USING WEB ONTOLOGY LANGUAGE TO UNDERSTAND SOFTWARE COMPONENTS BEHAVIOR

M. Rizwan Jameel Qureshi, Alla Defallah Alrehily
Faculty of Computing and Information Technology, King Abdulaziz University,
Jeddah, Kingdom of Saudi Arabia
anriz@hotmail.com, looliabc@yahoo.com

ABSTRACT: In software engineering field, reuse process is very effective because it improves productivity, reduces cost and saves time. Developer software cannot reuse a lot of components because he cannot understand component function. To solve this problem, the authors' use Web Ontology Language (OWL) is a Semantic Web language designed to define knowledge in formal way. Using this language, the developer can understand the description of component and also function of it in easy and clear way. The purpose of using OWL in order the effectiveness and fast the search and selection the component of the web repository, ease of modification on component, also reuse it after modification of the web repository and also OWL is flexibility to add new functions. In this paper, we observed of verification results of effectiveness using this language that aim of description component.

Key words: Component reuse, Component, Software Engineering, Repository, Web Ontology Language, Semantic Web.

1. INTRODUCTION

The Semantic Web defined a web that is able to describe things in a way that computers can understand. It describes the relationships between things and the properties of things. It will add more layers to World Wide Web are web of cooperation, web of meaning and web of trust. Web Ontology Language (OWL) is a Semantic Web language able to describe concepts in formal form. OWL represents six elements are classes (set of entities within a domain), attributes (set of attributes that belong to a class), Relations (interrelations between several classes), Instances (Object or individual of class) and axioms (explicit rules to constrain the use of classes) [1]. Ghobadi and Rahgozar [2] describe that several studies are conducted on human semantic interactions with Web resources but no considerable progresses are achieved. Ghobadi and Rahgozar [2] worked on an approach to understand semantics of HTML documents to retrieve the information automatically.

In semantic web field, the programmer builds ontology concept to share common understanding of structure of information among people or software agent, to enable the reuse of domain knowledge, to make domain assumptions explicit, to separate domain knowledge from operational knowledge, to analyse and describe domain knowledge [1]. Ontology language is used in human communication through common terminology between people in group, organization or community, used to system interoperability through making sure that interacting systems share the understanding of the concepts exchange and used in system engineering through identification of domain requirements.

According of uses and benefits ontology in semantic web field. The authors proposed use OWL in software engineering field because many of Components are stored in a repository to reuse any time but a software developer suffer from difficultly in understanding of the component function when reuse it of the repository. For this reason authors use OWL method to describe component in easy way and also become search process of it fastest in the web repository. The aim of the web repository to storage, organizes and retrieval of software components. The paper is organized as follows, the next section describes some brief literature review and in section 3, the authors have described the statement of the problem and in section 4, the solution is proposed of this problem. In section 5, the solution is validated by survey and conclusion is provided in section 6.

2. RELATED WORK

In [3], reusable software assets using product line approach are discussed to consider in initial investments to reduce costs, schedule and increased product quality. Reuse costs included all phases of a project life cycle, domain Analysis, Software Architecture Design, Tools that support of the reuse and training people on the use tools. There are prerequisites to creating reusable software based on product line approach requires commitment of Organization abilities and skills, technical skills in design and domain analysis.

Thus, understanding of initial investments and these prerequisites lead to create reusable software assets. Jalender et al. [4] describe that there are barriers for component based development To break these barriers for component based software using commercial security, ensure reliability and usability of component, knowing location and use of relevant components, reduce costs, the changes that add of component must define costs and schedules from component assembly, value in legacy software systems must take care, pricing and revenue process may be not compatible in market place. Basha et al. [5] offers these paper would be review of software reusability included types of reuse, reuse approaches, software reuse benefits and problems. The empirical study applied on the software reuse activity by expert designers in the context of object-oriented design.

Table 1 Summary of Related Work.

Title of paper	limitation
A pragmatic approach to software reuse [3]	1. Lack the method of calculating initial investments of reuse software process. 2. Software process models that help in the reuse are not support. 3. Creating phases the software asset are not defined. 4. The reasons failure of software reuse is not enough.
Breaking the Boundaries for Software Component Reuse Technology [4]	1. Industry affected of component based development. 2. Technology certain that suitable component based development. 3. The references are not enough to support component based development.
Software Reuse: Developers' Experiences and Perceptions [8]	1. Developers experiences and perceptions about software reuse different in each organization. 2. No solution suggests for impediments that faced developers. 3. The model conditions focused on reuse code or documents. 4. The developer cannot reuse the old code due to complexity.
Assessing Opportunities Of software Reuse for Companies [9]	1.Did not mention facts that make software engineering focused on original development of software 2. Systematic software reuse approach did not support by examples in software field. 3. The paper only applied two opportunities: reduce in cost and improve software quality. 4. This Journal has been done in little time. 5. The references are not enough to support software Reuse.
Review of software reusability [10]	1. The study was not writing in paper. 2. Systematic software reuse approach did not support by examples in software field. 3. No results achieved of these study. 4. The references are not enough to support software Reuse.
Reusability of the software[11]	1. The paper did not support by example represent components and how to interact with other units. 2. The references are not enough to support semantic Wiki concept and the software reuse repository. 3. The semantic Wiki must show example of web applications.
Designing code level reusable software components [7]	1. The developer spends more time for developing the requirements the software. 2. No example considered for build code level reusable.
Reusability Assessment of Open Source Components for Software Product Lines [11]	1. Each metric affected on certain attribute. 2. Discovery of these factors takes a lot of time. 3. This metrics only focused to measure java code. 4. Variability and scope coverage attributes not focused of software component.
Design of dynamic component reuse and reusability metrics library for reusable software components in context level [12]	1. This architecture may be not applied for different environments. 2. In Dynamic metrics library after each phase take the many of time to calculate reusability metrics. 3. The methods used in architecture may be not accurate.
Methodology to manage victim components using CBO measure [5]	1. Proposal Measures spend a great effort and a lot of time. 2. CBO calculation each component wastes of designer time.

However, study fulfillment time is not enough for reach of results and reuse approaches must be detailed to understand how to software reuse.

Wiki systems cannot provide software reuse repository to help people in reuse knowledge and reduce development costs [6]. Semantic Wiki is proposed adding metadata about artifacts and relations in order to solve this problem. A software reuse repository system is developed to achieve benefits like contextual presentation of pages, improved navigation to access of related contents, search by semantic and reasoning support [6].

Jalender et al. [4] describes that the best way to save the time, money and increase productivity and quality in the product by building code level reusable components using class libraries, functional libraries, design patterns and framework classes. Best design code reuse must share common classes and collections of functions, frameworks and procedures. Frameworks' concept will help programmers to design an application quickly. Reuse will reduce time and money while increasing productivity and quality in the product [7].

An exploratory study is presented about factors that effect on reusability of open source software like flexibility, maintainability, portability, scope coverage, stability, understandability, usage history and variability [8]. A reusability assessment model is presented that contains six attributes namely flexibility, maintainability, portability, scope coverage, understandability and variability regarded to the reusability of SPL component. These attributes and metrics are used to assess reusability.

In companies, reuse needs a lot of time to determine if the component qualified or not qualified for reusability from component library [1]. Architecture is proposed for component reusability in context level to identify, extract and qualify reusable components. A metric is used containing functional coverage report, extraction time and reuse frequency to measure the qualification of a component for reuse.

Components that are less reused in repository called victim components [5]. Basha et al. [5] applied weighted component, depth of inheritance tree and number of children measures on HR portal application component. Required component is determined that must be reconfigured to increase the reusability count. Coupling between object measure (CBOM) is proposed to identify one reconfigurable component that has highest measure. This methodology divides component into several parts to reuse in future.

3. Problem Statement

Difficulty in understanding the component behavior made it difficult for developers to reuse infrequently during the projects. Following is the research question identified based on the literature review [8].

"How to increase reuse of component or artifacts in software projects?"

4. The Proposed Solution

A Software developer suffers from difficultly in understanding the component when reuse it of a repository. Components are stored in the repository that like library system has graphical user interface (GUI) for searching and reuse the component. The repository provides storage, retrieval and organizes of software components.

The developer enters to system by GUI and makes reuse for any component but it might be difficult to understand functional description, code and documentation. Thus, the idea will be using Web Ontology Language (OWL) is an ontology language for the semantic web which defined meaning. OWL represents six elements are classes (set of entities within a domain), attributes (set of attributes that belong to a class), Relations (interrelations between several classes), Instances (Object or individual of class) and Axioms (explicit rules to constrain the use of classes) [1]. These elements are stored as semantic web. Programmers build ontologies to share common understanding of structure of information among people or software agent enabling the reuse of domain knowledge (to make domain assumptions explicit, to separate domain knowledge from operational knowledge, to analyse and describe domain knowledge) [1].

In our paper, ontology is used to describe components in an easy and fast way to search it using a web repository. A repository is proposed using OWL to select components having similar functionally as in one concept and also components which are not similar described in a unique concept. This description method makes easier for the developer understanding component when to reuse it and also help in searching, selecting the component of the repository making reusability effective and fast. OWL will describe component elements are objects, class's properties, constraint and class hierarchies. OWL defines a transitive property, a functional property, the inverse property and an inverse functional property to assists in understanding the functionality.

Here, we will depict how the developer will select components describing and searching using by OWL in a repository to reuse as shown in figure 1. Developer searches a component by insertion component name in GUI. A query string is passed to the web repository and matched a component to reuse. He can modify a component by adding new functions and properties using OWL. He can download OWL files easily to update or save.

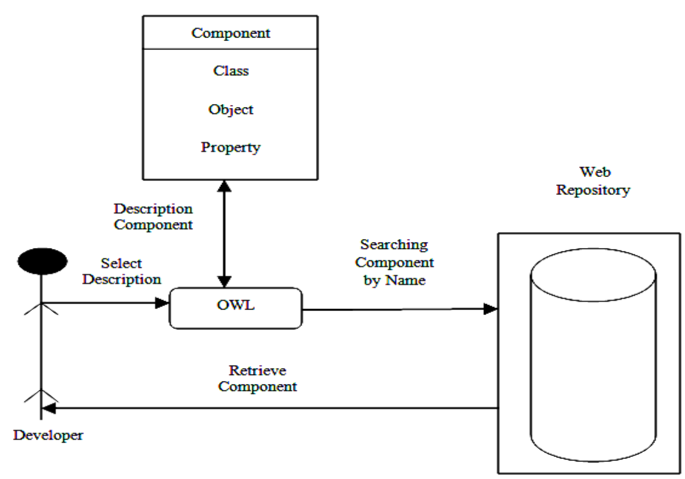

Figure 1 selection and search process

The web repository is a tool that is used to search components. In addition, it stored component name, a description text about Component, component type, domain, language component, component interface name, a platform that fits component and component code as represented in figure 2.

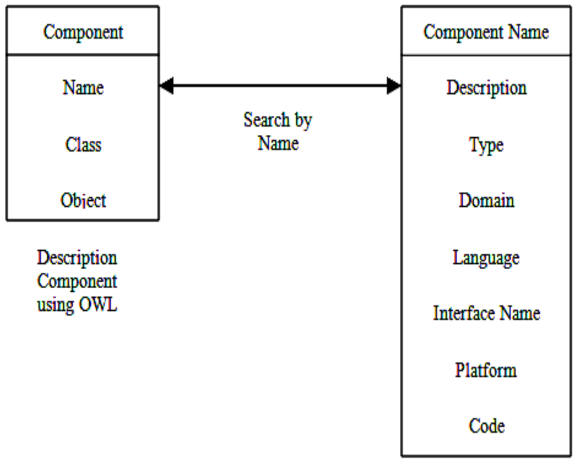

Figure 2 Description component in OWL and the web repository

5. Validation of the proposed solution

Validation of the proposed solution is conducted using a survey via Twitter, Face book and e-mail. This method does not consume time. The questionnaire is composed of 18 close ended questions further divided into 3 goals. Likert scale is ranging from 1 to 5 as: Strongly Disagree indicating 1; Disagree indicating 2; Neutral indicating 3; Agree indicating 4 and Strongly agree indicating 5. Statistical analysis is applied on data and the results are represented using frequency tables and bar charts.

6. Findings

We conclude the results through cumulative statistical analysis of three goals.

A. CUMULATIVE STATISTICAL ANALYSIS of GOAL 1 IS UNDERSTANDING COMPONENT EASILY THROUGH ITS DESCRIPTION USING OWL.

Description component using OWL will make developer understands functionally of component when he reused of web repository. The description is as follows the similar components functionally are described together in one concept and also components which not similar described in unique concept. The component elements must describe using OWL method to understand function. The cumulative result of survey for goal 1 is shown in Table 2.

Table 2 shows that 37.74% of the responders agreed on use OWL in describe component because they can understand component easily and 36.28% strongly agreed of it. 4.90% disagree of it and 0.98% strongly disagrees of it while 20.10% remained neutral as shown below in figure 3.

Q. number	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
1	0.00	0.00	17.65	32.35	50.00
2	2.94	5.88	8.82	35.29	47.06
3	2.94	0.00	29.41	50.00	17.65
4	0.00	5.88	17.65	32.35	44.12
5	0.00	5.88	14.71	26.47	52.94
6	0.00	11.76	32.35	50.00	5.88
Total	5.88	29.40	120.59	226.46	217.65
Percent	0.98%	4.90%	20.10%	37.74%	36.28%

Table 2 Cumulative statistical analysis of goal 1

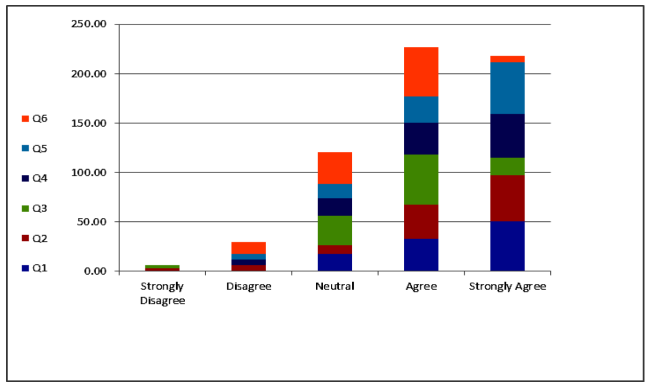

Figure 3 Cumulative results of questionnaire for goal 1

B. CUMULATIVE STATISTICAL ANALYSIS of GOAL 2 IS THE EFFECTIVENESS AND FAST THE SEARCH AND SELECTION THE COMPONENT OF THE WEB REPOSITORY.

Table 3 shows that the responders find that the searching and selecting the component of the web repository are effective and fast. 47.65% of them strongly agreed of it and 22.35% agreed of it. 8.23% disagree of it and 2.35% strongly disagree of it while 19.41% remained neutral as shown below in figure 4.

Table 3 Cumulative statistical analysis of goal 2

Q. number	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
7	2.94	11.76	11.76	20.59	52.94
8	0.00	2.94	14.71	20.59	61.76
9	2.94	17.65	50.00	20.59	8.82
10	2.94	0.00	14.71	23.53	58.82
11	2.94	8.82	5.88	26.47	55.88
Total	11.76	41.17	97.06	111.77	238.22
Percent	2.35%	8.23%	19.41%	22.35%	47.65%

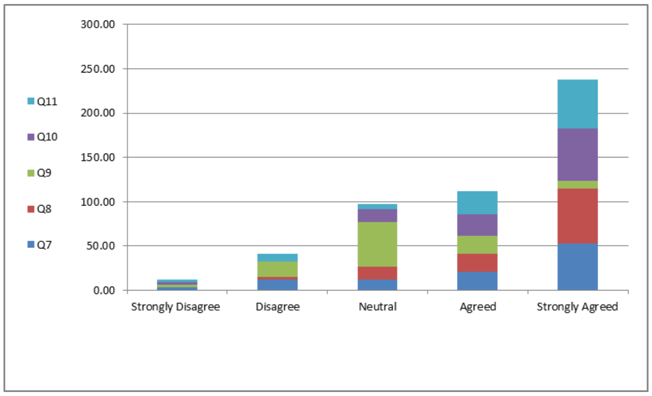

Figure 4 Cumulative results of questionnaire for goal 2

C. CUMULATIVE STATISTICAL ANALYSIS of GOAL 3 IS EASE OF MODIFICATION AND REUSE COMONENT OF THE WEB REPOSITORY.

Modification component means adding new functions and properties or deleting some elements. Using OWL will be easy of developer controlling in this feature. Thus, even after modification component will be reuse process is effective and fast. The cumulative result of survey for goal 1 is shown in Table 4.

Table 4 Cumulative statistical analysis of goal 3

Q. number	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
12	0.00	0.00	23.53	50.00	26.47
13	0.00	0.00	26.47	41.18	32.35
14	0.00	2.94	52.94	38.24	5.88
15	0.00	8.82	52.94	35.29	2.94
16	0.00	2.94	26.47	17.65	52.94
17	0.00	5.88	20.59	47.06	26.47
18	0.00	5.88	20.59	20.59	52.94
Total	0.00	26.46	223.53	250.01	199.99
Percent	0.00%	3.78%	31.93%	35.72%	28.57%

As observed in Table 4 that 35.72% of the responders agreed on ease of modification using OWL and reuse component of the web repository and 28.57% strongly agreed of it. 3.78% disagree of it and nobody strongly disagree of it while 31.93% remained neutral as shown below in figure 5.

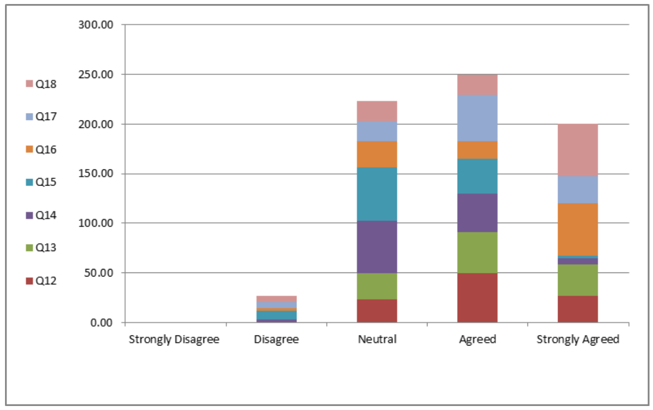

Figure 5 cumulative results of Questionnaire for goal 3

Finally, the cumulative result of survey for all goals is shown in Table 5.

Table 5 Cumulative evaluation of the goals 1 through 3

Q. number	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
1	0.00	0.00	17.65	32.35	50.00
2	2.94	5.88	8.82	35.29	47.06
3	2.94	0.00	29.41	50.00	17.65
4	0.00	5.88	17.65	32.35	44.12
5	0.00	5.88	14.71	26.47	52.94
6	0.00	11.76	32.35	50.00	5.88
7	2.94	11.76	11.76	20.59	52.94
8	0.00	2.94	14.71	20.59	61.76
9	2.94	17.65	50.00	20.59	8.82
10	2.94	0.00	14.71	23.53	58.82
11	2.94	8.82	5.88	26.47	55.88
12	0.00	0.00	23.53	50.00	26.47
13	0.00	0.00	26.47	41.18	32.35
14	0.00	2.94	52.94	38.24	5.88
15	0.00	8.82	52.94	35.29	2.94
16	0.00	2.94	26.47	17.65	52.94
17	0.00	5.88	20.59	47.06	26.47
18	0.00	5.88	20.59	20.59	52.94
Total	17.64	97.03	441.18	588.24	655.86
Percent	0.98%	5.39%	24.51%	32.68%	36.44%

As observed in Table 5, 36.44% of the responders strongly agreed proposed solution by authors and 32.68% agreed of it. 5.39% disagree of it and 0.98% strongly disagrees of it while 24.51% remained neutral as shown below in figure 6.

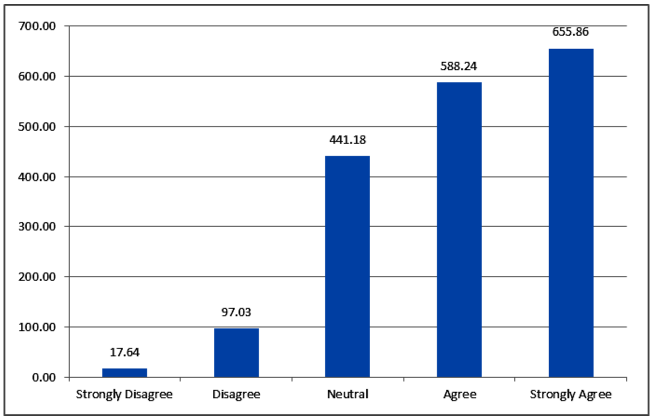

Figure 6 Final cumulative evaluation of the goals 1 through 3

7. CONCLUSION

In software engineering field, many of Components stored in a repository to reuse any time but a software developer suffer from difficultly in understanding the component function when reuse it of the repository. He must overcome of this difficultly .The proposed solution aims to easy of understanding component through its description using OWL, the effectiveness and fast the search and selection the component of the web repository and ease of modification and reuse component of the web repository. Reuse component will improve productivity of software, reduce cost and save time. In this paper, the author's proposed solution concluded of cumulative analysis of survey that 36.44% of people strongly agreed on importance description component using OWL and 32.68% agreed of it. 5.39% disagree of it and 0.98% strongly disagrees of it while 24.51% remained neutral. Thus, we observe of verification results that most responders have desire strongly of using OWL because easy of understanding component when reuse it of the web repository.

REFERENCES

[1] Pressman R., *Software Engineering*, 7th Ed, McGraw Hill, 2010.

[2] Ghobadi A., and Rahgozar, M., "An Ontology-based Semantic Extraction Approach for B2C eCommerce," *The International Arab Journal of Information Technology*, vol. 8, no. 2, pp. 163-169, 2011.

[3] Jalender B., G.ovardhan A., and Premchand P., "A Pragmatic Approach To Software Reuse," *Journal of Theoretical and Applied Information Technology*, vol. 14, no. 2, pp.87-96, 2010.

[4] Jalender B., Govardhan A., and Premchand P., "Breaking the Boundaries for Software Component," *International Journal of Computer Applications*, vol. 13, no. 6, pp.37-41, 2011.

[5] Basha N., and Moiz S., "Methodology To Manage Victim Components using CBO Measure," *International Journal of Software Engineering & Applications*, vol. 3, no. 2, pp.87-96, 2012.

[6] Singh S., Singh S., Singh G., "Reusability of the Software," *International Journal of Computer Applications*, vol. 7, no. 14, pp. 38-41, 2010.

[7] Jalender B., Govardhan A., and Premchand P., "Designing Code Level Reusable Software Components," *International Journal of Software Engineering & Applications*, vol. 3, no. 1, pp.219-229, 2012.

[8] Agresti W., "Software Reuse: Developers' Experiences and Perceptions," *Journal of Software Engineering and Applications*, vol. 4, no. 1, pp.48-58, 2011.

[9] Sigar K., "Assessing Opportunities Of software Reuse for Companies," *International Journal of Advanced Research in Computer Science*, vol. 4, no. 3, pp. 46-47, 2013.

[10] Budhija N., and Ahuja S., "Review of Software Reusability," *in Proceedings of the 1st International Conference on Computer Science and Information Technology*, India, pp. 113-115, 2011.

[11] Amin F., Mahmood A., Oxley A., "Reusability Assessment of Open Source Components for Software Product Lines," *International Journal on New Computer Architectures and Their Applications*, vol. 1, no. 3, pp. 519-533, 2011.

[12] Subedha V., and Sridhar S., "Design of Dynamic Component Reuse and Reusability Metrics Library for Reusable Software Components in Context Level," *International Journal of Computer Applications*, vol. 40, no. 9, pp.30-34, 2012.