Towards an Ontology for Software Product Quality Attributes

Ahmad Kayed, Nael Hirzalla
College of Computing
Fahad Bin Sultan Univ.
Tabuk, KSA
kayed@fbsc.edu.sa

Ahmad A. Samhan, Mohammed Alfayoumi
College of IT
Middle East University for Graduate Studies
Amman, Jordan
nhirzallah@fbsc.edu.sa

Abstract—**Recently, Quality Assurance concept has been developed increasingly to be included in many of our life existing fields; financial, industrial, trading, computing, etc. Software Quality Product Attributes (SWQAs) have been created as a matter of applying the QA concept on the results of web or desktop application development process, to fit the products with the organizational and global market standards and goals, and to provide it with a competitive advantage value. Web application or software product quality is composed of many attributes such as portability, usability, reliability, modularity. During the recent years, many researchers discussed and presented software attributes in their works which showed that till now there is a lack of consensus on the semantic of many of concepts and terminologies used in this field. Our work is focusing on studying Software Product Quality Attributes concepts and terminologies. We conduct several experiments to extract the main concepts for SWQAs. The results show that there is a number of concepts that are frequently used to describe these attributes. Summarizing and formalizing the semantic of the attributes into these concepts presents a common understanding and agreement on the semantic of SWQPAs which can be used by software engineers, researchers, practitioners, and stakeholders.**

Keywords- Web application quality, ontology, software product quality, terminologies

I. Introduction

Software quality attributes and measures are one of the key issues that made significant influences on software engineering; it plays a very important role in evaluating software programs or web applications. It is considered by practitioners and researchers to be the key factor for producing high quality competitive web or desktop applications to the markets, which is enforced by the appearance of quality assurance issues. As a matter of fact, many initiatives such as IEEE Standard Releases, ISO/IEC Releases, SPICE (Software Process Improvement and Capability Determination), and many quality models, such as McCall quality model, Boehm quality model, Dromey quality model, and others, consider software quality measures and attributes to be an important element of reaching a higher maturity levels and managing the quality of software programs [1, 2].

During the last decades, many developments in many fields have affected how the business is done. One of the most important issue that was revealed is the emerging of the internet and the globalization effect on the individuals and the organizations business processes. It created a need for sharing information and resources widely as a matter of collaboration to compete efficiently in the market. In order to achieve this collaboration, standards are created to provide agreed concepts and practices that make participants avoid inconsistencies in their business [1]. Researchers in this domain explained that there is no single standard that covers the area of software measurements in its totality, but rather, there are many different standards focus on specific areas, without considering a comprehensive framework, [3].

Recently, a lot of efforts from researchers and institutes are done to manipulate the symptoms that software quality measurements discipline suffers from; software quality measures concepts, principles, and terminologies are considered by those researchers and institutes to be in a stage that they are still being defined, consolidated, and agreed upon [1].

In this paper, ontology is used to reach an understandable unified semantic framework for web or desktop application quality measurements that their concepts and terminologies are inconsistency among the current studies and reports. In this work we studied and analyzed a total of 80 different documents, reports, and proposals concerned with software measures, attributes and quality. The first step was to exclude documents, reports or proposals that included more than 80% programming code, having weak relevancy to software quality versus general quality, or can be classified as redundant information. This process, which is more detailed and justified in [4], reduced the total number from 80 to a total of 34 documents, reports, and proposals. Then we extract various concepts, definitions, and terminologies from them. Our claim is that the semantic of these definitions can be condensed into a smaller set of concepts. If these concepts have enough and clear semantic, then they will achieve a common understanding for any other web application quality attributes.

This paper also provides a brief introduction to Ontology and some of its practical usage which appears in section two. Section three talks about our experiment where well-known Software Product Quality Attributes are selected, defined, and then used to present common concepts. Section four will discuss the findings and results of the experiment. Finally, section five concludes our work and recommends future work.

II. What is Ontology?

Ontology which was coined in 1613 included in many philosophical areas from the metaphysics of Aristotle to the object-theory of Alexius Meinong [5]. Philosophical Ontology handles the precise utilization of words as descriptors of entities; it gives an account for those words that belong to entities and those that do not [6]. In both Computer Science and Information Science, an ontology is a representation of a set of concepts within a domain and the relationships between those concepts. It is used to reason about the properties of the domain in question, and may be used to define the domain [7].

Recently, the term ontology was widely included in the field of computer and information science. When building frameworks for information representation, the data and knowledge base systems designers use a wide variety of terms and concepts. Studies showed that there is an inconsistency problem in the semantic of the terms that are used, e.g. identical databases labels are used but with different meanings, and same meaning could be expressed using different names. Thus, methods must be found to resolve the terminological and conceptual incompatibilities [6]. Ontology is, in this context, a dictionary of terms formulated in a canonical syntax and with commonly accepted definitions designed to yield a lexical or taxonomical framework for knowledge-representation, which can be shared by different information systems communities [6]. Ontology's are used in variety of current fields, such as artificial intelligence, software engineering, the semantic web, biomedical informatics, library science, and information architecture, as a form of knowledge representation.

In this era, the presence of consistent global information has become an important issue. In every domain researchers and practitioners need to share information to conduct their work in a professional manner. To do that in a correct way, inconsistencies between terms and concepts must be reduced. Ontology defines a common vocabulary for them. It contains machine-interpretable definitions of basic concepts in the domain and relations among them**.** Some of the reasons that motivate researchers and practitioners to develop an ontology, [8], are

- To share common understanding of the structure of information among people or software agents.
- To enable reuse of domain knowledge.
- To make domain assumptions explicit.
- To separate domain knowledge from the operational knowledge

Another reason for using ontology is that it contributes in reducing gabs among researchers created by conceptual confusion [9,10,11]. Ontology shows enormous potential in making software more efficient, adaptive, and intelligent. It is recognized as one of the areas which will bring the next breakthrough in software development.

The idea of ontology has been welcomed by visionaries and early adopters. Recently the semantic Web initiative, lead by W3C, has changed the ontology landscape completely. Researchers and developers joined forces to provide standard semantics markup languages based on XML, ontology management systems, and other useful tools.

Also, the Web provides interesting applications of ontology critical to daily life such as search and navigation. In addition, people rediscover the value of ontology in other important applications such as information and process integration [12,13].

Maria de los Angelos and Luis Olsina stated in there study in software metrics and indicators [3] that if we take a look at the current software engineering international standards produced by major standardization organizations such as IEE, ISO, and IEC we will see that the situation is the same there; terminology conflicts and inconsistencies are detected not only between standards issued from different organizations, but also among standards issued from the same organization.

In this work, we are building ontology to capture the conceptualization knowledge about Software Product Quality Attributes (SWPQAs) domain, which will achieve a significant successful solution for the semantic conflicts problem it suffers from.

III. The Experiment

The experiment starts first by formalizing the conceptualization of the SWPQAs in order to produce a coherent and consistent set of common terminologies and concepts as a common agreement software quality measurements pool of knowledge**.** This contribution aims at enabling software engineers, researchers, practitioners, and stakeholders to find out shared and common concepts and terms for describing software quality measures and attributes. This will remove gabs, inconsistencies, and terminology conflicts that can affect their SW quality to reach for a consensus knowledge domain.

To do so, we suggest the following steps:

a) First, we collect about 80 reports, documents, standards, and publications that are related to Software measures, attributes and quality. Then we filter them into 34 documents based on SWPQ attributes definitions.

b) From these filtered materials, we select the most 67 common discussed SWQP attributes and collect their definitions as presented in the materials (multi source definitions). Table 1 shows a sample of these collected definitions.

c) By using KAON's [14] extension software, we extract the terms, concepts, and relationships that are used in the selected SWPQ attributes context. Table 2 shows a sample of the resulted 496 concepts. We then insert all these concepts into a database along with their definitions.

d) Using SQL statements, we count the number of times each concept appears in the definition of other concepts. Table 3 shows a snapshot of the 496 concepts we worked on.

e) We modify the SQL statement to count the number of concepts each definition may contain out of the 496 concepts we have. A sample of the results is shown in Table 4

TABLE I. **PART OF SELECTED SWPQ ATTRIBUTES (OUT OF 159)**

Attribute	ID	Definition	Src.
Accuracy	1	Attributes of software that bare on the provision of right or agreed results or effects	19
		Those attributes of the software which provide the required precision in calculations and outputs.	20
		This quality factor addresses the concern that programs provide the precision required for each output. Accuracy is important because most computer manipulations are not exact, but are limited approximations	21
		A software product possesses accuracy to the extent that its outputs are sufficiently precise to satisfy their intended use	5
Consistency	9	Those attributes of the software which provide for uniform design and implementation techniques and notation	20
		This quality factor addresses the concern that the source code syntax and constructs in programs be implemented uniformly	21
		"Those characteristics of software which provide for uniform design and implementation techniques and notation"	21
Performance	19	This quality factor addresses the concern of how well a program attribute or function is implemented with respect to some standard. Often, this is related to the utilization of resources	21
		The degree to which a system or component accomplishes its designated functions within given constraints regarding processing time and throughput rate. such as speed, accuracy, or memory usage	9
		Performance as a software quality attribute refers to the timeliness aspects of how software systems behave	16
Stability	28	Attributes of software that relate to the risk of unexpected effect of modifications	19
		The capability of the software product to avoid unexpected effects from modifications of the software	19
Testability	30	the ease of testing the program, to ensure that it is error-free and meets its specification	2
		The degree to which a system or component facilitates the establishment of test criteria and the performance of tests to determine whether those criteria have been met	11
Usability	33	This characteristic express the ability of a component to be understood, learned, used, configured, and executed, when used under specified conditions	1
		A set of attributes that bear on the effort needed for use, and on the individual assessment of such use, by a stated or implied set of users	2
		The capability of the software to be understood, learned, used and attractive to the user when used under specified conditions	9

TABLE II. **SAMPLE OF THE EXTRACTED CONCEPTS**

Concept
Ability
Architecture
Capability
Data
Document
Environment
Fact
Function
Measure
Product
Requirement

TABLE III. **PART OF THE UNIQUE FREQUENCY FOR EACH CONCEPT OCCURRENCE IN ALL DEFINITIONS**

Concepts	Frequency
Software	57
Ability	48
Attribute	48

System	45
Component	37
Function	36
Use	36
Product	35
Form	33
Degree	32
Capability	30
Program	28
Characteristic	26
Concern	25
Perform	24

TABLE IV. PART OF THE COUNTS OF CONCEPTS IN EACH DEFINITION

Def. ID	Definition	Sum Of Concepts Match
16	Maintainability	89
11	Efficiency	78
33	Usability	65
19	Performance	60
20	Portability	58
27	Security	56
22	Reliability	55
32	Understandability	53
30	Testability	50
45	Integrity	44
61	Verifiability	39
37	Correctness	37
31	Traceability	36
44	Flexibility	36
38	Modifiability	34
1	Accuracy	33
43	Modularity	33
14	Interoperability	32
5	Availability	31

IV. DISCUSSION AND CONTRIBUTION

Table 3 shows the unique concepts occurrences in all definitions. We choose the highest fifteen occurrences since they appear on average in 54% of the 67 definitions. Table 3 shows that these 15 concepts have on average 36 occurrences in the definitions. This indicates that about 54 % of the studied SWPQ attributes are using these concepts. In other words, the semantic of these attributes (the definitions) needs these concepts. It also indicates that we need to agree about the semantic of these concepts in order to define the semantic of almost 36 attributes (out of 67).

After a deep study of these concepts, we can claim that if we don't agree about the semantic of these concepts we will not agree about the semantic of the 54% of the SWPQ attributes.

Fifty four percent of the semantics of the 67 SWPQ attributes can be condensed into these 15 concepts. We can't claim that these 15 concepts can replace all the semantics of all SWPQ attributes, but we can claim that these concepts are important for any of these attributes.

We try to find out the most important concepts to have good and clear definitions. We have done this by studying SWPQ attributes, terms, and their definitions. We extracted concepts from the context of these terms, counted the occurrences for each concept in all definitions, and counted the occurrences of concepts in each definition. We claim that these concepts are important to be in the semantic of the studied software product quality attributes definitions. Thus, we can condense the semantic of many software quality attributes into a smaller set of concepts.

Also we know that ontology is an explicit specification of conceptualization and this specification may summarize many concepts. We believe that a concept may symbolize many other concepts. Putting this in practice, the results show that this could be true.

In summary, this paper investigates new ways to enhance dealing with the semantic of software product quality attributes. We choose to use ontology as the foundation of the enhancement, because ontology's can represent conceptualization. Furthermore, ontology's capture the semantics of concepts which can be represented in a formal language, and can be used to store related metadata. The purpose of this paper is to suggest a framework that aims to identify some important SWPQ attributes concepts that are heavily used at different definitions. It aims to address the needs of two main kinds of interested audiences. The first kind is the software quality measurements researchers and standard developers (e.g., international standardization institutes and committees), who are responsible for producing concepts, terms, and standards in this field, and the second is the software quality measurements practitioners, who may be confused by the terminology differences and conflicts in the existing standards and proposals.

The main aim of this paper is to provide experts, mainly researchers, and practitioners in the field of software product quality with an Ontology to be considered as the base of a common agreement knowledge. This work provides the most common concepts for software product quality attributes. These concepts have been extracted from field documents and reports with a common, shared, and consistent semantic.

As far as text mining tools, the algorithm presented is based on KAON's [14], thus many of text mining tools are used by KAON. Moreover, interested readers may refer to

KANON in [14] for more details on text mining tools that have been used.

V. Conclusion

Software quality measures discipline is still in the emerging phase and it suffers from the typical symptoms of any relatively evolving disciplines. Software quality measures are currently in the phase in which terminologies, principles, and methods are still being defined, consolidated, and agreed upon. In particular, there is a lack of consensus on the concepts and terminologies used in this field. Studies showed that inconsistencies between the different research measurement proposals often occur.

In this work we focus on studying the most 67 common discussed software product quality attributes concepts and terminologies extracted from 34 current software measures, software quality reports and related documents. In this paper, we showed that the semantic of the definitions of these attributes can be condensed into a smaller set of concepts; thus, unifying many of these concepts.

As a future work, we are trying to deploy the ontology in some applications to show how effective, useful, and expressive is the proposed ontology to the audience in a context of software engineering domain and especially to the audience in the context of software quality measurements domain.

References

[1] F. Garcia, M. Bertoa, and C. Calero. “Towards a Consistent Terminology for Software Measurements,” Information of Software Technology, 48(8): 631-644, 27 June 2005.

[2] ISO/IEC. Software and Systems Engineering - Guidelines for the Application of ISO/IEC 9001:2000 to Computer Software. International Standards Organization, Geneva, Switzerland, 2004. International Standard ISO/IEC 90003.

[3] M. de los Martin and L. Olsina. “Toward An Ontology for SW Metrics and Indicators as the Foundation For Catalog Web System.” LA – WEB 2003: 103 -113, Santiago, Chile.

[4] Ahmad A. Samhan, “An Ontology for Software Product Quality Attributes” Middle East University for Graduate Studies Master’s thesis, 2008

[5] B. Smith and C. Welty, “Ontology: Towards a New Synthesis,” FOIS’01, October 17-19, 2001, Ogunquit, Maine, USA, 2001 ACM1-58113-377-4/01/0010.

[6] B. Smith, “Ontology,” Draft version of a chapter in “Blackwell Guide to the Philosophy of Computing and Information”, Oxford: Blackwell, 2003, 155-166.

[7] T. Gruber “Ontology,” to appear in the Encyclopedia of Database Systems, Ling Liu and M. Tamer Özsu (Eds.), Springer-Verlag, 2008.

[8] N. F. Noy, and D. L. McGuinness, “Ontology Development 101: A Guide to Creating Your First Ontology,” Stanford University, Stanford, CA, 94305, 2001.

[10] R. William., S. Pawlowski, and V. Volkov, "Requirements interaction management", ACM computing surveys, Vol35, No2, June 2003, pp 132-190.

[11] S. Walt., C. Jensen, J. Noll, and M. Elliott, "Multi-Modal Modeling, Analysis and Validation of Open Source Software Requirements Processes" ACM 2005.

[12] Z. Ganng, Y. Gao, and R. Meersman, "An ontology-based approach to business modeling" ACM 2005.

[13] A. Kayed, and R. Colomb. “Extracting Ontological Concepts for Tendering Conceptual Structures. Eng.” 40(1): 71-398(2002).

[14] A. Kayed, and R. Colomb. “Using BWW Model to Evaluate Building Ontologies in CGs Formalism.” Information. System. 30(5): 379-398(2005).

[15] Extensions to the Karlsruhe Ontology and Semantic Web Framework, KAON Extensions, Developer’s Guide for KAON Extensions 0.6, August 2003