ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645
**Indian Journal of Science and Technology,** *Vol 9(38), DOI: 10.17485/ijst/2016/v9i38/102970, October 2016*

# SWOT Analysis of Ontology Driven Software Engineering

**M. P. S. Bhatia[1], Akshi Kumar[2] and Rohit Beniwal[1]***

[1]Division of Computer Engineering, Netaji Subhas Institute of Technology, New Delhi - 110078, Delhi, India; mpsbhatia@nsit.ac.in, rohitbeniwal@yahoo.co.in
[2]Department of Computer Engineering, Delhi Technological University, Main Bawana Road - 110042, Delhi, India; akshi.kumar@gmail.com

## Abstract

In the past decade offshoring and outsourcing the software development phenomenon has been undeniably a key software engineering practice. The need to adapt to this new reality is obvious and is bound to have a long lasting influence on the software industry. This fosters the industry and researchers to look for intelligent supporting technologies and tools that can help interconnect and exchange Software Engineering knowledge. A rising trend to exploit ontologies for sharing and reusing information across web is well recognized. We examine the strategic alignment of ontologies to Software Engineering where the former can be used to improve and assist in intelligent software development process. The SWOT (Strengths, Weaknesses, Opportunities, and Threats) analysis is presented giving an insight to the use of ontologies to enrich and enhance Software Engineering processes.

**Keywords:** Ontology, Ontology Driven, SWOT Analysis, Semantic Web, Software Engineering

## 1. Introduction

"Semantic Web provides a common framework that allows data to be shared and reused across application, enterprise, and community boundaries"[1]. It is a machine processable "Web of Data"[2]. Semantic Web stack illustrates the architecture of the Semantic Web that encompasses of several languages or technologies. At the core of architecture is "Ontology, which is an explicit, formal specification of shared conceptualization"[3]. Ontologies are used to represent abstract model, where a domain is fixed with identified relevant concepts and relationship among those concepts. Ontology engineering in Semantic Web is principally supported by languages such as XML, RDF, RDFS and OWL[4].

At the same time, the conventional area of Software Engineering has come a long way in both research and practice. Formally, Software Engineering is "the application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software, and the study of these approaches; that is, the application of engineering to software"[5]. Software development is a multifaceted participative and collaborative task that comprises a lot of effort from various participants and yields considerable amount of information. Reusing extant pertinent information saves significant effort during the development and maintenance of software system. Moreover, according to the recent IT Outsourcing Statistics 2015/ 2016 report by Computer Economics[6] in 2015, around 62% of the Application development work has been outsourced either entirely or in part by IT organizations and in the past 5 years, over 60% of companies outsourced their application development work. Externalizing the software development would thus mean

*Author for correspondence

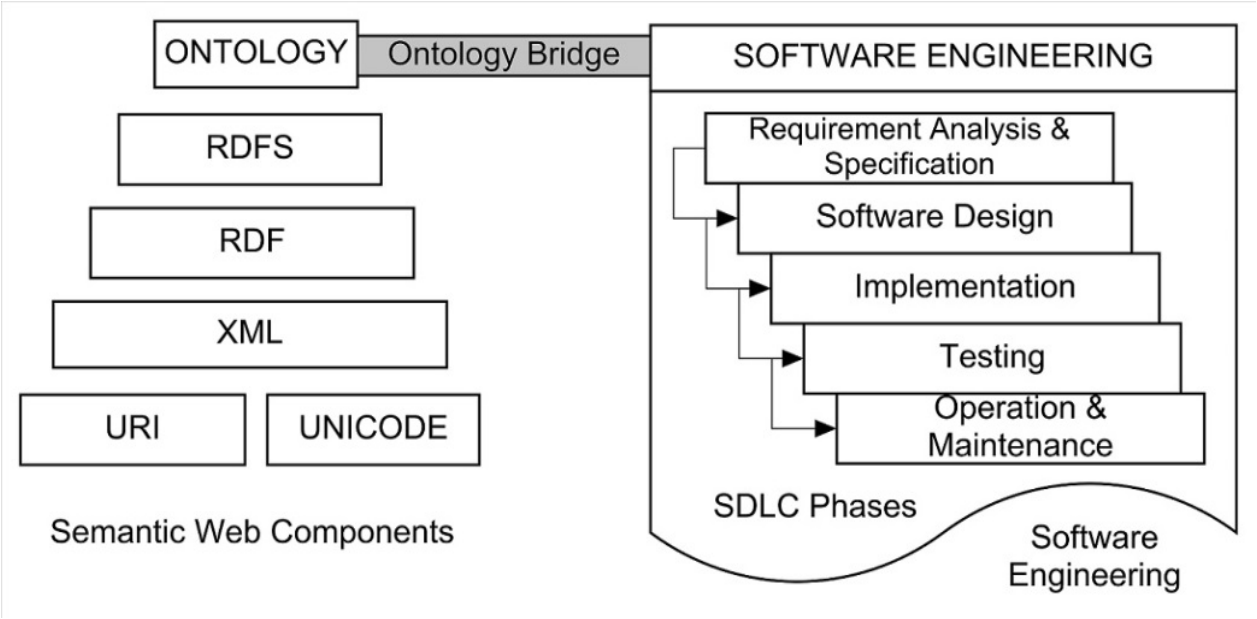


**Figure 1.** Relationship between ontologies and Software Engineering.

the development teams are diversely located and this can lead to the generation of inconsistent information leading to a development of incorrect, erroneous, and undesired software. Consequently, reusing and sharing Software Engineering knowledge becomes a key operative challenge, which motivates researchers to explore strategic and supporting technologies.

Recent studies show the consolidation among research fields of Semantic Web and Software Engineering, exemplifying the rewards of consolidating semantic techniques with Software Engineering[7, 8, 9, 10,–11]. Ontologies have come forth as a crucial player in this direction[8,9,12]. The relationship between ontologies and Software Engineering is depicted by Figure 1 where the usage ontologies have been found in each and every Software Development Life- Cycle (SDLC) phase[7].

The purpose of this study is to utilize the SWOT analysis framework to determine the benefits, impact, challenges, and risks of using ontologies for Software Engineering. This will help in providing an insight to short-term and long-term practical recommendations that can enhance the Software Engineering process and impact businesses/ individuals/ organizations/ groups in a prolific and strategic manner.

## 2. SWOT Analysis of Ontology Driven Software Engineering

SWOT is an acronym for Strengths, Weaknesses, Opportunities, and Threats. A SWOT Analysis (SWOT Matrix) is a structured planning method used to evaluate strengths, weaknesses, opportunities, and threats involved in a project[13] and is shown in Table 1. It views all positive and negative factors inside and outside the area/ field that affect the growth/scope[14].

The analytical technique of SWOT presented here, addresses and urges researchers/ organizations/ groups to make significant improvements in understanding, working and improving the Ontology Driven Software Engineering (ODSE) domain reviewing the strategic alignment between these two entities. The following subsections expound the details of the analysis undertaken to help comprehend the adaption and adoption of ontologies

**Table 1.** SWOT analysis matrix

| SWOT Analysis | Helpful (to achieve the objective) | Harmful (to achieve the objective) |
|---|---|---|
| **Internal factors (attributes of the organization)** | Strengths | Weaknesses |
| **External factors (attributes of the environment)** | Opportunities | Threats |

as new technology for driving the Software Engineering processes.

## 2.1 Strength

### *2.1.1 Platform to Share and Reuse Software Engineering Knowledge*

The presence of software development teams at different geographical, virtual, and cultural locations led to the problem of sharing and reusing Software Engineering knowledge[15–, [16], [17],18]. In this situation of distributed development environment, ontologies provide the platform to share Software Engineering knowledge. Once Software Engineering knowledge is represented by ontologies, it can be reused whenever required.

### *2.1.2 Availability of Software Engineering Knowledge in Human as well as in Machine Understanda-ble Form*

As we know that ontologies are formal specification, therefore, whatever information is represented by ontologies, it is available in human as well as in machine understandable form. When ontologies represent the Software Engineering knowledge, it becomes available in both forms which further increase the visibility of Software Engineering knowledge over the web.

### *2.1.3 Platform to Generate Consistent Information*

In distributed development environment, without sharing and reusing Software Engineering knowledge, the process leads to the generation of inconsistent information followed by the development of undesired software[15,16]. However, ODSE overcomes this inconsistency problem by enabling sharing and reuse of Software Engineering knowledge.

### *2.1.4 Effective Communication Channel*

As ontologies enable the sharing of Software Engineering knowledge, therefore, ODSE provides the effective communication channel through Semantic Web for all the stakeholders related to the project. This better communication channel results in improved software development with reduced terminological and conceptual mismatches.

## 2.2 Weakness

### *2.2.1 A Standard Way to Generate Ontologies for Software Engineering cannot be Defined*

One of the weaknesses of ODSE area is that there is no standard way to generate ontologies for Software Engineering, rather there are several different ways and the best way to generate the ontology depends on project to be developed.

### *2.2.2 An Ontology Generation May Take a Lot of Time*

Ontology development is an iterative process which goes through many cycles of revisions and refinement before it is finally shaped and can be used. The development of

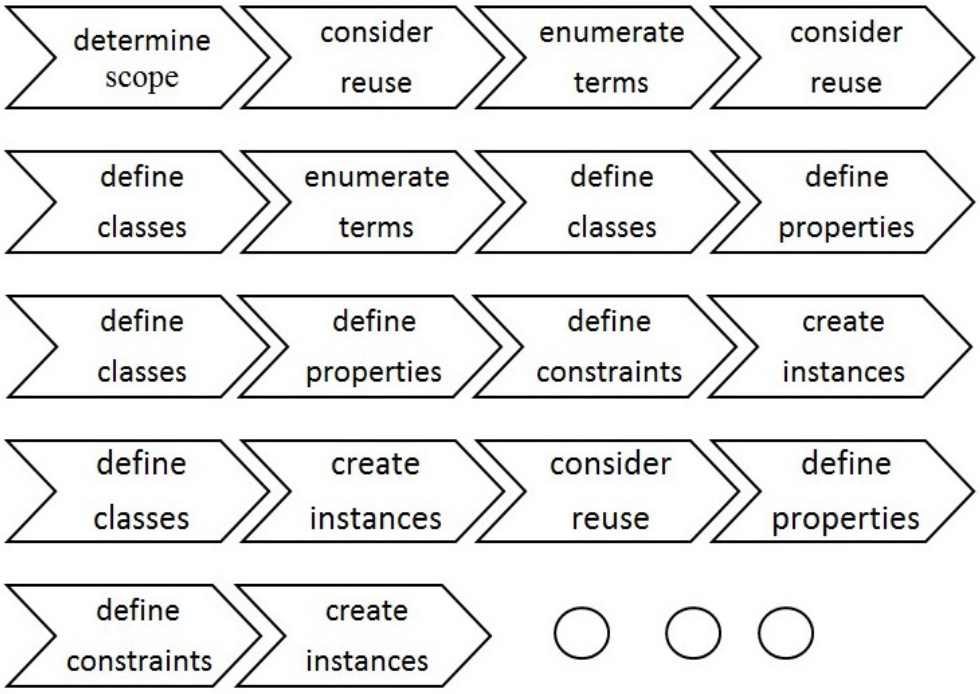


**Figure 2.** Ontology development process (iterative).

ontology is a step by step process which includes describing terms in the domain and relations among them i.e. defining classes, sub classes, attributes, properties, and instances. The following Figure 2 depicts this continuous, time consuming ontology development process which is applicable in the case of building ontologies for Software Engineering too.

### *2.2.3 Process may have Extra Cost*

The cost of ontology engineering includes ontology building, reuse and maintenance cost[19]. When an organization adopts ODSE practice for the very first time, which includes development of ontologies for Software Engineering from the scratch, it may consume a lot of human effort, thus, may result in extra cost. However, continuous and subsequent practice in this area can reduce the enhanced cost.

## 2.3 Opportunities

### *2.3.1 To Create Intelligent Support Tools that Facilitate Communication and Information Sharing*

According to Dillon et al.[20], in distributed development environments, communication and sharing becomes a problem between different development teams due to the following three reasons: (a) different training and practices between diverse cities and countries. This means the staff members who are in remote locations will struggle to exchange information and share their ideas, experiences and knowledge; (b) Software Engineering discipline is not followed and shared between different development locations, which create inconsistencies in the collection and presentation of requirements, and other design documents; and (c) inconsistency in understanding the

theories and applications of Software Engineering. In this case, different approaches and terminologies are applied to specify a same function, process or object, causing confusion and ambiguity. As a result, projects may result in a failure and consequently dissatisfaction to the end user[12].

Hence, in order to tackle the above problems, there is a need to develop intelligent support tools that can prevent errors in communication and information sharing.

### 2.3.2 Develop Intelligent Smart Tools to Effectively Manage the Different Life Cycle Phases of Soft-ware Development

There are several obstacles that need to be overcome to create intelligent smart tools that can effectively manage SDLC phases. According to Beckert et al.[21], there are few intelligent smart tools that can be applied in various domains; which can enhance the correctness, reliability, and security of the software being developed; and at the same time be flexible and useful for the developer[12].

One of techniques that received much attention from the Software Engineering community is known as formal methods[22]. Formal methods in Software Engineering are mathematical techniques that can be used to verify the correctness of requirements specification, design, coding, unit testing, integration, and system testing. They are used to verify the reliability and robustness of the software. However, the application of formal methods to check and verify software system is very labor intensive, and thus expensive. It requires expertise in mathematics and formal logic. Therefore, it is not a good idea to check each and every component of software system in detail due to the reasons of complexity and extensive money involved in the process. To be cost effective, it is better to determine the crucial components of software system and then verifying them using formal methods.

Formal methods can assume various forms and levels of rigor. To enable the use of formal methods, it is important to abstract and model systems and processes with an appropriate level of representation[22]. However, Atkinson et al.[23] indicate that the current generation of modeling tools lacks richer functions to properly represent these models. In particular, Atkinson and colleagues advocate the use of an ontological approach to enable the creation of intelligent smart tools that provide support to abstraction processes and systems modeling[12].

In this context, there is a need to develop search ontologies to facilitate the modeling of systems and processes in Software Engineering. Moreover, it is also necessary property of intelligent smart tools for modeling and managing different life cycle phases of software development. In particular, a major opportunity is to develop methods that use task ontologies in order to facilitate the modeling process and thus enable the use of formal methods with greater regularity.

The idea behind the development of task ontologies is there intrinsic contribution to generate independent domain models and having semantics to clarify the role of the concepts involved in the field. Thus, task ontology makes the separation between the domain knowledge, as knowledge of Software Engineering and the steps required to perform a task that can be used in the field. Hence, task ontologies allow us to create patterns, models and meta models that are both domain independent, and that give semantics to the domain concepts when used in context.

This domain independence characteristic of task ontologies helps reduce the complexity and constraints of using formal methods more extensively. Thus, it is possible to create formal tools that provide support to the developer throughout the development process. Furthermore, it is expected that it is possible to create procedures based on formal ontologies and methods for interconnecting all stages of the software development. That way, we can offer more reliable means of detecting human error or implementation in different life cycle phases of software development and ensure a higher quality of the final product.

### 2.3.3 Develop Methods, Techniques, and Environment to Facilitate the Production of Semantic Soft-ware using Interdisciplinary Approaches

One of the challenges in ODSE is related towards meeting the growing demand of interaction between information handling systems and web data in an intelligent way[24]. According to Shadbolt et al.[25], the answer to this challenge is the development of Semantic Web based systems. Moreover, Cardoso et al.[26] also indicate that semantic systems are already being incorporated by various organizations, particularly those that deal with large amounts of data. However, the Semantic Web area is still

new and therefore there are many opportunities to create and improve methods and techniques for analysis and specification of requirements and for development and implementation of software.

According to several researchers, one of the important factor to ensure quality of software being developed using Semantic Web technologies highly depends on the developer's training and the ability of interdisciplinary teams[26,27]. This is because for developing semantic software, developers deal with three concurrent factors: (a) the architecture and reference models for developing semantic applications; (b) interface between computer person and between computer computer; and (c) the information structure.

Therefore, in order to develop semantic systems, developer requires an interdisciplinary training about Software Engineering, Artificial Intelligence, and Human Computer Interface to handle complexity of architecture involved. A good semantic systems developer can understand how data is represented using ontologies, how to receive and transmit data using structured data and standards and how to create user friendly interfaces that handle such data[26,27].

The Semantic Web community still lacks intelligent smart tools to effectively manage the different life cycle phases of software development. Further, there is little research discussing the creation of role models that help in developing semantic applications and their use in real scenarios[28–, [29],30]. Another problem for developing semantic systems is the lack of smart tools and support development environment interfaces semantically dealing with the end user[12,31].

Thus, there is a need to create methods and management tools that utilize Semantic Web technologies. To be precise, we need to develop methods, techniques, and environment that facilitate the production of semantic software using interdisciplinary approaches such as Software Engineering, Artificial Intelligence, and Human Computer Interface.

## 2.4 Threats

### *2.4.1 Social, Legal, and Ethical Issues*

As ODSE enables the sharing and reusing of Software Engineering knowledge, it possesses the threat of violation of intellectual property rights belonging to a person, group or organization. Also, any stakeholder related to the project can leak the critical information which may cause hindrance to the successful completion of project. Therefore, any violation of social, legal, and ethical rules and regulations possesses a great threat[14].

### *2.4.2 Vendor Specific Solution*

When there is specific demand from the vendors to supply software or product using specific technologies, it can cause disruption to developers to apply this emerging technology effectively in both research and practice. Finding vital contracts and partners who can actually provide support and promote this technology may turn out as a challenge as the promoters may have zero knowledge about this field.

### *2.4.3 Loss of Key Staff*

As ODSE is an emerging area, finding key and skilled staff in this area is a significant challenge. Even if an organi-

**Table 2.** SWOT analysis of ontology driven Software Engineering

| Strengths | Weaknesses |
|---|---|
| • Platform to share and reuse Software Engineering knowledge | • A standard way to generate ontologies for Software Engineering cannot be defined |
| • Availability of Software Engineering knowledge in both human and machine understandable form: | • An ontology generation may take a lot of time |
| • Platform to generate consistent information | • Process may have extra cost |

| Opportunities | Threats |
|---|---|
| • To create intelligent support tools that facilitate communication and information sharing<br>• Develop smart tools to effectively manage the different life cycle phases of software development<br>• Develop methods, techniques, and environment to facilitate the production of semantic software using interdisciplinary approaches | • Social, Legal, and Ethical issues<br>• Vendor specific solution<br>• Loss of key staff |

zation spends too much resource (money and effort) to train its staff to learn this technology, loosing key staff will turn out as a big threat. Competitor in the similar field may try to employee the well trained staff of other organizations.

The following Table 2 summarizes the SWOT Analysis of ODSE.

## 3. Conclusion

The purpose of this study was to utilize the SWOT analysis framework to determine the use of ontologies for Software Engineering. The idea was primarily to capture and evaluate the practical use of ontologies in Software Engineering from business organization's perspective. It helped us to establish that ontologies as a supporting technology can surely be a platform to share and reuse Software Engineering knowledge in a consistent manner. A continuous need to create smart, intelligent tools that enhance the Software Engineering process is acknowledged that can help organizations to achieve business goals in a competitive way. The strength and opportunities of ODSE have an indicative advantage overlooking the trade-offs of people, skill, resource requirements and the time consuming task of ontology development.

## 6. References


1. W3C Semantic Web [Internet]. 2015 Nov. Available from: http://www.w3.org/2001/sw.
2. Berners-Lee T, Hendler J, Lassila O. The semantic web. Scientific American. 2001; 284:28–37.
3. Studer R, Benjamins VR, Fensel D. Knowledge engineering: principles and methods. Data Knowledge Engineering. 1998; 25:161–97.
4. Ding L, Kolari P, Ding Z, Avancha S. Using ontologies in the semantic web: A survey. Ontologies; 2007. p. 79–113.
5. Radatz J, Geraci A, Katki F. IEEE standard glossary of software engineering terminology. IEEE Standard. 1990; 610121990:3.
6. IT Outsourcing Statistics 2015/ 2016 [Internet]. 2015 Nov. Available from: http://www.computereconomics.com/page.cfm?name=outsourcing.
7. Gröner G, Pan JZ, Zhao Y, Kendall EF, Stojanovic L. Introduction to the Proceedings of the 9th International Workshop on Semantic Web Enabled Software Engineering (SWESE) 2013. Service-Oriented Computing–ICSOC 2013 Workshops; 2014. p. 223–24.
8. Ilyas QM. Ontology augmented software engineering. Software Development Techniques for Constructive Information Systems Design. 2013:406.
9. Aßmann U, Zivkovic S, Miksa K, Siegemund K, Bartho A, Rahmani T. Ontology-guided software engineering in the MOST Workbench. Ontology-Driven Software Development; 2013. p. 293–318.
10. Kumar A, Bhatia M, Beniwal R. Characterizing relatedness of web and requirements engineering, Webology. 2015; 12.
11. Bhatia M, Kumar A, Beniwal R. Ontologies for software engineering: Past, present and future. Indian Journal of Science and Technology. 2016; 9.
12. Isotani S, Bittencourt II, Barbosa EF, Dermeval D, Paiva ROA. Ontology driven software engineering: A review of

challenges and opportunities. Latin America Transactions, IEEE (Revista IEEE America Latina). 2015; 13:863–9.
13. SWOT analysis [Internet]. 2015 Nov. Available from: http://en.wikipedia.org/wiki/SWOT_analysis.
14. Kumar A, Sharma A. SWOT analysis of requirements engineering for web applications. Journal of Advances Research in Science and Engineering. 2015; 4:34–43.
15. Bhatia M, Kumar A, Beniwal R. Ontology based framework for automatic software's documentation. 2015 2nd International Conference on Computing for Sustainable Global Development (INDIACom); 2015. p. 421–4.
16. Bhatia M, Kumar A, Beniwal R. An ontology based framework for automatic detection and updation of requirement specifications. 2014 International Conference on Contemporary Computing and Informatics (IC3I); 2014. p. 238–42.
17. Bhatia M, Kumar A, Beniwal R. Ontology based framework for ambiguity detection in software requirement specifications. 2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom); 2016. p. 421–4.
18. Bhatia M, Kumar A, Beniwal R. Ontology based framework for reverse engineering of conventional softwares. 2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom); 2016. p. 421–4.
19. Bontas EP, Mochol M. Towards a cost estimation model for ontology engineering. Berliner XML Tage; 2005. p. 153–60.
20. Dillon TS, Chang E, Wongthongtham P. Ontology-based software engineering- Software Engineering 2.0. 19th Australian Conference on Software Engineering. ASWEC 2008; 2008. p. 13–23.
21. Beckert B, Hoare T, Hähnle R, Smith DR, Green C, Ranise S. Intelligent systems and formal methods in software engineering. Intelligent Systems. 2006; 21:71–81.
22. Hinchey M, Jackson M, Cousot P, Cook B, Bowen JP, Margaria T. Software engineering and formal methods. Communications of the ACM. 2008; 51:54–9.
23. Atkinson C, Gutheil M, Kennel B. A flexible infrastructure for multilevel language engi-neering. IEEE Transactions on Software Engineering. 2009; 35:742–55.
24. Heino N, Dietzold S, Martin M, Auer S. Developing semantic web applications with the onto wiki framework. Networked Knowledge-Networked Media; 2009. p. 61–77.
25. . Shadbolt N, Hall W, Berners-Lee T. The semantic web revisited, Intelligent Systems. IEEE. 2006; 21:96–101.
26. Cardoso J, Hepp M, Lytras MD. The semantic web: Real-world applications from in-dustry. Springer Science and Business Media. 2007; 6.
27. Hendler J, Shadbolt N, Hall W, Berners-Lee T, Weitzner D. Web science: An interdisci-plinary approach to understanding the web. Communications of the ACM. 2008; 51:60–9.
28. Devedžic V. Semantic web and education. Springer Science and Business Media. 2006; 6.
29. Holanda O, Isotani S, Bittencourt II, Elias E, Tenório T. JOINT: Java Ontology Inte-grated Toolkit. Expert Systems with Applications. 2013; 40:6469–77.
30. Isotani S, Mizoguchi R, Inaba A, Ikeda M. The foundations of a theory-aware authoring tool for CSCL design. Computers and Education. 2010; 54:809–34.
31. Huynh DF, Miller RC, Karger DR. Potluck: Data mash-up tool for casual users. Web Semantics: Science, Services and Agents on the World Wide Web. 2008; 6:274–82.