Formalize the Software Quality Measurement for Heterogeneous Requirements

Edwin Mit
Faculty of Computer Science and Information
Universiti Malaysia Sarawak
94300 Kota Samarahan, Sarawak.
edwin@fit.unimas.my

Cheah Wai Shiang
Faculty of Computer Science and Information
Universiti Malaysia Sarawak
94300 Kota Samarahan, Sarawak.
cwshiang@fit.unimas.my

***Abstract*—There are two main challenges in measuring quality of software requirements: (i) There is no single model that can precisely represent the properties of heterogeneous requirements, and (ii) how to derive the quantitative measurement of the requirement properties that can be used to measure the software requirements quality? The issue of quality measurement is to define the benchmark or baseline used to transform the qualitative measurement to the quantitative measurement, and to date there are lack of particular rules or properties used to assess whether the propose software requirements fulfills the criteria. Therefore, this paper presents and discusses the framework of a new formal platform for assessing the quality of heterogeneous software requirements, which are obtain from open innovations. Hence, with this model the reliability of the software to be produced is known. This is an important study which, lead to the reduction of software failure in particular for safety-critical system**

Keywords—software quality, formal methods, requirement measurement, open innovation

I. INTRODUCTION

Open innovation enables us innovate faster and continuously lowering the risk and expenses of bringing new products and services to market [1]. Agile methods, such as SCRUM and eXtreme Programming (XP) [2] has a similar goal, that is to bring the new product to the market faster. In the Agile approach, the interactive interaction between customers and developers is important in order to gain immediate feedback. Agile methods are in a way similar to open innovation, whereby customers are actively involved in the software development process, so that the development team can gather, compile and build open ideas in order to obtain the best solution. In order to increase the effectiveness of the customer on an eXtreme Programming (XP) project, patterns of roles and practices of customers has been defined in [3]. However, in Agile methods, their feedback and innovations are limited to the customers directly involved in the development project. C. Schmitt [4] had proposed the design of an Open Object Information Infrastructure to enable Open Innovation in the context of Ubiquitous Computing by enriching the semantics of virtual and physical object in the open innovation environment. In order to establish and maintain effective communication structures, a collection of patterns have been defined by C.Lescher [5], to help build a global team and to improve the effectiveness of communication and collaboration in global development projects. So far, there has been little discussion about the assessment of quality of the solutions from a variety of global talent sources. Recently there is a proposal by J. Mund [6] to investigate the possibilities and limitations of metrics in assessing the quality of (software) requirement specifications (SRS) and thereby aims to improve the understanding of the notion of quality for those specifications. Similar work has been carried out by Shahid Iqbal, [7] exploring how different metrics relating to different areas of software project, especially in requirement engineering. The focus of their research study is to evaluate and highlight the important of various performance metrics and propose additional metrics for requirement gathering and management and an effort to minimize the negative risk factors and improved adherence to quality assurance. In object-oriented approach, Martin Monperrus [8] define the requirements meta-model and use an automated measurement approach to specify requirements metrics. As software systems becoming more and more complex over time therefore software quality is also becoming major concern in software development [9]. For example, there are many service-oriented applications suffer from poor quality and are hard to evolve [10]. The quality problem is common problem in any development approaches as quality is measured entirely based on the requirement. The software components obtain from global talent such as open sources, being used and tested in different environment where the quality can be improved. However it lack of formal quantitative quality measurement to assess its quality metrics. Therefore, there is need for a more precise quality measurement model, which can formalize the software quality metrics as discuss in [6,7], so that the software requirement quality can be visualize in a quantitative manner. This is important for the developer to know earlier the quality of the software before it is being develop. And therefore can help developer improve quality metrics which may require high accuracy.

Therefore, this paper will present a new framework on how the quality of heterogeneous software requirements measurement is addressed in formal platform. The heterogeneous software requirements are the requirements that are captured from different domains. Section 2 will discussed the formulation of formal platform of the quality framework. Section 3 will discussed the formal platform used to measure the heterogeneous software requirements, and finally the conclusion and future work of this research.

II. Proposed Framework

The methodology used in deriving the formal platform of the quality framework in which the framework is used for assessing the quality of heterogeneous software requirement is divided into four phases (i) Formulate heterogeneous requirement properties, (ii) Construct a formal platform, (iii) Develop software quality measurement and (iv) Evaluation of formalize Problem and Solution Platform.

A. Formulate Heterogenous Requirement Properties

The first step is to gather and categorize the software requirements of different domains. This is carried out by capturing the software from different domain. The requirement to each domain then categorize into functional requirements, non-functional requirements and the requirements environment. Later, the typical properties for each category are formulated, which later will be used to measure the quality of solutions.

B. Construct a Formal Platform

At this stage the formal platform for open innovation is constructed. This carried out by defining the architecture for the problem and the solution platform. The solution platform is constructed by defining the formal relationships between problems and solutions. The mapping rules are defined which are used to specify the formal relationships.

C. Develop Software Quality Measurement

This phase is related to the development of a new approach transforming subjective quality measurement (e.g., can generate huge number of reports) to qualitative measurement (e.g., can generate one hundred reports per second) using fuzzy values. This phase, determine the fuzzy values of software quality metrics (e.g., completeness, preciseness, consistency and correctness) for each solution elements. This is carried out by determining fuzzy values (truth values) of the quality metrics based on expert recommendation. The requirement quality is then calculated by using similar formula as being used [11].The prototype is developed to automate the mapping process and also generating the fuzzy values of software quality metrics.

D. Evaluation of Formalize Problem and Solution Platform

The evaluation of the propose framework is conducted at this stage. This is carried by gathering research problems/questions or assignment/project questions. These are used as inputs to the propose models. Operational testing is conducted with different domain/heterogeneous problems. The propose model is then evaluated based on the expert evaluation. This will be carried out by comparing the output produced by the propose model with the subject expert (e.g., researcher/lecturer on their specialist field).

The propose framework for measuring the software quality is as shown in Figure 1.

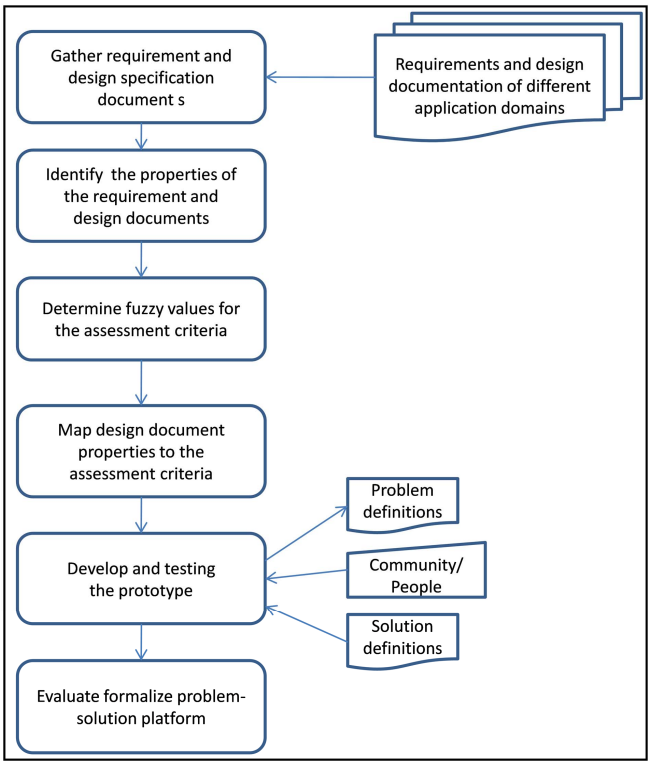

Fig. 1. Framework of Software Quality Measurement

This research is proposing the quantitative measurement of software quality model (formal platform) based on the requirements capture form the global talents (open innovation). The propose idea will be carried out by first identify the specifications of different software requirements and software design of different application domains (heterogeneous). This is expected to provide the typical properties and also the unique properties of the problem definitions. The platform of heterogeneous problems are build based on the problem definitions and its' propose solutions.

In order to define a precise model of the platform, formal specification language is used. The formal platform models includes: the problem definitions, the solution definitions and the assessment criteria. The new formal platform is expected to provide the functions to publish, store and assess the problems. The problems are published to the community/people whose has the same interest or those who would like to participate to propose the solution to the problem. The solutions from the community/people will be stored and assessed (quantitative measurement) based on the defined criteria in an attempt to seek for the best solution for the problem.

The quality of requirement specification is measured by using fuzzy logic. This research suggested that the measurement criteria (or quality metrics) are assigned with the quantitative values based on subjective measurement by domain expert in order to obtain the fuzzy values. Similar calculation algorithm defined in [11] which is based on average values will be used to derive the fuzzy values, which reflect the degree of software requirement quality.

To validate the result, the output from the propose model will be verified with the subject or domain expert. The involvement of domain expert and people in software development also expected to improve the understanding of the software development process and the solution to the problems.

III. Discussion and Framework Evaluation

This research is proposing the quantitative measurement of software quality model (formal platform) based on the requirements capture form the global talents (open innovation). The propose idea will be carried out by first identify the specifications of different software requirements and software design of different application domains (heterogeneous). This is expected to provide the typical properties and also the unique properties of the problem definitions. The platform of heterogeneous problems are build based on the problem definitions and its' propose solutions.

In order to define a precise model of the platform, formal specification language is used. The formal platform models includes: the problem definitions, the solution definitions and the assessment criteria. The new formal platform is expected to provide the functions to publish, store and assess the problems. The problems are published to the community/people whose has the same interest or those who would like to participate to propose the solution to the problem. The solutions from the community/people will be stored and assessed (quantitative measurement) based on the defined criteria in an attempt to seek for the best solution for the problem.

The quality of requirement specification is measured by using fuzzy logic. This research suggested that the measurement criteria (or quality metrics) are assigned with the quantitative values based on subjective measurement by domain expert in order to obtain the fuzzy values. Similar calculation algorithm defined in [11] which is based on average values will be used to derive the fuzzy values, which reflect the degree of software requirement quality.

To validate the result, the output from the propose model will be verified with the subject or domain expert. The involvement of domain expert and people in software development also expected to improve the understanding of the software development process and the solution to the problems, hence improve the quality of software to be produced.

This work is still at the initial stage. The framework had been defined. The requirements of different domains are study and the typical properties representing the requirement are defined. The requirements should be identified based on requirement unique identification number, the title and contents. The content then categorized into functional, non-functional requirements and requirements environments. These properties represented by using VDM++ data definition as shown in Figure 2 below.

```
types
reqdoc ::  fileFormat: FILE            -- type and author
           docFormat: DOCUMENT  -- ID, pageNo, font propertie
                                       -- requirement types, content
Quality :: precise : nat
           correct : nat
           consistent : nat
           complete : nat;
docQuality = map reqdoc to Quality;

instance variables
  quality : Quality
  inv quality = quality <=0 and quality >= 5;
```

Fig. 2. Document Properties

At this moment the works is focused on functional requirements. The functions are identified based on its name, and its purpose. The function quality is assessed based on the completeness, preciseness, consistency and correctness, as shown in Figure 3.

```
completeness : nat * nat ==> nat
Completeness ( categoryRequirement, totalFunctions ) ==  (
  dcl percentage : real := ( totalFunctions * 100 ) / categoryRequirement;
  dcl rate: nat;
  if (percentage = 100)
   then rate := 2
  else if ( percentage < 100 and percentage >= 50 )
    then rate := 1
  else rate := 0;
  return rate; );

consistency: nat * nat ==> nat
consistency ( SimilarActionTimes, totalSimilarOutput ) ==  (
  dcl PercentageOfConsistency : nat := (totalSimilarOutput/SimilarActionTimes)*100;
  dcl rate : nat;
  if (PercentageOfConsistency >= 80)
    then (rate:= 2;  )
  else if (PercentageOfConsistency < 80 and PercentageOfConsistency >= 50)
    then (rate:=1;  )
  else (rate:=0;  );
  return rate; );

correctness: nat * nat ==> nat
correctness (totalExpectedOutput, totalSameOutput)==  (
  dcl percentageOfCorrectness: nat:= (totalSameOutput/totalExpectedOutput)*100;
  dcl rate : nat;
  if (percentageOfCorrectness = 100)
    then ( rate:= 2; )
  else if (percentageOfCorrectness < 100 and percentageOfCorrectness >= 50)
    then ( rate:=1; )
  else ( rate:=0; );
  return rate;              );

preciseness : nat * nat ==> nat
preciseness (totalOfPreciseOutput, totalOutput) == (
  dcl percentage : real := (totalOfPreciseOutput * 100)/ totalOutput;
  dcl rate : nat;
  if (percentage = 100) then rate := 1
  else rate:= 0;
  return rate; );
```

Fig. 3. Initial Formal VDM++ Model

The software requirement quality is measured based on the fuzzy values assigned to the quality properties namely preciseness, correctness, consistency and completeness. Fuzzy values are calculated based on average values [11].

The quality properties of each document are representing as a set:

```
precise quality, PQ = {pq1,pq2,…, pqn}
correctness quality, CQ = {cq1,cq2,…, cqn}
consistency quality, TQ = {tq1,tq2,…, tqn}
completeness quality, LQ = {lq1,lq2,…, lqn}
```

Fuzzy values that representing the quality measurement is mapped to each of the document. For example to measure the quality of a document, the total fuzzy values of document is divided by number of quality attributes used to measure document one. This can be illustrated as below:

$$Quality = Q_{doc}(n) = \frac{\sum pq + cq + tq + lq}{n} \quad (1)$$

Where pq, cq, tq and lq are the fuzzy values for preciseness, correctness, consistency and completeness of the document respectively, and n is the number of quality properties. In this case the number of quality properties are four, therefore *n i*s equal to 4. The total score of these quality properties Q_{doc} indicate the degree of quality of the software requirement.

IV. Conclusion and Future Work

The completeness, correctness, preciseness and consistency of software requirement are very important to guarantee that software produced is reliable and fulfill user need. The software quality is measured based on either its fulfill the user requirements, and it is well known that the completeness, correctness, preciseness and consistency of software requirement contribute to software quality. The measurement of these criteria will help developer assess the quality of software being produced. Formal model is deployed to build the platform in order to create a rigour tool for measuring the quality of software requirements.

To date, the work is refining the formal model for representing and measuring the functional requirements. Next step is defining the formal platform for non-functional requirements and requirements environment. And later, the formal platform needs to be evaluated to assess its accuracy in measuring the requirements quality.

Acknowledgement

The authors would like to thank Universiti Malaysia Sarawak for providing the funding to publish and present this paper. This work is supported by Fundamental Research Grant Scheme: FRGS/02/2014/ICT07/UNIMAS/02/1

References

[1] Topcoder, Enterprise Open Innovation, http://community.topcoder.com/tc

[2] Kieran Conboy, Brian Fitzgerald, Method and developer characteristics for effective agile method tailoring: A study of XP expert opinion, Transactions on Software Engineering and Methodology (TOSEM), Volume 20 Issue 1, ACM Jun 2010.

[3] Angela Martin, James Noble, Robert Biddle, Programmers are from Mars, customers are from Venus: a practical guide for customers on XP projects, PLoP '06: Proceedings of the 2006 conference on Pattern languages of programs. ACM 2006.

[4] Christian Schmitt, Kai Fischbach, Detlef Schoder, Enabling open innovation in a world of ubiquitous computing, ADPUC '06: Proceedings of the 1st international workshop on Advanced data processing in ubiquitous computing (ADPUC 2006), ACM Nov 2006.

[5] Christian Lescher, Patterns for global development: how to build one global team?, EuroPLoP '10: Proceedings of the 15th European Conference on Pattern Languages of Programs, ACM Jul 2010.

[6] Jakob Mund, Quality Assessment of Requirement Specifications using Metrics– A Research Proposal – , IDoESE ’13 Baltimore, Maryland USA.

[7] Shahid Iqbal and M. Naeem Ahmed Khan, Yet another Set of Requirement Metrics for Software Projects, International Journal of Software Engineering and Its Applications Vol. 6, No. 1, January, 2012.

[8] Martin Monperrus, Benoit Baudry, Joël Champeau,Brigitte Hoeltzener, Jean-Marc Jézéquel, Automated Measurement of Models of Requirements, "Software Quality Journal 21, 1 (2013) 3-22" DOI : 10.1007/s11219-011-9163-6

[9] A. Goeb and K. Lochmann, A software quality model for SOA, WoSQ’11, September 4, 2011, Szeged, Hungary.

[10] G. Feuerlicht, "Simple Metric for Assessing Quality of Service Design Service-Oriented Computing." vol. 6568, E. Maximilien, et al., Eds., ed: Springer Berlin / Heidelberg, 2011, pp. 133-143.

[11] [11] Edwin Mit and Ng Bong Ding, Framework of Indigenous Knowledge Representation, Proceedings of 5th International Conference on Intelligent Systems, Modelling and Simulation, 26-29 Jan 2014, Langkawi Malaysia