

# **Ontology Model for Web Development Frameworks**



Thesis Submitted By:

**Afnan Siddique**

**01-243162-001**

Supervised By:

**Dr. Muhammad Asfand e Yar**

*A dissertation submitted to the Department of Computer Science, Bahria University, Islamabad as a partial fulfillment of the requirements for the award of the degree of Masters in Computer Science*

**Session Spring 2019**



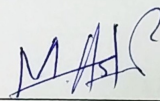
**Bahria University**  
Discovering Knowledge

**MS-13**

### **Thesis Completion Certificate**

It is to certify that the above student's thesis has been completed to my satisfaction and, to my belief, its standard is appropriate for submission for Evaluation. I have also conducted plagiarism test of this thesis using HEC prescribed software and found similarity index at 18% that is within the permissible limit set by the HEC for the MS/MPhil degree thesis.

I have also found the thesis in a format recognized by the BU for the MS/MPhil thesis.

**Principal Supervisor's Signature:** \_\_\_\_\_ 

**Date:** 25-06-2019 **Name:** Dr. Muhammad Asfand e Yar



**Bahria University**  
Discovering Knowledge

**MS-14A**

**Author's Declaration**

I, **Afnan Siddique** hereby state that my MS thesis titled **"Ontology Model for Web Development Framework"** is my own work and has not been submitted previously by me for taking any degree from this university **Bahria University, Islamabad Campus** or anywhere else in the country/world.

At any time if my statement is found to be incorrect even after my Graduate the university has the right to withdraw/cancel my MS degree.

Name of scholar: **Afnan Siddique**

Date: **25-June-2019**



**Bahria University**  
Discovering Knowledge

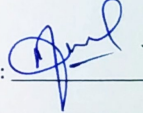
**MS-14B**

### **Plagiarism Undertaking**

I solemnly declare that research work presented in the thesis titled "**Ontology Model for Web development Frameworks**" is solely my research work with no significant contribution from any other person. Small contribution / help wherever taken has been duly acknowledged and that complete thesis has been written by me.

I understand the zero tolerance policy of the HEC and Bahria University towards plagiarism. Therefore I as an Author of the above titled thesis declare that no portion of my thesis has been plagiarized and any material used as reference is properly referred / cited.

I undertake that if I am found guilty of any formal plagiarism in the above titled thesis even after award of PhD degree, the university reserves the right to withdraw / revoke my PhD degree and that HEC and the University has the right to publish my name on the HEC / University website on which names of students are placed who submitted plagiarized thesis.

Student / Author's Sign: 

Name of the Student: **Afnan Siddique**

# Dedication

*"The darkest nights produces the brightest stars...!!!"*

This research work is dedicated to my beloved late sister who passed away after a long fight for about four years.

May her soul rest in peace, Ameen.

# Acknowledgments

I am very thankful to Almighty Allah Who gave me the ability to complete my research work on time. The research could not have been completed without the guidance of Dr. Muhammad Asfand-e-Yar, who not only helped me as my supervisor but also encouraged me throughout the thesis.

I would also like to thank my friends and family, who encouraged me and helped me during my Masters. Their encouragement enabled me to complete the research.

AFNAN SIDDIQUE

Bahria University Islamabad, Pakistan

June, 2019

# Contents

<b>1</b>	<b>Introduction</b>	<b>2</b>
1.1	Overview . . . . .	2
1.2	Web Development Frameworks . . . . .	2
1.3	Problem Statement . . . . .	3
1.4	Thesis Organization . . . . .	4
<b>2</b>	<b>Literature Review</b>	<b>6</b>
2.1	Related Work . . . . .	6
2.2	Comparative Analysis . . . . .	13
<b>3</b>	<b>Methodology</b>	<b>16</b>
3.1	Overview of Tools and Languages . . . . .	16
3.2	Methodology . . . . .	16
3.3	RDF Graphs for Web Framework . . . . .	17
3.3.1	Ontology for General Programing Concepts . . . . .	18
3.3.2	Ontology for ASP.Net and PHP Programming Concepts . . . . .	18
<b>4</b>	<b>Implementation</b>	<b>22</b>
4.1	Descriptive Logic (DL) for General Programming Concepts . . . . .	22
4.1.1	DL for Loops defined in ProgLanguage Ontology . . . . .	23
4.2	Descriptive Logic for PHP Concepts . . . . .	24
4.2.1	DL for Tags in PHP Semantics . . . . .	25
4.2.2	DL for Conditional Statements in PhpSemantics . . . . .	25
4.2.3	DL for Loops in PhpSemantics . . . . .	26
4.3	Descriptive Logic for ASP Concepts . . . . .	27
4.3.1	DL for Tags in ASP Semantics . . . . .	28
4.3.2	DL for Loops in ASP Semantics . . . . .	28
4.3.3	DL for Conditional Statement in ASP Semantic . . . . .	29

<b>5</b>	<b>Results and Analysis</b>	<b>32</b>
5.1	Queries on Ontologies . . . . .	33
5.1.1	containsSome . . . . .	33
5.1.2	isDefinedBy . . . . .	35
5.1.3	isTypeOf . . . . .	36
5.1.4	hasOne . . . . .	38
5.1.5	startsWith . . . . .	39
5.1.6	hasTermination . . . . .	40
5.1.7	isInitializedBy . . . . .	42
5.2	Analysis and Comparison . . . . .	43
5.2.1	Analysis using OntoClean . . . . .	44
5.2.2	Comparison with previous studies . . . . .	44
<b>6</b>	<b>Conclusions and Future Work</b>	<b>46</b>
6.1	Conclusion . . . . .	46
6.2	Future Work . . . . .	47



# List of Figures

1.1	Overview of Proposed Methodology . . . . .	4
2.1	Schemantic Overview of Software Engineering Knowledge Representation	8
2.2	Abstracted Structure Model of Web-based Application Source Code . . .	10
2.3	Schematic Overview of Software Engineering Knowledge Representation	10
2.4	Basic Classes and Subclasses of C Language Ontology . . . . .	12
3.1	Overview of Proposed Methodology . . . . .	17
3.2	Classes for Programming Language Ontology . . . . .	18
3.3	RDF Graph of Programming Language Ontology . . . . .	19
3.4	Classes for ASP Ontology . . . . .	19
3.5	Classes for PHP Ontology . . . . .	20
3.6	RDF Graph for ASP Ontology . . . . .	20
3.7	RDF Graph for PHP Ontology . . . . .	21
5.1	Object Properties of ProgLanguage Ontology . . . . .	32
5.2	Object Properties of PHP and ASP . . . . .	33
5.3	containsSome for ASP . . . . .	34
5.4	containsSome for PHP . . . . .	34
5.5	isDefinedBy ASP . . . . .	35
5.6	isDefinedBy PHP . . . . .	36
5.7	isTypeOf ASP . . . . .	37
5.8	isTypeOf PHP . . . . .	37
5.9	hasOne ASP . . . . .	38
5.10	hasOne PHP . . . . .	39
5.11	startsWith ASP . . . . .	40
5.12	startsWith PHP . . . . .	40
5.13	hasTermination ASP . . . . .	41
5.14	hasTermination PHP . . . . .	42

5.15 isInitiazliedBy ASP . . . . .	43
5.16 isInitiazliedBy PHP . . . . .	43
5.17 OntoClean for ProgLanguage . . . . .	44
5.18 OntoClean for PHPSemantics . . . . .	44
5.19 OntoClean for ASPSemantics . . . . .	45
5.20 Basic concepts of C Programming Language . . . . .	45

# List of Tables

1.1	Definition of terms used in Ontology . . . . .	3
2.1	Comparative analysis of related work . . . . .	15
4.1	Symbol in Descriptive Logic . . . . .	22
4.2	Descriptive Logic for General Programming Ontology Model . . . . .	23
4.3	Descriptive Logic for Loops . . . . .	24
4.4	Descriptive Logic for PHP Semantics/Ontology Model . . . . .	25
4.5	Descriptive Logic for PHP Tags . . . . .	26
4.6	Descriptive Logic for PHP Tags . . . . .	26
4.7	Descriptive Logic for Loops in Php Semantics . . . . .	27
4.8	Descriptive Logic for ASP Semantics Model . . . . .	28
4.9	Descriptive Logic for ASP Tags . . . . .	29
4.10	Descriptive Logic for Loops in ASP Semantics . . . . .	30
4.11	Descriptive Logic for PHP Tags . . . . .	31

# Abbreviations

OWL	Web Ontology Language
RDF	Resource Description Framework
SPARQL	SPARQL Protocol and RDF Query Language
ASP	Active Server Pages
PHP	Hypertext Preprocessor
DL	Descriptive Logic
API	Application Programming Interface

# Abstract

There are multiple scripting and programming languages for Web Development. To shift from one language to another, it uses various methods on which some work has been done but it leads to errors. Various parsing techniques are used for translating and transforming programming languages, such as LALR, LL(1), GLR, ANTLR, etc. But these techniques are ambiguous. These techniques are needed to be update because frameworks for programming environments are changes by time. Like, HTML and HTML5 have minor difference in tags but does the same work. Many researchers have formulated Semantics for different programming languages such as C, C++, C Sharp, Java etc. But they haven't used those Semantics for translating or transforming purposes.

In this research, I have used Ontologies that models different development frameworks and relate them with one another. Therefore, it could be easier for developers to translate and maintain the originality of methods and frameworks used in a certain program or application. Furthermore, a plugin is designed with an interface using which one directly translates from one language to another using Semantics.

Initially, I have worked on two mostly used languages for backend Web programming languages which are PHP and ASP.Net. This research work consists of three Ontologies based on Semantics. These includes a general programming Ontology, a PHP Ontology and an ASP.Net Ontology. Afterwards, I have formulated descriptive logic DL for each Ontology to make relationships between each Ontology which will help in formulating Semantics between each Ontology. These Ontologies and DL are further used to co-relate for translating one language to another based on Semantics.

# Chapter 1

## Introduction

### 1.1 Overview

Semantic Web is growing strong in many fields of relating and sharing of information in different formats. An Ontology is used as type casting, giving properties to different types of data and defining relations between arbitrary or real-world entities. The OWL (Web Ontology Language) is used to represent of the data and format accordingly that can easily read by machines and could be used for further research and development. In Web development, multiple languages and frameworks are used for development of Websites. During the process of shifting from one development framework to another is difficult as there will be errors and leads to misconceptions.

Various parsing techniques are used for translation between languages, which does not provide freedom of selecting between languages, as parsing need proper formatting and syntactical information. Parsing is limited to few frameworks as parsing is only limited to syntax level. The use of Semantics not only provide developers to easily understand complex structures of languages but also allow them to translate freely using defined Ontologies which are comparatively easy to update and modify without losing the original Semantics of languages. Protege is used for constructing Ontologies of different languages and for further verification and validation an interface in eclipse is developed for translation of languages. Following table 1.1 contains definitions of main terminologies used in constructing and defining Ontology.

### 1.2 Web Development Frameworks

There are many models for Web development, most popular ones are Asp.NET, PHP and JavaScript with addition of HTML as a support. ASP stands for Active Server Pages.

Table 1.1: Definition of terms used in Ontology

Creating	Modeling of an Ontology for a specific domain. Mostly, it is done by using Protégé.
Combining	Making a union of two (or more than two) Ontologies.
Merging & Integrating	The process of concatenation of Ontologies.
Aligning	Combination of Ontologies in such a way that they make a whole new Ontology that is correct and valid for further use.
Mapping	Merging the same concepts and relations from different domain.
Articulation	Relationship between aligned Ontologies.
Updating	Making changes in representation of concepts and relations while Semantics remain same.
Comparing	Looking for similarities or differences between Ontologies.
Versioning	Managing multiple instances of an Ontology.

ASP.NET is a .NET framework-based development environment with provides essential services for developing enterprise-class Web application using windows form<sup>1</sup>. PHP stands for Hypertext Preprocessor. PHP is a server-side scripting language that embeds with HTML to provide management for dynamic content and databases by session tracking<sup>2</sup>.

### 1.3 Problem Statement

*How to translate the application's development framework from one to another;  
while the originality of an application should not be disturbed.*

The main purpose of this research is to co-relate Web development frameworks in such a way that while translating from one framework to another, their originality of structure and methods which are implemented should not be disturbed. Using this technique will help developers work easier as if they want to switch between frameworks of Web development, such as ASP.NET to PHP or vice versa.

There are always problems while translating from one method or framework to another. In this proposal, an Ontology model will be defined using Web Semantics by which transforming from one Web development framework to another will be easier. There are multiple ways to perform any task while developing an application but when a Web application is implemented on a larger scale, the developer or the main stack holders tends

<sup>1</sup>M. Library, Microsoft Corporation, [Online]. Available: <https://msdn.microsoft.com/en-us/library/4w3ex9c2>. [Accessed 21 January 2018].

<sup>2</sup>T. P. Group. [Online]. Available: [php.net/manual/en/intro-what-is.php](http://php.net/manual/en/intro-what-is.php). [Accessed 21 January 2018].

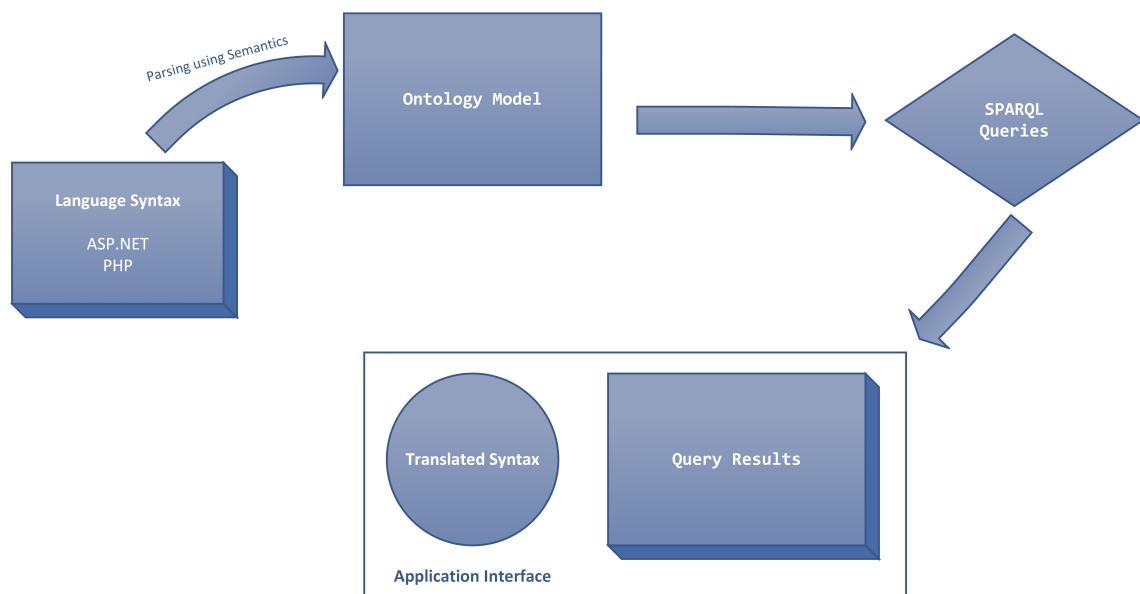


Figure 1.1: Overview of Proposed Methodology

to shift towards the trending framework or technique. These trends effect the market value of running application, as it is implemented using an older framework.

To enhance the translating methodology, an Ontology model is defined to make sure it does not alter the main functionality of already implemented application. This model is based on three Ontologies, a general model for programming, a model for ASP.Net and a model for PHP. Afterwards, an interface is used to verify these models and co-relate them for translation purpose. Interface is further detailed in coming chapters. These models can be modified for other frameworks. Overview of proposed system is shown in figure 1.1.

For an example, Bahria University' s CMS (Content Management System) is implemented using ASP.Net for its all three campuses, at Islamabad, Lahore and Karachi. At any time in future, if university officials want to transfer their system towards PHP framework for same purpose due to any issues in currently implemented application, there will be issues. These issues could be logical or syntactical level, which may alter basic functions of application. So, the solution will be a model having enough information about the similarities of languages and frameworks which will help developers to easily shift from one framework to another.

## 1.4 Thesis Organization

The thesis is structured as: Chapter 2 discusses about already existing methodologies and technique, comparative analysis of methodologies discussed, Chapter 3 briefs about



proposed system and methodology in detail, Chapter 4 is discussion of successfully implementation of proposed system, Chapter 5 shows results and other findings, and Chapter 6 summarizes the conclusion of work done and details about future research.

# Chapter 2

## Literature Review

In the chapter 2, work related to use of Ontology for structural break down in development of applications and programming languages are briefly discussed. They mainly focus on use of Ontology for getting ease in different steps of development and practical documentation.

Research work done in field of Ontologies with respect to programming languages and application development are discussed in below mentioned section. Later, a comparative study is done based on existing approaches and their findings.

### 2.1 Related Work

In Kaiya et.al [6], software requirement analysis strategies considering space Ontology method. The inspiration of this exploration is to build up a mapping between a software requirement specification and the Ontology that speaks to Semantics segments. The proposed Ontology comprises of a thesaurus and interface rules which includes space ideas and connections reasonable for Semantics handling. This Ontology permits requirement specialists to dissect a requirement as for the Semantics of the application area. The proposed Ontology demonstrates three Semantics processing, (1) Detecting incompleteness and inconsistency, (2) Measurement of quality of specification, and (3) Predicting requirement changes based on Semantics analysis about change history. Every requirement articulation ought to be deciphered in view of the learning of nuclear constituents of significance and Ontology is utilized all things considered information. By utilizing lexical deterioration system, every requirement articulation can be decayed into a few terms that are translated in same route by anybody. In the research, data mining methods on a few sorts of regular dialect depictions, for example, client manuals, changes chronicles, utilize case portrayals situations et cetera. Such sort of archives can be requirements.

Bensaber et.al [2] explains that Semantic Web guarantees mechanize summons, revelation and synthesis of Web services by upgrading services with Semantic portrayals. The philosophy is separated into three principle steps. The initial step we save designed WSDL reports into UML profile models that empower the utilization of abnormal state graphical models as an integration stage for Semantic Web services. In the second step, reasonable space Ontologies are utilized for the Semantic explanation of UML models. At last, in the third step a transformation device will create naturally the OWL-S depiction from these UML models [2].

Shen et.al [12] research shows that UML profile gives adaptability as it can express various Semantic Web benefit ideas. They have characterized and executed transformation rules from WSDL to UML to robotize the figuring out process. The classes in UML are mapped on existing OWL idea amid the comment procedure of UML outline. Their methodology uses existing aptitudes in UML displaying, which can significantly enhance the effectiveness of the Semantic Web benefit advancement work process <sup>1</sup>. In their approach, BPEL4WS specification is translated into an OWL-S specification.

Wongthongtham et.al [14] aims to present an Ontology model of software engineering to represent its knowledge. It gives an analysis of what software engineering Ontology is, the thing that it comprises of? Furthermore, what it is utilized for as use model situations? The use situations displayed in this research feature the qualities of the software engineering Ontology. The software engineering Ontology helps with characterizing the data for the trading of Semantic venture data and is utilized as an interchanges structure, shown in figure 2.1 [14]. Its clients are software engineers sharing area information and additionally occasion learning of the software engineering.

Software engineering Ontology comprises of cases speaking to task information, properties speaking to twofold relations held among software engineering ideas/examples and classes deciphered resources that contain particular undertaking information. The software engineering Ontology classes are developed of portrayals of software engineering ideas that determine the conditions that must be fulfilled by venture information for it to be an individual from the classes. In a research, characterized graphical documentations of displaying software engineering Ontology as an elective formalism. The demonstrating documentations are utilized to outline software engineering Ontology. Toward the end, they finished up there are numerous changes that can be made through future work, which could consider software engineering Ontology evolution [14].

In Imamoglu's et.al [5], author have proposed a rule-based decision support system to guide decision makers and software engineers in programming language selections. Right

---

<sup>1</sup>Wikipedia, "Ontology (Computer Science)," Wikipedia, the Free Encyclopedia, 21 06 2006. [Online]. Available: <https://en.wikipedia.org/wiki/Ontology>. [Accessed 29 04 2018].

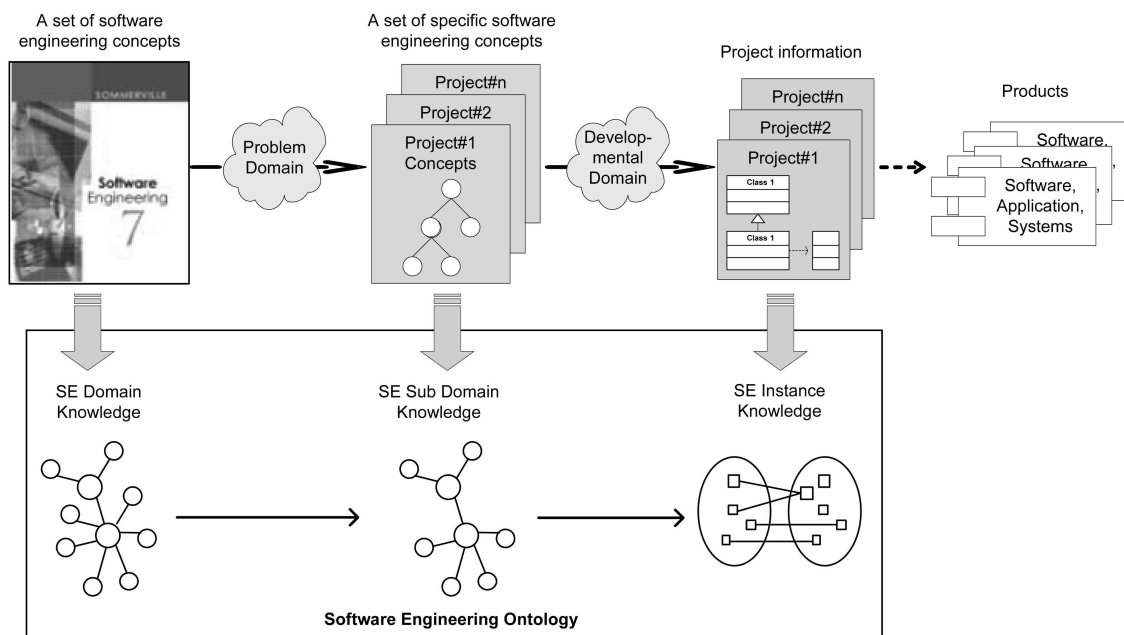


Figure 2.1: Schematic Overview of Software Engineering Knowledge Representation

off the bat, the system gives a component to manufacture and adjust a knowledge base. At that point, the system gives direction about programming language determination before the coding stage as per the project subtle elements. The target of this work is to give an instrument to construct and alter a knowledge base to about programming languages and in addition to develop a specialist system that works with this knowledge base to give direction to programming language choice as indicated by the project subtle elements [5].

A rule-based decision support system uses rules to build the knowledge base and to do the expert reasoning for solving problems. Each programming language has some essential building squares are characterized by the language definition as grammar and Semantics of the language. The primary inspiration of the research is to propose a rule-based decision support system to direct decision producers in programming language determination. Initially, give an instrument to construct and change a knowledge base about programming languages. At that point, develop a specialist system that works with the gave knowledge to give a rule to programming language determination as per the project subtle elements. The knowledge base incorporates the programming language definitions with essential properties and additional traits that can be related with the characterized languages [5].

The research done by Levy[8] et.al draws both empirical and theoretical parallels between the embedding and alignment literature. Cross-lingual word embedding algorithms try to represent the vocabularies of two or more languages in one common continuous vector space. They hypothesize that the information in bilingual sentence-aligned data

has been thoroughly mined by existing methods and suggest that future work explore additional sources of information to make substantial progress [8].

In [10], Posadas et.al proposes a methodology for Author Profiling by using syntactic highlights, for example, syntactic based n-gram of different sorts to anticipate different component of the author. The application of syntactic n-grams gives preferred outcomes over using traditional ones for the assignment authorship attribution. There are different sorts of syntactic n-grams depending on the information utilized for their construction. For this reason, a syntactic parser is required for our methodology. The sentences are parsed depending on their size. They have given only a couple of syntactic n-grams are generally identified with expressions that parsers don't process well.

To represent the information, vector space show approach, an instance is represented as a vector space, in which each dimension corresponds to a determine syntactic n-gram and the esteem is its frequency. The final arrangement of highlights for a mark is the union of all the chose highlights through the chi-square test. By the outcomes, we can train the SVM classifier using RBF kernel and run of the mill normalization of vectors. The methodology is rehashed for each mark and then every classifier is trained for each name. Their methodology misuses information contained in the dependency trees, its performance is influenced using external syntactic parsers [10].

In [15], Zhao et.al talks about a novel methodology for removing knowledge from Web-based application source code in supplementing and helping Ontology improvement from database. The relationship between the imperative parts of Web application source code and the back-end database schema with their distinctive structures are explicitly shown in detail. A knowledge processing and mix exhibit for extraction and joining the knowledge embedded in the source code for Ontology headway is then proposed [15].

Web-based applications, overall, contain a backend databased and dynamic server page which make Web content intensely from the backend database. They have portrayed a separated structure model of Web-based application source code for knowledge extraction for Ontology change and for demonstrating how each bit of the structure can be used for the Ontology picking up from databases, figure 2.2 [15].

Afterwards, they presented a model for knowledge extraction from Web-based application source code and for incorporating the extricated knowledge amid the procedure of Ontology improvement from the database schema. The source code that is implemented in different languages will be processed by one of the corresponding language processing using shown as JSP, ASP and PHP in the model, as shown in figure 2.3 [15].

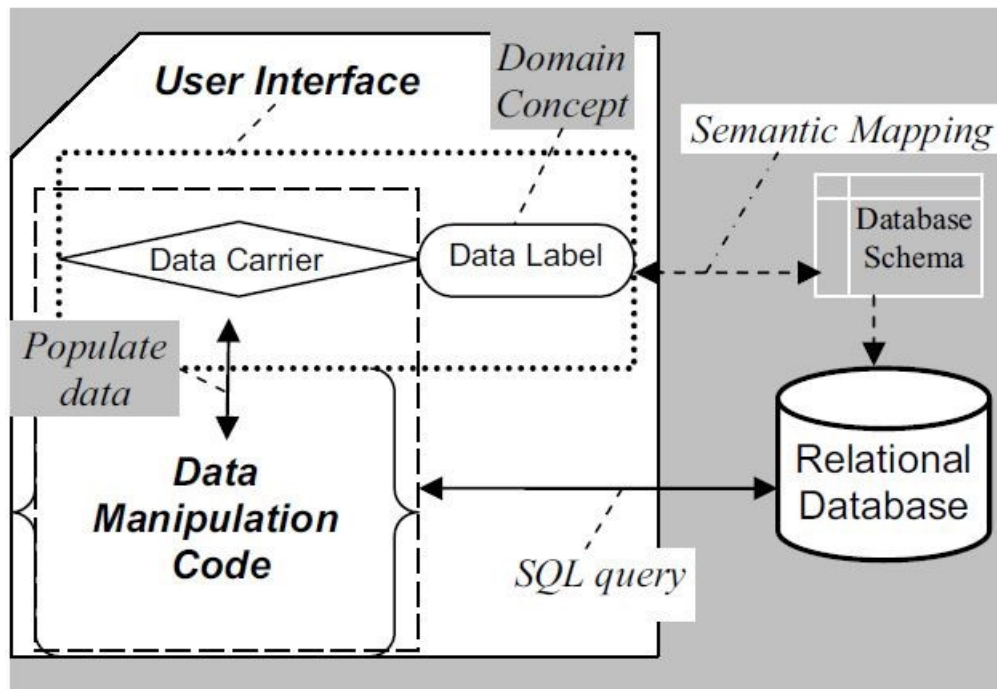


Figure 2.2: Abstracted Structure Model of Web-based Application Source Code

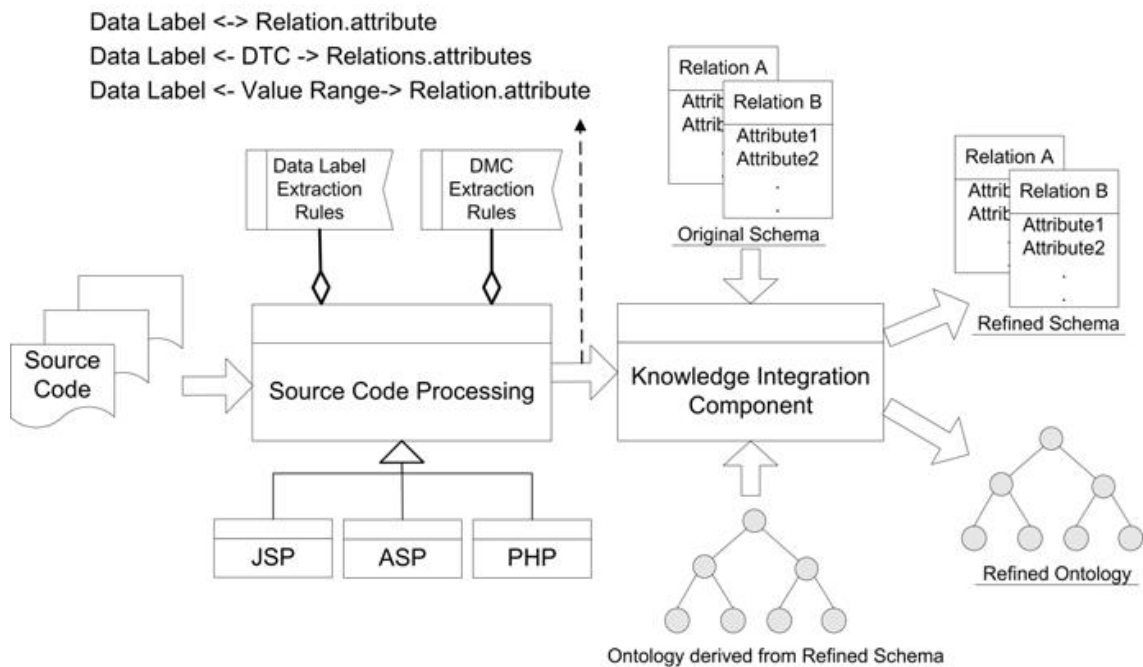


Figure 2.3: Schematic Overview of Software Engineering Knowledge Representation

Web applications have been become out of the blue in these ongoing years yet because of contrast between Web applications and programming applications, their improvement procedures vary from numerous points of view. In [7], R. Sharma et.al speaks to different development methodologies and methodologies proposed uncommonly for Web applications. Agile, object oriented, UML based and so forth are a few models of ways to deal with be utilized for transforming strategies for Web application development.

Web application utilizes a Web program as a client which request/post a few information from/on server which again perhaps centralized or distributed to enhance server response time. Client is a computer software application introduced at clients' end which executes Web application, created in a program supporting language (such as JavaScript combined with HTML). Web application utilizes a Web program as a client which request/post a few information from/on server which again perhaps centralized or distributed to enhance server response time. Client is a computer software application introduced at clients' end which executes Web application, created in a program supporting language [7].

Knowledge base of C programming language is displayed utilizing Ontology development. Ontology development standards, methods and focal points are breaking down as the qualities of knowledge structure and C programming language configuration course are additionally mulled over. In Y. Hu et.al [4] says, Ontology knowledge of C programming language is itemized. By utilizing this Ontology, the structure of C programming language is clear to understand and moderately less demanding to instruct understudies as it introduces the system model and the work stream of the C programming language, as shown in figure 2.4 [4].

However, there are some problems to be corrected, such as (1) the automation of knowledge and (2) the study on the coordination between knowledge reasoning and knowledge reasoning and knowledge base structure [4].

The learning of programming languages required the student to create systematic portrayals of the lexical develops and Semantic standards of the languages. In [9], Pierakeas et.al proposed a combination of Learning Objects and modeling the area ideas of programming language. An Ontology comprises a formalism for seeing and preparing data, sharing information, enabling it to be reused and in this way empowering correspondence among heterogeneous and distributed frameworks.

The fundamental inspiration for this examination work is to propose a novel methodology on building a course that could reduce the assignment of planning the learning procedure and presumably prompt more compelling learning ways. These methodologies depend on the thought which is utilized to speak to the system of ideas and relationships of the information space. The came about Ontologies furnish teachers with a clearer picture

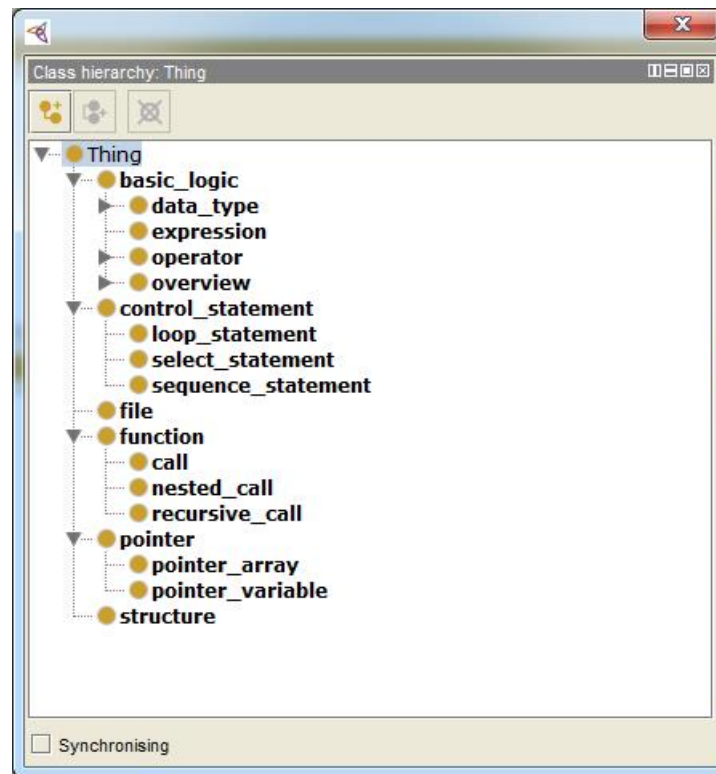


Figure 2.4: Basic Classes and Subclasses of C Language Ontology

about the space ideas is the field of Java and C, and in addition about their relationships [9].

The Semantic Web works on the existing Web which presents the meaning of information as well-defined vocabularies understood by the people. Semantic Search, at the same time, works on improving the accuracy of a search by understanding the intent of the search and providing contextually relevant results. The research describes a Semantic approach towards Web search through a PHP application. The goal was to parse through a user's browsing history and return Semantically relevant Web pages for the search query provided. The browser used for this purpose was Mozilla Firefox. The user's history was stored in a MySQL database, which, in turn, was accessed using PHP. The Ontology, created from the browsing history, was then parsed for the entered search query and the corresponding results were returned to the user providing a Semantically organized and relevant output [11].

Ontology have become a relevant representation formalism and many application domains are considering adopting them. This attention claims for methods for reusing domain knowledge resources in the development of domain Ontologies. Accordingly, in this research we discuss a general methodology to create domain Ontology for more



than one object-oriented language (OOP) like Java, PHP and C++. A lot of software development methods specially Web applications have presented most of these methods that are focusing on the structure of distributed systems and security, in which they are connected through networks and the internet; resulting in more valuable business and critical assets stored, searched and manipulated by World Wide Web. The aims of this study building domain Ontology for OOP language classes for different OOP languages or different versions of the same language is an exciting opportunity for researchers to access the information required under the constant increase in the volume of information disseminated on the Internet [1].

## 2.2 Comparative Analysis

Following table 2.1 shows the comparative analysis of included research works with fields of motivation and methodologies used in different fields.

Reference Paper	Problem Statement	Methodology	Techniques	Results	Limitation
H Kaiya et.al [6]	Ontology as Domain Knowledge	Software requirement analysis using domain ontology and natural language processing techniques	mapping between ontology concepts and specification requirements	Metrics for requirement documents	inconsistencies in natural language requirements limit automation of ontology development.
D. A. et.al [2]	UML profile Modeling, Domain ontologies and OWL-S description using UML models	model to facilitate OWL-S construction	uses MDA concepts and strategies for designing and implementation	Automation of Semantic linkage with web services	complex system requires more complexity in ontology development

P. Wongthongtham et.al [14]	BPEL4WS are linked to a process in OWL-S	Ontology model for software engineering and its components	defining info for semantics and communication framework	Application for mapping and translation	OWL-S needs more time to get mature in comparison of BPEL4WS
M Y Imamoglu et.al [6]	knowledge base for different programming languages used for decision making for development of any application	Knowledge base for programming language guidance	rule based decision system for selection of programming language	PL Expert application	Doesn't work in all scenarios
J. P. Posadas Duran et.al [10]	Syntactic N-gram and general classifications	Feature extraction for author profiling task	Used supervised machine learning for classification and N-gram markers for feature extraction	syntactic parser constructed dependency trees	need new heuristics and improved weight balancing
S. Zhao et.al [15]	Domain ontology and database schema for knowledge	Extracting knowledge from source code of web based application	defined abstract model of source code for knowledge extraction	Ontology derived from database schema based on extracted knowledge	Need improved data label extraction for better results

Y. Hu [4]	Ontology and relations in semantics	Ontology for C programming language for teaching resources	Knowledge base for C programming is presented by using Ontology	Complete Ontology of C programming language concepts	Automation of knowledge extraction for larger data and coordination btw base structure and knowledge
C. Pierakeas [9]	Ontology and Web Semantics	Learning of programming languages using lexical and semantics of language	Ontology for knowledge representation of Java and C languages	Java and C language ontologies	Semantic of languages may not be similar for all cases.

Table 2.1: Comparative analysis of related work

# Chapter 3

## Methodology

In chapter 3, tools and methodology used through out thesis is detailed. At first, overview of tools and languages which are selected for implementation are defined. In second and third part of the chapter, implementation steps and ontologies are explained briefly.

### 3.1 Overview of Tools and Languages

RDF provides the basics for defining and linking data in Web Semantics in the form of OWL, which can be processed using any interface. SPARQL is used for querying the data from an Ontology model [7]. There are many models for Web development, most popular ones are Asp.NET, PHP and JavaScript with addition of HTML as a support. ASP stands for Active Server Pages. ASP.NET is a .NET framework-based development environment with provides essential services for developing enterprise-class Web application using windows form<sup>1</sup>.PHP stands for Hypertext Preprocessor. PHP is a server-side scripting language that embeds with HTML to provide management for dynamic content and databases by session tracking<sup>2</sup>.

### 3.2 Methodology

The figure 3.1 describes steps to be done during the research period. Two Web development languages, which are ASP .Net and PHP, will be parsed into syntactical and Semantic form. After parsing the languages, RDF graphs and Ontology models will

---

<sup>1</sup>M. Library, Microsoft Corporation, [Online]. Available: <https://msdn.microsoft.com/en-us/library/4w3ex9c2>. [Accessed 21 January 2018].

<sup>2</sup>T. P. Group. [Online]. Available: [php.net/manual/en/intro-what-is.php](http://php.net/manual/en/intro-what-is.php). [Accessed 21 January 2018]

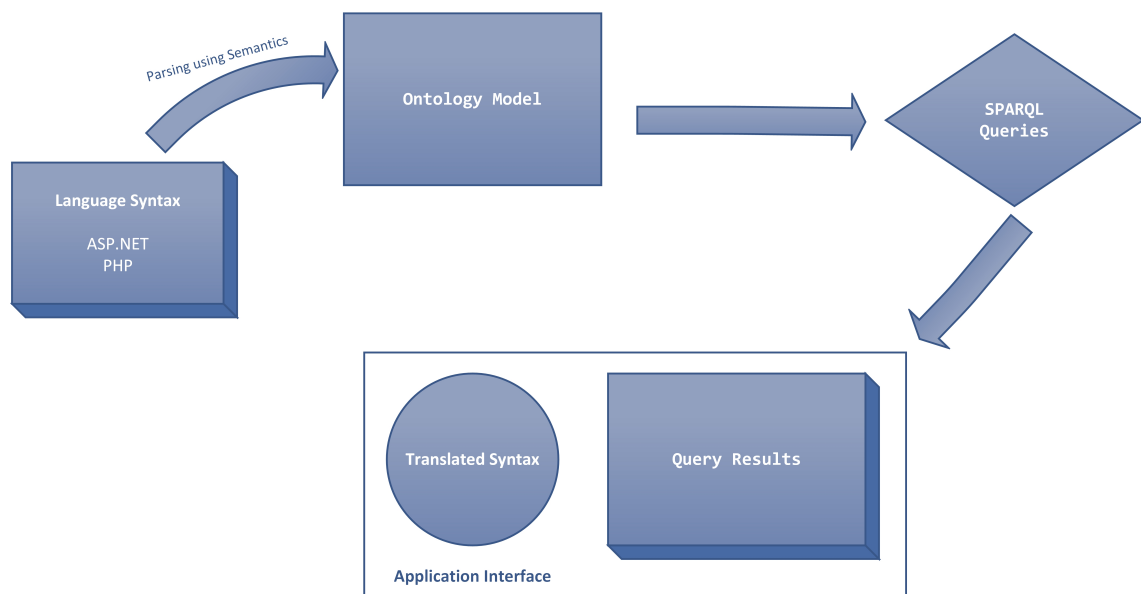


Figure 3.1: Overview of Proposed Methodology

be made along with the executable SPARQL queries. these graphs and queries will be further used in application interface by which users can easily translate the syntax from one language to another.

### 3.3 RDF Graphs for Web Framework

Each framework has different syntax for execution of similar loop structures, as given below:

- Definition of for-each loop in Asp.NET

```

foreach (var element in array)
{
    body in Asp.Net...
}
  
```

- Definition of for-each loop in PHP

```

foreach (array as element)
{
    body in PHP...
}
  
```

Similarly, an Ontology can be made for every method available in Web frameworks and relate them with each other for better understanding and easy learning. This also helps in translating between languages used by developers and transforming from one architecture to another more efficiently, without any lose in originality.

### 3.3.1 Ontology for General Programing Concepts

In the figure 3.2, classes used for general programing ontology are shown. The graph is an Ontology model for general programming concepts, such as defining general statement, conditional statements and different loops. In loops, it includes for loop, foreach loop and while loop. In definition of general statement, it includes a subclass of data collection along with a subclass of variable.

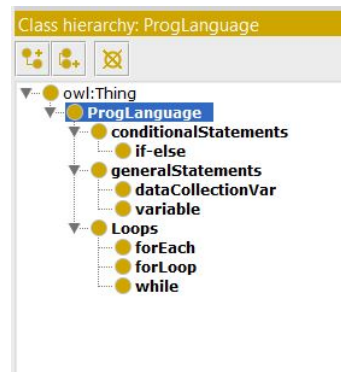


Figure 3.2: Classes for Programming Language Ontology

The RDF graph for above mentioned Ontology is shown in figure 3.3, it includes all the relations between classes and subclasses of Ontology as concepts, data properties and individuals.

### 3.3.2 Ontology for ASP.Net and PHP Programming Concepts

Similarly, an Ontology for ASP and PHP is shown in figures 3.4 and 3.5, respectively. These Ontologies include classes and subclasses based on the concepts used in General Programming Ontology. Such as Conditional Statements, General Statements and Loop. Furthermore, it includes the semantics for printing a variable and a string.

The RDF for above mention ontologies of ASP and PHP is shown in figures 3.6 and 3.7, respectively



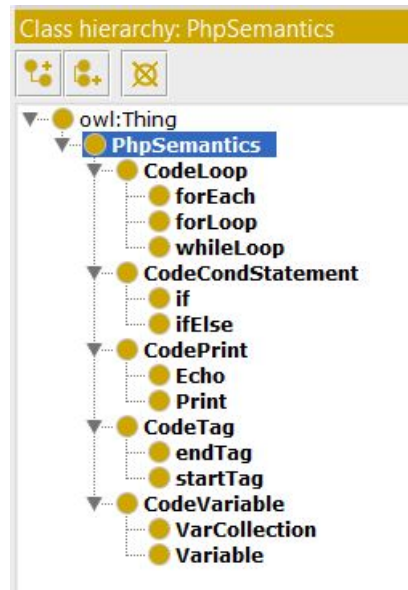


Figure 3.5: Classes for PHP Ontology

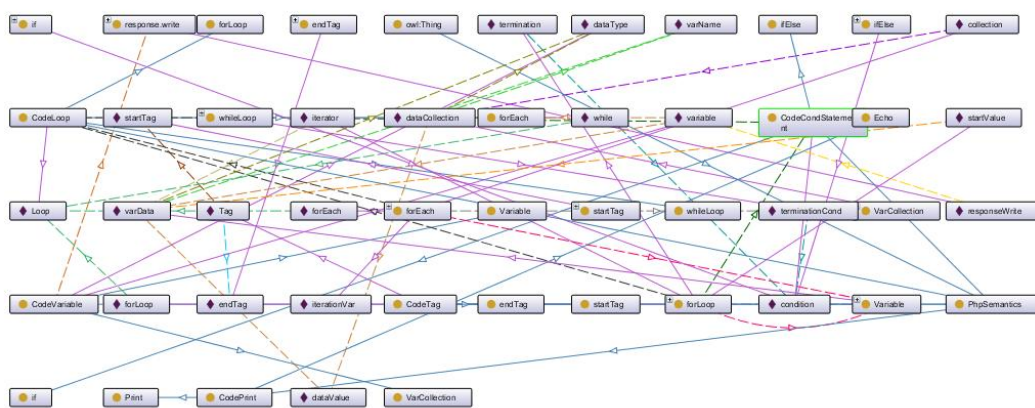


Figure 3.6: RDF Graph for ASP Ontology



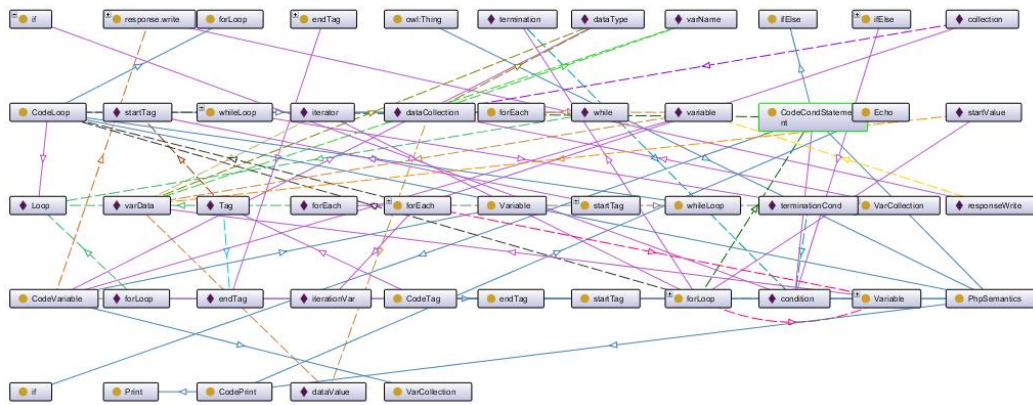


Figure 3.7: RDF Graph for PHP Ontology

# Chapter 4

## Implementation

In this chapter, steps involved in implementation are briefed. Based on Ontologies, as described in previous chapter, descriptive logics of each ontology with respect to their classes and sub classes are formulated.

Descriptive logic is used to co-relate concepts as classes and subclasses with roles as data properties and data restrictions based on a formal and meaningful definition between classes and roles. Following Table 4.1 shows description of symbols used in descriptive logic.

Table 4.1: Symbol in Descriptive Logic

Symbol	Meaning
$\subset$	Subclass
$\exists$	Existential Restriction
$T$	Thing
$\forall$	Universal Restriction

### 4.1 Descriptive Logic (DL) for General Programming Concepts

From the classes and sub classes. as described in previous chapter, the descriptive logic for General Programming is detailed in table 4.2. This table includes Subclass relations, Data Properties and Object Properties of different Classes and Objects.

Table 4.2: Descriptive Logic for General Programming Ontology Model

<b>ProgLanguage</b>	<b>Subclass Relations</b> ConditionalStatement $\subset$ ProgLanguage GeneralStatement $\subset$ ProgLanguage Loop $\subset$ ProgLanguage	
<b>ConditionalStatement</b>	<b>Subclass Relations</b> if-else $\subset$ ConditionalStatement	
	<b>Data Properties</b>	<b>Descriptive Logic</b>
	xsd:boolean	$\exists$ terminatedBy.T $\subset$ ConditionalStatement $T \subset \forall$ terminatedBy.xsd:boolean
<b>GeneralStatement</b>	<b>Subclass Relations</b> dataCollectionVar $\subset$ GeneralStatement variable $\subset$ GeneralStatement	
<b>Loop</b>	<b>Subclass Relations</b> forLoop $\subset$ Loop forEach $\subset$ Loop whileLoop $\subset$ Loop	
	<b>Data Properties</b>	<b>Descriptive Logic</b>
	xsd:string	$\exists$ loopName.T $\subset$ forEach $T \subset \forall$ loopName.xsd:string

#### 4.1.1 DL for Loops defined in ProgLanguage Ontology

Based on the table above, descriptive logic for Loops, such as forLoop, forEach and whileLoop, is shown in the table 4.3. Descriptive logic of these loops is formulated by viewing the data property relations and data restrictions within classes and individuals defined in the classes.

By using the table 4.3 DL for ProgLanguage Ontology as base, we can formulate DLs for PHP and ASP Semantics, which are detailed in next sections of the chapter.

Table 4.3: Descriptive Logic for Loops

<b>forEach</b>	<b>Subclass Relations</b> forEach $\subset$ Loop	
	<b>Data Properties</b>	<b>Descriptive Logic</b>
	xsd:anyURI	$\exists$ contains.T $\subset$ forEach $T \subset \forall$ contains.xsd:anyURI
	xsd:integer	$\exists$ iteratedBy.T $\subset$ forEach $T \subset \forall$ iteratedBy.xsd:integer
	<b>Object Properties</b>	<b>Descriptive Logic</b>
	hasOne	$\exists$ hasOne.T $\subset$ DataCollection $T \subset \forall$ hasOne.forEach
	isTypeOf	$\exists$ isTypeOf.T $\subset$ Loop $T \subset \forall$ isTypeOf.forLoop
isIteratedBy	$\exists$ isIteratedBy.T $\subset$ variable $T \subset \forall$ isIteratedBy.forEach	
<b>forLoop</b>	<b>Subclass Relations</b> forLoop $\subset$ Loop	
	<b>Data Properties</b> xsd:integer	<b>Descriptive Logic</b> $\exists$ iteratedBy.T $\subset$ forLoop $T \subset \forall$ iteratedBy.xsd:integer
	<b>Object Properties</b> hasTerminationCond	<b>Descriptive Logic</b> $\exists$ hasTerminationCond.T $\subset$ conditionalStatements $T \subset \forall$ hasTerminationCond.forLoop
	isTypeOf	$\exists$ isTypeOf.T $\subset$ Loop $T \subset \forall$ isTypeOf.forLoop
	isIteratedBy	$\exists$ isIteratedBy.T $\subset$ variable $T \subset \forall$ isIteratedBy.forLoop
<b>whileLoop</b>	<b>Subclass Relations</b> whileLoop $\subset$ Loop	
	<b>Data Properties</b> xsd:Boolean	<b>Descriptive Logic</b> $\exists$ isTerminated.T $\subset$ whileLoop $T \subset \forall$ isTerminatedBy.xsd:Boolean
	<b>Object Properties</b> hasTerminationCond	<b>Descriptive Logic</b> $\exists$ hasTerminationCond.T $\subset$ conditionalStatements $T \subset \forall$ hasTerminationCond.whileLoop
	isTypeOf	$\exists$ isTypeOf.T $\subset$ Loop $T \subset \forall$ isTypeOf.whileLoop

## 4.2 Descriptive Logic for PHP Concepts

All programming languages have almost similar structure and command flow patterns. Keeping this in mind, semantics for PHP are formulated. DL for those semantics is shown

in table 4.4.

Table 4.4: Descriptive Logic for PHP Semantics/Ontology Model

<b>PhpSemantics</b>	<b>Subclass Relations</b> $\text{CodeCondStatement} \subset \text{PhpSemantics}$ $\text{CodeLoop} \subset \text{PhpSemantics}$ $\text{CodePrint} \subset \text{PhpSemantics}$ $\text{CodeTag} \subset \text{PhpSemantics}$ $\text{CodeVariable} \subset \text{PhpSemantics}$	
<b>CodeVariable</b>	<b>Subclass Relations</b> $\text{CodeVariable} \subset \text{PhpSemantics}$ $\text{VarCollection} \subset \text{CodeVariable}$ $\text{Variable} \subset \text{CodeVariable}$	
	<b>Object Properties</b>	<b>Descriptive Logic</b>
	startsWith	$\exists \text{ startsWith.T} \subset \text{rdf:Literal}$ $\text{T} \subset \forall \text{ startsWith.CodeVariable}$
	definedBy	$\exists \text{ definedBy.T} \subset \text{rdf:Literal}$ $\text{T} \subset \forall \text{ definedBy.CodeVariable}$
	<b>Data Properties</b>	<b>Descriptive Logic</b>
	rdf:Literal	$\exists \text{ definedBy.T} \subset \text{rdfs:Literal}$ $\text{T} \subset \forall \text{ definedBy.CodeVariable}$
	xsd:anyURI	$\exists \text{ someValue.T} \subset \text{CodeVariable}$ $\text{T} \subset \forall \text{ someValue.xsd:anyURI}$
<b>CodeLoop</b>	<b>Subclass Relations</b> $\text{forLoop} \subset \text{CodeLoop}$ $\text{forEach} \subset \text{CodeLoop}$ $\text{whileLoop} \subset \text{CodeLoop}$	

### 4.2.1 DL for Tags in PHP Semantics

Php uses starting and ending tag for scripting. Descriptive logic of these tags for Ontology is detailed in table 4.5.

### 4.2.2 DL for Conditional Statements in PhpSemantics

Conditional statements play a vital role in any programming language. They are used in loops, building conditions in the flow and many more. For PhpSemantics, descriptive logic to detail conditional statement and its relations is shown in table 4.6.

Table 4.5: Descriptive Logic for PHP Tags

<b>CodeTag</b>	<b>Subclass Relations</b> startTag $\subset$ CodeTag endTag $\subset$ CodeTag	
	<b>Object Properties</b>	<b>Descriptive Logic</b>
	hasEndTag	$\exists$ hasEndTag.T $\subset$ endTag $T \subset \forall$ hasEndTag.CodeTag
	hasStartTag	$\exists$ hasStartTag.T $\subset$ startTag $T \subset \forall$ hasStartTag.CodeTag
	<b>Data Properties</b>	<b>Descriptive Logic</b>
	rdf:Literal	$\exists$ endTag.T $\subset$ rdfs:Literal $T \subset \forall$ hasEndTag.rdfs:Literal
rdf:Literal	$\exists$ startTag.T $\subset$ rdfs:Literal $T \subset \forall$ hasStartTag.rdfs:Literal	

Table 4.6: Descriptive Logic for PHP Tags

<b>CodeCondStatement</b>	<b>Subclass Relations</b> if $\subset$ CodeCondStatement ifElse $\subset$ CodeCondStatement	
	<b>Object Properties</b>	<b>Descriptive Logic</b>
	hasCondition	$\exists$ hasCondition.T $\subset$ condition $T \subset \forall$ hasCondition.CodeCondStatement
	<b>Data Properties</b>	<b>Descriptive Logic</b>
	xsd:Boolean	$\exists$ condition.T $\subset$ CodeCondStatement $T \subset \forall$ condition.xsd:Boolean

### 4.2.3 DL for Loops in PhpSemantics

Like ProgLanguage, class of Loop in PHP can be divided into three subclasses. These subclasses are of forLoop, forEach and whileLoop. They are further detailed in the following table 4.7.

Table 4.7: Descriptive Logic for Loops in Php Semantics

<b>forEach</b>	<b>Subclass Relations</b> forEach $\subset$ CodeLoop	
	<b>Data Properties</b>	<b>Descriptive Logic</b>
	xsd:anyURI	$\exists$ contains.T $\subset$ forEach T $\subset$ $\forall$ contains.xsd:anyURI
	xsd:integer	$\exists$ iteratedBy.T $\subset$ forEach T $\subset$ $\forall$ iteratedBy.xsd:integer
	<b>Object Properties</b>	<b>Descriptive Logic</b>
	hasOne	$\exists$ hasOne.T $\subset$ VarCollection T $\subset$ $\forall$ hasOne.forEach
	isTypeOf	$\exists$ isTypeOf.T $\subset$ CodeLoop T $\subset$ $\forall$ isTypeOf.forLoop
isIteratedBy	$\exists$ isIteratedBy.T $\subset$ CodeVariable T $\subset$ $\forall$ isIteratedBy.forEach	
<b>forLoop</b>	<b>Subclass Relations</b> forLoop $\subset$ CodeLoop	
	<b>Data Properties</b>	<b>Descriptive Logic</b>
	xsd:integer	$\exists$ iteratedBy.T $\subset$ forLoop T $\subset$ $\forall$ iteratedBy.xsd:integer
	<b>Object Properties</b>	<b>Descriptive Logic</b>
	hasTerminationCond	$\exists$ hasTerminationCond.T $\subset$ CodeCondStatement T $\subset$ $\forall$ hasTerminationCond.forLoop
isTypeOf	$\exists$ isTypeOf.T $\subset$ CodeLoop T $\subset$ $\forall$ isTypeOf.forLoop	
isIteratedBy	$\exists$ isIteratedBy.T $\subset$ CodeVariable T $\subset$ $\forall$ isIteratedBy.forLoop	
<b>whileLoop</b>	<b>Subclass Relations</b> whileLoop $\subset$ CodeLoop	
	<b>Data Properties</b>	<b>Descriptive Logic</b>
	xsd:Boolean	$\exists$ isTerminated.T $\subset$ whileLoop T $\subset$ $\forall$ isTerminatedBy.xsd:Boolean
	<b>Object Properties</b>	<b>Descriptive Logic</b>
hasTerminationCond	$\exists$ hasTerminationCond.T $\subset$ CodeCondStatement T $\subset$ $\forall$ hasTerminationCond.whileLoop	
isTypeOf	$\exists$ isTypeOf.T $\subset$ CodeLoop T $\subset$ $\forall$ isTypeOf.whileLoop	

### 4.3 Descriptive Logic for ASP Concepts

Based on ProgLanguage semantics, ASP Semantics are formulated which are very alike with PHP. Classes, subclasses and their relationships of ASPSemantics concepts are

detailed in the table 4.8 below.

Table 4.8: Descriptive Logic for ASP Semantics Model

<b>ASPSemantics</b>	<p><b>Subclass Relations</b></p> <p>CodeCondStatement <math>\subset</math> ASPSemantics  CodeLoop <math>\subset</math> ASPSemantics  CodePrint <math>\subset</math> ASPSemantics  CodeTag <math>\subset</math> ASPSemantics  CodeVariable <math>\subset</math> ASPSemantics</p>	
<b>CodeVariable</b>	<p><b>Subclass Relations</b></p> <p>CodeVariable <math>\subset</math> ASPSemantics  VarCollection <math>\subset</math> CodeVariable  Variable <math>\subset</math> CodeVariable</p>	
	<b>Object Properties</b>	<b>Descriptive Logic</b>
	startsWith	$\exists$ startsWith.T $\subset$ rdf:Literal $T \subset \forall$ startsWith.CodeVariable
	definedBy	$\exists$ definedBy.T $\subset$ rdf:Literal $T \subset \forall$ definedBy.CodeVariable
	<b>Data Properties</b>	<b>Descriptive Logic</b>
rdf:Literal	$\exists$ definedBy.T $\subset$ rdfs:Literal $T \subset \forall$ definedBy.CodeVariable	
xsd:anyURI	$\exists$ someValue.T $\subset$ CodeVariable $T \subset \forall$ someValue.xsd:anyURI	
<b>CodeLoop</b>	<p><b>Subclass Relations</b></p> <p>forLoop <math>\subset</math> CodeLoop  forEach <math>\subset</math> CodeLoop  whileLoop <math>\subset</math> CodeLoop</p>	

### 4.3.1 DL for Tags in ASP Semantics

Php uses starting and ending tag for scripting. Descriptive logic of these tags for Ontology is detailed in table 4.9.

### 4.3.2 DL for Loops in ASP Semantics

As defined in previous section, ASP loops are defined in almost similar way. Following table 4.10 shows descriptive logic and relations of different loops used.



Table 4.9: Descriptive Logic for ASP Tags

<b>CodeTag</b>	<b>Subclass Relations</b> startTag $\subset$ CodeTag endTag $\subset$ CodeTag	
	<b>Object Properties</b>	<b>Descriptive Logic</b>
	hasEndTag	$\exists$ hasEndTag.T $\subset$ endTag T $\subset \forall$ hasEndTag.CodeTag
	hasStartTag	$\exists$ hasStartTag.T $\subset$ startTag T $\subset \forall$ hasStartTag.CodeTag
	<b>Data Properties</b>	<b>Descriptive Logic</b>
	rdf:Literal	$\exists$ endTag.T $\subset$ rdfs:Literal T $\subset \forall$ hasEndTag.rdfs:Literal
	rdf:Literal	$\exists$ startTag.T $\subset$ rdfs:Literal T $\subset \forall$ hasStartTag.rdfs:Literal

### 4.3.3 DL for Conditional Statement in ASP Semantic

Conditional statements are essential for coding and developing application. In table 4.11, shows the DL for conditional statement in ASP.

Table 4.10: Descriptive Logic for Loops in ASP Semantics

<b>forEach</b>	<b>Subclass Relations</b> forEach $\subset$ CodeLoop	
	<b>Data Properties</b>	<b>Descriptive Logic</b>
	xsd:anyURI	$\exists$ contains.T $\subset$ forEach $T \subset \forall$ contains.xsd:anyURI
	xsd:integer	$\exists$ iteratedBy.T $\subset$ forEach $T \subset \forall$ iteratedBy.xsd:integer
	<b>Object Properties</b>	<b>Descriptive Logic</b>
	hasOne	$\exists$ hasOne.T $\subset$ VarCollection $T \subset \forall$ hasOne.forEach
	isTypeOf	$\exists$ isTypeOf.T $\subset$ CodeLoop $T \subset \forall$ isTypeOf.forEach
isIteratedBy	$\exists$ isIteratedBy.T $\subset$ CodeVariable $T \subset \forall$ isIteratedBy.forEach	
<b>forLoop</b>	<b>Subclass Relations</b> forLoop $\subset$ CodeLoop	
	<b>Data Properties</b>	<b>Descriptive Logic</b>
	xsd:integer	$\exists$ iteratedBy.T $\subset$ forLoop $T \subset \forall$ iteratedBy.xsd:integer
	<b>Object Properties</b>	<b>Descriptive Logic</b>
	hasTerminationCond	$\exists$ hasTerminationCond.T $\subset$ CodeCondStatement $T \subset \forall$ hasTerminationCond.forLoop
isTypeOf	$\exists$ isTypeOf.T $\subset$ CodeLoop $T \subset \forall$ isTypeOf.forLoop	
isIteratedBy	$\exists$ isIteratedBy.T $\subset$ CodeVariable $T \subset \forall$ isIteratedBy.forLoop	
<b>whileLoop</b>	<b>Subclass Relations</b> whileLoop $\subset$ CodeLoop	
	<b>Data Properties</b>	<b>Descriptive Logic</b>
	xsd:Boolean	$\exists$ isTerminated.T $\subset$ whileLoop $T \subset \forall$ isTerminatedBy.xsd:Boolean
	<b>Object Properties</b>	<b>Descriptive Logic</b>
hasTerminationCond	$\exists$ hasTerminationCond.T $\subset$ CodeCondStatement $T \subset \forall$ hasTerminationCond.whileLoop	
isTypeOf	$\exists$ isTypeOf.T $\subset$ CodeLoop $T \subset \forall$ isTypeOf.whileLoop	

Table 4.11: Descriptive Logic for PHP Tags

<b>CodeCondStatement</b>	<b>Subclass Relations</b>	
	if $\subset$ CodeCondStatement	
	ifElse $\subset$ CodeCondStatement	
	<b>Object Properties</b>	<b>Descriptive Logic</b>
	hasCondition	$\exists$ hasCondition.T $\subset$ conditon T $\subset$ $\forall$ hasCondition.CodeCondStatement
<b>Data Properties</b>	<b>Descriptive Logic</b>	
xsd:Boolean	$\exists$ condition.T $\subset$ CodeCondStatement T $\subset$ $\forall$ condition.xsd:Boolean	

## Chapter 5

# Results and Analysis

SPARQL is an RDF query language and protocol produced by the W3C RDF Data Access Working Group (DAWG). It was released as a W3C<sup>1</sup> Recommendation in January of 2008<sup>2</sup>. By using SPARQL, queries and results are formulated from the Ontologies defined in Chapter 03. These queries are based on object properties between classes and subclasses. Object properties for ProgLanguage are shown in figure 5.1.

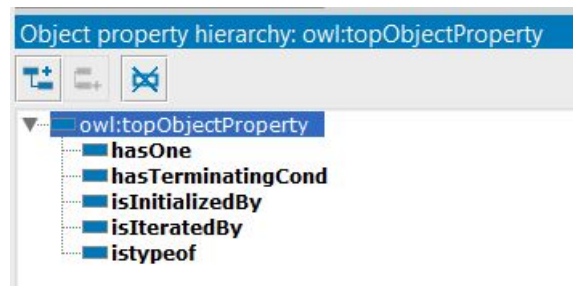


Figure 5.1: Object Properties of ProgLanguage Ontology

Based on ProgLanguage, PHP and ASP semantics were formulated. Both ontologies have same object properties, as they have same structure of formulation. These object properties are shown in figure 5.2.

Query results from SPARQL are in the form of triple based on object, predicate and subject. In the following sections, few results of SPARQL queries on Ontologies defined in Chapter 3 are shown.

<sup>1</sup>website: <https://www.w3.org>

<sup>2</sup>SPARQL, W3C [Online]. Available: <https://www.w3.org/wiki/SPARQL>. [Accessed 12 December 2018].

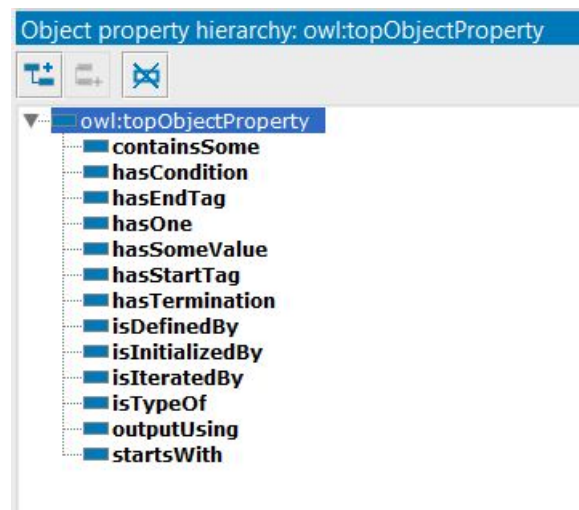


Figure 5.2: Object Properties of PHP and ASP

## 5.1 Queries on Ontologies

Queries executed on Ontologies of ASP and PHP are listed below with results obtained. The structure and results of these queries are same due to similarities in the Ontology.

### 5.1.1 containsSome

This query is used to extract information about subjects and classes that have object property as containsSome. Results for ASP and PHP are shown in figures 5.3 and 5.4, respectively.

The results for ASP are read as:

- Variable containsSome dataValue as varData and data.
- VarCollection containsSome dataCollection as dataValue.
- CodeVariable containsSome variable as varData and dataCollection

The results for PHP are read as:

- Variable containsSome dataValue as varData and data.
- VarCollection containsSome dataCollection as dataValue.
- CodeVariable containsSome variable as varData and dataCollection
- Print and Echo containsSome variable to be printed.

SPARQL query:

```

PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX ex: <http://www.semanticweb.org/afnan-pc/ontologies/2019/ASPSemanticsOnto#>
SELECT ?subject ?object ?class
  WHERE { ?subject ex:containsSome ?object.
          ?subject rdf:type ?class
          Filter (?class !=owl:NamedIndividual).
          Filter (?class !=owl:DatatypeProperty).
          Filter (?class !=owl:Class)      }

```

subject	object	class
varData	dataValue	Variable
dataCollection	dataValue	VarCollection
variable	varData	CodeVariable
variable	dataCollection	CodeVariable

Figure 5.3: containsSome for ASP

SPARQL query:

```

PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
prefix ex: <http://www.semanticweb.org/afnan-pc/ontologies/2019/PhpSemanticsOnto#>
SELECT ?subject ?object ?Loop
  WHERE { ?subject ex:containsSome ?object.
          ?subject rdf:type ?Loop
          Filter (?Loop !=owl:NamedIndividual)
          }

```

subject	object	Loop
varData	dataValue	Variable
dataCollection	dataValue	VarCollection
variable	varData	CodeVariable
print	variable	Print
echo	variable	Echo
variable	dataCollection	CodeVariable

Figure 5.4: containsSome for PHP

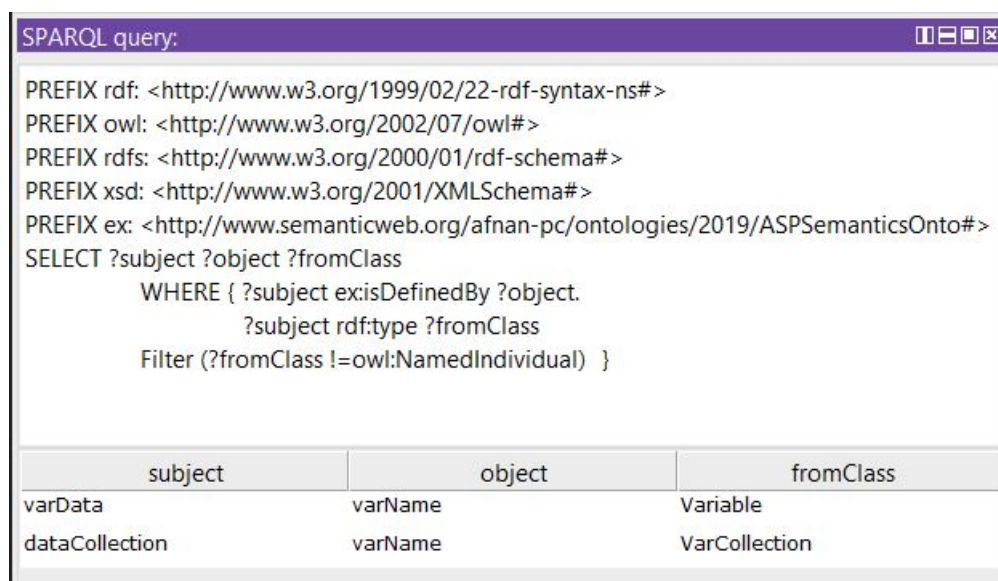
### 5.1.2 isDefinedBy

The object property named as `isDefinedBy` relates to names given for variables and data collections. Query results for this property are shown in figure 5.5 and 5.6 for ASP and PHP, respectively. The results for ASP are read as:

- Variable and `varCollection` `isDefinedBy` `varName`.

The results for PHP are read as:

- Variable and `varCollection` `isDefinedBy` `varName`.



The screenshot shows a SPARQL query window with a purple title bar. The query text is as follows:

```

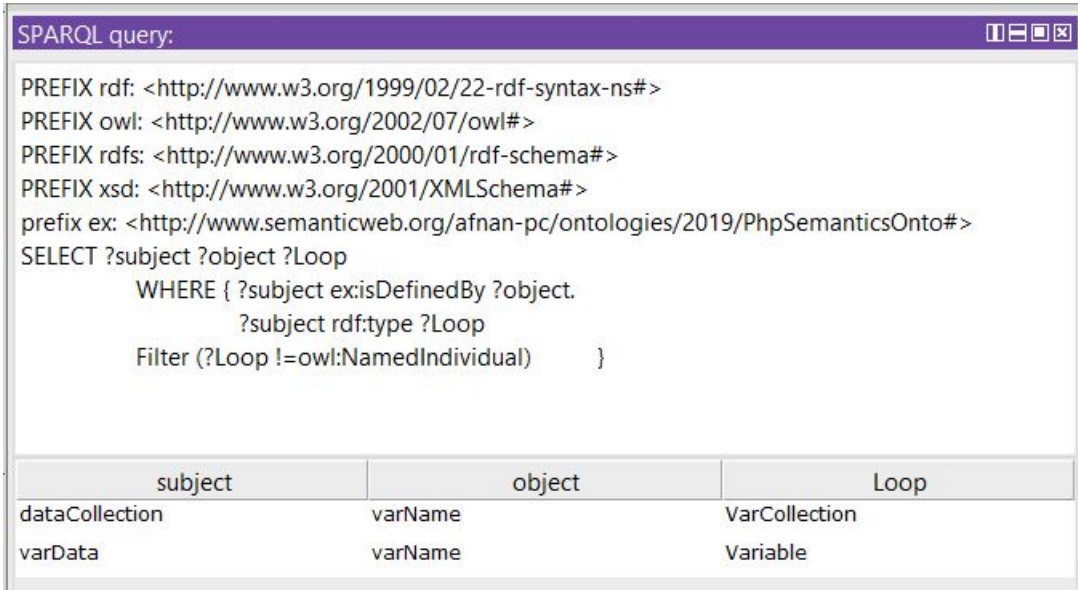
SPARQL query:
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX ex: <http://www.semanticweb.org/afnan-pc/ontologies/2019/ASPSemanticsOnto#>
SELECT ?subject ?object ?fromClass
  WHERE { ?subject ex:isDefinedBy ?object.
          ?subject rdf:type ?fromClass
          Filter (?fromClass !=owl:NamedIndividual) }

```

Below the query, a table displays the results:

subject	object	fromClass
varData	varName	Variable
dataCollection	varName	VarCollection

Figure 5.5: `isDefinedBy` ASP



SPARQL query:

```

PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
prefix ex: <http://www.semanticweb.org/afnan-pc/ontologies/2019/PhpSemanticsOnto#>
SELECT ?subject ?object ?Loop
  WHERE { ?subject ex:isDefinedBy ?object.
          ?subject rdf:type ?Loop
          Filter (?Loop !=owl:NamedIndividual)  }

```

subject	object	Loop
dataCollection	varName	VarCollection
varData	varName	Variable

Figure 5.6: isDefinedBy PHP

### 5.1.3 isTypeOf

The property isTypeOf is used make subclasses of loops. I have used three loops in my ontologies, which are shown in the results below as figure 5.7 and figure 5.8. The results for ASP are read as:

- whileLoop isTypeOf Loop.
- forEach isTypeOf Loop.
- for isTypeOf Loop.

The results for PHP are read as:

- whileLoop isTypeOf Loop.
- forEach isTypeOf Loop.
- for isTypeOf Loop.



SPARQL query:

```

PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX ex: <http://www.semanticweb.org/afnan-pc/ontologies/2019/ASPSemanticsOnto#>
SELECT ?subject ?object ?Loop
  WHERE { ?subject ex:isTypeOf ?object.
          ?subject rdf:type ?Loop
          Filter (?Loop !=owl:NamedIndividual).
          Filter (?Loop !=owl:DatatypeProperty).
          Filter (?Loop !=owl:Class)      }

```

subject	object	Loop
while	Loop	whileLoop
forEach	Loop	forEach
forLoop	Loop	forLoop

Figure 5.7: isTypeOf ASP

Individuals by class x DL Query x OntoGraf x SPARQL Query x

Active Ontology x Entities x

SPARQL query:

```

PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
prefix ex: <http://www.semanticweb.org/afnan-pc/ontologies/2019/PhpSemanticsOnto#>
SELECT ?subject ?object ?Loop
  WHERE { ?subject ex:isTypeOf ?object.
          ?subject rdf:type ?Loop
          Filter (?Loop !=owl:NamedIndividual).
          Filter (?Loop != owl:DatatypeProperty).
          Filter (?Loop != owl:Class)
          }

```

subject	object	Loop
while	Loop	whileLoop
forLoop	Loop	forLoop
forEach	Loop	forEach

Figure 5.8: isTypeOf PHP

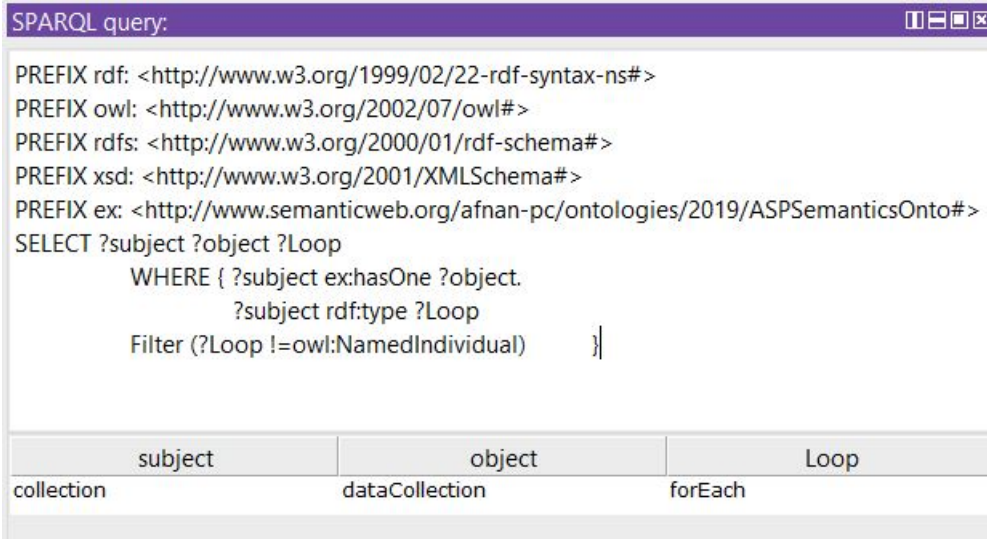
### 5.1.4 hasOne

The property `hasOne` refers to an iterator or a collection used in any loop. Results shows that in PHP, `forLoop` hasOne iterator as `varData` and in ASP, `forEach` hasOne collection as `DataCollection`. Figures 5.9 and 5.10 show execution of query. The results for ASP are read as:

- `forEach` hasOne collection as `dataCollection`

The results for PHP are read as:

- `forLoop` hasOne iterator as `varData`.



The screenshot shows a SPARQL query editor window with a purple title bar. The query text is as follows:

```
SPARQL query:
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX ex: <http://www.semanticweb.org/afnan-pc/ontologies/2019/ASPSemanticsOnto#>
SELECT ?subject ?object ?Loop
  WHERE { ?subject ex:hasOne ?object.
          ?subject rdf:type ?Loop
          Filter (?Loop !=owl:NamedIndividual)    }|
```

Below the query, a table displays the results:

subject	object	Loop
collection	dataCollection	forEach

Figure 5.9: hasOne ASP

The screenshot shows a SPARQL query editor window with a purple title bar. The query text is as follows:

```

SPARQL query:
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
prefix ex: <http://www.semanticweb.org/afnan-pc/ontologies/2019/PhpSemanticsOnto#>
SELECT ?subject ?object ?Loop
  WHERE { ?subject ex:isIteratedBy ?object.
          ?subject rdf:type ?Loop
          Filter (?Loop !=owl:NamedIndividual)
        }

```

Below the query text is a table with three columns: 'subject', 'object', and 'Loop'. The first row of data shows 'iterator' under 'subject', 'varData' under 'object', and 'forLoop' under 'Loop'.

subject	object	Loop
iterator	varData	forLoop

Figure 5.10: hasOne PHP

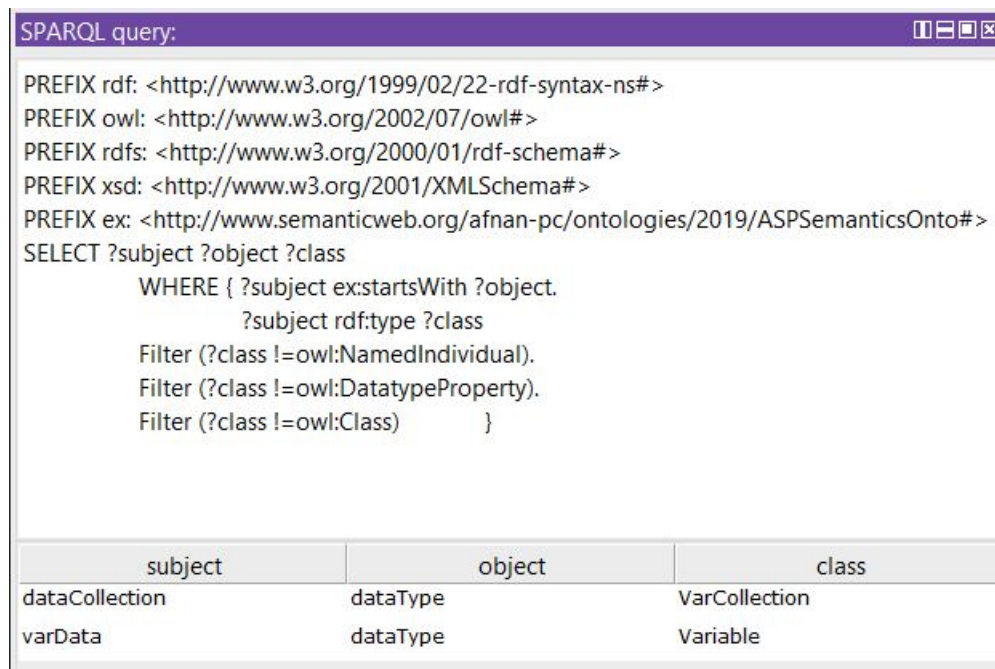
### 5.1.5 startsWith

Variables and data collections have some datatype or some sign in the start. The property startsWith refers to the dataType and dollar sign used in ASP and PHP, respectively. Query and results are shown in figure 5.11 and 5.12. The results for ASP are read as:

- dataCollection and varData startsWith dataType.

The results for PHP are read as:

- varData and dataCollection startsWith dollarSign (\$).



SPARQL query:

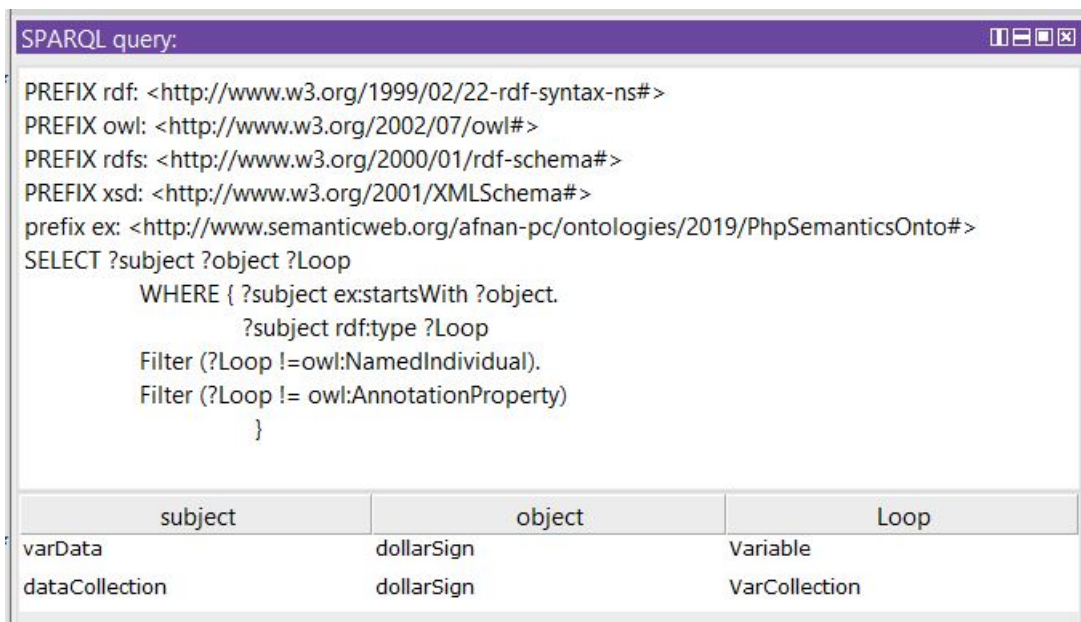
```

PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX ex: <http://www.semanticweb.org/afnan-pc/ontologies/2019/ASPSemanticsOnto#>
SELECT ?subject ?object ?class
  WHERE { ?subject ex:startsWith ?object.
          ?subject rdf:type ?class
          Filter (?class !=owl:NamedIndividual).
          Filter (?class !=owl:DatatypeProperty).
          Filter (?class !=owl:Class)      }

```

subject	object	class
dataCollection	dataType	VarCollection
varData	dataType	Variable

Figure 5.11: startsWith ASP



SPARQL query:

```

PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
prefix ex: <http://www.semanticweb.org/afnan-pc/ontologies/2019/PhpSemanticsOnto#>
SELECT ?subject ?object ?Loop
  WHERE { ?subject ex:startsWith ?object.
          ?subject rdf:type ?Loop
          Filter (?Loop !=owl:NamedIndividual).
          Filter (?Loop != owl:AnnotationProperty)
          }

```

subject	object	Loop
varData	dollarSign	Variable
dataCollection	dollarSign	VarCollection

Figure 5.12: startsWith PHP

### 5.1.6 hasTermination

For and while loops have some termination conditions to step out of loop execution. For this, hasTermination property was added. Queries and executed results are shown in

figures 5.13 and 5.14. The results for ASP are read as:

- forLoop hasTermination condition as termination.
- whileLoop hasTermination condition as terminationCond.

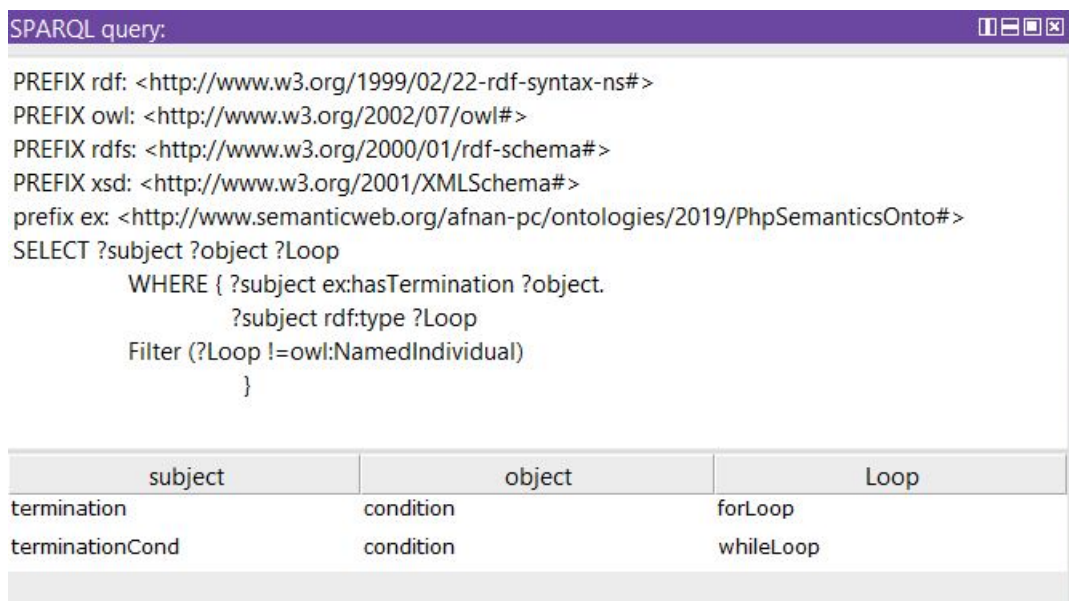
The results for PHP are read as:

- forLoop hasTermination condition as termination.
- whileLoop hasTermination condition as terminationCond.

```
SPARQL query:
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX ex: <http://www.semanticweb.org/afnan-pc/ontologies/2019/ASPSemanticsOnto#>
SELECT ?subject ?object ?Loop
  WHERE { ?subject ex:hasTermination ?object.
          ?subject rdf:type ?Loop
          Filter (?Loop !=owl:NamedIndividual)
        }
```

subject	object	Loop
termination	condition	forLoop
terminationCond	condition	whileLoop

Figure 5.13: hasTermination ASP



SPARQL query:

```

PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
prefix ex: <http://www.semanticweb.org/afnan-pc/ontologies/2019/PhpSemanticsOnto#>
SELECT ?subject ?object ?Loop
  WHERE { ?subject ex:hasTermination ?object.
         ?subject rdf:type ?Loop
         Filter (?Loop !=owl:NamedIndividual)
        }

```

subject	object	Loop
termination	condition	forLoop
terminationCond	condition	whileLoop

Figure 5.14: hasTermination PHP

### 5.1.7 isInitializedBy

Initializing of For loop is done by some variable. This can be defined by using isInitialziedBy property. Query and execution results are shown in figures 5.15 and 5.16. The results for ASP are read as:

- forLoop has startValue which isInitializedBy varData.

The results for PHP are read as:

- forLoop has startValue which isInitializedBy varData.

```

SPARQL query:
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX ex: <http://www.semanticweb.org/afnan-pc/ontologies/2019/ASPSemanticsOnto#>
SELECT ?subject ?object ?Loop
  WHERE { ?subject ex:isInializedBy ?object.
          ?subject rdf:type ?Loop
          Filter (?Loop !=owl:NamedIndividual)
          Filter (?Loop !=owl:DatatypeProperty)  }

```

subject	object	Loop
startValue	varData	forLoop

Figure 5.15: isInitiazliedBy ASP

```

SPARQL query:
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
prefix ex: <http://www.semanticweb.org/afnan-pc/ontologies/2019/PhpSemanticsOnto#>
SELECT ?subject ?object ?Loop
  WHERE { ?subject ex:isInializedBy ?object.
          ?subject rdf:type ?Loop
          Filter (?Loop !=owl:NamedIndividual).
          Filter (?Loop != owl:DatatypeProperty)
          }

```

subject	object	Loop
startValue	varData	forLoop

Figure 5.16: isInitiazliedBy PHP

## 5.2 Analysis and Comparison

The Ontologies, which are described and queries in above chapters and sections, can be validated using OntoClean and compared with an Ontology formulated by Yahui et.al [4] for classes and subclasses of Ontologies modeled in this research.

### 5.2.1 Analysis using OntoClean

OntoClean is a methodology for analyzing ontologies based on formal, domain-independent properties of classes developed by Nicola Guarino and Chris Welty [3]. OntoClean have four basic notations, named as Essence, Rigidity, Identity and Unity. OntoClean was first proposed by Guarino and Welty in 2000 with main purpose of formal foundation for ontological analysis. Keeping in view the study of [13], I have formulated taxonomic structure of my Ontologies as shown in Figures 5.17, 5.18 and 5.19.

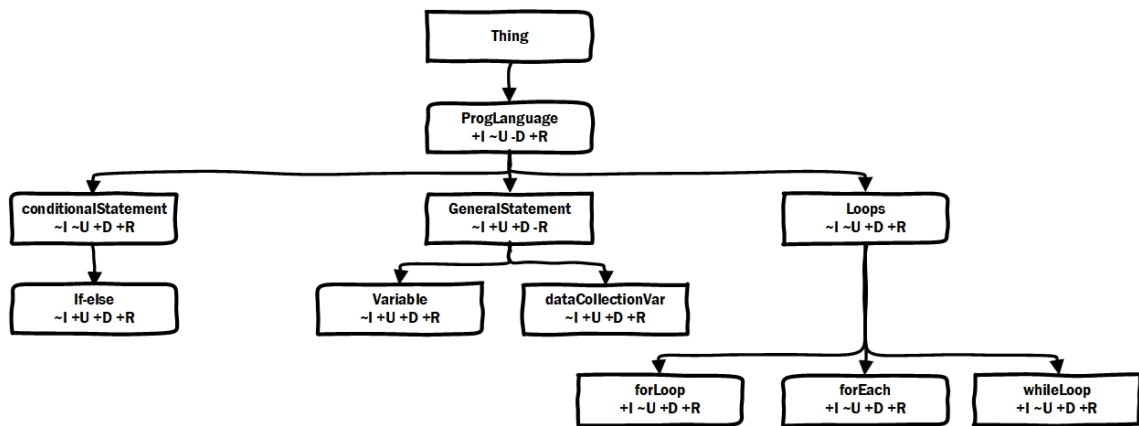


Figure 5.17: OntoClean for ProgLanguage

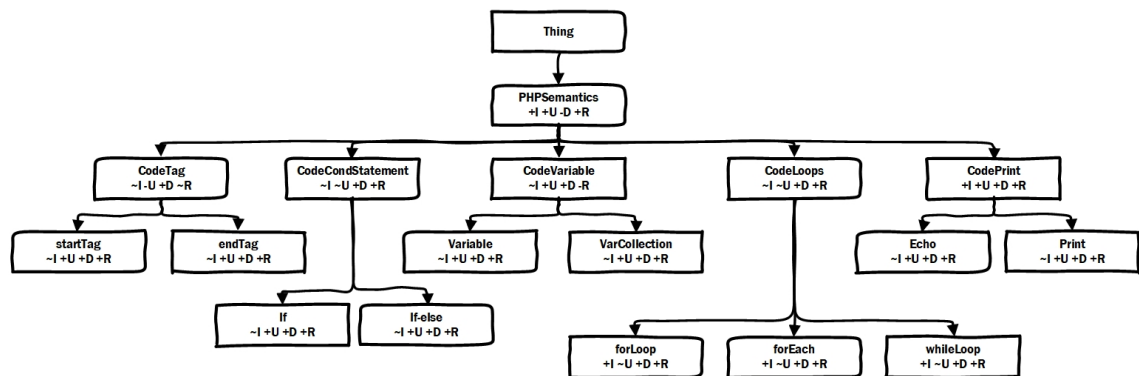


Figure 5.18: OntoClean for PHPSemantics

### 5.2.2 Comparison with previous studies

In research by Yahui et.al [4], they formulated an ontology of C programming language for purpose of learning and deep understanding. They discussed classes and sub classes along with work flow of C programming language, basic concepts of Ontology is shown in figure 5.20.



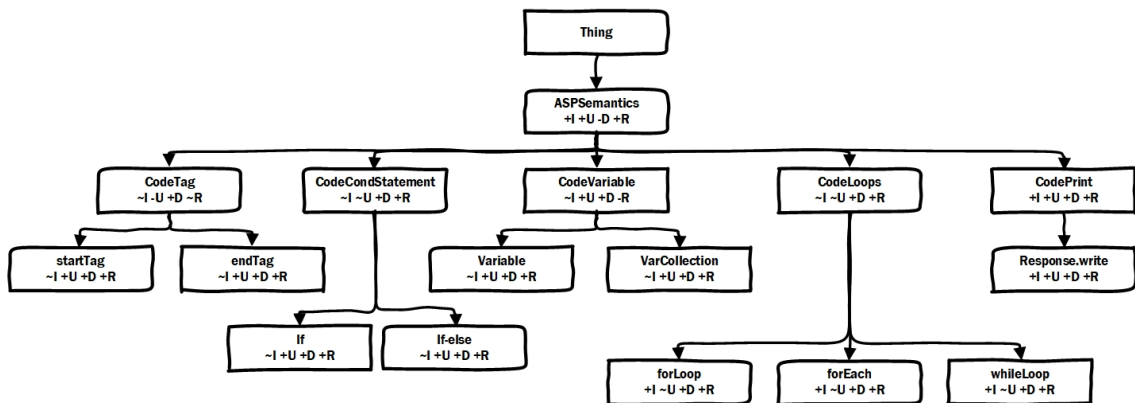


Figure 5.19: OntoClean for ASPSemantics

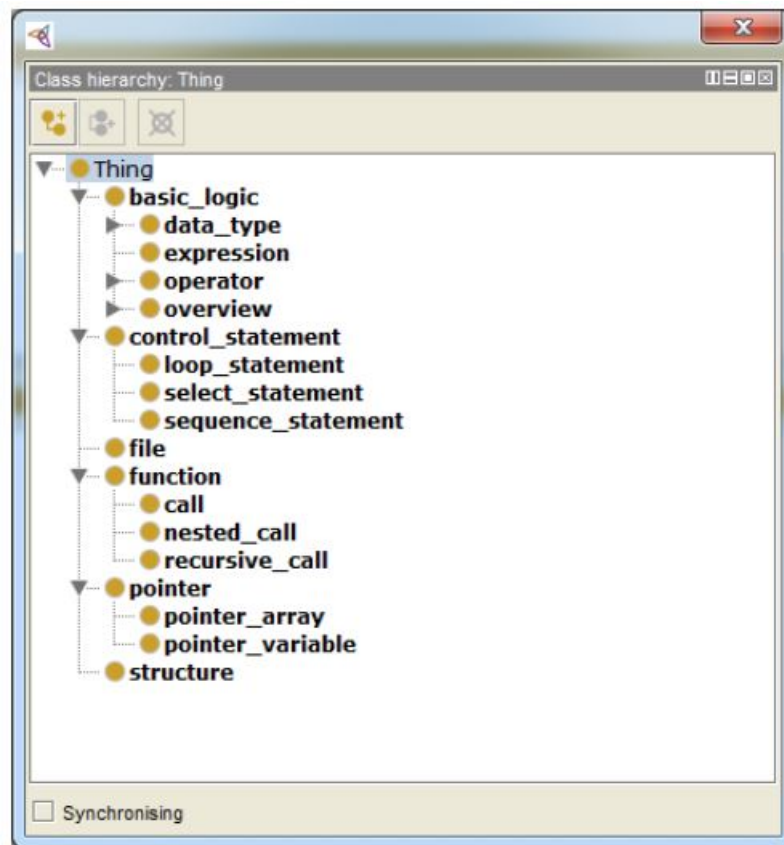


Figure 5.20: Basic concepts of C Programming Language

As shown in figure 5.20, classes and sub classes have some similarities such as loops and variable definition. There are more than few differences such as function calls, structure statements, filing, pointers etc.

# Chapter 6

## Conclusions and Future Work

### 6.1 Conclusion

Ontology and semantic web is a growing field these days. It is used in relationship based applications integrated with a data-set or a database. RDF graphs are used for visualizing and describing relationships in Ontologies. They can be used for comparison of different languages which can be computer programming languages or web/mobile development frameworks. In this research, ASP and PHP are selected as web development frameworks. ASP and PHP, both frameworks work on similar structure yet different syntactical formats.

Parsing from one language into another language is easier when you know the whole working and structure of the language. ASP and PHP have quiet similar programming structure and also same work flow but different syntax for writing their codes. Ontologies can be used in multiple ways for relating different concepts and managing them. For this purpose, I have made three ontologies, named as General ProgLanguage, PHP Semantics and ASP Semantics. General ProgLanguage is used as base ontology for developing other two ontologies of PHP and ASP. PHP and ASP have same concepts with different definition of classes and subclasses.

These Ontologies are further used for different SPARQL queries for relating data and object properties of ASP and PHP for getting results. Limitation of this research is interface, I have not used any interface for better representation of these ontologies and their queries. Furthermore, we have to make complete and meaningful semantics for better relationships, as if we miss one concept or make any mistake in one of these relations it won't give foreseeable results.

## **6.2 Future Work**

In future, I desire to make a proper interface using Jena API and integrating these semantics with Jena API by making an application using Java. Furthermore, this can be done for many other available programming languages.

# Bibliography

- [1] IAO Abuhassan and Akram MO AlMashaykhi. Domain ontology for programming languages. *Journal of Computations & Modelling*, 2(4):75–91, 2012. Cited on p. [13](#).
- [2] D Amar Bensaber, Djamel Benslimane, and Mimoun Malki. Ontology development for web services: Reverse engineering approach. In *2008 Second International Conference on Research Challenges in Information Science*, pages 433–440. IEEE, 2008. Cited on pp. [7](#) and [13](#).
- [3] Nicola Guarino and Christopher A Welty. An overview of ontoclean. In *Handbook on ontologies*, pages 151–171. Springer, 2004. Cited on p. [44](#).
- [4] Yahui Hu, Yamin Hu, and Lejiang Guo. Construction of c programming language based on ontology knowledge base. In *Proceedings of the Second International Conference on Innovative Computing and Cloud Computing*, page 23. ACM, 2013. Cited on pp. [11](#), [15](#), [43](#), and [44](#).
- [5] Meltem Yıldırım İmamoğlu and Deniz Çetinkaya. A rule based decision support system for programming language selection. In *2017 2nd International Conference on Knowledge Engineering and Applications (ICKEA)*, pages 71–75. IEEE, 2017. Cited on pp. [7](#) and [8](#).
- [6] Haruhiko Kaiya and Motoshi Saeki. Ontology based requirements analysis: lightweight semantic processing approach. In *Fifth International Conference on Quality Software (QSIC'05)*, pages 223–230. IEEE, 2005. Cited on pp. [6](#), [13](#), and [14](#).
- [7] Shipra Ravi Kumar, Ravi Sharma, and Keshav Gupta. Strategies for web application development methodologies. In *2016 International Conference on Computing, Communication and Automation (ICCCA)*, pages 160–165. IEEE, 2016. Cited on p. [11](#).

- [8] Omer Levy, Anders Søgaard, and Yoav Goldberg. A strong baseline for learning cross-lingual word embeddings from sentence alignments. *arXiv preprint arXiv:1608.05426*, 2016. Cited on pp. 8 and 9.
- [9] Christos Pierrakeas, Georgia Solomou, and Achilles Kameas. An ontology-based approach in learning programming languages. In *2012 16th Panhellenic Conference on Informatics*, pages 393–398. IEEE, 2012. Cited on pp. 11, 12, and 15.
- [10] Juan-Pablo Posadas-Durán, Iliia Markov, Helena Gómez-Adorno, Grigori Sidorov, Ildar Batyrshin, Alexander Gelbukh, and Obdulia Pichardo-Lagunas. Syntactic n-grams as features for the author profiling task. *Working Notes Papers of the CLEF*, 2015. Cited on pp. 9 and 14.
- [11] Monica Shekhar et al. Semantic web search based on ontology modeling using protege reasoner. *arXiv preprint arXiv:1305.5827*, 2013. Cited on p. 12.
- [12] J Shen, Y Yang, C Zhu, and C Wan. From bpel4ws to owl-s: Integrating e-business process definitions. In *Proceedings of the 3rd International Conference on Web Services (ICWS)*, 2005. Cited on p. 7.
- [13] Chris Welty. Ontowclean: Cleaning owl ontologies with owl. In *FOIS*, volume 150, pages 347–359, 2006. Cited on p. 44.
- [14] Pornpit Wongthongtham, Elizabeth Chang, Tharam Dillon, and Ian Sommerville. Development of a software engineering ontology for multisite software development. *IEEE Transactions on knowledge and Data Engineering*, 21(8):1205–1217, 2008. Cited on pp. 7 and 14.
- [15] Shuxin Zhao, Elizabeth Chang, and Tharam Dillon. Knowledge extraction from web-based application source code: An approach to database reverse engineering for ontology development. In *2008 IEEE International Conference on Information Reuse and Integration*, pages 153–159. IEEE, 2008. Cited on pp. 9 and 14.

## Thesis

### ORIGINALITY REPORT

**18%**

SIMILARITY INDEX

**14%**

INTERNET SOURCES

**11%**

PUBLICATIONS

**6%**

STUDENT PAPERS

### PRIMARY SOURCES

<b>1</b>	<b>www.scienpress.com</b> Internet Source	<b>2%</b>
<b>2</b>	<b>espace.curtin.edu.au</b> Internet Source	<b>2%</b>
<b>3</b>	<b>arxiv.org</b> Internet Source	<b>2%</b>
<b>4</b>	<b>eprints.bournemouth.ac.uk</b> Internet Source	<b>2%</b>
<b>5</b>	<b>ceur-ws.org</b> Internet Source	<b>2%</b>
<b>6</b>	Shipra Ravi Kumar, Ravi Sharma, Keshav Gupta. "Strategies for web application development methodologies", 2016 International Conference on Computing, Communication and Automation (ICCCA), 2016 Publication	<b>1%</b>
<b>7</b>	D. Amar Bensaber, D. Benslimane, M. Malki. "Ontology Development for Web Services: Reverse engineering approach", 2008 Second	<b>1%</b>