundational ontology interchangeability.
- **Evaluating ontological alignments by users:** As discussed above in Section 6.2, the users agreed with over 40% of the alignments. However real disagreement was less than 10% due to 'Unsure' and 'Skip' responses.
- **Evaluating functionality by comparison with other repositories:** From this evaluation in Section 6.3, we conclude that when compared to existing ontology repositories, ROMULUS does provide advanced functionality.

Table 6.2.: Comparison of ROMULUS's features with those of other repositories.

	ROMULUS	BioPortal and OOR	TONES	COLORE	Ontohub
Browse	Uses WebProtégé. Hierarchical ontology view. Displays classes, object properties, individuals and annotations of an ontology. The initial load takes a few seconds but thereafter, browsing is immediate without delay.	Hierarchical ontology view. Displays classes and class local neighbourhood. Has advanced visualization support. Browsing, overall is slow as it takes a few seconds to load subclasses and expand the hierarchy.	Browsing is currently unavailable.	No support.	Sequential ontology view. Lists of classes, properties, individuals and sentences.
Mediation	Alignments and mappings between its ontologies, merged and higher-level foundational ontologies, mapping inconsistencies and a method for foundational ontology interchangeability.	Some mappings between ontologies. Users may specify mappings.	No support.	No support.	No support.
Metadata	Metadata lists for ontologies based on OMV, OM ² R, and own model.	Metadata lists for ontologies based on OMV, but some missing metadata.	Metadata for metrics of each ontology.	Metadata exists in the ontology files.	Metadata lists for ontologies, but some missing metadata.
Ontology selection	ONSET, a foundational ontology selection tool, available to download and execute locally.	An ontology recommender system that allows one to calculate which ontologies are most relevant for a corpus.	No support.	No support.	No support.
Search	Allows a user to search for alignments, mappings and metadata.	Advanced search. Allows a user to search for entity names, ids, synonyms, properties and filter search.	No support.	No support.	Search for ontology name and ontology symbol.
Ontology view	Views in description logics and natural language.	No support.	No support.	Views in common logic.	No support.
Comparison	Ontology comparison in terms of different categories of criteria.	No support.	No support.	No support.	No support.
Ontology access	Users can view and download ontologies and tools.	Users can upload, edit own ontologies and download ontologies.	Users can download ontologies.	Users can view and download ontologies.	Users can upload, edit own ontologies and download ontologies.

6.5. Discussion

It is unlikely that ontology developers will commit to using a single unified foundational ontology for ontology development because different foundational ontologies, with sometimes conflicting philosophical distinctions and commitments exist (descriptive vs. realist, multiplicative vs. reductionist). These philosophical distinctions affect the way that entities are modelled, to a certain extent. Descriptive ontologies capture concepts based on human common-sense and understanding. Realist ontologies capture the logical world of what can be seen, and excludes cognitive aspects such as phenomena, belief etc. As such, the former allows for abstract entities, while the latter does not. The distinction between possibilism and actualism affects the ontologies in the same way. Rather than trying to enforce a worldwide ontology, it is achievable to enable interoperability among the existing foundational ontologies by performing ontology mediation. While the differences in philosophical choices affect some processes of ontology mediation, in most cases it is possible to align entities independently of the foundational ontologies' philosophies. Concerning the current alignments, we have decided to ignore certain aspects of the underlying philosophies of each foundational ontology, because else it would result in few or no alignments. For instance, an OWL file is agnostic about whether, e.g., an OWL class really refers to a universal in reality or not and some differences in the descriptions of the entities are not reflected in the OWL file due to language feature limitations.

Given the size of the ontologies and our high tolerance by ignoring underlying philosophies, the amount of alignments and the amount of mappings is less than one may have expected; or: once investigated in detail, the foundational ontologies are, at present, not particularly interchangeable even at the logical level. Only six pairwise mappings exist, i.e., they being, essentially, equivalent throughout all three examined foundational ontologies.

On a positive note, the systematised list of issues now can be taken up by ontologists. While some of the inconsistencies found are quite elaborate, others should be easier to resolve both ontologically (philosophically) and where in the ontology the entity is positioned; e.g., the notion of a mathematical Set is fairly well investigated already but appears in different positions in DOLCE and GFO ontologies. As such, the results presented here provide a solid foundation for ample ontological investigations. Regardless of the issues, we now know that there are some mappings resulting in some foundational ontology interoperability.

Based on these mappings and interoperability, ROMULUS, a repository of foundational ontologies was created with the vision of assisting with semantic operations such as foundational ontology interchangeability. It contains foundational ontologies with different ontological commitments and functionality. The WFOL was envisioned to be a library of related foundational ontologies which Semantic Web applications can commit to, reflecting different commitments and purposes. The vision behind ROMULUS is similar to that of the WFOL. We now explore the goals and requirements of both systems.

The first main goal of the WFOL is for it to serve as a starting point for building new ontologies by providing a high-level view of entities that are to be modelled. The second main goal of the WFOL is for it to be a reference point for easy and rigorous comparisons among differ-

ent ontological approaches. Lastly, the third main goal of the WFOL is for it to be a common framework for analyzing, harmonizing and integrating existing ontologies and metadata standards. ROMULUS meets the first main goal of the WFOL by providing a higher-level ontology, FFO, containing only the common entities of DOLCE, BFO and GFO ontologies. The FFO serves as a starting point for modelling entities in ontology development. Secondly, ROMULUS meets the goal of being a reference point for providing comparisons between different ontological approaches in the form of its online multi-dimensional criteria comparison of selected foundational ontologies. Lastly, ROMULUS is a shared framework for analyzing, harmonizing and integrating existing ontologies and metadata standards thanks to its multi-dimensional criteria comparison of selected foundational ontologies, alignments, mapping ontologies, merged ontologies, and extensive metadata for each ontology.

The three requirements for the WFOL is for it to be minimal, rigorous and extensively researched. ROMULUS is minimal in that it only contains foundational ontologies and related modules, and an even higher-level minimal ontology, FFO. Secondly, ROMULUS meets the requirement of being rigorous by containing the OWL versions of each foundational ontology which is characterized by means of rich axiomatisations. Furthermore, human-readable views of these axiomatisations are available in ROMULUS to aid in foundational ontology usage. For the last requirement, ROMULUS is extensively researched because its foundational ontologies were selected after an extensive research and in-depth comparison into widely used and maintained foundational ontologies. In addition to these WFOL requirements, ROMULUS meets the functional requirements from Section 5.1.1: modular foundational ontologies, mediated foundational ontologies, online browsing library, extensive foundational ontology comparisons, ontology views, multi-dimensional metadata criteria, and access to download ontologies and related resources.

By comparing the philosophy, goals and requirements of ROMULUS to WFOL, we realise that their philosophies are the same and that ROMULUS meets all goals and requirements of the envisioned WFOL and meets other important requirements. ROMULUS is clearly useful in enabling semantic interoperability by providing infrastructure to assist with foundational ontology interchangeability.

Conclusions and future work

The problem of semantic interoperability when using different foundational ontologies has been successfully solved in ROMULUS. We have realised a solution to overcome the issues posed in foundational ontology interchangeability. ROMULUS offers the ontology developer a practical solution of having the freedom to use a preferred foundational ontology in development and acquiring semantic interoperability by linking their ontology to heterogenous systems. In order for ROMULUS to be applied to diverse Semantic Web applications, we have selected foundational ontologies with different philosophies and ontological commitments that have been applied to a wide range of subject domains and applications.

ROMULUS provides infrastructure: ontological alignments, mappings, merged ontologies, a higher-level ontology, and a method for foundational ontology interchangeability to aid with interoperability. The ontological alignments, which form the basis to perform mapping, have been identified after careful consideration by using existing tools and documentation, and manually. The mappings and merged ontologies, which may be used together with the method for foundational ontology interchangeability, assists a user in converting between foundational ontologies and linking different domain ontologies that use different foundational ontologies. The method for foundational ontology interchangeability has been evaluated, by converting a domain ontology from BFO to DOLCE, with the resulting ontology containing all the original domain classes, demonstrating that the proposed method does in fact assist a user in foundational ontology interchangeability. The higher-level ontology, FFO, a single ontology containing the most general concepts from the foundational ontologies in ROMULUS, promotes interoperability. Furthermore, ROMULUS has a number of other features aimed at promoting foundational ontology usage. The metadata criteria for each ontology in ROMULUS enables ontology reuse. ROMULUS has human-readable views for each foundational ontology which aims at enabling better understanding and ease of use of foundational ontologies. The online browsing feature in ROMULUS means that users need not install or execute any software to browse through foundational ontologies. Ontology comparison in ROMULUS provides the user with an extensive, structured comparison of foundational ontologies in the form of tables and lists. Ontology selection allows a user to automatically select a foundational ontology, based on their requirements.

To the best of our knowledge, ROMULUS is the first online repository of machine-processable, modularised, aligned, and merged foundational ontologies, with the goal of facilitating semantic interoperability by allowing for foundational ontology interchangeability.

7.1. Future Work

Our significant achievement of providing a foundational ontology repository meeting all functional requirements and successfully solving the issue of semantic interoperability when using different foundational ontologies, opens a plethora of avenues for further research and investigation questions to expand on this work.

Including other foundational ontologies in ROMULUS will improve foundational ontology interchangeability and usage by improving the versatility of the repository and providing the ontology developers with a greater variety of foundational ontologies to interchange between. If the method for performing foundational ontology interchangeability was automated, it would be easier for the user to perform interchangeability and enhance semantic interoperability and foundational ontology usage. We would like to provide infrastructure to allow users to upload ontologies and create alignments that the community could evaluate. At present, only a few subsumption relations have been used for alignment. Extending this to a complete set of subsumption relation alignments will improve foundational ontology interchangeability. Rather than limiting the ontologies to that of OWL language, we feel the need to include ontologies in other languages.

The ontology metadata in ROMULUS, can be improved by including the OWL formalism of OMV and OM²R in each foundational ontology in ROMULUS. The ontology view can be improved to include tables comparing OWL axioms to ACE concepts [38], generated by the OWL verbaliser tool. ROMULUS's current search function allows a user to perform filtered searches through the alignments and metadata. It would be useful to expand this search throughout the repository. There is some difficulty in maintaining ROMULUS with regard to its ontologies. Many of the ontologies in ROMULUS are related (the modules, mapping and merged ontologies). As such, when changes occur to one of the ontologies, each related module is affected e.g., changing an alignment in the DOLCE-BFO mapping ontology may have an effect on the FunctionalParticipation-BFO mapping, the SpatialRelation-BFO mapping and the TemporalRelation-BFO mapping ontologies. It is rather time-consuming and prone to error to change each of these files and ensure that the entities and axioms are consistent throughout. An ontology library or module management system would greatly assist with this task.

The evaluation of alignments discussed previously in Section 6.2 revealed that the participants agreed with over 40% of the alignments. It is useful to conduct an in-depth investigation into the motivations for the participant's choices to reveal reasons for disagreement and uncertainty, and thereby further improve existing alignments.

7.2. Summary of contributions

The new types of foundational ontology modules presented herein were created to aid developers in performing specific functions, where usage of an entire foundational ontology is not required. A content comparison between DOLCE, BFO and GFO was performed. Ontology mediation performed for the foundational ontologies resulted in ontological alignments, mappings, merged ontologies, a higher level ontology, and a method to perform foundational ontology interchangeability. The ontological alignments form the basis for creating mappings. From the alignment process, we have identified and provided both accurate and approximate alignments. From the mapping process, we have identified ontological inconsistencies for a number of entities which may be useful to foundational ontology developers to improve their foundational ontologies to achieve a higher level of interoperability. We provide fixes to some of these ontological inconsistencies. The mapping and merged ontologies, which may be used together with the method for performing foundational ontology interchangeability, are useful in cases where one wishes to convert between the three foundational ontologies and to achieve interoperability between a heterogeneous system: one may link a particular ontology using a foundational ontology to an ontology using a different foundational ontology. Interoperability may also be achieved if the higher-level ontology is used as it solely encompasses entities and relational properties from all three foundational ontologies. We have created metadata lists for each foundational ontology in the repository with the hope of enabling ontology reuse. Mapping and merged ontologies and foundational ontology modules have additional metadata criteria. The ontology selection that is integrated into ROMULUS is aimed at assisting the user with performing ontology selection, and further assisting by providing links to features of ROMULUS that meet a user's requirement.

Bibliography

- [1] Protégé. <http://protege.stanford.edu/>. Accessed on 07/05/2012. (Cited on page 19.)
- [2] The TONES ontology repository. <http://owl.cs.manchester.ac.uk/repository/browser>. Accessed on 22/12/2012. (Cited on pages 16 and 88.)
- [3] AHMAD, F. AND LINDGREN, H. 2010. Selection of foundational ontology for collaborative knowledge modeling in healthcare domain. In *14th international conference on Artificial intelligence: methodology, systems, and applications*. AIMSAS'10. Springer-Verlag, Berlin, Heidelberg, 261–262. (Cited on pages 2 and 12.)
- [4] ANTONIOU, G., , ANTONIOU, G., ANTONIOU, G., HARMELEN, F. V., AND HARMELEN, F. V. 2003. Web ontology language: OWL. In *Handbook on Ontologies in Information Systems*. Springer, 67–92. (Cited on page 12.)
- [5] BACLAWSKI, K. AND SCHNEIDER, T. 2009. The open ontology repository initiative: Requirements and research challenges. In *Proceedings of the Workshop on Collaborative Construction, Management and Linking of Structured Knowledge (CK2009), collocated with the 8th International Semantic Web Conference (ISWC-2009)*. CEUR Workshop Proceedings, vol. 514. CEUR-WS.org. Washington DC, USA, 25 October, 2009. (Cited on page 88.)
- [6] BEISSWANGER, E., SCHULZ, S., STENZHORN, H., AND HAHN, U. 2008. BioTop: An upper domain ontology for the life sciences - a description of its current structure, contents, and interfaces to OBO ontologies. *Applied Ontology* 3, 4, 205–212. (Cited on page 2.)
- [7] BORGIO, S. 2011. Goals of modularity: A voice from the foundational viewpoint. In *WoMO 2011 - Modular Ontologies. Proceedings of the Fifth International Workshop*. Frontiers in Artificial Intelligence and Applications, vol. 230. IOS Press, Ljubljana, Slovenia, 1–6. (Cited on page 18.)
- [8] BURHANS, D. T., CAMPBELL, A. E. R., AND SKUSE, G. R. 2003. Exploring the role of knowledge representation and reasoning in biomedical text understanding. In *SIGIR Workshop on Text Analysis and Search for Bioinformatics*. ACM, Toronto, Canada. (Cited on page 1.)
- [9] CASTANO, S., FERRARA, A., AND MONTANELLI, S. 2003. H-MATCH: an algorithm for dynamically matching ontologies in peer-based systems. In *The first International Workshop*

- on *Semantic Web and Databases (SWDB'03)*. Humboldt-Universitt, Berlin, Germany, 231–250. (Cited on page 22.)
- [10] CEUSTERS, W. AND SMITH, B. 2010a. Foundations for a realist ontology of mental disease. *Journal of biomedical semantics* 1, 1, 10. (Cited on page 1.)
- [11] CEUSTERS, W. AND SMITH, B. 2010b. Malaria diagnosis and the plasmodium life cycle: the BFO perspective. *Nature Precedings* 3, 25–34. (Cited on page 1.)
- [12] CUENCA GRAU, B., HORROCKS, I., KAZAKOV, Y., AND SATTTLER, U. 2008. Modular reuse of ontologies: Theory and practice. *Journal of Artificial Intelligence Research (JAIR)* 31, 273–318. (Cited on page 19.)
- [13] DANG, T. T., GABRIEL, A., HERTLING, S., ROSKOSCH, P., WLOTZKA, M., ZILKE, J. R., JANSSEN, F., AND PAULHEIM, H. 2012. HotMatch results for OEAI 2012. In *Seventh International Workshop on Ontology Matching (OM'12)*. CEUR-WS, vol. 946. (Cited on page 21.)
- [14] D'AQUIN, M., SABOU, M., AND MOTTA, E. 2006. Modularization: a Key for the Dynamic Selection of Relevant Knowledge Components. In *1st International Workshop on Modular Ontologies (WoMO'06), co-located with the International Semantic Web Conference (ISWC'06)*. November 5, Athens, Georgia. (Cited on page 20.)
- [15] D'AQUIN, M., SCHLICHT, A., STUCKENSCHMIDT, H., AND SABOU, M. 2009. Criteria and evaluation for ontology modularization techniques. In *Modular Ontologies*. Lecture Notes in Computer Science. Springer, 67–89. (Cited on pages 18 and 20.)
- [16] DE BRUIJN, J., EHRIG, M., FEIER, C., MARTÍNS-RECUERDA, F., SCHARFFE, F., AND WEITEN, M. 2006. Ontology mediation, merging, and aligning. In *Semantic Web Technologies*. Wiley Online Library, 1–20. (Cited on pages 21 and 34.)
- [17] DEAN, M. AND YIM, P. SOCoP. <http://socop.oor.net>. Accessed on 20/05/2013. (Cited on page 16.)
- [18] DECAENE, D. jowl 1.0. <http://jowl.ontologyonline.org/>. Accessed on 10/10/2012. (Cited on page 23.)
- [19] ERMOLAYEV, V., KEBERLE, N., AND MATZKE, W.-E. 2008. An upper level ontological model for engineering design performance domain. In *27th International Conference on Conceptual Modeling (ER'08)*. Lecture Notes in Computer Science, vol. 5231. Springer, 98–113. (Cited on page 2.)
- [20] EUZENAT, J. AND SHVAIKO, P. 2007. *Ontology Matching*. Springer. (Cited on pages 21, 34, and 35.)
- [21] FARRAR, S. AND LANGENDOEN, T. 2003. A linguistic ontology for the semantic web. *Glott International* 7, 3, 97–100. (Cited on page 9.)
- [22] GANGEMI, A., FISSEHA, F., KEIZER, J., LEHMANN, J., LIANG, A., PETTMAN, I., SINI, M., AND TACONET, M. 2004. A core ontology of fishery and its use in the fishery ontology

- service project. In *Proceedings of the EKAW*04 Workshop on Core Ontologies in Ontology Engineering*. CEUR Workshop Proceedings, vol. 118. CEUR-WS.org. Northamptonshire (UK), October 8, 2004. (Cited on page 1.)
- [23] GIUNCHIGLIA, F., SHVAIKO, P., AND YATSKEVICH, M. 2004. S-Match: an algorithm and an implementation of semantic matching. In *The Semantic Web: Research and Applications, First European Semantic Web Symposium, (ESWS'04)*. Lecture Notes in Computer Science, vol. 04391. Springer, 61–75. Heraklion, Crete, Greece, May 10-12, 2004,. (Cited on page 21.)
- [24] GRENON, P. 2003. BFO in a nutshell: A bi-categorical axiomatization of BFO and comparison with DOLCE. IFOMIS Report 06/2003, Institute for Formal Ontology and Medical Information Science (IFOMIS), University of Leipzig, Leipzig, Germany. (Cited on page 12.)
- [25] GRENON, P., SMITH, B., AND GOLDBERG, L. 2004. Biodynamic ontology: Applying BFO in the biomedical domain. *Ontologies in medicine 102*, 20. (Cited on page 1.)
- [26] GRUNINGER, M. COLORE. <http://code.google.com/p/colore/>. Accessed on 21/05/2013. (Cited on pages 17 and 88.)
- [27] GRUNINGER, M. 2009. COLORE: Common Logic Ontology Repository. http://ontolog.cim3.net/file/work/OOR-Ontolog-Panel/2009-08-06_Ontology-Repository-Research-Issues/Colore--MichaelGruninger_20090806.pdf. Slides. (Cited on page 18.)
- [28] GUIZZARDI, G. AND WAGNER, G. 2004. Towards ontological foundations for agent modelling concepts using the unified foundational ontology (UFO). In *6th International Bi-Conference Workshop on Agent-Oriented Information Systems II (AOIS'04)*. Lecture Notes in Computer Science. Springer, 110–124. USA, New York, July 20, 2004. (Cited on page 1.)
- [29] HARTMANN, J., SURE, Y., HAASE, P., PALMA, R., AND DEL CARMEN SUÁREZ-FIGUEROA, M. 2005. OMV - Ontology Metadata Vocabulary. In *Ontology Patterns for the Semantic Web (OPSW) workshop collocated with the 4th International Semantic Web Conference (ISWC-2005)*. Galway, Ireland, November, 2005. (Cited on pages 23 and 24.)
- [30] HERRE, H. 2010. General Formal Ontology (GFO): A foundational ontology for conceptual modelling. In *Theory and Applications of Ontology: Computer Applications*. Springer, Heidelberg, Chapter 14, 297–345. (Cited on pages xi, 1, 38, 39, and 133.)
- [31] HERTLING, S. 2012. Hertuda results for OEAI 2012. In *Seventh International Workshop on Ontology Matching (OM'12)*. CEUR-WS, vol. 946. (Cited on page 22.)
- [32] HITZLER, P., KRÖTZSCH, M., PARSIA, B., PATEL-SCHNEIDER, P. F., AND RUDOLPH, S., Eds. 27 October 2009. *OWL 2 Web Ontology Language: Primer*. W3C Recommendation. Available at <http://www.w3.org/TR/owl2-primer/>. (Cited on page 12.)
- [33] HOEHNDORF, R., NGONGA NGOMO, A.-C., AND HERRE, H. 2009. Developing consistent and modular software models with ontologies. In *8th International Conference on Software Methodologies, Tools and Techniques (SoMeT'09)*. Frontiers in Artificial Intelligence and Applications, vol. 199. IOS Press, 399–412. (Cited on page 1.)

- [34] HOEHNDORF, R., PRFER, K., BACKHAUS, M., VISAGIE, J., AND KELSO, J. 2006. The design of a wiki-based curation system for the ontology of functions. In *Proceedings of the Joint BioLINK and 9th Bio-Ontologies Meeting (JBB 2006)*. Fortaleza, Brazil, Aug 5. (Cited on page 1.)
- [35] JANOWICZ, K. AND COMPTON, M. 2010. The Stimulus-Sensor-Observation Ontology Design Pattern and its Integration into the Semantic Sensor Network Ontology. In *3rd International Workshop on Semantic Sensor Networks*. CEUR Workshop Proceedings, vol. 668. CEUR-WS.org. November 7, Shanghai, China. (Cited on page 16.)
- [36] JIMÉNEZ-RUIZ, E. AND CUENCA GRAU, B. 2011. LogMap: Logic-based and scalable ontology matching. In *10th International Semantic Web Conference (ISWC'11)*. Lecture Notes in Computer Science, vol. 7031. Springer, 273–288. (Cited on page 21.)
- [37] JURETA, I. J., MYLOPOULOS, J., AND FAULKNER, S. 2009. A core ontology for requirements. *Applied Ontology* 4, 3-4 (Aug.), 169–244. (Cited on pages 2 and 12.)
- [38] KALJURAND, K. 2007. Attempto Controlled English as a Semantic Web Language. Ph.D. thesis, Faculty of Mathematics and Computer Science, University of Tartu. (Cited on pages 27 and 94.)
- [39] KALYANPUR, A., PARSIA, B., SIRIN, E., CUENCA GRAU, B., AND HENDLER, J. A. 2006. Swoop: A web ontology editing browser. *Journal of Web Semantics* 4, 2, 144–153. (Cited on pages 19 and 20.)
- [40] KEET, C. M. 2005. Factors affecting ontology development in ecology. In *Data Integration in the Life Sciences 2005 (DILS'05)*. Lecture Notes in Bioinformatics, vol. 3615. Springer Verlag, 46–62. San Diego, USA, 20-22 July 2005. (Cited on pages 2 and 13.)
- [41] KEET, C. M. 2010. Dependencies between ontology design parameters. *International Journal of Metadata, Semantics and Ontologies* 5, 4, 265–284. (Cited on page 1.)
- [42] KEET, C. M. 2011. The use of foundational ontologies in ontology development: an empirical assessment. In *8th Extended Semantic Web Conference (ESWC'11)*. Lecture Notes in Computer Science, vol. 6643. Springer, 321–335. Heraklion, Crete, Greece, 29 May-2 June, 2011. (Cited on page 1.)
- [43] KEET, C. M. 2012a. Detecting and revising flaws in OWL object property expressions. In *18th International Conference on Knowledge Engineering and Knowledge Management (EKAW'12)*, A. ten Teije et al., Ed. Lecture Notes in Artificial Intelligence, vol. 7603. Springer, 252–266. Oct 8-12, Galway, Ireland. (Cited on page 30.)
- [44] KEET, C. M. 2012b. Transforming semi-structured life science diagrams into meaningful domain ontologies with DiDOn. *Journal of Biomedical Informatics*, (in press). (Cited on page 2.)
- [45] KHAN, Z. AND KEET, C. M. 2012. ONSET: Automated foundational ontology selection and explanation. In *18th International Conference on Knowledge Engineering and Knowledge Management (EKAW'12)*, A. ten Teije et al., Ed. Lecture Notes in Artificial Intelligence, vol.

7603. Springer, 237–251. Oct 8-12, Galway, Ireland. Status: published (after peer-review). Acceptance rate: 15% for long papers. (Cited on pages 12, 29, 75, and 77.)
- [46] KOLLI, R. AND DOSHI, P. 2008. OPTIMA: tool for ontology alignment with application to semantic reconciliation of sensor metadata for publication in sensormap. In *Proc. of ICSC'08*. 484–485. (Cited on page 22.)
- [47] KONEV, B., LUTZ, C., WALTHER, D., AND WOLTER, F. 2009. Formal properties of modularisation. *Modular Ontologies 5445*, 25–66. (Cited on page 19.)
- [48] KOZAKI, K., KITAMURA, Y., IKEDA, M., AND MIZOGUCHI, R. 2002. Hozo: An environment for building/using ontologies based on a fundamental consideration of “role” and “relationship”. In *13th International Conference on Knowledge Engineering and Knowledge Management (EKAW'02)*. Lecture Notes in Computer Science, vol. 2473. Springer, 213–218. (Cited on page 10.)
- [49] LARSON, S. D., FONG, L. L., GUPTA, A., CONDIT, C., BUG, W. J., AND MARTONE, M. E. 2007. A formal ontology of subcellular neuroanatomy. *Front Neuroinformatics 1*, 3. (Cited on page 85.)
- [50] LINDBERG, D. A., HUMPHREYS, B. L., MCCRAY, A. T., ET AL. 1993. The Unified Medical Language System. *Methods of information in medicine 32*, 4, 281. (Cited on page 22.)
- [51] MALONE, J., HOLLOWAY, E., ADAMUSIAK, T., KAPUSHESKY, M., ZHENG, J., KOLESNIKOV, N., ZHUKOVA, A., BRAZMA, A., AND PARKINSON, H. E. 2010. Modeling sample variables with an experimental factor ontology. *Bioinformatics 26*, 8, 1112–1118. (Cited on page 28.)
- [52] MASCARDI, V., CORD, V., AND ROSSO, P. 2007. A comparison of upper ontologies. Technical Report DISI-TR-06-21, University of Genova, Italy. 55-64. (Cited on page 12.)
- [53] MASOLO, C., BORGIO, S., GANGEMI, A., GUARINO, N., AND OLTRAMARI, A. 2003. Ontology Library. WonderWeb Deliverable D18 (ver. 1.0, 31-12-2003). <http://wonderweb.semanticweb.org>. (Cited on pages 1, 2, 3, 7, 8, 13, and 64.)
- [54] MASUYA, H. AND MIZOGUCHI, R. 2009. Toward fully integration of mouse phenotype information. In *Proceedings of the Second Interdisciplinary Ontology Meeting*. 35–44. Tokyo, Japan, February 28 - March 1, 2009. (Cited on page 2.)
- [55] MCCOMB, D. 2011. 14 Ways to Use Your Enterprise Ontology. <http://www.irmac.ca/1011/14%20Ways%20to%20Use%20Your%20Enterprise%20Ontology%20final.pdf>. Slides. (Cited on page 11.)
- [56] MIZOGUCHI, R. 2010. YAMATO: yet another more advanced top-level ontology. In *Proceedings of the Sixth Australasian Ontology Workshop*. Conferences in Research and Practice in Information. 1–16. Sydney : ACS. (Cited on pages 2, 11, and 42.)
- [57] MIZOGUCHI, R. AND KITAMURA, Y. 2009. A functional ontology of artifacts. *The Monist 92*, 3, 387–402. (Cited on page 2.)

- [58] MIZOGUCHI, R., KOU, H., ZHOU, J., KOZAKI, K., IMAI, K., AND OHE, K. 2009. An advanced clinical ontology. In *Proceedings of International Conference on Biomedical Ontology (ICBO'09)*. 119–122. Buffalo, NY, June 24 - 26, 2009. (Cited on page 2.)
- [59] MORRISON, N., ASHBURNER, M., FIELD, D., LEWIS, S., MUNGALL, C., SCHRIML, L., AND SMITH, B. 2009. The environment ontology linking environmental data. In *Proceedings of European conference TOWARDS eENVIRONMENT*. Prague, Czech Republic, March 25-27, 2009. (Cited on page 1.)
- [60] MOSSAKOWSKI, T., KUTZ, O., AND LANGE, C. 2012. Three semantics for the core of the distributed ontology language. In *Seventh International Conference on Formal Ontology in Information Systems (FOIS'12)*. Frontiers in Artificial Intelligence and Applications, vol. 239. IOS Press, 337–352. Gray, Austria, July 24-27, 2012. (Cited on page 12.)
- [61] MUTHAIYAH, S. AND KERSCHBERG, L. 2008. A hybrid ontology mediation approach for the semantic web. *International Journal of EBusiness Research* 4, December, 79–91. (Cited on page 21.)
- [62] NEWMAN, D., BECHHOFFER, S., AND ROURE, D. D. 2009. myExperiment: An ontology for e-Research. In *Semantic Web Applications in Scientific Discourse*. October 25. (Cited on page 18.)
- [63] NGO, D. AND BELLAHSENE, Z. 2012. YAM++: A multi-strategy based approach for ontology matching task. In *18th International Conference on Knowledge Engineering and Knowledge Management (EKAW'12)*. LNAI, vol. 7603. Springer, 421–425. Oct 8-12, Galway, Ireland. Demo. (Cited on page 21.)
- [64] NILES, I. AND PEASE, A. 2001. Towards a standard upper ontology. In *Second International Conference on Formal Ontology in Information Systems (FOIS'01)*. IOS Press. Ogunquit, Maine, October 17-19, 2001. (Cited on pages 1 and 42.)
- [65] NILES, I. AND PEASE, A. 2003. *Linking Lexicons and Ontologies: Mapping WordNet to the Suggested Upper Merged Ontology*. CSREA Press, 412–416. (Cited on pages 1 and 9.)
- [66] NOY, N. F. AND MUSEN, M. A. 2000. PROMPT: Algorithm and tool for automated ontology merging and alignment. In *Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence (AAAI/IAAI)*. AAAI Press / The MIT Press, 450–455. (Cited on pages 20 and 22.)
- [67] OBERLE, D. 2006. *Semantic Management of Middleware*. Semantic Web and Beyond, vol. 1. Springer, New York. (Cited on page 12.)
- [68] PARENT, C. AND SPACCAPRIETA, S. 2009. An overview of modularity. In *Modular Ontologies*. LNCS, vol. 5445. Springer, Chapter 2, 5–23. (Cited on page 12.)
- [69] RECTOR, A. L. AND ROGERS, J. 2006. Ontological and Practical Issues in Using a Description Logic to Represent Medical Concept Systems: Experience from GALEN. In *Reasoning Web, Second International Summer School*. Lecture Notes in Computer Science, vol. 4126. Springer, 197–231. September 4-8, Lisbon, Portugal. (Cited on page 28.)

- [70] ROITMAN, H. AND GAL, A. 2006. OntoBuilder: Fully automatic extraction and consolidation of ontologies from web sources using sequence semantics. In *EDBT Workshops. Lecture Notes in Computer Science*, vol. 4254. Springer, 573–576. Munich, Germany, March 26-31, 2006. (Cited on page 21.)
- [71] SCHEFFCZYK, J., BAKER, C. F., AND NARAYANAN, S. 2008. Ontology-Based reasoning about lexical resources. In *Ontologies and Lexical Resources for Natural Language Processing*. Cambridge Studies in Natural Language Processing. Cambridge University Press, Cambridge, MA. (Cited on page 1.)
- [72] SCHNEIDER, L. 2003. Designing foundational ontologies: The object-centered high-level reference ontology OCHRE as a case study. In *22nd International Conference on Conceptual Modeling (ER'03)*. Lecture Notes in Computer Science, vol. 2813. Springer, 91–104. (Cited on page 3.)
- [73] SCHULZ, S., BOEKER, M., AND STENZHORN, H. 2008. How granularity issues concern biomedical ontology integration. *Studies In Health Technology And Informatics* 136, 863–868. (Cited on page 2.)
- [74] SEMY, S. K., PULVERMACHER, M. K., AND OBRST, L. J. 2004. Toward the use of an upper ontology for us government and us military domains: An evaluation. Technical Report MTR 04B0000063, The MITRE Corporation. (Cited on pages 2 and 12.)
- [75] SEYED, A. P. 2009. BFO/DOLCE primitive relation comparison. In *The 12th Annual Bio-Ontologies Meeting Colocated with Intelligent Systems for Molecular Biology (ISMB'09)*. Stockholm, Sweden, 28 June 2009. (Cited on pages 13, 38, and 39.)
- [76] SILVA, V. S., CAMPOS, M. L. M., SILVA, J. C. P., AND CAVALCANTI, M. C. 2011. An approach for the alignment of biomedical ontologies based on foundational ontologies. *Journal of Information and Data Management (JIDM)* 2, 3, 557–. (Cited on page 1.)
- [77] SIMON, J., DOS SANTOS, M. C., FIELDING, J. M., AND SMITH, B. 2006. Formal ontology for natural language processing and the integration of biomedical databases. *International Journal of Medical Informatics* 75, 3-4, 224–231. (Cited on page 1.)
- [78] SINHA, G. AND MARK, D. 2010. Toward a foundational ontology of the landscape. Extended abstracts of Geographic Information Science (GIScience), Zürich, Switzerland, September 14-17, 2010. (Cited on page 2.)
- [79] SMITH, B., ASHBURNER, M., ROSSE, C., BARD, J., BUG, W., CEUSTERS, W., GOLDBERG, L. J., EILBECK, K., IRELAND, A., MUNGALL, C. J., OBI CONSORTIUM, LEONTIS, N., ROCCA-SERRA, P., RUTTENBERG, A., SANSONE, S.-A. A., SCHEUERMANN, R. H., SHAH, N., WHETZEL, P. L., AND LEWIS, S. 2007. The OBO Foundry: Coordinated evolution of ontologies to support biomedical data integration. *Nature Biotechnology* 25, 11, 1251–1255. (Cited on pages 1 and 2.)
- [80] SMITH, B., CEUSTERS, W., KLAGGES, B., KOHLER, J., KUMAR, A., LOMAX, J., MUNGALL, C., NEUHAUS, F., RECTOR, A., AND ROSSE, C. 2005. Relations in biomedical ontologies. *Genome Biology* 6, 5, 46. (Cited on page 36.)

- [81] STUCKENSCHMIDT, H. AND SCHLICHT, A. 2009. Structure-based partitioning of large ontologies. In *Modular Ontologies: Concepts, Theories and Techniques for Knowledge Modularization*. Lecture Notes in Computer Science, vol. 5445. Springer, 187–210. (Cited on page 20.)
- [82] TEMAL, L., ROSIER, A., DAMERON, O., AND BURGUN, A. 2010. Mapping BFO and DOLCE. *Studies in Health Technology and Informatics 160*, Pt 2, 1065–9. (Cited on pages xi, 13, 38, 39, and 134.)
- [83] THIRD, A., WILLIAMS, S., AND POWER, R. 2011. OWL to English : a tool for generating organised easily-navigated hypertexts from ontologies. 10th International Semantic Web Conference (ISWC 2011), 23 - 27 Oct 2011, Bonn, Germany. (Cited on pages 27 and 29.)
- [84] THOMAS, H., BRENNAN, R., AND O’SULLIVAN, D. 2012. Using the OM2R meta-data model for ontology mapping reuse for the ontology alignment challenge - a case study. In *Seventh International Workshop on Ontology Matching (OM’12) collocated with the 11th International Semantic Web Conference (ISWC’12)*. CEUR Workshop Proceedings, vol. 946. CEUR-WS.org. (Cited on pages 23 and 25.)
- [85] TUDORACHE, T., VENDETTI, J., AND NOY, N. F. 2008. Web-Protege: A lightweight OWL ontology editor for the web. In *Fifth OWLED Workshop on OWL: Experiences and Directions, collocated with the 7th International Semantic Web Conference (ISWC-2008)*. CEUR Workshop Proceedings, vol. 432. CEUR-WS.org. (Cited on page 29.)
- [86] VALE, D. C., CLAUSEN, C., KOHORST, T., AND BECKER, I. Ontohub. <http://ontohub.informatik.uni-bremen.de/>. Accessed on 30/07/2012. (Cited on pages 17 and 88.)
- [87] VARZI, A. 2012. Mereology. In *The Stanford Encyclopedia of Philosophy*, Winter 2012 ed., E. N. Zalta, Ed. (Cited on page 7.)
- [88] WEIBEL, S., KUNZE, J., LAGOZE, C., AND WOLF, M. 1998. Dublin core metadata for resource discovery. *Internet Engineering Task Force RFC 2413*, 222. (Cited on page 23.)
- [89] WHETZEL, P. L., NOY, N. F., SHAH, N. H., ALEXANDER, P. R., NYULAS, C., TUDORACHE, T., AND MUSEN, M. A. 2011. Biportal: enhanced functionality via new web services from the national center for biomedical ontology to access and use ontologies in software applications. *Nucleic Acids Research 39*, Web-Server-Issue, 541–545. (Cited on page 88.)

Complete set of accurate alignments

The alignments in bold font are those that result in successful mappings.

	Class from BFO	Alignment relation	Class from GFO
1.	Entity	equivalence	Entity
2.	IndependentContinuant	equivalence	Persistent
3.	DependentContinuant	equivalence	Dependent
4.	MaterialEntity	equivalence	Material_persistent
5.	Object	equivalence	Material_object
6.	ObjectBoundary	equivalence	Material_boundary
7.	Function	equivalence	Function
8.	Role	equivalence	Role
9.	Occurrent	equivalence	Occurrent
10.	Process	equivalence	Process
11.	SpatialRegion	equivalence	Spatial_region
12.	TemporalRegion	equivalence	Temporal_region
13.	Quality	equivalence	Property

Figure A.1.: Alignments between BFO and GFO ontologies.

	Class from BFO	Alignment relation	Class from GFO-Basic
1.	Entity	equivalence	Entity
2.	IndependentContinuant	equivalence	Persistent
3.	Object	equivalence	Material_object
4.	ObjectBoundary	equivalence	Material_boundary
5.	Role	equivalence	Role
6.	Occurrent	equivalence	Occurrent
7.	Process	equivalence	Process
8.	SpatialRegion	equivalence	Spatial_region
9.	TemporalRegion	equivalence	Temporal_region
10.	Quality	equivalence	Property

Figure A.2.: Alignments between BFO and GFOBasic ontologies.

	Class from DOLCE-Lite	Alignment relation	Class from BFORO
1.	endurant	equivalence	IndependentContinuant
2.	physical-endurant	equivalence	materialEntity
3.	physical-object	equivalence	Object
4.	perdurant	equivalence	Occurrent
5.	process	equivalence	Process
6.	quality	equivalence	Quality
7.	spatio-temporal-region	equivalence	SpatioTemporalRegion
8.	temporal-region	equivalence	TemporalRegion
9.	space-region	equivalence	SpatialRegion
	Object Property from DOLCE-Lite	Alignment relation	Object Property from BFORO
1.	part	equivalence	has_part
2.	part-of	equivalence	part_of
3.	proper-part	equivalence	has_proper_part
4.	proper-part-of	equivalence	proper_part_of
5.	participant	equivalence	has_participant
6.	participant-in	equivalence	participates_in
7.	generic-location	equivalence	located_in
8.	generic-location-of	equivalence	location-of

Figure A.3.: Alignments between DOLCE-Lite and BFORO ontologies.

	Class from FunctionalParticipation	Alignment relation	Class from BFORO
1.	endurant	equivalence	IndependentContinuant
2.	physical-endurant	equivalence	materialEntity
3.	physical-object	equivalence	Object
4.	perdurant	equivalence	Occurrent
5.	process	equivalence	Process
6.	quality	equivalence	Quality
7.	spatio-temporal-region	equivalence	SpatioTemporalRegion
8.	temporal-region	equivalence	TemporalRegion
9.	space-region	equivalence	SpatialRegion
	Object Property from FunctionalParticipation	Alignment relation	Object Property from BFORO
1.	part	equivalence	has_part
2.	part-of	equivalence	part_of
3.	proper-part	equivalence	has_proper_part
4.	proper-part-of	equivalence	proper_part_of
5.	participant	equivalence	has_participant
6.	participant-in	equivalence	participates_in
7.	generic-location	equivalence	located_in
8.	generic-location-of	equivalence	location-of

Figure A.4.: Alignments between FunctionalParticipation and BFORO ontologies.

	Class from BFORO	Alignment relation	Class from GFO
1.	Entity	equivalence	Entity
2.	IndependentContinuant	equivalence	Persistent
3.	DependentContinuant	equivalence	Dependent
4.	MaterialEntity	equivalence	Material_persistent
5.	Object	equivalence	Material_object
6.	ObjectBoundary	equivalence	Material_boundary
7.	Function	equivalence	Function
8.	Role	equivalence	Role
9.	Occurrent	equivalence	Occurrent
10.	Process	equivalence	Process
11.	SpatialRegion	equivalence	Spatial_region
12.	TemporalRegion	equivalence	Temporal_region
13.	Quality	equivalence	Property
	Object Property from BFORO	Alignment relation	Object Property from GFO
1.	has_part	equivalence	has_part
2.	part_of	equivalence	part_of
3.	has_proper_part	equivalence	has_proper_part
4.	proper_part_of	equivalence	proper_part_of
5.	has_participant	equivalence	has_participant
6.	participates_in	equivalence	participates_in
7.	occupies	equivalence	located_in
8.	occupied_by	equivalence	location_of
9.	has_agent	equivalence	has_agent
10.	agent_in	equivalence	agent_in

Figure A.5.: Alignments between BFORO and GFO ontologies.

	Class from BFORO	Alignment relation	Class from GFO-Basic
1.	Entity	equivalence	Entity
2.	IndependentContinuant	equivalence	Persistent
3.	Object	equivalence	Material_object
4.	ObjectBoundary	equivalence	Material_boundary
5.	Role	equivalence	Role
6.	Occurrent	equivalence	Occurrent
7.	Process	equivalence	Process
8.	SpatialRegion	equivalence	Spatial_region
9.	TemporalRegion	equivalence	Temporal_region
10.	Quality	equivalence	Property
	Object Property from BFORO	Alignment relation	Object Property from GFO-Basic
1.	has_part	equivalence	has_part
2.	part_of	equivalence	part_of
3.	has_proper_part	equivalence	has_proper_part
4.	proper_part_of	equivalence	proper_part_of
5.	has_participant	equivalence	has_participant
6.	participates_in	equivalence	participates_in
7.	occupies	equivalence	located_in
8.	occupied_by	equivalence	location_of

Figure A.6.: Alignments between BFORO and GFOBasic ontologies.

	Class from SpatialRelations	Alignment relation	Class from BFORO
1.	endurant	equivalence	IndependentContinuant
2.	physical-endurant	equivalence	materialEntity
3.	physical-object	equivalence	Object
4.	perdurant	equivalence	Occurrent
5.	process	equivalence	Process
6.	quality	equivalence	Quality
7.	spatio-temporal-region	equivalence	SpatioTemporalRegion
8.	temporal-region	equivalence	TemporalRegion
9.	space-region	equivalence	SpatialRegion
	Object Property from SpatialRelations	Alignment relation	Object Property from BFORO
1.	part	equivalence	has_part
2.	part-of	equivalence	part_of
3.	proper-part	equivalence	has_proper_part
4.	proper-part-of	equivalence	proper_part_of
5.	participant	equivalence	has_participant
6.	participant-in	equivalence	participates_in
7.	generic-location	equivalence	located_in
8.	generic-location-of	equivalence	location-of

Figure A.7.: Alignments between SpatialRelations and BFORO ontologies.

	Class from TemporalRelations	Alignment relation	Class from BFORO
1.	endurant	equivalence	IndependentContinuant
2.	physical-endurant	equivalence	materialEntity
3.	physical-object	equivalence	Object
4.	perdurant	equivalence	Occurrent
5.	process	equivalence	Process
6.	quality	equivalence	Quality
7.	spatio-temporal-region	equivalence	SpatioTemporalRegion
8.	temporal-region	equivalence	TemporalRegion
9.	space-region	equivalence	SpatialRegion
	Object Property from TemporalRelations	Alignment relation	Object Property from BFORO
1.	part	equivalence	has_part
2.	part-of	equivalence	part_of
3.	proper-part	equivalence	has_proper_part
4.	proper-part-of	equivalence	proper_part_of
5.	participant	equivalence	has_participant
6.	participant-in	equivalence	participates_in
7.	generic-location	equivalence	located_in
8.	generic-location-of	equivalence	location-of
9.	precedes	equivalence	precedes
10.	follows	equivalence	preceded_by

Figure A.8.: Alignments between TemporalRelations and BFORO ontologies.

	Class from DOLCE-Lite	Alignment relation	Class from BFO
1.	endurant	equivalence	IndependentContinuant
2.	physical-endurant	equivalence	materialEntity
3.	physical-object	equivalence	Object
4.	perdurant	equivalence	Occurrent
5.	process	equivalence	Process
6.	quality	equivalence	Quality
7.	spatio-temporal-region	equivalence	SpatioTemporalRegion
8.	temporal-region	equivalence	TemporalRegion
9.	space-region	equivalence	SpatialRegion

Figure A.9.: Alignments between DOLCE-Lite and BFO ontologies.

	Class from DOLCE-Lite	Alignment relation	Class from GFO
1.	particular	equivalence	Individual
2.	endurant	equivalence	Presential
3.	physical-endurant	equivalence	Material_persistent
4.	physical-object	equivalence	Material_object
5.	amount-of-matter	equivalence	Amount_of_substrate
6.	perdurant	equivalence	Occurrent
7.	process	equivalence	Process
8.	state	equivalence	State
9.	abstract	equivalence	Abstract
10.	set	equivalence	Set
11.	quality	equivalence	Property
12.	quale	equivalence	Property_value
13.	quality-space	equivalence	Value_space
14.	time-interval	equivalence	Chronoid
15.	space-region	equivalence	Spatial_region
16.	temporal-region	equivalence	Temporal_region
	Object Property from DOLCE-Lite	Alignment relation	Object Property from GFO
1.	generic-constituant	equivalence	has_constituant_part
2.	generic-constituant-of	equivalence	constituant_part_of
3.	generic-dependent	equivalence	necessary_for
4.	generically-dependent-on	equivalence	depends_on
5.	part	equivalence	abstract_has_part
6.	part-of	equivalence	abstract_part_of
7.	proper-part	equivalence	has_proper_part
8.	proper-part-of	equivalence	proper_part_of
9.	participant	equivalence	has_participant
10.	participant-in	equivalence	participates_in
11.	has-quale	equivalence	has_value
12.	quale-of	equivalence	value_of
13.	boundary	equivalence	has_boundary
14.	boundary-of	equivalence	boundary_of
15.	q-present-at	equivalence	exists_at
16.	temporary-participant	equivalence	has_agent
17.	temporary-participant-in	equivalence	agent_in
18.	exact-location	equivalence	occupies
19.	exact-location-of	equivalence	occupied_by

Figure A.10.: Alignments between DOLCE-Lite and GFO ontologies.

	Class from DOLCE-Lite	Alignment relation	Class from GFO-Basic
1.	particular	equivalence	Individual
2.	endurant	equivalence	Presential
3.	physical-object	equivalence	Material_object
4.	amount-of-matter	equivalence	Amount_of_substrate
5.	perdurant	equivalence	Occurrent
6.	process	equivalence	Process
7.	state	equivalence	State
8.	abstract	equivalence	Abstract
9.	quality	equivalence	Property
10.	time-interval	equivalence	Chronoid
11.	space-region	equivalence	Spatial_region
12.	temporal-region	equivalence	Temporal_region
13.	event	equivalence	Event
	Object Property from DOLCE-Lite	Alignment relation	Object Property from GFO-Basic
1.	generic-dependent	equivalence	necessary_for
2.	generically-dependent-on	equivalence	depends_on
3.	part	equivalence	abstract_has_part
4.	part-of	equivalence	abstract_part_of
5.	proper-part	equivalence	has_proper_part
6.	proper-part-of	equivalence	proper_part_of
7.	participant	equivalence	has_participant
8.	participant-in	equivalence	participates_in
9.	boundary	equivalence	has_boundary
10.	boundary-of	equivalence	boundary_of
11.	q-present-at	equivalence	exists_at
12.	exact-location	equivalence	occupies
13.	exact-location-of	equivalence	occupied_by
14.	has-quality	equivalence	has-property

Figure A.11.: Alignments between DOLCE-Lite and GFOBasic ontologies.

	Class from FunctionalParticipation	Alignment relation	Class from BFO
1.	endurant	equivalence	IndependentContinuant
2.	physical-endurant	equivalence	materialEntity
3.	physical-object	equivalence	Object
4.	perdurant	equivalence	Occurrent
5.	process	equivalence	Process
6.	quality	equivalence	Quality
7.	spatio-temporal-region	equivalence	SpatioTemporalRegion
8.	temporal-region	equivalence	TemporalRegion
9.	space-region	equivalence	SpatialRegion
10.	role	equivalence	Role

Figure A.12.: Alignments between FunctionalParticipation and BFO ontologies.

	Class from FunctionalParticipation	Alignment relation	Class from GFO
1.	particular	equivalence	Individual
2.	endurant	equivalence	Presential
3.	physical-endurant	equivalence	Material_persistent
4.	physical-object	equivalence	Material_object
5.	amount-of-matter	equivalence	Amount_of_substrate
6.	perdurant	equivalence	Occurrent
7.	process	equivalence	Process
8.	state	equivalence	State
9.	abstract	equivalence	Abstract
10.	set	equivalence	Set
11.	quality	equivalence	Property
12.	quale	equivalence	Property_value
13.	quality-space	equivalence	Value_space
14.	time-interval	equivalence	Chronoid
15.	space-region	equivalence	Spatial_region
16.	temporal-region	equivalence	Temporal_region
17.	situation	equivalence	Situation
18.	social-role	equivalence	Social_role
19.	concept	equivalence	Concept
20.	role	equivalence	Role
21.	action	equivalence	Action

Figure A.13.: Class alignments between FunctionalParticipation and GFO ontologies.

	Object Property from FunctionalParticipation	Alignment relation	Object Property from GFO
1.	generic-constituant	equivalence	has_constituant_part
2.	generic-constituant-of	equivalence	constituant_part_of
3.	generic-dependent	equivalence	necessary_for
4.	generically-dependent-on	equivalence	depends_on
5.	part	equivalence	abstract_has_part
6.	part-of	equivalence	abstract_part_of
7.	proper-part	equivalence	has_proper_part
8.	proper-part-of	equivalence	proper_part_of
9.	participant	equivalence	has_participant
10.	participant-in	equivalence	participates_in
11.	has-quale	equivalence	has_value
12.	quale-of	equivalence	value_of
13.	boundary	equivalence	has_boundary
14.	boundary-of	equivalence	boundary_of
15.	q-present-at	equivalence	exists_at
16.	temporary-participant	equivalence	has_agent
17.	temporary-participant-in	equivalence	agent_in
18.	exact-location	equivalence	occupies
19.	exact-location-of	equivalence	occupied_by

Figure A.14.: Object property alignments between FunctionalParticipation and GFO ontologies.

	Class from FunctionalParticipation	Alignment relation	Class from GFO-Basic
1.	particular	equivalence	Individual
2.	endurant	equivalence	Presential
3.	physical-object	equivalence	Material_object
4.	amount-of-matter	equivalence	Amount_of_substrate
5.	perdurant	equivalence	Occurrent
6.	process	equivalence	Process
7.	state	equivalence	State
8.	abstract	equivalence	Abstract
9.	quality	equivalence	Property
10.	time-interval	equivalence	Chronoid
11.	space-region	equivalence	Spatial_region
12.	temporal-region	equivalence	Temporal_region
13.	event	equivalence	Event
14.	role	equivalence	Role
15.	social-role	equivalence	Social-role
16.	concept	equivalence	Concept
	Object Property from FunctionalParticipation	Alignment relation	Object Property from GFO-Basic
1.	generic-dependent	equivalence	necessary_for
2.	generically-dependent-on	equivalence	depends_on
3.	part	equivalence	abstract_has_part
4.	part-of	equivalence	abstract_part_of
5.	proper-part	equivalence	has_proper_part
6.	proper-part-of	equivalence	proper_part_of
7.	participant	equivalence	has_participant
8.	participant-in	equivalence	participates_in
9.	boundary	equivalence	has_boundary
10.	boundary-of	equivalence	boundary_of
11.	q-present-at	equivalence	exists_at
12.	exact-location	equivalence	occupies
13.	exact-location-of	equivalence	occupied_by
14.	has-quality	equivalence	has-property
15.	inherent-in	equivalence	property_of

Figure A.15.: Alignments between FunctionalParticipation and GFOBasic ontologies.

	Class from SpatialRelations	Alignment relation	Class from BFO
1.	endurant	equivalence	IndependentContinuant
2.	physical-endurant	equivalence	materialEntity
3.	physical-object	equivalence	Object
4.	perdurant	equivalence	Occurrent
5.	process	equivalence	Process
6.	quality	equivalence	Quality
7.	spatio-temporal-region	equivalence	SpatioTemporalRegion
8.	temporal-region	equivalence	TemporalRegion
9.	space-region	equivalence	SpatialRegion

Figure A.16.: Alignments between SpatialRelations and BFO ontologies.

	Class from SpatialRelations	Alignment relation	Class from GFO
1.	particular	equivalence	Individual
2.	endurant	equivalence	Presential
3.	physical-endurant	equivalence	Material_persistent
4.	physical-object	equivalence	Material_object
5.	amount-of-matter	equivalence	Amount_of_substrate
6.	perdurant	equivalence	Occurrent
7.	process	equivalence	Process
8.	state	equivalence	State
9.	abstract	equivalence	Abstract
10.	set	equivalence	Set
11.	quality	equivalence	Property
12.	quale	equivalence	Property_value
13.	quality-space	equivalence	Value_space
14.	time-interval	equivalence	Chronoid
15.	space-region	equivalence	Spatial_region
16.	temporal-region	equivalence	Temporal_region
	Object Property from SpatialRelations	Alignment relation	Object Property from GFO
1.	generic-constituant	equivalence	has_constituant_part
2.	generic-constituant-of	equivalence	constituant_part_of
3.	generic-dependent	equivalence	necessary_for
4.	generically-dependent-on	equivalence	depends_on
5.	part	equivalence	abstract_has_part
6.	part-of	equivalence	abstract_part_of
7.	proper-part	equivalence	has_proper_part
8.	proper-part-of	equivalence	proper_part_of
9.	participant	equivalence	has_participant
10.	participant-in	equivalence	participates_in
11.	has-quale	equivalence	has_value
12.	quale-of	equivalence	value_of
13.	boundary	equivalence	has_boundary
14.	boundary-of	equivalence	boundary_of
15.	q-present-at	equivalence	exists_at
16.	temporary-participant	equivalence	has_agent
17.	temporary-participant-in	equivalence	agent_in
18.	exact-location	equivalence	occupies
19.	exact-location-of	equivalence	occupied_by

Figure A.17.: Alignments between SpatialRelations and GFO ontologies.

	Class from SpatialRelation	Alignment relation	Class from GFO-Basic
1.	particular	equivalence	Individual
2.	endurant	equivalence	Presential
3.	physical-object	equivalence	Material_object
4.	amount-of-matter	equivalence	Amount_of_substrate
5.	perdurant	equivalence	Occurrent
6.	process	equivalence	Process
7.	state	equivalence	State
8.	abstract	equivalence	Abstract
9.	quality	equivalence	Property
10.	time-interval	equivalence	Chronoid
11.	space-region	equivalence	Spatial_region
12.	temporal-region	equivalence	Temporal_region
13.	event	equivalence	Event
	Object Property from SpatialRelations	Alignment relation	Object Property from GFO-Basic
1.	generic-dependent	equivalence	necessary_for
2.	generically-dependent-on	equivalence	depends_on
3.	part	equivalence	abstract_has_part
4.	part-of	equivalence	abstract_part_of
5.	proper-part	equivalence	has_proper_part
6.	proper-part-of	equivalence	proper_part_of
7.	participant	equivalence	has_participant
8.	participant-in	equivalence	participates_in
9.	boundary	equivalence	has_boundary
10.	boundary-of	equivalence	boundary_of
11.	q-present-at	equivalence	exists_at
12.	exact-location	equivalence	occupies
13.	exact-location-of	equivalence	occupied_by
14.	has-quality	equivalence	has-property
15.	inherent-in	equivalence	property_of

Figure A.18.: Alignments between SpatialRelations and GFOBasic ontologies.

	Class from TemporalRelations	Alignment relation	Class from BFO
1.	endurant	equivalence	IndependentContinuant
2.	physical-endurant	equivalence	materialEntity
3.	physical-object	equivalence	Object
4.	perdurant	equivalence	Occurrent
5.	process	equivalence	Process
6.	quality	equivalence	Quality
7.	spatio-temporal-region	equivalence	SpatioTemporalRegion
8.	temporal-region	equivalence	TemporalRegion
9.	space-region	equivalence	SpatialRegion

Figure A.19.: Alignments between TemporalRelations and BFO ontologies.

	Class from TemporalRelations	Alignment relation	Class from GFO
1.	particular	equivalence	Individual
2.	endurant	equivalence	Presential
3.	physical-endurant	equivalence	Material_persistent
4.	physical-object	equivalence	Material_object
5.	amount-of-matter	equivalence	Amount_of_substrate
6.	perdurant	equivalence	Occurrent
7.	process	equivalence	Process
8.	state	equivalence	State
9.	abstract	equivalence	Abstract
10.	set	equivalence	Set
11.	quality	equivalence	Property
12.	quale	equivalence	Property_value
13.	quality-space	equivalence	Value_space
14.	time-interval	equivalence	Chronoid
15.	space-region	equivalence	Spatial_region
16.	temporal-region	equivalence	Temporal_region

Figure A.20.: Class alignments between TemporalRelations and GFO ontologies.

	Object Property from TemporalRelations	Alignment relation	Object Property from GFO
1.	generic-constituant	equivalence	has_constituant_part
2.	generic-constituant-of	equivalence	constituant_part_of
3.	generic-dependent	equivalence	necessary_for
4.	generically-dependent-on	equivalence	depends_on
5.	part	equivalence	abstract_has_part
6.	part-of	equivalence	abstract_part_of
7.	proper-part	equivalence	has_proper_part
8.	proper-part-of	equivalence	proper_part_of
9.	participant	equivalence	has_participant
10.	participant-in	equivalence	participates_in
11.	has-quale	equivalence	has_value
12.	quale-of	equivalence	value_of
13.	boundary	equivalence	has_boundary
14.	boundary-of	equivalence	boundary_of
15.	q-present-at	equivalence	exists_at
16.	temporary-participant	equivalence	has_agent
17.	temporary-participant-in	equivalence	agent_in
18.	exact-location	equivalence	occupies
19.	exact-location-of	equivalence	occupied_by

Figure A.21.: Object property alignments between TemporalRelations and GFO ontologies.

	Class from TemporalRelations	Alignment relation	Class from GFO-Basic
1.	particular	equivalence	Individual
2.	endurant	equivalence	Presential
3.	physical-object	equivalence	Material_object
4.	amount-of-matter	equivalence	Amount_of_substrate
5.	perdurant	equivalence	Occurrent
6.	process	equivalence	Process
7.	state	equivalence	State
8.	abstract	equivalence	Abstract
9.	quality	equivalence	Property
10.	time-interval	equivalence	Chronoid
11.	space-region	equivalence	Spatial_region
12.	temporal-region	equivalence	Temporal_region
13.	event	equivalence	Event
	Object Property from TemporalRelations	Alignment relation	Object Property from GFO-Basic
1.	generic-dependent	equivalence	necessary_for
2.	generically-dependent-on	equivalence	depends_on
3.	part	equivalence	abstract_has_part
4.	part-of	equivalence	abstract_part_of
5.	proper-part	equivalence	has_proper_part
6.	proper-part-of	equivalence	proper_part_of
7.	participant	equivalence	has_participant
8.	participant-in	equivalence	participates_in
9.	boundary	equivalence	has_boundary
10.	boundary-of	equivalence	boundary_of
11.	q-present-at	equivalence	exists_at
12.	exact-location	equivalence	occupies
13.	exact-location-of	equivalence	occupied_by
14.	has-quality	equivalence	has-property
15.	inherent-in	equivalence	property_of

Figure A.22.: Alignments between TemporalRelations and GFOBasic ontologies.

Alignments from existing tools and documentations

We provide the alignments given by each tool and documentation.

B.1. H-Match’s alignments

Table B.1.: H-Match’s equivalence alignments between DOLCE-Lite and BFO ontologies.

	DOLCE-Lite	BFO
1.	accomplishment	Occurrent
2.	achievement	Entity
3.	endurant	Entity
4.	event	Entity
5.	feature	Occurrent
6.	process	Process
7.	proposition	Disposition
8.	quale	Role
9.	quality	Quality
10.	region	SpatialRegion
11.	set	Entity
12.	spatio-temporal-region	SpatioTemporalRegion
13.	state	Role
14.	stative	Entity
15.	temporal-region	TemporalRegion
16.	time-interval	Occurrent

Table B.2.: H-Match's equivalence alignments between DOLCE-Lite and GFO ontologies.

	DOLCE-Lite	GFO
1.	abstract	Abstract
2.	abstract-quality	Abstract
3.	abstract-region	Abstract
4.	accomplishment	Entity
5.	achievement	Item
6.	arbitrary-sum	Item
7.	dependent-place	Dependent
8.	endurant	Entity
9.	event	Entity
10.	feature	Set
11.	perdurant	Persistent
12.	process	Process
13.	proposition	Individual
14.	quale	Role
15.	quality	Entity
16.	quality-space	Space
17.	region	Spatial_region
18.	relevant-part	Entity
19.	set	Individual
20.	space-region	Space
21.	spatio-temporal-region	Spatial_region
22.	state	State
23.	stative	State
24.	temporal-region	Temporal_region
25.	time-interval	Time

Table B.3.: H-Match's equivalence alignments between BFO and GFO ontologies.

	BFO	GFO
1.	Entity	Mass_entity
2.	Continuant	Continuous
3.	Disposition	Item
4.	FiatObjectPart	Material_boundary
5.	Function	Function
6.	MaterialEntity	Entity
7.	Object	Set
8.	ObjectAggregate	Item
9.	OneDimensionalRegion	Space
10.	Quality	Item
11.	RealizableEntity	Entity
12.	Role	Set
13.	Site	Item
14.	SpatialRegion	Spatial_region
15.	ThreeDimensionalRegion	Space
16.	TwoDimensionalRegion	Space
17.	ZeroDimensionalRegion	Space
18.	ConnectedTemporalRegion	Temporal_region
19.	FiatProcessPart	Process
20.	Occurrent	Occurrent
21.	Process	Process
22.	ProcessAggregate	Process
23.	ProcessBoundary	Process
24.	ProcessualContext	Process
25.	ProcessualEntity	Processual_role
26.	ScatteredSpatiotemporalRegion	Time
27.	ScatteredTemporalRegion	Time
28.	SpatiotemporalRegion	Temporal_region
29.	TemporalInstant	Time
30.	TemporalInterval	Time
31.	TemporalRegion	Temporal_region

B.2. PROMPT's alignments

Table B.4.: PROMPT's equivalence alignments between DOLCE-Lite and BFO ontologies.

	DOLCE-Lite	BFO
1.	quality	Quality
2.	state	Site
3.	quale	Roll
4.	process	Process
5.	state	Role
6.	stative	Entity
7.	temporal-region	TemporalRegion
8.	time-interval	Occurrent

Table B.5.: PROMPT's equivalence alignments between DOLCE-Lite and GFO ontologies.

	DOLCE-Lite	GFO
1.	state	State
2.	abstract	Abstract
3.	temporal-region	Temporal_region
4.	process	Process
5.	quale	Role
6.	particular	Item
7.	particular	Dependent

Table B.6.: PROMPT’s equivalence alignments between BFO and GFO ontologies.

	BFO	GFO
1.	SpatialRegion	Spatial_region
2.	Site	Space
3.	Site	Surface
4.	Site	Situoid
5.	Site	State
6.	Role	Role
7.	Function	Function
8.	Function	Action
9.	Process	Process
10.	TemporalRegion	Temporal_Region
11.	Occurrent	Occurrent
12.	Entity	Entity

B.3. LogMap’s alignments

Table B.7.: LogMap’s equivalence alignments between DOLCE-Lite and BFO ontologies.

	DOLCE-Lite	BFO
1.	quality	Quality
2.	process	Process

Table B.8.: LogMap’s equivalence alignments between DOLCE-Lite and GFO ontologies.

	DOLCE-Lite	GFO
1.	state	State
2.	process	Process

Table B.9.: LogMap's equivalence alignments between BFO and GFO ontologies.

	BFO	GFO
1.	Entity	Entity
2.	DependentContinuant	Dependent
3.	Function	Function
4.	IndependentContinuant	Independent
5.	Object	MaterialObject
6.	Role	Role
7.	Occurrent	Occurrent
8.	SpatialRegion	Spatial_region
9.	Process	Process

B.4. YAM++'s alignments

Table B.10.: YAM++'s equivalence alignments between DOLCE-Lite and BFO ontologies.

	DOLCE-Lite	BFO
1.	process	Process
2.	quality	Quality
3.	spatio-temporal-region	SpatiotemporalRegion
4.	temporal-region	TemporalRegion

Table B.11.: YAM's equivalence alignments between BFO and GFO ontologies.

	BFO	GFO
1.	Entity	Entity
2.	Function	Function
3.	IndependentContinuant	Independent
4.	Object	Material_object
5.	Occurrent	Occurrent
7.	Process	Process
6.	SpatialRegion	Spatial_region

Table B.12.: YAM++’s equivalence alignments between DOLCE-Lite and GFO ontologies.

	DOLCE-Lite	GFO
1.	abstract	Abstract
2.	boundary	has_boundary
3.	boundary-of	boundary_of
4.	constant-participant	has_agent
5.	endurant	Presential
6.	generic-constituent	has_sequence_constituent
7.	generic-constituent-of	sequence_constituent_of
8.	immediate-relation	abstract_has_part
9.	immediate-relation-i	abstract_part_of
10.	part	has_part
11.	part-of	part_of
12.	participant	has_participant
13.	particular	Item
14.	perdurant	Occurrent
15.	process	Process
16.	proper-part	has_proper_part
17.	proper-part-of	proper_part_of
18.	quality-space	Space
19.	set	Set
20.	spatio-temporal-particular	Individual
21.	state	State
22.	temporal-region	Temporal_region
23.	temporary-proper-part	has_constituent_part
24.	temporary-proper-part-of	constituent_part_of
25.	time-interval	Chronoid

B.5. HotMatch’s alignments

Table B.13.: HotMatch’s equivalence alignments between DOLCE-Lite and BFO ontologies.

	DOLCE-Lite	BFO
1.	process	Process
2.	quality	Quality
3.	temporal-region	TemporalRegion

Table B.14.: HotMatch's equivalence alignments between DOLCE-Lite and GFO ontologies.

	DOLCE-Lite	GFO
1.	abstract	Abstract
2.	boundary	has_boundary
3.	boundary-of	boundary_of
4.	part	has_part
5.	part-of	part_of
6.	participant	has_participant
7.	process	Process
8.	proper-part	has_proper_part
9.	proper-part-of	proper_part_of
10.	set	Set
11.	state	State
12.	temporal-region	Temporal_region

Table B.15.: HotMatch's equivalence alignments between BFO and GFO ontologies.

	BFO	GFO
1.	Entity	Entity
2.	Function	Function
3.	Occurrent	Occurrent
4.	Process	Process
5.	Role	Role
6.	SpatialRegion	Spatial_region
7.	TemporalRegion	Temporal_region

B.6. Hertuda's alignments

Table B.16.: Hertuda's equivalence alignments between DOLCE-Lite and BFO ontologies.

	DOLCE-Lite	BFO
1.	process	Process
2.	quality	Quality
3.	temporal-region	TemporalRegion

Table B.17.: Hertuda's equivalence alignments between DOLCE-Lite and GFO ontologies.

	DOLCE-Lite	GFO
1.	abstract	Abstract
2.	boundary	has_boundary
3.	boundary-of	boundary_of
4.	part	has_part
5.	part-of	part_of
6.	participant	has_participant
7.	participant-in	has_participant
8.	process	Process
9.	proper-part	has_proper_part
10.	proper-part-of	proper_part_of
11.	set	Set
12.	state	State
13.	temporal-region	Temporal_region

Table B.18.: Hertuda's equivalence alignments between BFO and GFO ontologies.

	BFO	GFO
1.	Entity	Entity
2.	Function	Function
3.	Occurrent	Occurrent
4.	Process	Process
5.	Role	Role
6.	SpatialRegion	Spatial_region
7.	TemporalRegion	Temporal_region

B.7. Optima's alignments

Table B.19.: Optima's equivalence alignments between DOLCE-Lite and BFO ontologies.

	DOLCE-Lite	BFO
1.	dependent-place	Site
2.	non-physical-object	Object
3.	physical-object	Object
4.	physical-region	ConnectedSpatiotemporalRegion
5.	process	Process
6.	quality	Quality
7.	region	SpatiotemporalRegion
8.	relevant-part	Function
9.	relevant-part	Role
10.	space-region	SpatiotemporalInterval
11.	state	ProcessBoundary
12.	temporal-region	TemporalRegion
13.	space-region	SynonymType

Table B.20.: Optima's equivalence alignments between BFO and GFO ontologies.

	BFO	GFO
1.	Continuous	Continuant
2.	Entity	Entity
3.	Function	Function
4.	Item	Continuant
5.	Material_object	Object
6.	Material_structure	FiatObjectPart
7.	Occurrent	Occurrent
8.	Point	Continuant
9.	Process	Process
10.	Role	Role
11.	Situation	Site
12.	Space_time	ObjectBoundary
13.	SpatialRegion	Spatial_region
14.	TemporalRegion	Temporal_region
15.	Space	SynonymType
16.	has_agent	has_agent
17.	has_proper_part	ObsoleteProperty

Table B.21.: Optima’s equivalence alignments between DOLCE-Lite and GFO ontologies.

	DOLCE-Lite	GFO
1.	abstract	Abstract
2.	amount-of-matter	Property_value
3.	atomic-part	has_proper_part
4.	atomic-part	has_sequence_constituent
5.	atomic-part-of	has_proper_part
6.	atomic-part-of	has_sequence_constituent
7.	boundary	has_spatial_boundary
8.	boundary-of	boundary_of
9.	dependent-place	Dependent
10.	dependent-place	Point
11.	feature	Property
12.	identity-c	depends_on
13.	identity-n	depends_on
14.	immediate-relation	abstract_has_part
15.	immediate-relation-i	abstract_has_part
16.	mediated-relation	abstract_has_part
17.	mediated-relation-i	abstract_has_part
18.	part	has_proper_part
19.	part	has_sequence_constituent
20.	part-of	has_proper_part
21.	part-of	has_sequence_constituent
22.	particular	Item
23.	partly-compresent	categorial_part_of
24.	physical-region	Time_boundary
25.	process	Process
26.	proper-part	has_proper_part
27.	proper-part	has_sequence_constituent
28.	proper-part-of	has_sequence_constituent
29.	proper-part-of	has_proper_part
30.	quality-space	Space
31.	region	Spatial_region
32.	relevant-part	Function
33.	relevant-part	Role
34.	set	Set
35.	sibling-part	abstract_part_of
36.	space-region	Space
37.	state	State
38.	temporal-region	Temporal_region
39.	time-interval	Time

B.8. GFO documentation's alignments

Table B.22.: Equivalence alignments between DOLCE-Lite and GFO ontologies from the GFO documentation [30].

	DOLCE-Lite	GFO
1.	particular	Individual
2.	endurant	Presential
3.	endurant	Persistent
4.	physical-endurant	Material_structure
5.	amount-of-matter	Amount_of_substrate
6.	Feature	Material_boundary
7.	physical-object	Material_object
8.	non-physical-endurant	Levels
9.	mental-object	Concept
10.	social-agent	Social_role
11.	perdurant	Occurrent
12.	event	Change
13.	achievement	Achievement
14.	accomplishment	Accomplishment
15.	stative	Process
16.	state	State
17.	quality	Property
18.	abstract	Space_time or Set or Fact
19.	fact	Fact
20.	set	Set
21.	region	Space_time
22.	region	Measurement_system
23.	time-interval	Chronoid
24.	space-region	Spatial_region
25.	temporal-region or space-region	Space_time
26.	region	Space_region
27.	abstract	Abstract
28.	endurant or perdurant or quality	Concrete
29.	quale	Property_value
30.	temporal-region	Temporal_region
31.	entity	Entity

B.9. Temal et al.'s alignments

Table B.23.: Equivalence alignments between DOLCE-Lite and BFO ontologies from Temal et al.'s alignment [82].

	DOLCE-Lite	BFO
1.	achievement	InstantaneousTemporalBoundaries
2.	physical-quality	Quality
3.	temporal-region	TemporalRegion
4.	time-interval	Interval
5.	space-region	SpatialRegion
6.	perdurant	ProcessualEntities
7.	achievement	Events

ROMULUS documentation

ROMULUS is a web-based repository aimed at promoting foundational ontology usage for achieving semantic interoperability. To access ROMULUS, go to <http://www.thezfiles.co.za/ROMULUS/home.html>

The header of ROMULUS has a menu bar containing ROMULUS’s functions. See Fig. C.1.



Figure C.1.: ROMULUS’s menu bar with different functions.

The **Home** page (Fig. C.2) introduces a user to ROMULUS, its goals and functions.

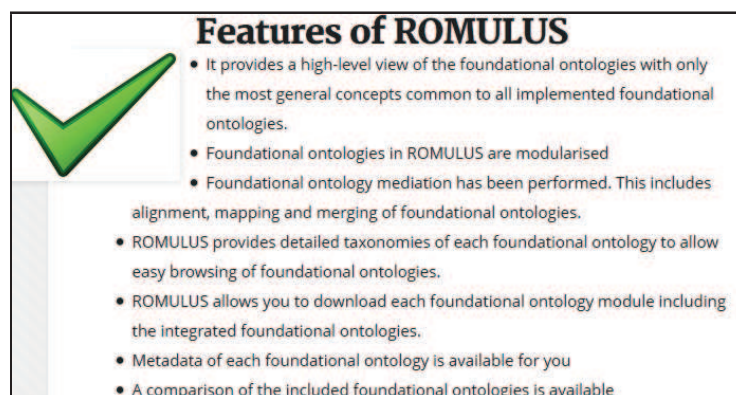
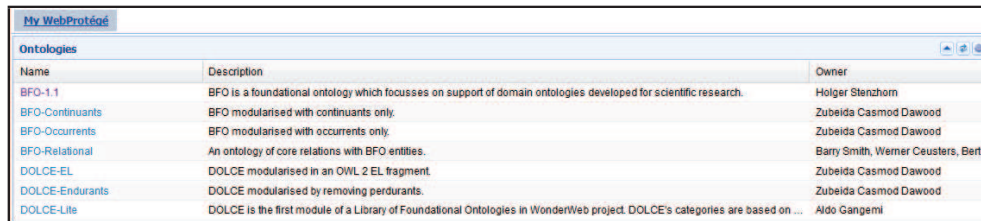


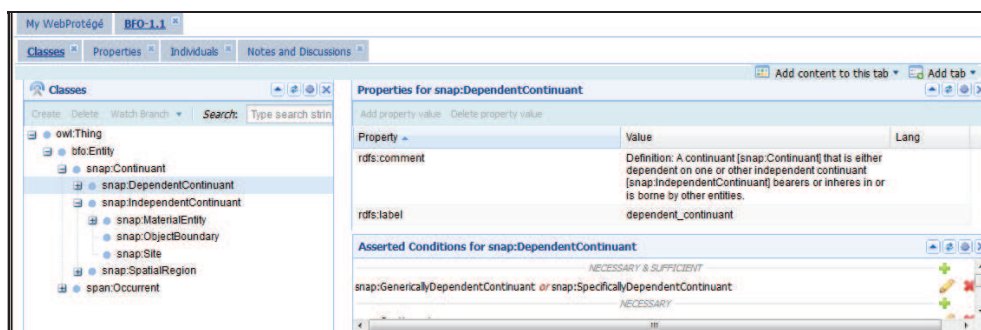
Figure C.2.: ROMULUS’s home page.

A user may browse the ontologies in the repository, online by opening up the **Browse ontologies** page. See Figures C.3 and C.4 for this.



Name	Description	Owner
BFO-1.1	BFO is a foundational ontology which focusses on support of domain ontologies developed for scientific research.	Holger Stenzhorn
BFO-Continuants	BFO modularised with continuants only.	Zubeida Casmod Dawood
BFO-Occurrents	BFO modularised with occurrents only.	Zubeida Casmod Dawood
BFO-Relational	An ontology of core relations with BFO entities.	Barry Smith, Werner Ceusters, Bert...
DOLCE-EL	DOLCE modularised in an OWL 2 EL fragment.	Zubeida Casmod Dawood
DOLCE-Endurants	DOLCE modularised by removing perdurants.	Zubeida Casmod Dawood
DOLCE-Lite	DOLCE is the first module of a Library of Foundational Ontologies in WonderWeb project. DOLCE's categories are based on ...	Aldo Gangemi

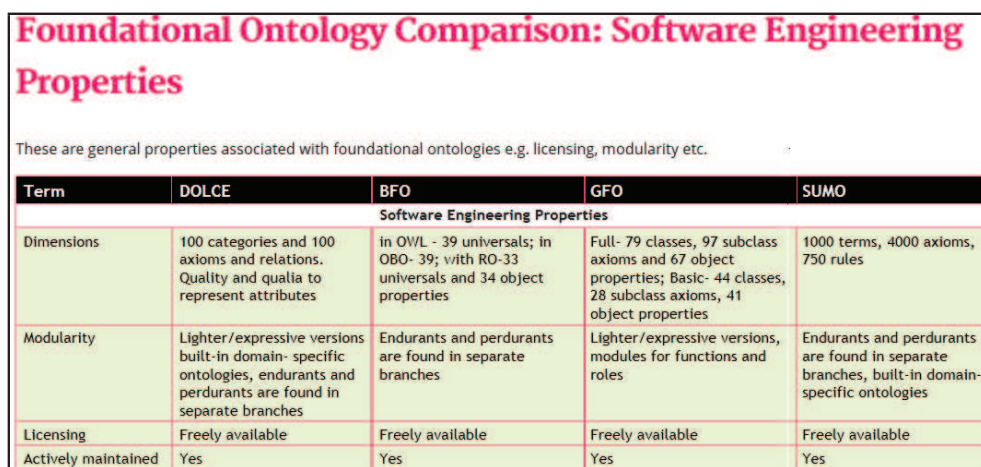
Figure C.3.: ROMULUS's browse ontology page.



The screenshot shows the 'Classes' tab for the BFO-1.1 ontology. On the left, a tree view shows the hierarchy: owl:Thing, bfo:Entity, snap:Continuant, snap:DependentContinuant, snap:IndependentContinuant, snap:MaterialEntity, snap:ObjectBoundary, snap:Site, snap:SpatialRegion, and span:Occurrent. The right pane shows the 'Properties for snap:DependentContinuant', including 'rdfs:comment' and 'rdfs:label'. Below this, 'Asserted Conditions for snap:DependentContinuant' are listed, showing 'NECESSARY & SUFFICIENT' and 'NECESSARY' conditions.

Figure C.4.: Browsing through the BFO ontology in ROMULUS.

The **Ontology Comparison** page provides a multi-dimensional comparison of foundational ontologies. It is spread out to different pages: **Ontological Commitments**, **Representation Language**, **Software Engineering Properties**, **Subject Domains**, and **Applications**. One of the pages, **Software Engineering Properties** is displayed in Fig. C.5



The screenshot shows a table titled 'Foundational Ontology Comparison: Software Engineering Properties'. Below the title, it states: 'These are general properties associated with foundational ontologies e.g. licensing, modularity etc.' The table compares five ontologies: DOLCE, BFO, GFO, and SUMO across four properties: Dimensions, Modularity, Licensing, and Actively maintained.

Term	DOLCE	BFO	GFO	SUMO
Software Engineering Properties				
Dimensions	100 categories and 100 axioms and relations. Quality and qualia to represent attributes	in OWL - 39 universals; in OBO- 39; with RO-33 universals and 34 object properties	Full- 79 classes, 97 subclass axioms and 67 object properties; Basic- 44 classes, 28 subclass axioms, 41 object properties	1000 terms, 4000 axioms, 750 rules
Modularity	Lighter/expressive versions built-in domain- specific ontologies, endurants and perdurants are found in separate branches	Endurants and perdurants are found in separate branches	Lighter/expressive versions, modules for functions and roles	Endurants and perdurants are found in separate branches, built-in domain-specific ontologies
Licensing	Freely available	Freely available	Freely available	Freely available
Actively maintained	Yes	Yes	Yes	Yes

Figure C.5.: The Software Engineering Properties comparison page.

The **Ontology View** page provides different human-readable views on the axioms of the ontology for the users. Currently it provides a natural language, and description logic view. The user decides on a view before proceeding with viewing the ontology. Description logic view provides the axioms of the ontology in description logic while natural language view provides the axioms in natural language sentences, in alphabetical order of classes, relational properties and individuals. Fig. C.6 displays these views.

Ontology Verbalisation

There are two views of the foundational ontologies taken by ROMULUS. The axiomatisation of each foundational ontology is shown in description logic view. The formalism of each foundational ontology is shown in natural language view. Click on a link below to proceed with a view.

- Description logic view:** The axioms are expressed in DL language. e.g., $\text{Chocolate_cake} \sqsubseteq \text{cake}$
- Natural language view:** The axioms are expressed in natural language in alphabetical order e.g., A chocolate_cake is a cake.

Natural Language View: GFO

ABSTRACT (class)

Typology	An abstract is an individual .
Distinctions	No abstract is a space time , or a concrete .

ACTION (class)

Typology	An action is an occurent .
Description	An action has as agent a presential .

Description logic view: GFO

Classes

Abstract

$\text{Abstract} \sqsubseteq \text{Individual}$
 $\text{Abstract} \sqsubseteq \neg \text{Space_time}$
 $\text{Abstract} \sqsubseteq \neg \text{Concrete}$

Action

$\text{Action} \sqsubseteq \text{Occurent}$
 $\text{Action} \sqsubseteq \exists \text{has_agent Presential}$

Figure C.6.: A screenshot of the ontology view page with snippets from natural language and description logic views for GFO ontology.

The **Ontology mediation** page is spread out onto five pages: **Alignment**, **Mapping**, **Merging**, **Search**, **Foundational Ontology Interchangeability**, and **Mapping Inconsistencies**. The **Alignment** pages (Fig. C.7) have tables of ontological alignments.

Ontology Alignment		
Alignments between DOLCE-Lite and BFO		
	Class from DOLCE-Lite	Class from BFO
1.	endurant	equivalence IndependentContinuant
2.	physical-endurant	equivalence materialEntity
3.	physical-object	equivalence Object
4.	perdurant	equivalence Occurent
5.	process	equivalence Process
6.	quality	equivalence Quality
7.	spatio-temporal-region	equivalence SpatioTemporalRegion
8.	temporal-region	equivalence TemporalRegion
9.	space-region	equivalence SpatialRegion

Figure C.7.: A table of ontological alignments.

The **Search** page allows you to search for a particular alignment. See Fig. C.8.

The screenshot displays the 'Ontology Mediation Search' web interface. At the top, there is a heading 'Ontology Mediation Search' in red. Below it, a sub-header reads 'Search for alignments and mappings.' There are three radio buttons: 'Alignments' (selected), 'Mappings', and 'Inconsistencies'. A search input field contains the text 'Set', and a 'Search' button is to its right. Below the search section, the heading 'Mediation Search Results' is shown in red, followed by 'Search results'. A message states 'Matches for the term : Set.' Below this is a table with four columns: 'Entity', 'Relation', 'Entity', and 'Link to table'. The table contains four rows of results, each showing 'set' as the entity, 'equivalence' as the relation, and 'Set' as the entity, with links to various GFO modules.

Entity	Relation	Entity	Link to table
set	equivalence	Set	temporalrelations-gfo.php
set	equivalence	Set	spatialrelations-gfo.php
set	equivalence	Set	functionalparticipation-gfo.php
set	equivalence	Set	dolcelite-gfo.php

Figure C.8.: Ontology alignment search.

The **Mapping** and **Merging** pages have similar functionality as the **Browse ontologies** page in that they allow a user to browse through the mapping and merged ontologies. The **Foundational Ontology Interchangeability** page contains a method that may be used to perform foundational ontology interchangeability. The **Mapping Inconsistencies** page provides explanations for mapping inconsistencies that arise in cases where alignments cannot be mapped due to various logical reasons.

The **Ontology Metadata** page has compiled lists of metadata for each foundational ontology module in ROMULUS. See Fig. C.9 for the page that shows the metadata list of the BFO-Continuants module. There is also an **Ontology Metadata Search** page, and the output of this is shown in Fig. C.10.

The **Downloads** page has download links for each foundational ontology module, as well as for additional resources for ROMULUS and ONSET.

The **Ontology Selection** page provides the user with some insight on ONSET, a foundational ontology selection tool, and has a link for the user to download ONSET.

The **Contact** page has the details of ROMULUS developers.

For further details about ROMULUS, feel free to contact:

Zubeida Casmod Dawood email: zkhan@csir.co.za

C.Maria keet email: keet@ukzn.ac.za

Entity	Value
Ontology details	
Ontology Name	Basic Formal Ontology (Continuants)
Ontology Acronym	BFO-Continuants
Ontology ID	13
Ontology description	BFO modularised with continuants only
Ontology creation date	27 July 2012
Ontology latest modified date	27 July 2012
Ontology version	1
Ontology URI	http://www.cs.ukzn.ac.za/zubeida/ontologies/bfo-continuants.owl
Ontology languages	All OWL species
Ontology licence	Free
Organisation details	
Ontology documentation page	
Ontology creators contact details	Zubeida Casmod Dawood zkhan@csir.co.za
Organisation name	University of KwaZulu-Natal (UKZN) and Centre for Artificial Intelligence Research (CAIR), South Africa
Organisation homepage	http://cair.za.net/
Metrics	
Number of classes in the ontology	22
Number of individuals in the ontology	0
Number of properties in the ontology	0
Number of axioms in the ontology	53
Modularity	
Module type	Separate branches of 3D and 4D entities
Original ontology	BFO 1.1

Figure C.9.: Metadata list for BFO-Continuants.

Ontology Metadata Search

To search through the metadata, please select an attribute type and type in a keyword below.

☐ Ontology Name/ Acronym
☐ Ontology Language
☐ Organisation Name
☒ Ontology Developer

Metadata Search Results

Search results

ontologydeveloperMatches for the term : **Loebe**.

Ontology Developer Name	Link to metadata
Frank Loebe	gfo
Frank Loebe	gfo-basic

Figure C.10.: Ontology metadata search.