Knowledge-Based Systems 000 (2016) 1–9

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

Software test quality rating: A paradigm shift in swarm computing for software certification

Saad M. Darwish*

Department of Information Technology, Institute of Graduate Studies and Research, Alexandria University, 163 Horreya Avenue, El Shatby, P.O. Box 832, Alexandria 21526, Egypt

ARTICLE INFO

Article history:
Received 19 April 2015
Revised 15 July 2016
Accepted 16 July 2016
Available online xxx

Keywords:
Software quality
Quality assurance
Software certification model
ACO

ABSTRACT

Recently, software quality issues have emerged to be recognized as a fundamental point as we actualize an extensive growth of organizations involved in software industries. Still, these organizations cannot ensure the quality of their products; therefore abandoning customers in uncertainties. Software certification is the branch of quality by means that quality requires to be measured prior to certification admitting process. However, creating an official certification model is difficult due to the deficiency of data in the domain of software engineering. This research participates in solving the problem of assessing software quality by introducing a model that handles a fuzzy inference engine to mix both of the processes–driven and application-driven quality assurance procedures. The fundamental purpose of the suggested model is to enhance the compactness and the interpretability of the system's fuzzy rules via engaging an ant colony optimization algorithm (ACO), which attempts to discover a good rule description by a set of compound rules initially represented with traditional single rules. The proposed model is a fitting one that can be seen as practicing certification models that have already been created from software quality domain data and modifying them to a context-specific data. The model has been tested by a case study and the results have confirmed feasibility and practicality of the model in a real environment.

1. Introduction

The past decade has realized the rapid evolution and dispersion of software that has become necessary in everyday life; consequently, the quality of the software is a large disquiet, spirited and acute. IEEE expresses software quality as the suitability for use of the software product and to obey to the requirements and to offer valuable services [1]. Software companies are contending to create software that is appealed to be good and achieve user's hope and requests. As a general law, quality is in the eye of the perceiver because different people may understand quality in their own viewpoints. Software quality assurance touches both direct productivity and long-term custody of client attention.

In the earlier engendering of software evolution, software quality is ranked through static estimation of code's structure [1]. Conveniently, a new development generation comprehends that software quality is more than just static features; it should also encompass non-functional, behavioral and human characteristics [2]. People who are embodied in software such as users, developers are fitting more attention on the other perspectives or interpretations of quality. As a rule, worthy software development procedures do not promise the brilliant quality of the product. Thus, assessment of the end product must be independent of the development procedure. Assessing software quality at the early stages of the design and development process is very complicated since most of the software quality features are not directly computable. Nonetheless, quality can be acquired from other measurable attributes. For this scope, software quality models have been broadly employed.

In the literature, software quality can be estimated via three classes of assessments [3]: (1) Internal measures: depend on internal characteristics that associating to how the software product is established; (2) External measures: based on external attributes that classically quantifying the performance of the code when executed; and (3) Quality is use measures: include elementary set of characteristics that influence the software like productivity and gratification. Even though there are more than a few models of software quality accessible from literature (see [2] for more details); it is still supposed that quality is a composite notion, since the quality is nothing more than a recommendation some like it excellent, good or bad. Thus, there can be no single, simple evaluation of software quality appropriate for everyone. In this context, the academics express of software quality assessment models that can be hired to form associations between the quantifiable attributes on the one hand and the software quality characteristic of interest on the other.

* Corresponding author.
E-mail address: saad.darwish@alex-igsr.edu.eg, saad.darwish@gmail.com

http://dx.doi.org/10.1016/j.knosys.2016.07.022
0950-7051/

Certification of software supplies societies more certainty and assurance about the software. Certification of software assistances software purchases, procurement, and can be used to confirm statutory obedience or to realize adequate deliverables in outsourcing. The evolution of software certification methodology is a new invention in the field of software assessment, which will increase user sureness toward the quality of the software [4,5]. ISO outlines certification as a process by which a third party offers printed agreement that a product, process or service obeys detailed physiognomies [5]. Software certification can be seen in three different viewpoints: personal, processes and product known as a certification triangle [6]. The combination of these views will create the best results. At the recent time, societies are starting to approve the certification notion in the software industry.

Two reasons that trust in software certification must arise from someone other than the software publisher. The motives are [5,6]: (1) by authorizing a third party to license software certificates, publishers move the accountable on responsibility concerning quality onto someone else, and (2) fair assessment from the independent firm may assist end users. The approach on certifying software via independent third party organization has a potential to adjust the way in which software is gauged, categorized and retailed. One tactic to demeanor this approach is through the participation of end users in the process. In this approach, the independent certification body gathers relevant information from the user's environment and how the product is working. The respective agency then gives the report based on substantial functioning practice generated by the users. The second approach is through self-certification by the developer. Each of these methods to measure software quality has its benefits and shortcomings.

A few certification models have been established and constructed in the literature that linked with two different approaches [7–15]. The main objective of the first approach is in confirming that the software development technique is carried out successfully and skillfully to meet the predictable quality criteria. It is chiefly focused on significant aspects such as the quality of the process, the quality of the elaborated people, the use of development technology, the constancy of working environment, and finally the project circumstances. The second approach emphases on product quality perception. This approach lays stress on aggregating quality factors into several working zones. It consists of four key elements: pragmatic quality, assessment team, certification representation, and product certification repository. Each of these specific parts has distinct quality criteria and implements separate procedures and jobs in the assessment and certification process.

In general, the above-mentioned models are envisioned to be comprehensive and recognize both the inner and outer quality attributes of the software. But, unusually these attributes have been prearranged into a kind of systematic framework. At the same time, there are no strategies on how to contribute a complete assessment of quality rather than centering on the user interpretation of the software. There is a further issue that joins to the quality; there is precedence between attributes that entail conveying a weight (by the owner) to reflect the business requirements. In ISO 9126 all attributes are equally important [6]. A reader looking for more information regarding the survey and state-of-the-art software certification models can refer to [16]. The related works strengthen the demand for better quality evaluation and certification that can be fashioned to confirm that users will be receiving software packages that meet ordinary and contracted quality.

The fuzzy logic-a mathematical mechanism for dealing with vagueness and information granularity- gives major benefits over other approaches due to its aptitude to logically symbolize the qualitative feature of examination data and implement adaptable inference rules. Recently, fuzzy logic is used in many software development applications [17–22]. These efforts were mainly focused on the applicability of the fuzzy logic concept either in the fuzzification of the software characteristics to describe the features in a linguistics term or through building fuzzy inference rules to merge different characteristics to assess the quality of the software. Still, there are some limitations of fuzzy logic-based software development; the most important of which is that the knowledge does not always completely exist and the manual tuning of all the base variables takes time. Also, the absence of portability of the rule bases when the sizes of the software scale modification make the difficulty still more serious. To deal with these difficulties, the fuzzy rule learning (*FRL*) technique has been presented [23].

1.1. Problem statement and objective

Evaluating software quality at the early stages of the design and development process is very problematic since most of the software quality characteristics are not instantly assessable. However, they can be inferred from other quantifiable attributes. The quality of software is related with a number of quality attributes and assessment of software quality includes wide interpretations and innumerable viewpoints that might encompass regular explanation, in linguistic terms. Semantic terms are more suitable to use when human declare the subjectivity and fuzziness of their evolving but these linguistic variables embroil uncertainty and indistinctness. For this purpose, software quality certifying models have been extensively applied. However, building accurate certifying models is hard due to the lack of data in the domain of software engineering. As a result, the certification models built on one data set show a significant deterioration of their accuracy when they are used to certify new, unseen software. Since fuzzy logic deals with the ambiguity, imprecision and vagueness, therefore, the objective of this paper is to present a bio-inspired fuzzy rule learning approach for developing quality estimation framework with the aim of optimizing the accuracy of certifying models when used to classify new software data.

1.2. Novelty

Although, several routines and third party mechanisms have been recommended in the past for software certification procedure, but to the best of my knowledge (based on Goggle scholar) no certification scheme automatically manages both of the preferences between attributes and the compactness of the rule bases within a kind of a systematic fuzzy rule learning framework. In contrast with state-of-the-art certification models that entail specifying quality properties weight by hand to catch the business requirements; the weight values established in the suggested model are based on a bio-inspired calculation that verbalizes software engineering problems as optimization problems and utilizes metaheuristics technique, ant colony optimization, to resolve them.

This pioneering research builds a scalable certification model based on ant colony optimization (ACO) to find a set of compound rules to enhance the compactness (density) and the interpretability of the model's fuzzy rules with the aim of accurately certify new software. This set of compound rules can accommodate any number of good characteristics without being accompanied by a large number of rules and thus the result of certification is more general and comprehensive. This is suitable for the domain of software quality since there are a number of different aspects of quality properties that are known to positively influence the software quality and hence certification models built from one data set are hard to generalize and reuse on new data.

Please cite this article as: S.M. Darwish, Software test quality rating: A paradigm shift in swarm computing for software certification, Knowledge-Based Systems (2016), http://dx.doi.org/10.1016/j.knosys.2016.07.022