# An Ontological Approach to Model Software Quality Assurance Knowledge Domain

Nada O. Bajnaid, Rachid Benlamri, Algirdas Pakstas, and Shahram Salekzamankhani

***Abstract*—Software Quality Assurance (SQA) becomes one of the most important objectives of software development and maintenance activities, and many SQA standards have emerged as part of the Software Engineering discipline. However, despite the effort made to improve consistency and coherency among SAQ standards, still there is no single standard that covers the whole SQA knowledge area. To contribute to this effort, this paper presents a framework of an ontological model to describe and define both domain and operational knowledge of SQA. International standards (SWEBOK, IEEE, and ISO) were the main sources of the terminology and semantic relations of the proposed SQA conceptual model. Different approaches have been used to evaluate the developed SQA ontology. The ultimate goal was to develop an ontology that faithfully models the SQA discipline as practiced in the software development life cycle.**

***Index Terms*—Domain modeling, knowledge representation, ontology, ontology evaluation, semantic web, software engineering, software quality assurance.**

## I. Introduction

Many areas of human activities such as communication, transportation, health, finances, and education are highly dependent on software applications that range from simple to highly complex life critical systems. This requires software of high quality. Software quality is a rather complex concept; some authors have defined the entire discipline of SE as the production of quality software [1]. Therefore, Software Quality Assurance (SQA) becomes one of the most important objectives of software development and maintenance activities, and many SQA standards have emerged as part of the Software Engineering (SE) discipline.

Although Software Quality Assurance (SQA) becomes one of the most important objectives of software development and maintenance activities, yet there is no consensus among the SQA community of most of the domain terminology and concepts. Despite the efforts in research and international standardization, inconsistency and terminology conflicts appear between standards even within the same organization. A well-defined, complete and disciplined SQA process can be helpful to improve communication and collaboration among project participants and can serve as a standard when there is a disagreement.

Ontologies provide a common understanding and sharing of knowledge by using a general agreement on terminology among all interested people. SE domain ontologies are very useful in developing high quality, reusable software by providing an unambiguous terminology that can be shared through various software development processes. Ontologies also help in eliminating ambiguity, increasing consistency and integrating distinct user viewpoints [2]-[5].

Using ontology to model the SE knowledge shortens the development time, improves productivity, decreases cost, and increases product quality. Ontologies provide better understanding of the required changes and the system to be maintained [6].

There was an effort by different bodies to develop Software Engineering standards followed by the forming of the ISO/IEC Joint Technical Committee 1 (JTC1) workgroup in order to guarantee consistency and coherency among standards. This work is motivated by the need for having consistent terminology and agreed upon concepts among existing taxonomies of the SQA domain, where these taxonomies are mainly found in standard documents. The next section introduces the use of the development of the SQA ontology using agreed standards. Evaluation of the developed SQA ontology is presented in Section III. Section IV presents enhanced version of the SQA otology based on latest standards and results of the evaluation process. A case study showing the deployment of the SQA ontology in an e-learning system is presented in Section V, while Section VI concludes and summaries the findings of this research.

## II. SQA Ontology Model

Higher quality ontologies can be easier reused and shared with confidence among applications and domains. Additionally in case of re-use, the ontology may help to decrease maintenance costs [7]. The SQA ontology must contain well-defined, structured and organized knowledge of the SQA domain including the type of software process, its SQA requirements, quality attributes, and corresponding SQA measurements and metrics.

### A. Conceptualization

There are various vocabularies to describe the SQA domain knowledge. In fact, there is no single standard which embraces the whole software quality assurance knowledge. Different standards and proposals have used different terminologies for the same term. Similarly, the same term may be used to refer to different concepts. This issue has been recognized by the International Standards Organization (ISO) and in 1987 the ISO/IEC has established the Joint Technical Committee 1 (JTC1) workgroup to guarantee consistency and coherency among standards. Also the IEEE computer society

Manuscript received September 16, 2015; revised February 12, 2016.

Nada O. Bajnaid is with King Abdulaziz University, Saudi Arabia (e-mail: nbajnaid@kau.edu.sa).

Rachid Benlamri is with Lakehead University, Canada (e-mail: rbenlamr@lakeheadu.ca).

Algirdas Pakstas and Shahram Salekzamankhani are with London Metropolitan University, UK (e-mail: a.pakstas@londonmet.ac.uk, s.salekzamankhani@londonmet.ac.uk).

and the ISOJTC1-SC7 agreed to harmonize terminology among their standards.

The primary source of the SQA ontology is the *Software Engineering Body of Knowledge* SWEBOK guide [8] in addition to the above-mentioned ISO and IEEE standards (ISO 9126, IEEE 12207, IEEE 610.12, IEEE 00100, PMBOK 2008, CMMI v1.2). An enhanced SQA ontology has also been developed in our previous research [9].

We used the above-mentioned software engineering knowledge sources aided by domain experts to build the vocabulary and relationships of the SQA ontology. Ontology properties are used to describe relationships among individuals classes. Various properties are used to describe both static and dynamic aspects of the SQA knowledge, such as SQA-processes and related SQA issues. The ontology provides a formal description for SQAProcess which may have Quality Attributes (QAs) that can be measured. Various quality assurance processes, such as *Validation*, *Verification*, and *Audit* can be instantiated as shown in Table I.

Measurement plays an important part in software development. It can be used to indicate the quality of the product being developed [10]. According to Pressman's categorization of software metrics, quality metrics, which measure customer requirements fulfillment, indicate how closely software conforms to explicit and implicit customer requirements. In this study, software measurements and metrics are at the heart of the SQA ontology design. All aspects of SQA measurements and metrics as described in the ISO/IEC 9126 standard [11] are reflected in the proposed SQA ontology. Fig. 1 illustrates the top level of the SQA ontology model.

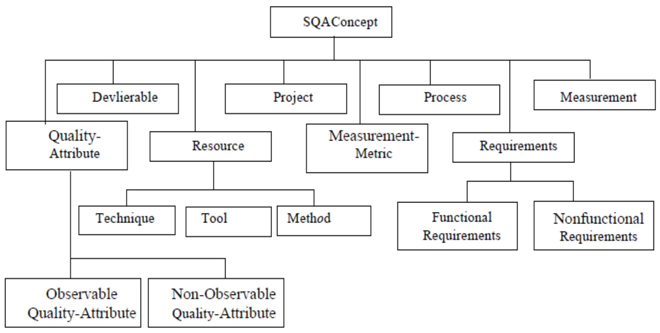


Fig. 1. Top level of the SQA ontology.

The proposed model may include some overwhelmed or unnecessary content. Ontology axioms, a declaratively and rigorously represented knowledge that has to be accepted without proof, were added to prevent unnecessary knowledge. In ontology representation, axioms can be used to represent the meaning of concepts carefully, and to answer questions on the capability of the built ontology using ontology concepts.

TABLE I: THE SQA ONTOLOGY PROPERTIES

| Name | Domain | Range | Cardinality | Inverse Property |
|---|---|---|---|---|
| hasProcess | Project | Process | Multiple: a project may have more than one process | - |
| enforces | Process | Quality-Attribute | Multiple: a process may enforces (ensures) more than one attribute | enforcedBy |
| Uses | Process | Resource | Multiple: a process may use more than one resource | isUsedBy |
| isInputTo | Deliverable | Process | Multiple: a process may have more than one deliverable as input | isInputTo |
| invokes | Process | Process | Multiple: a process might invoke other process (es) | - |
| hasProcess | Project | Process | Multiple: a project may have more than one process | - |
| enforces | Process | Quality-Attribute | Multiple: a process may enforces (ensures) more than one attribute | enforcedBy |
| Uses | Process | Resource | Multiple: a process may use more than one resource | isUsedBy |
| isInputTo | Deliverable | Process | Multiple: a process may have more than one deliverable as input | isInputTo |
| invokes | Process | Process | Multiple: a process might invoke other process (es) | - |

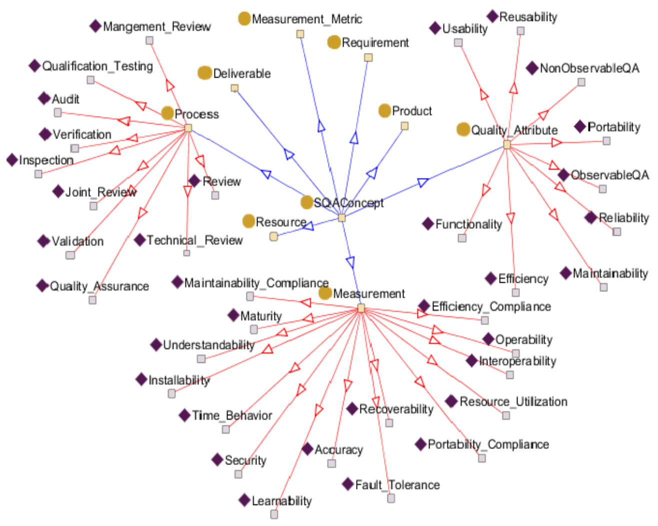


Fig. 2. Class hierarchy of the SQA ontology.

## *B. Implementation*

The conceptual model resulted from the previous step is transformed into formal OWL ontology. The Prot ég éediting tool is used to translate the SQA conceptual model into machine processable ontology represented in OWL language [12]. The Jambalay tab, a Prot ég é plug used for ontology visualization generates graphical representation of the ontology. Fig. 2 shows a class hierarchy of the software quality domain ontology. The figure shows classes and individuals of the SQA ontology where blue arrows represent the subclass relationships and the red arrows represent individuals of the class. Note that in the figure not all individuals of the classes are shown due to space limitation.

Moreover, the Prot ég é checker is used to verify the ontology consistency while the Racer Pro-reasoner is used as a Prot ég é plug in to check the consistency of the developed ontology.

## III. ONTOLOGY EVALUATION

Evaluating the ontology (its concepts definitions, taxonomy and axioms) is important and worthwhile task [7].

Mistakes and omissions in ontologies can lead to applications not realizing the potential of exchanging data. In addition, ontology evaluation increases the availability and thus reusability of the ontology and decreases maintenance costs. Ontology evaluation assesses the quality of the ontologies and thus encourages their publication and reusability since the confidence of the re-users in the quality of these ontologies increases.

Evaluating ontology is not an evidence of the absence of problems, but it will make its use safer. The main efforts towards evaluating ontology content were made by Gómez-Pérez [13], [14] in the framework of METHONTOLOGY and by Welty and Guarino [15] with the OntoClean method. A survey on evaluation methods and tools can be found in [16].

According to [16], ontology evaluation requires:

- ***Verification*** which refers to building the ontology correctly;
- ***Validation*** which refers to whether the ontology definitions really model the domain for which the ontology was created. Ontology validation ensures that the correct ontology was built. The goal is to show that the world model is compliant with the formal model;
- ***Assessment*** which focuses on judging the ontology from users' points of view (human judgment).

In this work, ontology evaluation is limited to the criteria identified by Gómez-Pérez [14] such as: ***completeness***: where all knowledge that is expected to be in the ontology is either explicitly stated in it or can be inferred; ***consistency***: refers to the absence (or not) of contradictory information in the ontology; ***conciseness***: checks if the ontology is free from any unnecessary, useless, or redundant definition; and ***expandability***: refers to the ability to add new definitions without altering the already stated semantic.

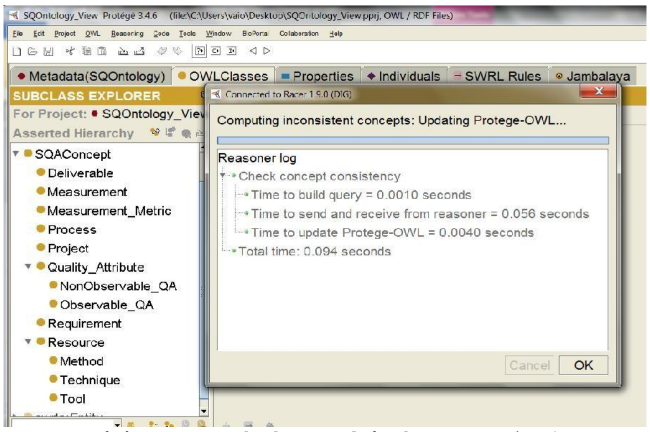
Different ontology evaluation approaches have been considered in literature depending on the purpose of the evaluation and the type of the ontology being evaluated. Brank and colleagues [17] classify ontology evaluation approaches as follows:

1) Those based on comparing the ontology to a "golden standard" which might be an ontology itself;
2) those based on using the ontology in an application and evaluating the results or application-based ontology evaluation;
3) those involving comparison with a source of data (e.g. a collection of documents) about the domain to be modeled by the ontology; and
4) those where evaluation is done by humans who try to assess how well the ontology meets a set of predefined criteria, standards, requirements, etc.

The first approach is not applicable due to the lack of a "golden standard" or upper-level Software Engineering ontology. However, the second approach has been adopted in this study and an application-based ontology evaluation was conducted using a prototype system which was implemented for this purpose (see Section V).

The third approach was held during development of the ontology when the evolving conceptual model was compared to the sources of knowledge. Recall that the goal of validating the ontology is to show that the world model is compliant with the formal model, i.e. the formal OWL representation of the ontology is compliant with the defined conceptual model.

Moreover, during implementation, the developed ontology was verified for consistency using the Protégé consistency checker tool which automatically checks the consistency and conciseness of the developed ontology. Only inconsistent classes will be displayed by the tool. Fig. 3 shows the result generated by Protégé and the Racer Pro reasoning for the consistency checking where no inconsistence classes are listed. Syntax checking is performed by Protégé OWL plugin, which generates OWL statements during creation of the ontology using the Graphical User Interface. The plugin ensures that the generated OWL statements adhere to the rules of the OWL language.


SQOntology_View Protégé 3.4.6 (file:\C:\Users\vaio\Desktop\SQOntology_View.pprj, OWL / RDF Files)

Connected to Racer 1.9.0 (DIG)

Computing inconsistent concepts: Updating Protege-OWL...

```
Reasoner log
Check concept consistency
    Time to build query = 0.0010 seconds
    Time to send and receive from reasoner = 0.056 seconds
    Time to update Protege-OWL = 0.0040 seconds
Total time: 0.094 seconds
```

Fig. 3. Protégé consistency checking result for the SQA ontology's concepts.

The fourth approach included usage of the ontology assessment questionnaire which was distributed among SE specialists to evaluate the quality of the ontology. The use of the conceptual model eases the assessment process in this work where the domain specialists can validate wither the model matches the purpose it was built for. The conceptual model with a link to the questionnaire has been sent to domain specialists inviting them to participate in the SQA ontology assessment process to verify its coverage of the SQA domain, structure, clarity, and extendibility.

Although, there is no such a single ontology that can unanimously represent any knowledge area, especially for an evolving domain like SQA, the survey shows a high level of agreement around the major assessment criteria. This is despite the fact that each participant responds based on their own view, background and context. Fig. 4 summaries results of the assessment process.

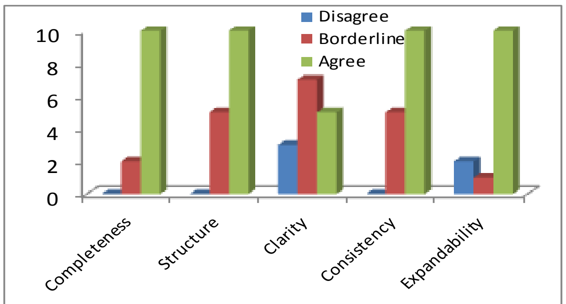


Fig. 4. Participants' assessments of the SQA ontology.

Although, there is no such a single ontology that can unanimously represent any knowledge area, especially for an evolving domain like SQA, the survey shows a high level of agreement around the major assessment criteria. This is despite the fact that each participant responds based on their own view, background and context.

## IV. Enhanced Version of the SQA Ontology

Based on the results and findings of the ontology evaluation process, enhanced version of the ontology is developed. In the new version, the ontology concepts "Quality Attribute" and "Measurement" are renamed "Quality Characteristic" and "Quality Sub-characteristic" respectively. The concept "Measurement Metric" is also renamed "Measure" to follow the transformation from the ISO/IEC 9126 [11] to the last quality standard ISO/IEC 25010 [18].

Comparison of the quality characteristics and sub-characteristics in the two standards as adopted from the ISO/IEC 25010 [18] is used in addition to the ISO/IEC 25023 [19] standard for development of a new enhanced SQA ontology as illustrated in Fig. 5.

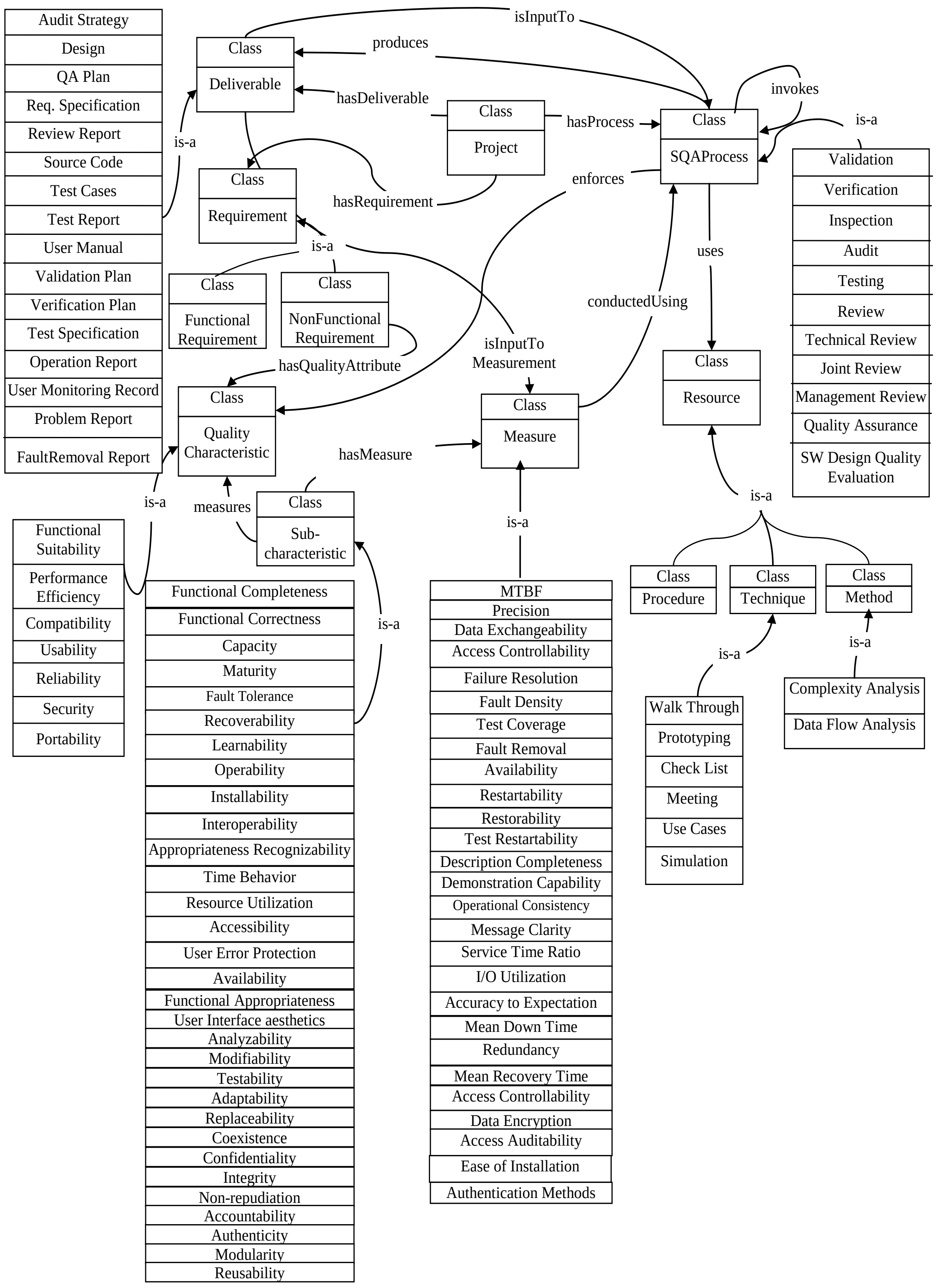


Fig. 5. Conceptual model of the SQA ontology according to ISO/IEC 25010.

## V. SQAES: Case Study

Application-based (or task-based) evaluations offer a useful framework for measuring practical results of ontology conciseness such as responses provided by the system and the ease of use of the query component [20]. A querying prototype consisting of an SQA E-Learning System (SQAES) has been designed and implemented [21] to evaluate the impact of ontologies on the information retrieval application where semantic search is combined with keyword- based search.

The prototype system aims at guiding software developers (e- learning in the workplace) or student (in traditional learning scenario) through the necessary QA practices by providing resources that deal with SQA related aspects of the software process in hand and hence improves product quality.

The main components of the SQA e-learning system (SQAES) are: the learning recommendation generator, the process discovery unit and the ontology reasoning unit as illustrated in Fig. 6 [22].

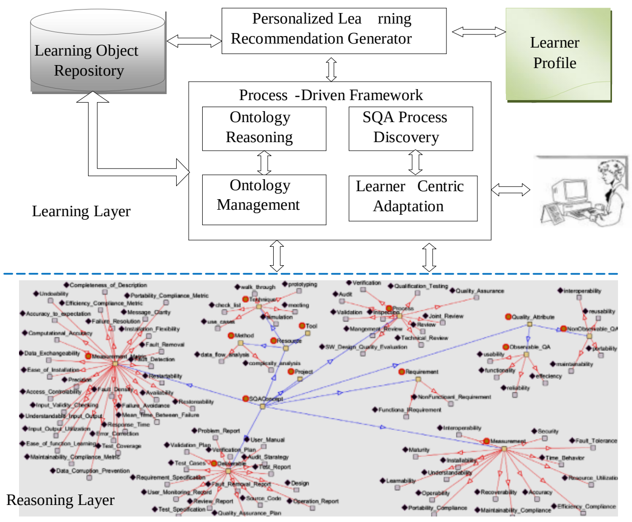

Learning Object Repository
Personalized Lea rning Recommendation Generator
Learner Profile
Process -Driven Framework
Ontology Reasoning
SQA Process Discovery
Ontology Management
Learner Centric Adaptation
Learning Layer
Reasoning Layer

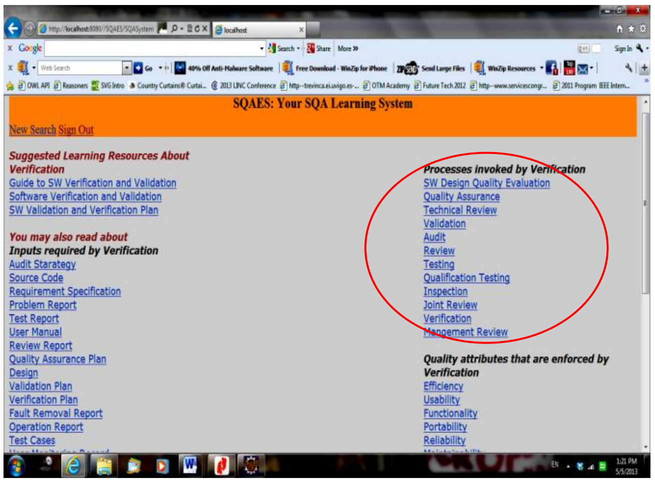
Fig. 6. SQAES structure.

### A. Adding Axioms to the SQA Ontology

The prototype system provides the learner with a recommendation list based on the initial query. However, this list may include some overwhelmed Learning Objects (LOs) or unnecessary content. Ontology axioms, a declaratively and rigorously represented knowledge which has to be accepted without proof, were added to prevent unnecessary knowledge. In ontology representation, axioms can be used to represent the meaning of concepts carefully, and to answer questions on the capability of the built ontology using the ontology concepts.

Consider the case when the user queries the *Verification* concept, which is a process according to the SQA ontology, the system retrieves the core LOs associated with the *Verification* concept from the LOs repository. Related concepts represent the list of recommended SQA concepts to be provided to the user for further investigation. However, this list may include some overwhelmed or unnecessary contents. In the example of Verification, by firing the Invokes rule, LOs associated with all SQA processes will be added to the list of recommendation as illustrated in Fig. 7.

In theory (i.e. as per IEEE 12207 standard) [23], only those processes that are associated with Review and Audit should have been added to the list (Fig. 8).


SQAES: Your SQA Learning System

New Search Sign Out

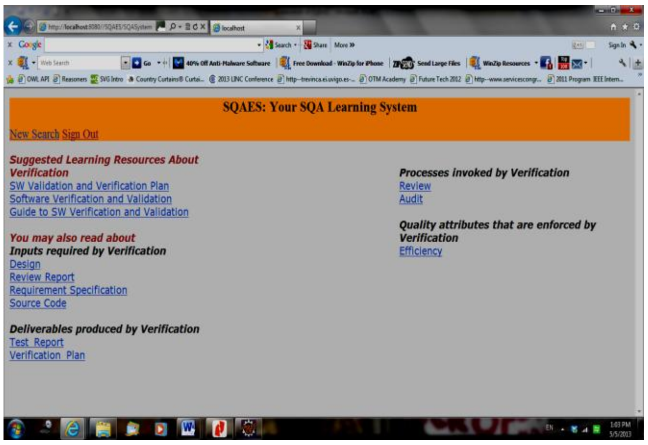
Suggested Learning Resources About Verification
Guide to SW Verification and Validation
Software Verification and Validation
SW Validation and Verification Plan

You may also read about
Inputs required by Verification
Audit Starategy
Source Code
Requirement Specification
Problem Report
Test Report
User Manual
Review Report
Quality Assurance Plan
Design
Validation Plan
Verification Plan
Fault Removal Report
Operation Report
Test Cases

Processes invoked by Verification
SW Design Quality Evaluation
Quality Assurance
Technical Review
Validation
Audit
Review
Testing
Qualification Testing
Inspection
Joint Review
Verification
Mangement Review

Quality attributes that are enforced by Verification
Efficiency
Usability
Functionality
Portability
Reliability

Fig. 7. System response without using axioms.


SQAES: Your SQA Learning System

New Search Sign Out

Suggested Learning Resources About Verification
SW Validation and Verification Plan
Software Verification and Validation
Guide to SW Verification and Validation

You may also read about
Inputs required by Verification
Design
Review Report
Requirement Specification
Source Code

Deliverables produced by Verification
Test Report
Verification Plan

Processes invoked by Verification
Review
Audit

Quality attributes that are enforced by Verification
Efficiency

Fig. 8. System response using axioms.

To prevent such situation, recommendation refining is guaranteed by adding ontology axioms to the ontology model. By referring back to our example related to *Verification* concept and according to ISO/IEC 9126 standard, a *Verification* process produces *Test Report* and *Verification Plan* and requires *Requirement Specification*, *Source Code*, *Review Report* and *Design* as inputs. In addition, *Verification* has *Efficiency* as quality attributes. The above knowledge can be represented with the following axioms added to the *Verification* concept of the SQA ontology model:

∀ `produces only (Test_Report or Verification_Plan)`

∀ `invokes only (Review or Audit)`

∀ `ensuresQA only Efficiency`

∀ `uses only (Use_case or Measurement or Prototyping)`

∀ `hasInput only (Requirement_Specification or Source_Code or Review_Report or Design)`

## VI. Conclusion

A well-defined, complete and disciplined SQA process can be helpful to improve communication and collaboration among project participants and can serve as a standard when there is a disagreement. This research has designed and developed a Software Quality Assurance ontology that at the first time represents both domain and operational knowledge of the SQA knowledge area. The ontology provides consistent terminology that aims to support communication between people and software agents. The common vocabulary and relationships modeled in the developed ontology is an attempt to resolve the problem of inconsistency among current standards and proposals. Different ontology evaluation approaches were conducted to validate and assess the SQA ontology. This research defines a framework of building ontology- based application for SQA e-learning. The presented framework can be easily transformed to reflect new standards in the domain. This research area is very rich and many ideas can be developed as extension to this research this may include merging the developed SQA ontology with other ontologies in the Software Engineering knowledge domain.

## REFERENCES

[1] E. Mnkandla and B. Dwolatzky, "Defining agile quality assurance," in *Proc. International Conference on Software Engineering Advances*, 2006.

[2] M. Uschold and M. Gruninger, "Ontologies: Principles, methods, and applications," *Knowledge Engineering Review*, vol. 11, no. 2, 1996.

[3] A. Perez and V . Benjamins, "Overview of knowledge sharing and reuse components: Ontologies and problem solving methods," in *Proc. Workshop on Ontologies and Problem-Solving Methods (KRR5)*, Stockholm, Sweden, August 2, 1999.

[4] P. Spyns, R. Meersman, and M . Jarrar, "Data modelling versus ontology engineering," *ACM SIGMOD Record*, vol. 31, no. 4, December 2002.

[5] Z. Yajing, D. Jing, and P. Tu, "Ontology classification for semantic-web- based software engineering," *IEEE Transactions on Services Computing*, vol. 2, no. 4, pp. 303-317, 2009.

[6] C. Calero, F. Ruiz, and M. Piattini, *Ontologies in Software Engineering and Software Technology,* Springer, 2006.

[7] D. Vrandečić, *Ontology Evaluation, Handbook on Ontologies, International Handbooks in Information Systems*, 2nd ed. Springer, Heidelberg, 2009, pp. 293-313.

[8] SWEBOK. (2004). Guide to the Software Engineering Body of Knowledge. IEEE Computer Society Press. [Online]. Available: http://www.swebok.org

[9] N. Bajnaid, R. Benlamri, A. Pakstas, and S. Salekzamankhani "Software quality assurance ontology: from Development to Evaluation," in *Proc. 25th International Conference on Software Engineering and Knowledge Engineering,* Boston, USA, pp. 27-29, June 2013.

[10] R. S. Pressman, *Software Engineering: A Practitioner's Approach,* 6th ed. McGraw-Hill Inc, 2005.

[11] *Software Engineering – Product Quality, Part1: Quality Model*, 2001.

[12] M. Smith, C. Welty, and D . McGuinness. (2004). OWL web ontology language guide. *W3C Recommendation*. [Online]. Available: http://www.w3.org/TR/owl-guide/#owl_Class

[13] A. Gómez-Pérez, M. Fernández-López, and A. de Vicente, "Towards a method to conceptualize domain ontologies," in *Proc. Workshop on Ontological Engineering*, Budapest, Hungary, 1996, pp. 41–52.

[14] A. Gómez-Pérez, "Evaluation of ontologies," *International Journal of Intelligent Systems,* vol. 16, no. 3, pp. 391–409, 2001.

[15] C. Welty and N. Guarino, "Supporting ontological analysis of taxonomic relationships," *Data and Knowledge Engineering*, vol. 39, no. 1, pp. 51–74, 2001.

[16] A. Gómez-Pérez, M. Fernandez-López, and O. Corcho, *Ontological Engineering: with Examples from the Areas of Knowledge Management, e-Commerce and the Semantic Web,* Springer-Verlag, New York, London, 2004.

[17] J. Brank, M. Grobelnik, and D. Mladenic, "A survey of ontology evaluation techniques," in *Proc. Conference on Data Mining and Data Warehouses (SiKDD 2005)*, Ljubljana, Slovenia, 2005.

[18] *Systems and Software Engineering-Systems and Software Quality Requirements and Evaluation-System and Software Quality Models*, 2011.

[19] *Systems and Software Engineering-Systems and Software Quality Requirements and Evaluation-Measurement of System and Software Product Quality*, 2011.

[20] L. Obrst *et al*., "The evaluation of ontologies: Toward improved semantic interoperability," *Semantic Web: Revolutionizing Knowledge Discovery in the Life Sciences,* Springer, 2007.

[21] N. Bajnaid, R. Benlamri, and B. Cogan, "Context-aware SQA e-learning system," in *Proc. the Sixth International Conference on Digital Information Management*, Melbourne, Australia, 26-28 Sept., 2011, pp. 327-331.

[22] N. Bajnaid, A. Pakstas, S. Salekzamankhani, and R. Benlamri, "Ontology-based personalized SQA e-learning system," in *Proc. Centeral Europian Conference on Information and Intelligent Systems*, Varazdin, Croatia, Sep. 2013, pp. 18-20.

[23] *IEEE Std 12207-2008: System and Software Engineering-Software Life Cycle Processes*, ISO/IEC 1220.

**Nada O. Bajnaid** was born in 1971 in Riyadh, SA. In 1992, Nada has earned a bachelor degree in computer science from King Saud University, SA. In 2003, she earned a master degree in computer science from the University of Wisconsin Milwaukee, USA and a PhD from London Metropolitan University, UK in 2014.

She is currently an assistant professor at King Abdulaziz University and the chairperson of the Computer Science Department (female section).

Nada's research interest include ontology, semantic web, context awareness, e-learning, software engineering, software quality and agile methodology.

**Rachid Benlamri** was born in Constantine, Algeria 1961. He is a professor of software engineering at the Faculty of Engineering at Lakehead University, Canada. He is the head of the Semantic Web and Mobile Computing Lab at Lakehead University. His research interests are in the area of Semantic web, context-aware computing, ubiquitous computing, and mobile knowledge management. He supervised over 70 students and postdoctoral fellows. He served as keynote speaker for many international conferences. Prof. Benlamri is an associate editor for the International Journal of Ubiquitous Learning, and member of the editorial board of many other journals such as the International Journal of Learning Technologies, the International Journal of Mobile Communications, the International Journal of Emerging Technologies in Web Intelligence, and the International Journal of Electronic Government.

**Algirdas Pakštas** was born in Irkutsk, Russia in 1958. He received the M.Sc. degree in radiophysics and electronics from the Irkutsk State University in 1980 and the Ph.D. degree in systems programming from Trapeznikov Institute of Control Sciences, Moscow, Russia in 1987.

From 1980 to 1983, he was a senior software engineer with the Institute of Solar-Terrestrial Physics, Irkutsk. Since 1987, he has been a head of the Department of Distributed Computing Systems with the Institute of Mathematics and Informatics, Vilnius, Lithuania. In 1994 he joined Agder University, Grimstad, Norway as a professor of telematics. He joined University of Sunderland in 1998 and moved to the University of North London (now London Metropolitan University) in 2000. He is the author of two research monographs, more than 150 articles and co-editor of one book. His research interests include variety of topics related to software engineering, network planning, modeling and simulation as well as most recently the history of computer science.

**Shahram Salekzamankhani** was born in Tehran, Iran in 1971. He received his B.Sc. degree in applied physics from the Arak University in 1995 and the Ph.D. degree in network security and modelling from Faculty of Computing of London Metropolitan University in 2011.

From 1996 to 2000 he was a freelance IT consultant. In 2001, he joined University of North London as a senior lecture. In 2009 he became a Cisco academy manager at London Metropolitan University. In 2012, he became the manager of Cisco Academy Training Center (ITC) and Cisco Academy Support Centre (ASC).

His research interests covers various topics related to network security, network planning and design, network simulation, network traffic optimization, routing protocols as well as most recently ontology based modeling and evaluation.

Since 2001, Dr. Shahram Salekzamankhani achieved several prestige Industrial professional certifications. From 2005 to 2014, he'd published several scientific papers in the field of Network security, modeling and evaluation.