Modular ontology modeling meets upper ontologies: the upper ontology alignment tool

by

Abhilekha Dalal

B.Tech., Amity University, India, 2017

A REPORT

submitted in partial fulfillment of the
requirements for the degree

MASTER OF SCIENCE

Department of Computer Science
Carl R. Ice College of Engineering

KANSAS STATE UNIVERSITY
Manhattan, Kansas

2020

Approved by:

Major Professor
Dr. Pascal Hitzler

Copyright

Abstract

Ontology modeling has become a primary approach to schema generation for data integration and knowledge graphs in many application areas. The quest for efficient approaches to model useful and re-useable ontologies has led to different ontology creation proposals over the years. The project focuses on two major approaches, modeling using a top-level ontology, and the other is modular ontology modeling.

The traditional approach is based on top-level ontology, and the strategy is to utilize ontology that is comprehensive enough to cover a broad spectrum of domains through their universal terminologies. In this way, all domain ontologies share a common top-level formal ontology in which their respective root nodes can be defined, and hence consistency is assured across the knowledge graph. Nevertheless, the most recent approach is quite different and is a refinement of the eXtreme Ontology Design methodology based on the ontology design patterns. Whole ontology is viewed as a collection of interconnected modules, and modules are developed around the classified fundamental notions according to experts' terminology or the use-case. Having developed modules in a fashion of divide and conquer, these modules are shareable and reusable among some other ontology if needed, and consequently, the ontology being FAIR is justified (findable, accessible, interoperable, and reusable).

Although, it has been argued that there are advantages to either paradigm, it is possible to have a combination of both approaches mentioned earlier, depending upon the use-case or the preferences of the ontology engineers. We provide an extension to the Protégé - based modular ontology engineering tool CoModIDE, in order to make it possible for ontology engineers to follow traditional, ad-hoc ontology modeling approach, alongside more modern paradigms such as modular ontology engineering. The project focuses on domain-level ontology developers or organizations dealing with ontology development, which may get help through the plugin in minimizing the tooling gap to unite paradigms and develop robust,

flexible ontologies suitable to their needs. As a bridge between the more recently proposed modular ontology modeling approach and more classical ones based on foundational ontologies, it enables a best-of-both-worlds approach for ontology engineering.

Table of Contents

List of Figures

List of Tables

Acknowledgments

This work was supported by the U.S. Department of Commerce, National Institute of Standards and Technology, under award number 70NANB19H094.

I am especially indebted to my major advisor Dr. Pascal Hitzler for believing in my abilities and for his constant support from my second semester here at Kansas State University. I am thankful to him for introducing me to the field of Data Semantics and Ontologies that equipped me with the knowledge to implement this project.

I am also grateful to Dr. Cogan Shimizu for being an amazing mentor during my semesters and throughout my project. I express my gratitude for time and effort spent by Dr. Pascal and Dr. Cogan on providing positive critique and suggestions that molded the outcome of the project and report.

I would like to express my sincere gratitude to my committee members Dr. Doina Caragea and Dr. Lior Shamir for taking the time to serve on my committee and their support in the process of this project.

I would like to extend a huge thanks to all my family members without whom this journey would not have been possible. And last, but in no way least, I would like to thank my friends for their support and friendship. The past two years would not have been so memorable and fun without all of them.

Dedication

I dedicate this project to my parents, Surender Singh Dalal and Yoginderi Dalal, and my little brother, Aryan Dalal, who have always been my greatest support and strength.

Chapter 1

Introduction

This chapter presents a brief overview of this project, starting with the problem statement with objectives and synopsis of this project. It gives an overview of relevant existing methods for the semantic web application task of ontology engineering and a justification for the experimental approach taken and the evaluation methods used.

1.1 Problem Definition and Objective

Nowadays, ontologies are achieving more reputation as there is a tremendous amount of data and knowledge worldwide. In many application areas, ontology modeling has become a primary approach to schema generation for data integration and knowledge graphs.[1–3]

Significant ontologies are built such that they are generic enough to serve different conceptual representations. They shall accommodate extensions towards additional related details (without knowing what these future extensions would exactly look like), questioning around ontology shall be possible. Lately, the policies of Findability, Accessibility, Interoperability, and Reusability (FAIR) have been formulated as essential goals that data receptacles should meet to enhance their data holdings' usefulness.[4] The quest for efficient approaches to model

useful and reusable ontologies has, over the years, led to different proposals for ontology creation processes and tooling.

One of the most classic approaches is based on so-called upper or foundational ontologies[5–7]. Central to this paradigm is to utilize ontologies that are generic and/or large and as such cover a wide swath of domains, such as BFO[5], DOLCE[8], SUMO[9]. In this approach to modeling, a new (domain) ontology is created in accordance with the mindset or structure conveyed by these upper or foundational ontologies. Technically, alignment of the domain ontology classes and relations to the upper/foundational ontology entities – meaning creating appropriate sub-class and sub-property relationships so that relevant structure and/or axioms are inherited – play a prominent role.[3]

Another, more recent approach to ontology modeling is based on an apparently quite different mindset: Modular ontology modeling[10] is based on the idea that an ontology may best be viewed as a collection of interconnected *modules*, each of which correspond to a key notion according to the terminology used by a domain expert. The approach is related to other recent proposals to approach ontology modeling in a *divide and conquer* fashion[11;12] and is a refinement of the eXtreme Ontology Design methodology[13] based on Ontology Design Patterns[14]. In its original conception, and the corresponding tooling, in particular the CoModIDE Protégé plug-in[10], the approach de-emphasizes sub-class and sub-property relationships, and in particular does not account for upper or foundational ontologies.[3]

However, approaches considered beforehand seem to be quite beneficial in having efficient ontologies; unfortunately, they have some shortcomings when practiced individually. Manually aligning ontologies is labor expensive, time-consuming, error-prone, and does not leave room for having self-sufficient reusable modules in the model as they are created in accordance with the mindset or structure conveyed by these upper or foundational ontologies. Also, merging different organizations' ontologies requires a high degree of cooperation, standardization, and commitment. Everyone engaged in the project has to agree about, for

example, what the resulting ontology will look like, which ontology to be used as top-level. On the other hand, we already have a collection of independent domain ontologies. Considering the increasing usability of domain ontologies, multiple groups of researchers are creating incompatible domain ontologies focused on their specific local needs, resulting in new information silos, additional problems of inaccessible, non-shareable data, and non-optimal use of resources.

This leads to the question of what can be done to improve the reusability of the knowledge graph as a whole, and this is the overarching research question. First, we posit it is possible to use a combination of modular ontology modeling and upper ontology modeling. Combining best of both worlds can help ensure consistent development of ontologies across multiple domains, in much the same way that all engineering students share knowledge of the basic sciences helps ensure that they can communicate with each other as they acquire different sorts of specialist knowledge. It will increase the flexibility of training; it will allow more effective governance and quality assurance of ontology development, and it will promote the degree to which multiple different groups of ontology developers and users can inspect and critique.

Therefore, rather than modifying either of the methodologies or saying which one is best, we provide a plugin that makes both the approaches compatible. We extend CoModIDE with additional functionality, the upper alignment tool that incorporates alignments with upper or foundational ontologies (or any ontology) along with modular ontology engineering. We initially suppose CoModIDE, a graphical paradigm is sufficient for modeling modular ontologies. As CoModIDE does not have alignment functionality already, a user would need to use the default protege experience to load, identify, and align classes and properties from an upper (or any other) ontology. Thus, we hypothesize that top-level ontology alignment with any modular ontology using the plugin developed is comparatively easier than doing alone with Protege. The plugin enables an engineer to pick the best of both worlds. It has been developed keeping domain-level ontology developers or organizations dealing in ontol-

ogy alignment, which may get help through the plugin to minimize the tooling gap to unite paradigms and develop robust, flexible ontologies suitable to their needs.[3]

1.2 Implementation and Objectives

This project aims to have a tool that allows users to load an ontology – which may be an upper or foundational ontology. The project also aimed at studying its usability and impact on the correctness of the results. The UOA tool is a view for Protégé; however, the view's functionality is dependent on the CoModIDE. The view extracts all of the classes and properties (excluding annotation properties) from the loaded ontology. The view will automatically construct and add the pertinent SubClass or SubProperty axioms to the ontology by selecting, or un-selecting, the checkboxes next to these entities. CoModIDE detects these additions and will display the added relationships.

For user study evaluation, participants will be asked to perform modeling tasks. Results will be evaluated based on the study of a few metrics. Factors to be considered are the time taken for the tasks, how difficult or easy it was to learn the system and the result's correctness. There is no prerequisite to the study in terms of knowledge in aligning modular ontology with foundational ontologies. Variables to be studied are experience with ontology modeling, familiarity with top-level ontology family, experience and knowledge about Protege and ComodIDE tools, time is taken to complete both the tasks, the correctness of the produced result, subjective assessment. Usability of aligning functionality(classes/properties with the model) of the software tool.

1.3 Synopsis

The rest of the report is organized as follows. Section 2 comprises a review of the supporting literature. It will discuss ontology engineering strategies, advantages of employing them, sustaining tools, or methodology. Section 3 presents our Upper Ontology Alignment Tool in detail, and the study design used to evaluate the plugin. Variables and metrics will be discussed in detail used for the evaluation. In sections 4 and 5, we discuss the results or findings from the usability study and conclude our work.

Based on our research work, software demo-paper ***Modular Ontology Modeling Meets Upper Ontologies: The Upper Ontology Alignment Tool*** *is to appear at the* ***19th ISWC 2020*** *(International Semantic Web Conference).*

Chapter 2

Literature Review

The intention behind having a tool(UOA) that supports the interactive visual alignment of ontologies with modular ontology methodology is to enhance ontology engineers experience with merging modular ontology with foundational ontologies. We have recognized a few tools that support visualization of ontology merging, along with that will also summarize methodologies/ tools that contribute to having modular ontology and how our work continues on them.

2.1 eXtreme Design

eXtreme Design (XD)[15] was initially proposed to emphasize waterfall methodologies in ontological engineering to introduce a new, more flexible thinking in ontological engineering. It was originally inspired by software engineering methods such as eXtreme Programming (XP)[16] and the experience factory approaches[17]. With the growth of the ontology, instead of a one-time process, an emphasis is placed on the iterative delivery approach to success. XD is denoted as task-focused, promotes pair design, believes in reusing ontological best practices via ODP.

The methodology can be divided into three parts. (1) a project initiation and definition phase that is executed only once at the beginning of the project (however may have to be

revisited if requirements change). It is about collecting requirements; for requirements to be realistic, they must also be based on stories that come directly from customers, ie, they were not conceived by ontology engineers. Because domain knowledge is essential, it is equally necessary to involve domain experts as customers in the development process to confirm the correctness of domain functionality and coverage and the adequacy of terminology and other non-functional requirements.

(2) XD emphasizes the divide and conquer paradigm that takes requirements piece by piece and create modules for each requirement by reusing ODP building blocks, ODP is adapted and integrated into the ontology module under development, and like this, all requirements are covered through a development loop that iteratively produces new modules. The module is tested against the selected requirements to ensure that it covers them adequately.

(3) The methodology is iterative and creates the end result step by step. This means that the methodology provides a tangible result in the initial phase and then extends this result with each iteration. As soon as all requirements have been managed, the module is released and integrated into the overall solution. XD have been classified as requirement-driven, inherently modular methodology which focuses on creating reusable modules and play a part to reduced failures in ontologies[18;19]. However, the findings also indicate that pitfalls are associated with the possibility of over-reliance on ODP projects, as discussed in[20].

2.2 Ontology Design Patterns.

Gangemi[21], and Blomqvist and Sandkuhl[22] introduced Ontology Design Models (ODP) in 2005 to simplify ontology development. ODPs are designed to guide unskilled users by consolidating best practices into reusable building blocks that these users accommodate and specialize in individual ontology development projects. Presutti et al.[23] define a typology of ODPs, including reasoning patterns, naming, transformation, etc. The eXtreme Design methodology[15] describes how ontological engineering projects can be broken down into dis-

crete sub-tasks to be solved using ODP. Previous studies have shown that ODP's use can reduce the number of modeling errors and inconsistencies in ontologies and that they are found to be useful and helpful by users.[18;19]

2.3 CoModIDE

CoModIDE[24] has been developed as a plug-in for the versatile and conventional Protege Ontology engineering environment. The plug-in presents three Protege views and a tab that stores these views. The Schema Editor view provides a graphical overview of the structure of the ontology, including ontology classes, their subclass relationships, and the object type and data type properties of the ontology that associate these classes with data types. All of these objects can be graphically edited by dragging and dropping. The pattern library view offers a number of integrated ontology design patterns from various projects and from the ODP Community Wiki[1]. The user can drag and drop design models from the library to the drawing area in order to display these models as modules in their ontology. In the configuration view, the user can set the behavior of other CoModIDE views and their components.

When a pattern is dragged over to the canvas, the constructs of that pattern are copied into the ontology (optionally by updating their IRIs to match the target ontology's namespace). But, they are also annotated using the OPLa vocabulary to indicate that they belong to a module based on a particular pattern and which model this module implements. In this way, the module origination is preserved, and the modules can be controlled (folded, unfolded, deleted, commented) as required.

However, the approach de-emphasizes sub-class and sub-property relationships, and in particular does not account for alignment with upper or foundational ontologies.

[1] http://ontologydesignpatterns.org/

2.4 Prompt-Viz

Prompt-Viz[25] is a visualization tool for Protege Prompt[26] plug-in that extends PROMPT-Diff[27] with information visualization techniques to provide advanced cognitive support for understanding the differences between versions of ontologies. It provides one single visual representation of ontologies merged utilizing histograms coloring mode within a treemap layout[28]. This visualization aims to provide users the ability to determine the Location, Impact, Type, and Extent (LITE questions) of the changes that have occurred to the ontology. The histograms bars represent the percentage of descendants classified as unchanged, appended, removed, moved-from, moved-to, and directly edited, respectively. It is divided into four linked frames:[29]

(1) an expandable horizontal tree layout of the ontology showing the differences; it contains a search tool to locate specific concepts quickly.
(2) a treemap layout of the ontology installed in a zoomable user interface;
(3) a path window shows the location of currently selected concepts in the ontology within the is-a hierarchy and serves as a navigation aid for the treemap component. The treemap view can be zoomed in to show all of the boundaries for each route's level.
(4) a comprehensive list of changes that have happened to the currently chosen concept, for instance, the classification of the change, the change procedure, and the reference frame in the previous and new versions of the ontology.

Prompt-Viz does not show sufficient detail about the source ontologies. With Prompt-Viz, the user can visualize the result of two merged ontologies. However, when examining the visualization, it is not easy to understand why specific merge actions occurred and its consequences. While this tool offers an exciting approach to visualization, it loses some intuitive aspects of a graphical visualization (e.g., hierarchical relationships between concepts). Using a single visualization to represent the two ontologies, the properties of the source ontologies lose their clarity, which can be sufficient to merge, but prevents the alignment of the ontologies.

2.5 OWL Lite Alignment

OLA (OWL Lite Alignment), a self-contained program, follows the similarity-based paradigm. It relies on a universal measure to compare the entities of two ontologies that homogeneously combine the totality of knowledge used in entity descriptions. The measure is calculated using an iterative process based on fixed points that subsequently approach the target solution. Use graphical visualizations to represent ontologies; specifically, an extended JGraph API is used. The graph structure of the OLA makes explicit the relationships between the elements of the language, e.g., if class c refers to another class c 'through an owl:allValuesFrom restriction, a labeled path is displayed between the corresponding nodes in the OL-graph, so that the connection between the two classes is perceived intuitively. In addition to general subclass relationships, the user can enable the display of edges between reverse, symmetrical, or transitive objects.

OLA does not have the overview and zoom mechanisms required for the interactive visualization of ontologies. OLA shows all details about the concepts, properties, and instances of the ontology simultaneously without an overview function. If there are more than 20 entities, a relatively small number, the user will only see parts of the graphs, and the context will be lost. Currently, the view is limited to displaying one ontology at a time. For the alignment visualization, it is expected that two of these graphics to be displayed side by side. Too many overlapping margins and labels are the current challenges for OLA. However, the main problem is that the diagram becomes cluttered as one view covers all the details without the zoom function.

UOA(Upper Ontology Alignment) tool draws influence from all of these works and incorporates the alignment of ODP-based modular ontology modeling with foundational ontology leveraging graphical visualization.

Chapter 3

Methodology

This chapter presents our Upper Ontology Alignment tool in detail and the study design used to evaluate the plugin, starting with the motivation[10] behind having plugin in the first place. It gives a detailed explanation for the implementation, which will describe the pluginś layout and features. It will discuss the study used to evaluate the plugin, its design, metrics, and variables used to conclude the studyś result.

3.1 Motivator

The Upper Ontology Alignment plugin is intended to simplify ontology engineering for users who are not experts in alignment, such as domain-level ontology developers or organizations dealing with ontology alignment. Experience indicates that these non-specialists rarely need to delve into alignment relationships, techniques, and approaches. Instead, they usually, at least initially, want to model relatively simple aligned semantics using more frequently used alignment relationships, such as equivalence or subsumption relationships. Usually, construction of modeling or even alignment is preferred to be done in a way that all involved participants, regardless of their level of ontological engineering skill, can understand and contribute graphically, whether on whiteboards, in vector drawing software, even on paper.

We have CoModIDE[24], a versatile and established Protégé ontology engineering envi-

ronment plugin. CoModIDE is an interface for directly authoring ontologies by drawing a schema diagram on a graphical canvas. It supports intuitive and agile visual modeling, directly outputting OWL ontologies and reusing ODPs to create and maintain ODP-based modules. In addition to this, it would help if there was a tool that continues to do

1. full integration with Protégé and CoModIDE along with leveraging the graphical user interface of CoModIDE and the creation of pattern-based modules,

2. easy loading of any ontology as a higher ontology and extraction of its concepts and relations, and

3. simple selection of checkboxes to define superior associations with modular ontology.

The tool's design criteria derive from the points discussed above. Therefore, Upper Ontology Alignment Tool.

3.2 Implementation

The Upper Ontology Alignment (UOA) tool[3] is an extension to the Protégé based modular ontology engineering tool CoModIDE. The plugin provides four Protégé views: schema editor, pattern library, configuration view, UOA, and a tab that hosts these views. The UOA is the bottom right view labeled as 1.(see figure 3.1) Classes are represented as cells, and properties are represented as edges. The highlighted box shows the resultant alignment model with the GFO.owl file. Figure 3.2 and 3.3 shows the zoom-in user interface showing the list of classes and properties.

Since UOA provides additional functionality to CoModIDE, one of the leading design criteria is that it must be compatible and support the graphical representation of an ontology created with CoModIDE based on the model of classes, properties, and data types. It should be consistent across all reboots, instruments, and operating systems or versions of Protégé.

The UOA view allows a user to load an ontology file using a load button – which may be an upper or foundational ontology – directly into the view (which is kept isolated from the

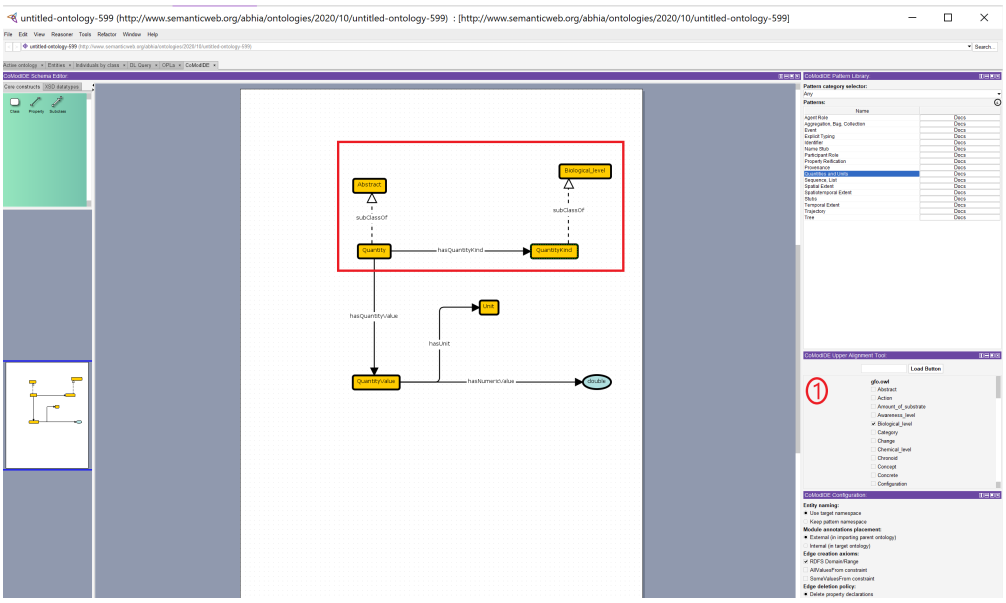

Figure 3.1: CoMOdIDE plugin views- UOA, schema editor, pattern library and configuration view

ontology active in Protégé). The view extracts all of the classes and properties (excluding annotation properties) from the loaded ontology. The user then selects classes (cells) or object/data properties (edges) on the graphical canvas. The UOA tool then displays the pertinent entities depending on which glyph is selected on the graphical canvas. The view will automatically construct and add the pertinent SubClass or SubProperty axioms to the ontology by selection or deselection of the checkboxes next to these entities. CoModIDE detects these additions and will display the added relationships.

Besides, the view provides some supporting functionality for ease and clarity of use: the view will display the currently selected entity, automatically select checkboxes for axioms that are already present in the ontology (e.g., if some entity is already a subclass of the Perdurant class, that particular checkbox will be selected), will display the currently loaded ontology file name, allows for different ontologies to be loaded (i.e., a user is not limited to a single upper ontology), and provides descriptive logging in the case of failure.

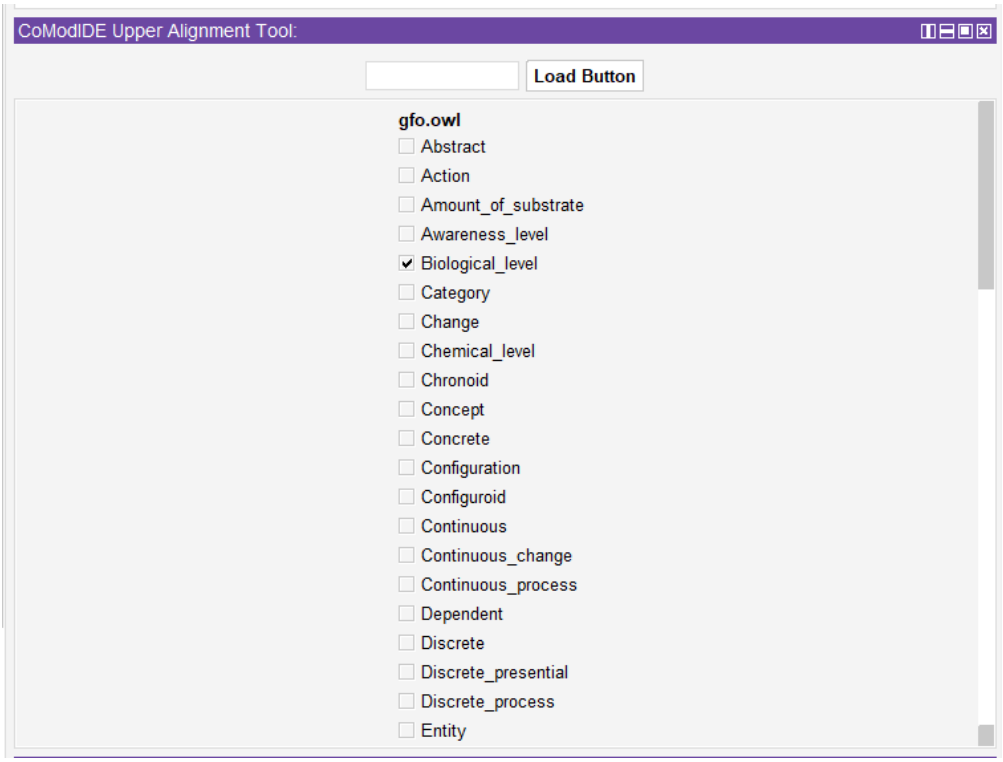

Figure 3.2: Checklist for classes when a "cell" is selected in the graphical canvas.

3.3 User study Design

User study experiment consists of four phases: a questionnaire survey to collect necessary data on the subject (familiarity with ontological knowledge and tools), two modeling tasks, and a follow-up questionnaire survey to collect information on the usability of Protégé and UOA. The tasks were designed to imitate a standard ontological engineering process in which a conceptual design is developed and approved through whiteboard prototyping. A developer is then tasked with linking the specified schema diagram with a higher ontology in an OWL ontology.

During each modeling task, participants are asked to create an appropriate and correct OWL file for the proposed tasks. To avoid a learning effect, the two tasks use two different schematic diagrams; one is an instance of the pattern, and the other is a small model. The specific order in which the user used which tool first and next was randomized among participants (some used Protégé first for the first activity and others used UOA) to avoid bias differences in the activity's complexity. The precision of the developed OWL files and

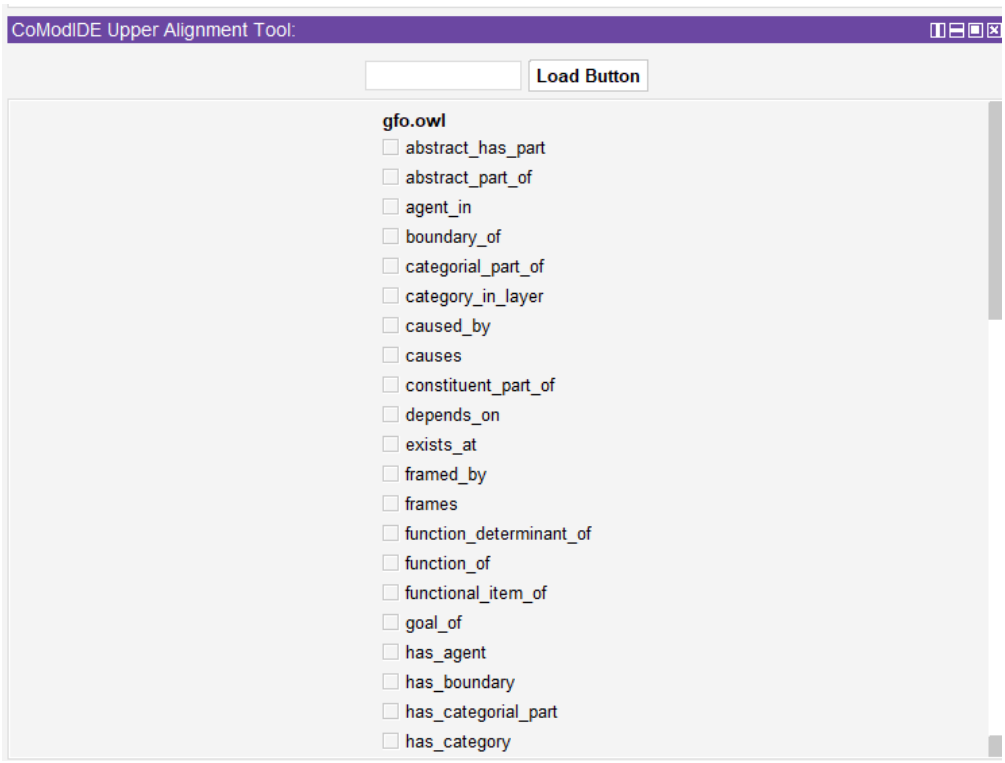

Figure 3.3: Checklist for properties when a "edges" is selected in the graphical canvas.

the time required to complete each task were recorded (the latter being limited to 20 minutes per task). Each step of the study has been explained below.

3.3.1 Introductory Tutorial

We intend to improve ontology modelingś approachability by making it more accessible to those without knowledge engineering expertise. As such, there were no prerequisites for participants with top-level ontological familiarity or alignment knowledge for this assessment. However, in order to establish a common basic understanding for the basic concepts of modeling, we provided a 10 min tutorial explaining ontologies concepts, top-level ontologies, classes, properties, domains, and ranges, sub-class relations.

3.3.2 Prior-Questionnaire Survey

The idea of a prior questionnaire survey was to collect information relating to the participantś basic level knowledge and experience with topics related to ontology modeling or alignment of ontologies to be used as control variables in later evaluation. We also asked the participants if they belong to Computer Science Background or Non-CS Background; this will normalize the user study range. We used a 5-point Likert scale for the rating of the following statements.

1. I have done ontology modeling before.
2. I am familiar with Ontology Design Patterns.
3. I am familiar with Manchester Syntax.
4. I am familiar withTop-level ontology.
5. I am familiar with Protégé.
6. I am familiar with the CoModIDE pattern library.

In the end, we asked the participants to describe their relationship to the test leader, (for example like a student, colleague, same research lab, not familiar).

3.3.3 Modeling Task A

In task A, there are two parts. Participants were asked to develop an ontology to model an event module for an organization and then align the entities or properties with an identified upper-level ontology. The scenario identified the following design pattern to use – Event, and the foundational ontology - GFO. For the first part, participants were asked to perform the modeling task using the Upper Ontology Alignment Tool, and for the second, they were asked to do it using Protégé alone. Figure 3.4 shows how the resultant ontology will look alike (highlighted part shows the alignment); after the event pattern's alignment with the foundational ontology, participants could take reference from the schema diagram.

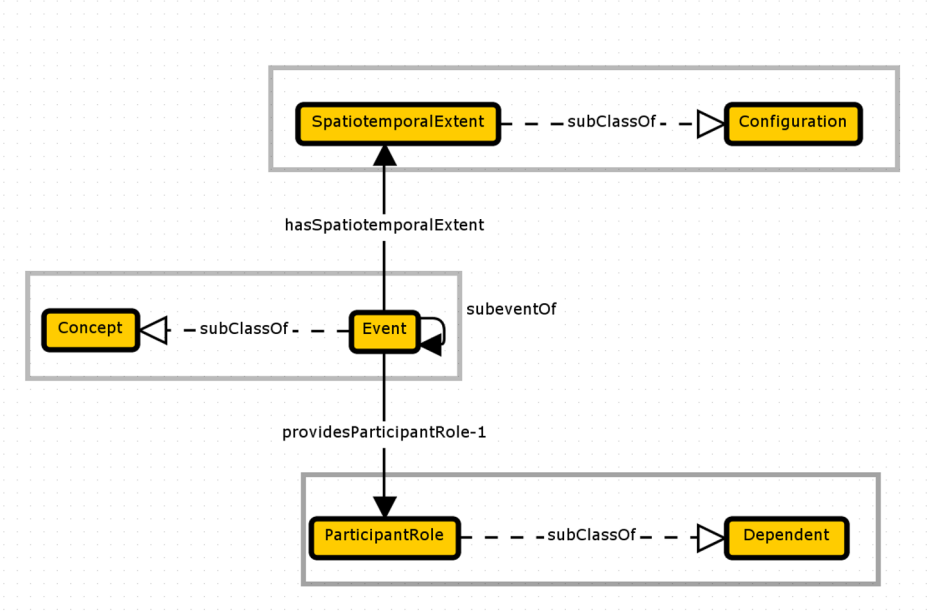

Figure 3.4: Schema Diagram showing the results of aligning the Event-pattern with GFO.

3.3.4 Modeling Task B

Similarly, for task B, there are two parts, participants were to develop an ontology to capture the selling and buying of dogs. The scenario needs to capture the event and role for dog purchase and later align the model with an identified foundational ontology. For the first part, participants were asked to perform the modeling task using the Upper Ontology Alignment Tool, and for the second, they were asked to do it using Protégé alone. Figure 3.5 shows how the resultant ontology will look alike (highlighted part shows the alignment); after the model's alignment with the foundational ontology, participants could take reference from the schema diagram.

3.3.5 Follow-up Questionnaire Survey

The follow-up survey included the SUS evaluations for both Protégé and UOA (Upper Ontology Alignment Tool). The SUS is a ubiquitous "quick and dirty" yet reliable tool for measuring a system's usability. It consists of 10 questions, the responses of which are used to calculate an overall usability score from 0 to 100.

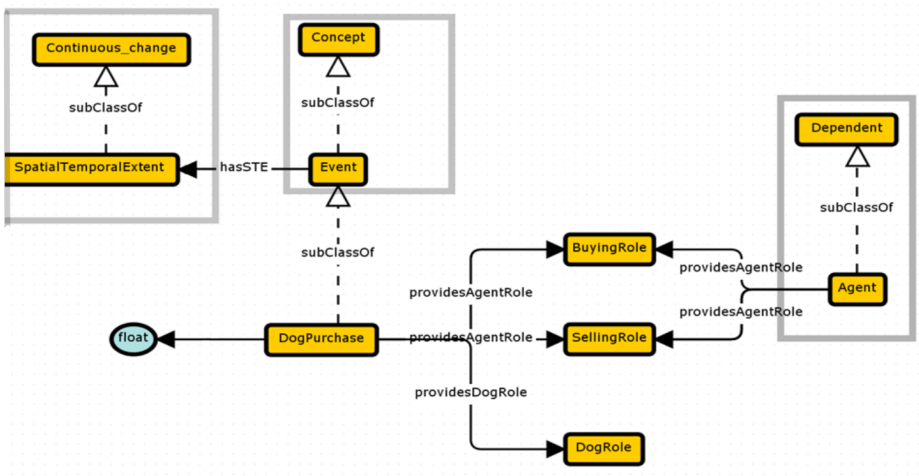

Figure 3.5: Schema Diagram showing the results of aligning the given model with GFO.

1. I think that I would like to use this system frequently.
2. I found the system unnecessarily complex.
3. I thought the system was easy to use.
4. I think that I would need the support of a technical person to be able to use this system.
5. I found the various functions in this system were well integrated.
6. I thought there was too much inconsistency in this system.
7. I would imagine that most people would learn to use this system very quickly.
8. I found the system very cumbersome to use.
9. I felt very confident using the system.
10. I needed to learn a lot of things before I could get going with this system.

The purpose of the selected questions was to capture the tool's learnability, feasibility, effectiveness, and efficiency, which are the main components of the usability goals. Additional

information on the SUS and its included items can be found online.[1] Additionally, we inquire about UOA-specific features. These statements are also scored using a Likert scale.

1. It was easy to find the classes from the loaded ontology.
2. It was easy to add classes to the ontology.
3. It was easy to do the subClassOf axioms.
4. I used the documentation to understand what I was doing.
5. It was easy to verify the correctness of the result produced.
6. It was easy to re-load ontology, if needed

However, this data has not been used in our evaluation, except to evaluate additions and inform our future work, as described in Section- Conclusion and Future Work. In the end of the survey, there was space for the participants to put any free comments on UOA's features or their experience with the tool.

With the above mentioned user-study design and implementation, we would like to show that UOA improves the approachability of knowledge graph development by verifying that users take less time to produce correct and reasonable output by using UOA than when using Protege; also, the user will find UOA to have a higher SUS score.

[1] https://www.usability.gov/how-to-and-tools/methods/system-usability-scale.html

Chapter 4

Results

This chapter includes the results of the experiments mentioned in Chapter 3 Section 3.3 and the evaluation metrics used to evaluate the performance of those machine learning models.

4.1 Participant Distribution

The total number of subjects who participated in the user study evaluation was 21, out of which 13 stated that they knew the author. At the same time, the rest did not report any such a relationship. The user study was not limited to subjects having a CS background to have diversity and capture usability goals across different backgrounds. Though the number of such subjects was not significant, few belong to background other than CS like entomology, agriculture engineering, and biochemistry. For self-presented ontological engineering knowledge, the answers are shown in Table 4.1. The responses differ significantly, with a relative standard deviation (σ \ mean) of 59-71%.

4.2 Metric Evaluation

The metrics that we considered for the result calculation are as follows:-

– Time Taken: number of minutes for each modeling task to run were recorded and

Table 4.1: Mean, median, standard deviation and relative standard deviation responses to a prior questionnaire

	mean	median	σ	relative σ
CV1: I have done ontology modelling before	2.14	1	1.46	68%
CV2: I am familiar with Ontology Design Patterns	2.14	2	1.35	63%
CV3: I am familiar with Manchester Syntax	1.52	1	1.03	68%
CV4: I am familiar with Top-level Ontology	1.76	1	1.04	59%
CV5: I am familiar with Protégé	2.24	1	1.58	71%
CV6: I am familiar with CoModIDE pattern library	2.10	1	1.41	67%

rounded to the nearest full minute and limited to 20 minutes for a task to run due to practical limitations;

– Correctness: it refers to the structural accuracy of the output generated. The complete structurally correct file received 2 points on examining URIs, axioms generated, alignments, 1 point was awarded for a partially correct file (e.g., one or two incorrect linkages, incorrect axiom creation, labels); and 0 points for incorrect files(e.g., absence of axioms, alignment).

For the metrics defined, we calculated simple statistics through which data of each modeling task is described. Table 4.2a and 4.2b each show the mean, median, and standard deviation of time-taken, and the output's accuracy for each modeling activity through Protégé as well as UOA (Upper Ontology Alignment).

Also, we examine the effects of our control variables (CVs). This analysis is vital because it provides the context for the representation or bias of our dataset. The analysis is illustrated in Table 4.2c, where CV1-CV6 corresponds precisely to the questions asked during the prior questionnaire survey, as described in section 4. We calculated each CVś bivariate correlation between the sample data and the self-reported data in the survey. For the readerś convenience, we would like to briefly discuss the correlation concept, also known as the Pearson correlation coefficient.

Simple bivariate correlation is a statistical technique used to determine relationships between two different variables (i.e., X and Y) in a single value between -1 and +1. It shows

Table 4.2: Summary of statistics comparing Protégé and UOA.

	mean	median	σ
Protégé	17.29	18	4.11
UOA	13.81	15	4.76

(a) Mean, median and standard deviation of **total time-taken** to complete both modeling task.

	mean	median	σ
Protégé (task A)	0.71	1	0.78
Protégé (task B)	0.52	0	0.74
UOA (task A)	1.38	2	0.86
UOA (task B)	1.05	1	0.86

(b) Mean, median and standard deviation of **outputś correctness** for both modeling task.

	CV1	CV2	CV3	CV4	CV5	CV6
TT (P)	-0.26	-0.07	-0.36	-0.47	-0.23	-0.16
Corr(P)	0.05	-0.02	0.12	0.22	0.05	-0.03
TT (UOA)	0.05	0.18	-0.13	-0.15	0.10	0.18
Corr(UOA)	0.08	-0.03	-0.02	0.03	0.19	0.12

(c) Correlations control variables (CV) on the Time Taken (TT) and Correctness of Output (Corr-(task A or B)) for both tools Protégé (P) and UOA.

	CV1	CV2	CV3	CV4	CV5	CV6
SUS (P)	0.28	0.27	0.28	0.28	0.24	0.26
SUS (UOA)	-0.0009	0.02	0.01	0.10	0.13	0.01

(d) Correlations with control variables (CV) on the SUS scores for both tools Protégé (P) and UOA.

how much value of X will change when there is a change in Y value. Correlations between variables can be positive, negative, curvilinear, or non-existent. When the two grow together, there is a positive relationship between the variables. In the case of a negative correlation, the observations of one variable increase while the other decreases. Curved-line relationships are slightly more complicated than typical positive or negative relationships because this correlation has both positive and negative relationships. If there is no relationship between the variables, they are said to be unrelated or non-existent.[30]

We believe calculating correlation has a reasonable measure of impact on the effect, as our sample's limited size, is not suitable for partitioning. The partitions (based on the prior questionnaire survey responses) could have been tested in pairs for statistical significance. However, sadly, the partitions would have been too small to perform the proper statistical tests. However, we emphasize that the sample size strongly influences the correlation effects.

SUS scores are analyzed in the same way. The Table 4.2d shows our observed correlations of the SUS score for both the tools with our control variables whereas, Table 4.3 shows the mean, median, and standard deviation of the data set. When using the SUS scale, the maximum number of points is 100.

Table 4.3: Mean, median and standard deviation for SUS score of each tool. The maximum score is 100.

	mean	median	σ
Protégé	44.05	42.5	21.04
UOA	71.79	72.5	13.06

Lastly, we compare each metric (time taken and accuracy of output) for one tool against the other. In other words, we want to know if our results are statistically significant - as can be seen from the statistics in Table 4.2, UOA does certainly perform better for evaluations of both the metrics and SUS. To make the comparison, we calculate the probability p for the null hypothesis that the samples in each dataset come from different underlying distributions. A standard tool that is used here is the paired (two-tail) T-Test. It is assumed that it is a powerful tool for quantitative analysis of limited data size and that the underlying data are widely distributed. The threshold p-value that is generally considered for indicating assurance that the difference is significant is $p < 0.05$. Table 4.4 summarizes these results.

Table 4.4: Significance of results.

Time-taken	Correctness (task-A)	Correctness (task-B)	SUS Evaluation
$p \approx 0.010 < 0.05$	$p \approx 0.004 < 0.05$	$p \approx 0.012 < 0.05$	$p \approx 0.0000015 < 0.001$

4.3 Additional Free-Text Responses

11 of the 21 participants decided to leave free-text comments in the comment field, which are available to participants at the end of the questionnaire. We applied qualitative coding

and analysis based on the fragments to these comments. We divide the comments based on the line breaks entered by the topics, read the details, and create a simple category chart. We then re-read the fragments and applied these categories to the fragments (with at most one category per fragment)[31]. Participants left 2-5 fragments for a total of 35 fragments to be analyzed, 25 of which were encoded, as shown in Table 4.5.

Table 4.5: Free text comment fragments per category

Category	Fragments no.
User-Interface	5
Tree Structure Layout	2
Bugs	5
Additional features	4
Valuable statements	9

Chapter 5

Discussion

This chapter reviews the finding of the experiment mentioned in Chapter 4 and presents a conclusion based on those findings or calculations.

5.1 Participant Distribution

The data show no correlation (bivariate correlation $\leq \pm 0.1$) between the reported familiarity of the subjects and the reported SUS values; for example, this would have happened if the subjects who knew the author were biased. The high relative standard deviation of knowledge level responses from the prior questionnaire survey shows that our subjects are very diverse in skills. In other words, they are not entirely made up of a limited-experience class or from a particular background of users that UOA will hopefully support at some point. This variation and diversity of education help us evaluate and compare the user's performance and the toolś usability more impartially.

5.2 Metric Evaluation

To analyze the correlations coefficient between our results and the control variables score collected in the prior-survey, we have used threshold values presented through the rule of thumb for the correlation coefficients |r|: 0 to 0.19 very weak(-0.19 to 0), 0.20 to 0.39 weak(-

0.39 to -0.20), 0.40 to 0.59 moderate(-0.59 to -0.40), 0.60 to 0.79 strong(-0.79 to -0.60), 0.80 to 1.00(-1.00 to -0.80) very strong.

The correlation coefficient estimates the strength and direction of the relationship between the variables. Strength is examined through the coefficient's absolute value, whereas the sign represents the relationship's direction. As depicted through the Table 4.2c, the metric time-taken used to complete tasks using the tool Protégé correlates negatively with each of the control variables (taken from the prior survey), and the strength of the relationship varies from very weak to moderate. In contrast, the accuracy metric correlates weakly positively as the absolute value ranges from 0 to 0.22 except CV2 and CV6. Analysis for Protégé indicates that familiarity with ontology modeling, top-level ontology, related concepts, and the tool decreases the time required to finish the task and, to any degree, improves the output's accuracy.

However, for the metrics (time-taken and output's correctness) concerning UOA, the relationship's strength and direction are dubious since there are only very weak correlations with control variables varying from 0-0.19. We may interpret that familiarity with ontology modeling does not have much influence and that performance when using UOA is mostly skeptical of the studyś control variables.

UOA reports having better scores when considering the mean and median for the metric time-taken as described in the Table 4.2a. When examining the underlying data (4.4), the significance of the p-value $\approx 0.010 < 0.05$. Subsequent, consider Table 4.2b for the correctness of both the tasks, UOA performs better for both mean and median score than Protégé. Comparing underlying data for correctness, the statistical significance of the p-value is $p \approx 0.004 < 0.05$ and $p \approx 0.012 < 0.05$.

Considering both the comparisons, we reject the null hypothesis and confirm that a **a user produce correct and reasonable output in less time when using UOA than when using Protégé alone.**

From the above analysis of the correlation coefficient where we observe a very weak correlation between the familiarity of ontology modeling and UOA performance results and the confirmation that the user performs better in terms of time required and outputś accu-

racy when using UOA rather than Protégé, it indicates that UOA has delivered increased accessibility and learnability.

Further, Table 4.3 illustrates that the SUS scores for UOA have a greater mean, greater median, and shorter σ, attaining a substantial statistical significance of p-value $\approx 0.0000015 <$ 0.001. Hence, we confirm that the **user finds UOA to have a higher SUS score than when using Protégé alone** from the evaluation. On examining SUS scores (Table 4.2d), evaluation states that Protégé correlates strongly positively by control variables. The absolute values indicate that subjects do not find the tool very useful in terms of usability goals, including adequate to use, easy to learn, and suitable. In contrast, the SUS correlation coefficient for UOA suggests that there is either a very weak or weak correlation with the CVs.

By proving that in less time, users produce accurate output when using UOA and users find UOA to have higher scores, we can say that UOA improves usability and approachability to the knowledge graphś development, especially for those unfamiliar with ontological modeling concerning our participantś distribution.

5.3 Additional Free-Text Responses

The fragments summarized in Table 4.5 paint a little consistent picture of the advantages and disadvantages of UOA recognized by subjects as follows:

– *User Interface*: The UOA tool's design format is confusing and less-informative; the button used for loading files into the tool does not serve the purpose much and reduces approachability.

– *Tree Structure Layout*: The users find the view crowded and uncomfortable, not easy to find the classes or properties down in the list.

– *Bugs*: Graphical display on canvas is faulty; checking boxes stopped adding alignments to the model.

– *Additional Features*: There should be a search box to find the classes/properties from the list; there should be prompts for the user in case of error; zooming is requested.

– *Valuable Statements*: Users appreciate graphical modeling with the additional feature of alignment with upper ontologies. e.g. *"The Upper Ontology Alignment tool made it much easier to add classes with specific sub-class axiom relations", "This system is very useful and easier to use", "The tool efficiently reduces the manual steps and easy to use. I loved the concept and would highly rate it."*

Some users opt to leave comments related to their performance in the experiment or knowledge about the tool, hence containing no codable fragments.

We find that there is almost an agreement among participants that UOA adds some value to graphical modeling and is intuitive and useful.When users criticize the UOA tool, these critiques are usually aimed at specific and more simple bugs or missing UI functionality.

Chapter 6

Conclusion and Future Work

Finally, we showed how the UOA tool allows ontology engineers to develop ontologies with the option of combining modular ontology modeling with modeling approaches based on upper/foundational ontologies, more correctly and faster than Protégé, irrespective of their previous knowledge level with alignment. That UOA is more user-friendly and has improved usability goals (SUS score) than the standard Protégé and that UOA concerns affecting users, as opposed to methodological or modeling problems, mainly derive from simple faults in the tool. Overall, this means that the modular graphical ontological engineering with alignment to upper ontology using the tool is a practical way to improve ontological engineering accessibility.

6.1 Future Work

We have come across many points through the user study evaluation that may help increase the toolś approachability and usability. Possible extensions of this work include:
-automatization or semi-automatization of upper alignments
-provisions of more complex alignment capabilities beyond sub-classes or sub-properties
-enhancement of namespace prefixes or labels
-introducing a search box and improvising the tree structure of the classes/properties.[3]

Bibliography

[1] Cogan Shimizu, Pascal Hitzler, Quinn Hirt, Dean Rehberger, Seila Gonzalez Estrecha, Catherine Foley, Alicia M. Sheill, Walter Hawthorne, Jeff Mixter, Ethan Watrall, Ryan Carty, and Duncan Tarr. The Enslaved Ontology: Peoples of the historic slave trade. *J. Web Semant.*, 63:100567, 2020.

[2] Randi Vita, James A. Overton, Christopher J. Mungall, Alessandro Sette, and Bjoern Peters. FAIR principles and the IEDB: short-term improvements and a long-term vision of OBO-Foundry mediated machine-actionable interoperability. *Database*, 2018:bax105, 2018.

[3] Abhilekha Dalal, Cogan Shimizu, and Pascal Hitzler. Modular ontology modeling meets upper ontologies: The upper ontology alignment tool. In *The 19th International Semantic Web Conference*, volume 2721, pages 119–124, 10/2020 2020.

[4] Mark D Wilkinson, Michel Dumontier, IJsbrand Jan Aalbersberg, Gabrielle Appleton, Myles Axton, Arie Baak, Niklas Blomberg, Jan-Willem Boiten, Luiz Bonino da Silva Santos, Philip E Bourne, et al. The fair guiding principles for scientific data management and stewardship. *Scientific data*, 3, 2016.

[5] Robert Arp, Barry Smith, and Andrew D Spear. *Building ontologies with basic formal ontology*. Mit Press, 2015.

[6] Daniel Oberle, Anupriya Ankolekar, Pascal Hitzler, Philipp Cimiano, Michael Sintek, Malte Kiesel, Babak Mougouie, Stephan Baumann, Shankar Vembu, and Massimo Romanelli. DOLCE ergo SUMO: on foundational and domain models in the SmartWeb Integrated Ontology (SWIntO). *J. Web Semant.*, 5(3):156–174, 2007.

[7] Barry Smith. Classifying processes: an essay in applied ontology. *Ratio*, 25(4):463–488, 2012.

[8] Aldo Gangemi, Nicola Guarino, Claudio Masolo, Alessandro Oltramari, and Luc Schneider. Sweetening ontologies with DOLCE. In Asunción Gómez-Pérez and V. Richard Benjamins, editors, *Knowledge Engineering and Knowledge Management. Ontologies and the Semantic Web, 13th International Conference, EKAW 2002, Siguenza, Spain, October 1-4, 2002, Proceedings*, volume 2473 of *Lecture Notes in Computer Science*, pages 166–181. Springer, 2002.

[9] Ian Niles and Adam Pease. Towards a standard upper ontology. In *2nd International Conference on Formal Ontology in Information Systems, FOIS 2001, Ogunquit, Maine, USA, October 17-19, 2001, Proceedings*, pages 2–9. ACM, 2001.

[10] Cogan Shimizu, Karl Hammar, and Pascal Hitzler. Modular graphical ontology engineering evaluated. In Andreas Harth, Sabrina Kirrane, Axel-Cyrille Ngonga Ngomo, Heiko Paulheim, Anisa Rula, Anna Lisa Gentile, Peter Haase, and Michael Cochez, editors, *The Semantic Web – 17th International Conference, ESWC 2020, Heraklion, Crete, Greece, May 31-June 4, 2020, Proceedings*, volume 12123 of *Lecture Notes in Computer Science*, pages 20–35. Springer, 2020.

[11] David Osumi-Sutherland, Mélanie Courtot, James P. Balhoff, and Christopher J. Mungall. Dead simple OWL design patterns. *J. Biomedical Semantics*, 8(1):18:1–18:7, 2017.

[12] Martin G. Skjæveland, Daniel P. Lupp, Leif Harald Karlsen, and Henrik Forssell. Practical ontology pattern instantiation, discovery, and maintenance with reasonable ontology templates. In Denny Vrandecic, Kalina Bontcheva, Mari Carmen Suárez-Figueroa, Valentina Presutti, Irene Celino, Marta Sabou, Lucie-Aimée Kaffee, and Elena Simperl, editors, *The Semantic Web – ISWC 2018 – 17th International Semantic Web Conference, Monterey, CA, USA, October 8-12, 2018, Proceedings, Part I*, volume 11136 of *Lecture Notes in Computer Science*, pages 477–494. Springer, 2018.

[13] Valentina Presutti, Enrico Daga, Aldo Gangemi, and Eva Blomqvist. eXtreme Design with content ontology design patterns. In Eva Blomqvist, Kurt Sandkuhl, François Scharffe, and Vojtech Svátek, editors, *Proceedings of the Workshop on Ontology Patterns (WOP 2009) , collocated with the 8th International Semantic Web Conference (ISWC-2009), Washington D.C., USA, 25 October, 2009.*, volume 516 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2009. URL http://ceur-ws.org/Vol-516/pap21.pdf.

[14] Aldo Gangemi and Valentina Presutti. Ontology design patterns. In Steffen Staab and Rudi Studer, editors, *Handbook on Ontologies*, International Handbooks on Information Systems, pages 221–243. Springer, 2009.

[15] Eva Blomqvist, Karl Hammar, and Valentina Presutti. Engineering ontologies with patterns-the extreme design methodology. *Ontology Engineering with Ontology Design Patterns*, (25):23–50, 2016.

[16] James Shore et al. *The Art of Agile Development: Pragmatic guide to agile software development.* " O'Reilly Media, Inc.", 2007.

[17] Victor R Basili, Gianluigi Caldiera, and H Dieter Rombach. Experience factory. *Encyclopedia of software engineering*, 2002.

[18] Eva Blomqvist, Valentina Presutti, Enrico Daga, and Aldo Gangemi. Experimenting with extreme design. In *International Conference on Knowledge Engineering and Knowledge Management*, pages 120–134. Springer, 2010.

[19] Eva Blomqvist, Aldo Gangemi, and Valentina Presutti. Experiments on pattern-based ontology design. In *Proceedings of the fifth international conference on Knowledge capture*, pages 41–48, 2009.

[20] Karl Hammar. Ontology design patterns in use: lessons learnt from an ontology engineering case. In *Workshop on Ontology Patterns in conjunction with the 11th International Semantic Web Conference 2012 (ISWC 2012)*, 2012.

[21] Aldo Gangemi. Ontology design patterns for semantic web content. In *International semantic web conference*, pages 262–276. Springer, 2005.

[22] Eva Blomqvist and Kurt Sandkuhl. Patterns in ontology engineering: Classification of ontology patterns. In *ICEIS (3)*, pages 413–416, 2005.

[23] Valentina Presutti, Aldo Gangemi, Stefano David, G Aguado de Cea, MC Surez-Figueroa, Elena Montiel-Ponsoda, and M Poveda. Neon deliverable d2. 5.1. a library of ontology design patterns: reusable solutions for collaborative design of networked ontologies. *NeOn Project. http://www. neon-project. org*, 2008.

[24] Cogan Shimizu and Karl Hammar. Comodide–the comprehensive modular ontology engineering ide. In *ISWC 2019 Satellite Tracks (Posters & Demonstrations, Industry, and Outrageous Ideas) co-located with 18th International Semantic Web Conference (ISWC 2019) Auckland, New Zealand, October 26-30, 2019.*, volume 2456, pages 249–252. CEUR-WS, 2019.

[25] David Stephen John Perrin. Prompt-viz: Ontology version comparison visualizations with treemaps. 2004.

[26] Natalya F Noy and Mark A Musen. The prompt suite: interactive tools for ontology merging and mapping. *International journal of human-computer studies*, 59(6):983–1024, 2003.

[27] Natalya Fridman Noy, Mark A Musen, et al. Promptdiff: A fixed-point algorithm for comparing ontology versions. *AAAI/IAAI*, 2002:744–750, 2002.

[28] Ben Shneiderman. Tree visualization with tree-maps: 2-d space-filling approach. *ACM Transactions on graphics (TOG)*, 11(1):92–99, 1992.

[29] Monika Lanzenberger and Jennifer Sampson. Alviz-a tool for visual ontology alignment. In *Tenth International Conference on Information Visualisation (IV'06)*, pages 430–440. IEEE, 2006.

[30] Mike Allen. *The SAGE encyclopedia of communication research methods*. Sage Publications, 2017.

[31] Philip Burnard. A method of analysing interview transcripts in qualitative research. *Nurse education today*, 11(6):461–466, 1991.