International Journal of Computer Science Engineering and Information Technology Research (IJCSEITR)
ISSN(P): 2249-6831; ISSN(E): 2249-7943
Vol. 4, Issue 2, Apr 2014, 225-242

DEVELOPMENT OF AN AUTOMATED FRAMEWORK TO RESOLVE SOFTWARE TESTING ISSUES

CHITTINENI ARUNA[1] & R. SIVA RAM PRASAD[2]

[1]Research Scholar, Acharya Nagarjuna University, Assistant Professor, Department of CSE, KKR & KSR Institute of Technology and Sciences, Guntur, Andhra Pradesh, India

[2]Research Guide, Department of CSE and Head, Department of IBS, Acharya Nagarjuna University, Guntur, Andhra Pradesh, India

ABSTRACT

The growing importance of Software in the present scenario can be attributed to the fact that software is utilized in several different ways for different issues to provide different types of solutions. The subject of Software Engineering is so vast and varied that it encompasses several areas such as manufacturing of Refrigerators, Air Conditioning Systems, Automobiles, Space Engineering, Wireless and Mobile Communications. Therefore, in most cases, software usually need to adhere to a specifications so that they can perform to produce only those results that are expected and enunciated.

In general, keeping in view of the present scenario, a software professional goes through a certain process to establish that the software conforms to a given specification. This process, verification and validation (V & V), ensures that the software conforms to its specification and that the customers ultimately receive what they ordered. Software testing is one of the techniques to use during V & V. To be able to use resources in a better way, computers should be able to help out in the "art of software testing" to a higher extent, than is currently the case today. The main issue is to not retrench human resources from the process of software testing itself altogether but to consider the fact that software engineering is an art and science of identifying suitable solutions to critical problems by not just involving computers alone to solve the testing issues but software engineers should themselves participate and come out with better and suitable solutions to those problems where computers are clearly not that good to provide appropriate solutions.

This research work presents a systematic and methodical approach aimed at examining, classifying and improving the concept of automated software testing and is built upon the assumption that software testing could be automated to a higher extent. Throughout this thesis an emphasis has been put on "real life" applications and the testing of these applications.

One of the contributions in this thesis is the research aimed at uncovering different issues with respect to automated software testing. The research is performed through a series of case studies and experiments which ultimately also leads to another contribution—a model for expressing, clarifying and classifying software testing and the automated aspects thereof. An additional contribution in this thesis is the development of framework which in turns acts as a broad substratum for a framework for object message pattern analysis of intermediate code representations. This is achieved keeping in view of the procedures relating to code optimization and effective code generation at all levels. The results that are expected in this thesis indicate how software testing can be improved, extended and better classified with respect to automation aspects. The main contribution lays in the investigations and the improvements related issues with regard to automated software testing. A comprehensive study has been made by the researchers to tackle the very crust of the problem of software testing methodologies.

It has to be noted that testing software in an automated way has been a goal for researchers and industry during the last few decades. Nevertheless, the success rate has not been readily available. Some tools that are partly automated have evolved, while at the same time the methodologies for testing software has improved, thus leading to an overall improvement. Even today, much of the software testing procedures are performed manually which is not reliable, prone to errors, inefficient and also a costly affair.

Even today several questions still remain unanswered. For instance, we have the question of when and how to stop the process of testing software when is it not economically feasible to continue to test software? It is very much evident that investing so much of time and money for testing an application which will be used only for a few times, in a non-critical environment, is probably a waste of resources.[1] At the same time, when software engineers develop software that will be placed in a critical domain and extensively used, an answer to that question needs to be found.

Secondly, there is the problem of resources. It is to be noted that small- and medium-sized companies are today, as always, challenged by their resources, or to be more precise, the lack thereof. Deadlines must be kept at all costs even when, in some cases, the cost turns out to be the actual reliability of their products. In addition to the growing complexity of software, the size of the problem becomes more and more complex and huge.

Keeping in view of the question of complexity and to be more precise based on the fact that software has grown in complexity and size which have become very much part of the problem of software testing and the type of research work that needs to be done to improvise on these issues is to be given a methodical and systematic approach.

KEYWORDS: Structural Testing, Commercial-off-the-Shelf, Component Based Software Engineering, Automated Software Testing, Empirical Evaluation, Verification and Validation