Software Component Quality Evaluation

Mebarka YAHLALI
Computer Science Department, USTO-MB University
Yahlali@univ-usto.dz

Abdellah CHOUARFIA
Computer Science Department, USTO-MB University
Chouarfia@univ-usto.dz

Abstract – Component-Based Software Engineering (CBSE) provides for developers the ability to easily reuse and assemble software entities to build complex software. It is based on the composition of prefabricated software entities called components. In this context, the selection step is very important. It consists of searching and selecting appropriate software components from a set of candidate components in order to satisfy the developer-specific requirements. In the selection process, both functional and non-functional requirements are generally considered.

In this paper we present a method enabling the evaluation of software components quality. This method allows us choosing the best component in term of non-functional needs.

Keywords – Software components quality, Component quality model, Quality evaluation, components selection.

1. INTRODUCTION

The components approach has become an important alternative for building software applications, and specially distributed systems. This approach tries to improve the flexibility, re-usability and maintainability of applications, and helps develop complex and distributed applications deployed on a wide range of platforms, by plugging commercial off-the-shelf (COTS) components, rather than building them from scratch.

Software component. One of the definitions most often quoted is given by Szyperski and Pfister [1][2]: "A Software component is a unit of composition with contractually specified interfaces and explicit context dependencies only. A Software component can be deployed independently and is subject to composition by third parties". A software component has mainly three elements [3]: (a)Functional interfaces and configuration properties: The required functional interfaces must be satisfied when a component instance is created so that this later can be used through the provided interfaces.(b) Control interfaces (provided/required): are the set of methods which allow managing the component instances' life cycle during the execution. These methods are intended to be called by the execution environment of the components model. (c) Dependences and deployment properties: the dependences are specific to each implementation of a component.

Currently, we can find several similar components .i.e. they provide the same functional requirements. The problem is "how to choose a component of good quality?". In this article we try to estimate the quality of each component in order to select the best among several equivalent propositions. Summarize the main function of our evaluation methods is to produce a single numerical value representing the quality offered by each component.

This paper is organized in three sections. After this introduction, section 2 presents an outline on software quality and in section 3 we present our assessment process.

2. THE QUALITY

2.1. Standardized Aspects of Quality Evaluation.

Factor or Characteristic: Software characteristic which contributes to its quality [5]. It relates to the used of characteristics. The factors translate the external vision [6].

Criteria or Subcharacteristic: a factor can be evaluated via these elements [5].

Attributes: The quality criteria are connected to attributes which are a posteriori measurements.

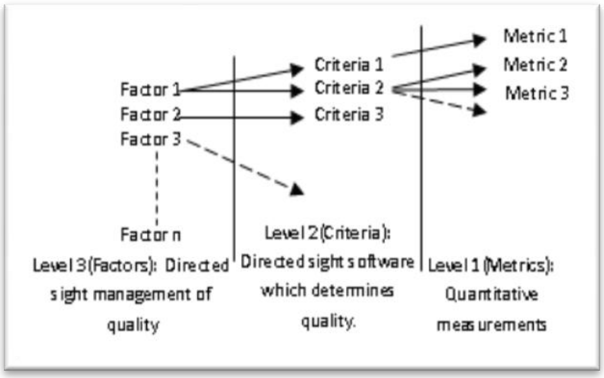

Figure 1. Quality levels

2.2. Software Component quality model

A Quality Model is defined as: "The set of characteristics and the relationships between them which provide the basis for specifying and evaluating quality requirements."[7].

There are more than 300 developed standards and maintained by more than 50 different organizations[7].The first quality model considered as a standard was developed and published by the International Standardization Organization in 1991 as ISO 9126 [8]. Ten years A new international initiative Software product QUality Requirements and Evaluation (SQuaRE) was set up aiming to develop set of norms ISO/IEC 25000 [9]. This new approach is perceived as new generation of software quality models.

Characteristics	Sub-characteristics
Functionality	Accuracy, Security, Suitability, Interoperability ,Compliance
Reliability	Faults Tolerance, Recoverability, Maturity , Compliance
Usability	Configurability, Understability, Learnability , Attractiveness, Operability, Compliance
Efficiency	Time Behavior,Resource Behavior, Compliance
Maintainability	Stability , Changeability , Testability , Analysability , Compliance
Portability	Deployability, Replaceability , Adaptability, Reusability

Table1. SQuaRE quality model [10]

3. SOFTWARE COMPONENT Quality

The objective of this work is to calculate the software component quality value.

In such quality evaluation process, the presence of a quality characteristics' description of each component is crucial; that means to be able to satisfy the quality it is important to add the necessary quality attributes to the product description [11]. For this we have proposed in [12] syntax for specifying the software component quality. The following example illustrates the specification of a quality of a component according to the syntax defined in [12].

Example 1:

DEFINE-QUALITY
Reliability ={
Maturity = {Volatility =0,25;
Failureremoval=0.49}
FaultTolerance= { Mechanismavailability=0.6;
Mechanism_Efficiency =0.31};
Recoverability={ Error –Handling=0.20}
};
Usability = {
Operability={ Provided_interface =0.55;
Required_interface=0.36};
Configurability ={ Configuration Effort =0,3 }
}.

3.1 Component Quality Evaluation

Let assume that:
F: a set of component factors: $F= \{f_{i\ /\ i=1..n}\}$
C: a set of criteria for F: $C= \{c_{j\ /j=1..m}\}$
A: a set of attributes relating to C: $A= \{a_{l/\ l=1..k}\}$

The quality model is decomposed into 03 levels: the first is the attributes, the second criteria and the third represents the factors. For this we define two relations:

1. "**presented by**": this relation is defined as follow:

E_i is presented by $e_{j\ j=1..n}$: means that the element E_i is presented by the set $(e_1,e_2,e_3\ldots e_n)$
For this work two cases can be distinguished:
$\forall$ $f_j \in F$ each factor fj is presented by a set of criteria
$\forall$ $c_i \in C$, each criterion cj is presented by a set of attributes

2. " **Evaluated by** ": if an element E_i is presented by $e_{j\ j=1..n}$ the quality value of E_i : $V(E_i)$ can be calculated based on the quality values of the elements e_j: $V(e_j)$. Whereas:

If E_i is presented $e_{j\ j=1..n}$ $\Rightarrow$ $V(E_i)$ is evaluated by $V(e_j)$ / j=1..n

3.1. Evaluation Process

Our evaluation process contains 03 main steps:
Step 1: In this step the user must classify the quality characteristics (factors) according to his non functional needs.

Step 2: Factors weights are calculated depending on the user classification using ROC (Rank Order Centroid) concept [4]. Centroids classification is a way to convert from the ranks (1st, 2nd, 3rd) in notes or weights which are numerical value. If n is the number of attributes, the weight (W) of attribute k (A_k) is [5] :

$$W(Ak) = \frac{\left(\sum_{i=k}^{n} \frac{1}{i}\right)}{n}$$

For Criteria and attributes weights we assign equal weights, that is to say if an element E has n sub-element E', the weight of each element E' is: $W(E)=\frac{1}{n}$

Step 3: Evaluation : Figure.2 summarizes the calculating process, with the following rules:

$$Qual\_V = \sum_{i=1}^{n} \left(V(Fi) * W(Fi)\right) \qquad \text{(R1)}$$

$$V(Fi) = \sum_{j=1}^{m} \left(V(Cj) * W(Cj)\right) \qquad \text{(R2)}$$

$$V(Ci) = \sum_{p=1}^{k} \left(V(Al) * W(Al)\right) \qquad \text{(R3)}$$

Such as:

- Qual_V : overall quality.
- $V(F_i)$: value of factor F_i.
- $V(C_j)$: value of criteria C_j.
- $V(A_l)$: value of attribute A_l.
- W: weight.

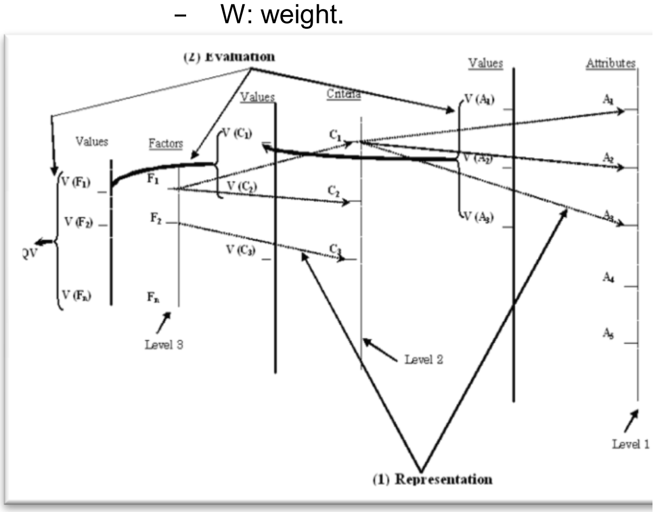

Figure 2. Component quality Evaluation

3.2. Case study

We will apply our evaluation process on the example 1 presented in Section 3. In this example, there are two quality factors: reliability and usability. These factors are defined as follows:

1. Reliability is presented by (Maturity, FaultTolerance, Recoverability).
2. Usability is presented by (Learnability, Operability, Configurability)

Factor	Criteria	Attribute	Val
Reliability	Maturity	Volatility	0.25
		Failure _removal	0.49
	Fault Tolerance	Mechanism-availability	0.6
		Mechanism_ Efficiency	0.31
	Recoverability	Error -Handling	0.20
Usability	Learnability	time_effor_to_ configure	0.69
	Operability	Provided_interface	0.55
		Required_interface	0.36
	Configurability	Configuration Effort	0.3

a) Weightings calculation:

For this evaluation, we consider that Usability is more important. In this case weightings are calculated as follows:

1. Usability weight :

$$W(Usability) = \sum_{i=1}^{2} ({}^{1}/_{i})/2$$

=[(1/1)+(1/2)]/2 =0.75

Usability is presented by three criteria, weight of each one equal to 1/3=0.33. For level 1 (attributes) we used the same principle.

Criteria	weight	Attribute	weight
Maturity	0.33	Volatility	0.5
		Failure _removal	0.5
Fault Tolerance	0.33	Mechanism-availability	0.5
		Mechanism_ Efficiency	0.5
Recoverability	0.33	Error -Handling	1

2. Reliability weight :

$$W(Reliability) = \sum_{i=2}^{2} ({}^{1}/_{i})/2$$

= [(1/2)]/2=0.25

The Reliability criteria and attributes' weights are calculated as follow:

Criteria	weight	Attributes	weight
Learnability	0.33	time_effor_to_configure	1
Operability	0.33	Provided_interface	0.5
		Required_interface	0.5
Configurability	0.33	Configuration Effort	1

b) Quality values calculation:

To calculate the global quality value , the criteria and the factors values must be calculated starting from the attributes values:

Usability value:

Criteria values: applying R3

- **V(Operability)=**V(Provided_interface)*W(Provided_interface)+V(Required_interface)*W(Required_interface)= 0.5*0.55+0.5*0.36=0.45

V(Operability)= 0.45

Apply the same rule (R3) for learnability and Opérability , we find:

- **V(Learnability)=** 0.69*1=0.69
- **V(Configurablity)=**0.3*1=0.3

Now, we use R2 to calculate Usability value:
V(Usability)=V(operability)*W(operability)+V(Learnability)*W(Learnability)+ V(Configurablity)*W(Configurablity) = 0.33*0.45+0.33*0.69+0.33*0.3=0.47
V(Usability)=0.47

Reliability value: We will apply the same calculation process, we find: **V(Reliability)=0.34**

Overall quality: for this level we use R1;
V_qual =V(Reliability)*W(Reliability)+V(Usability)*W(Us-ability)=0.34*0.25+0.47*.75
V_qual =0.43

4. Conclusion and future work

This paper presents a method allowing comparison of software components in term of quality. We have proposed a method for assessing the component quality. This method is Applicable to any quality model, it facilitates the comparison of components by producing a digital value represents the quality that a component can offer, and it minimizes the time to compare a few equivalent components. In order to enhance this methodology, we proposed a generalization of the proposed process of assembling software system based on the assembly of several components, i.e. to know the value of the quality offered by an application from its components.

5. REFERENCES

[1] A. Brown, K. Wallnau,(1999), “The Current State of CBSE”. IEEE Software, 37_46.

[2] C.Szyperski and C. Pfister,(1996), “COP’96 Workshop Repor”, International Workshop on Component-Oriented Programming, University of Linz, Austria.

[3] Humberto CERVANTES, “Towards a components directed services model to support the availability dynamic“, Doctoral Thesis, 29 March 2004, University of Joseph Fourier, GrenoubleI; France.

[4] McCaffrey, J., & Koski, N. Competitive Analysis Using MAGIQ. MSDNMagazine. Octobre 2006.

[5] McCaffrey, J.D,Using the Multi-Attribute Global Inference of Quality (MAGIQ) Technique for Software Testing, Information Technology: New Generations, 2009.

[7] Neelam Bawane nd C. V. Srikrishna (2010): A Novel Method for Quantitative Assessment of Software Quality, International Journal of Computer Science and Security, Volume 3: Issue (6)

[8]ISO/IEC 9126 (1991) “JTC1/SC7, Information Technology – Software Product Quality”, International Standardization Organization,

[9] ISO/IEC 25000(2005) “ JTC1/SC7, Software Engineering - Software product Quality Requirements and Evaluation (SQuaRE)”, International Standardization Organization,

[10] Dave Zubrow,(2004),”Software Quality Requirements and Evaluation, the ISO 25000 Series”, Carnegie Mellon University.

[11] Antkiewicz, M., K. Bąk, A. Murashkin, R. Olaechea, J. Liang, K. Czarnecki**,**2013**, “**Clafer Tools for Product Line Engineering”, Software Product Line Conference, Tokyo, Japan.

[12] Chouarfia Abdallah, Yahlali Mebarka,(2009), “Software Components Assembly with an Appreciated Qos” , Aug 2009, *Journal of Software*, Vol 4, No 6.