Chapter 6

CONCLUSION

6.1 Introduction

In today’s world, technology and its advancements are source of differentiation from competitors. The competition to produce more, with a very limited input value and be reactive to the endlessly changing needs and wants of internal and external customers. That is why flexibility, ability and price cutting are main reasons that are pushing increasingly a lot of firms to adopt Open supply Enterprise Resource Planning systems. The prime objectives of this research work are to analyze whether or not Open Source ERP systems will satisfy the requirements of enormous organizations additionally as Small to Medium Enterprises (SMEs). The thesis tries additionally to answer the question of whether Open supply ERP vendors provide adequate level of support to their purchasers.

First, the background of the analysis work and also the motivations behind it are reviewed. The conception of Software Reliability, Open Source, its evolution and also the contribution of Open Source ERP systems are conferred. In order to verify the analysis claim, a comprehensive review of the literature discussing the software reliability, empirical approaches as well as ERPs choice criteria is discussed. This review resulted in a very set of dimensions that served to make the analysis model. Another part of the analysis is that the “features” that were compiled by looking at the feature giving of the various ERP systems. This model served as a guiding principle once evaluating the chosen Open source ERP systems. However, the chosen Open Source ERPs conferred limitations once it came to their use in massive organizations: those limitations will be summarized to their quantifiability as there are still doubts regarding the power of these systems to handle massive volumes of users or requests, and their ability to be scaled in cluster-like modes. Another limitation impacting massive organizations is that the lack of support for multiple international accounting laws that is essential for in public listed organizations.

The proposed research work is totally concentrated on the modern software development world where two universally accepted and followed ways of software
development are proprietary and open source. Both these two different software development types have one very crucial element in common: the success of the end product. Even though the business success of open source can be judged in terms of profit or loss, the developed software and the project itself are subject to criteria such as reliability, maintainability, and security as success indicators. The proposed system will only term a software project and the developed software as reliable only if the end product can satisfy the requirements of metrics measuring these criteria. The proposed thesis work will focus on analyzing a real time ERP open source for evaluation of different reliability models and current proposed metrics for the open source software development in order to establish a software reliability model for the free and open source reliability models and metrics to be compared and contrasted and a classification of different open source paradigm is highlighted. These findings are also analyzed and verified on different case studies on open source software reliability evaluation considering ERP package e.g. OFBiz as a case study.

6.2 Limitations

The limitations of the proposed research work are as follows:

- The study is focused purely on OFBiz; however, the model for analysis is evaluated with many standard distribution models in order to maintain the flexibility of usage of the model in other ERP packages too. Other applications of CRM, SCM, Payroll processes that run in OSS are not considered in the study.

- The experimental test environment is medium scale. The complete experiment on reliability of OFBiz is done on Windows (2000/XP/Vista), Linux (Red hat), and Mac OS. However, experimenting on various hardware standards (processor, RAM, buffer, plugin etc) are very tedious task to accomplish. Hence virtualization software (VMWare) is used to overcome the issue.

- The proposed model will have optimal result only on java based scripts (Java Server Pages) and its associated java environment, however, focus was not towards other programming scripts (OpenERP-Python, WebERP-PHP etc)
6.3 Recommendations

In this thesis study different software reliability metrics, related constituent based on these metrics and different OFBiz paradigms were analyzed targeting to achieve reliability in OFBiz. All these themes are developed to determine whether an OFBiz application being developed is ready for use and how it performs compared to similar peers. Even after performing these metrics, following the proposed models and methods, it is really hard to decide that an OFBiz is fully reliable and to choose the viable software. We have also seen that the current state of Software Reliability definition needs to be improved for OFBiz. The case-studies also showed that evaluating the reliability of OFBiz is a tough process and it is almost impossible to find a unified approach to evaluate each and every OFBiz product. Following my analysis on the state of art and available proposals about the OFBiz paradigms we can summarize the properties of reliable OFBiz based on the categories provided by Metcalfe [Metcalfe, 2004] and I will suggest that a successful open source software reliability model should take the following points into consideration:

- **Reputation**: Successful software brings up reputation for its name and the developers among the similar alternatives in the market. In the open source case, the reliability of the software increases with the contribution of developers with high reputation. The quality of the output increases as we discussed about the developer contributions and community influence.

- **Ongoing Effort**: The evidence of ongoing effort is an indicator of an effort of creating quality software. Number of bug fix requests, fixes applied, implemented features and active communication channels should be considered as input data for reliability metrics. These ongoing efforts also encourage the participation of further developers, resulting in easier bug spotting, faster code implementations and finally leading to a more robust and reliable software.

- **Standards and interoperability**: The target product should implement the open standards and the product lifecycle and characteristics should be planned and based on these standards. In addition to this, interoperability with other software is crucial since an ongoing open source project can be embedded into further projects or used simultaneously with another one. As we discussed in
the reliability metrics and models such as reliability blocks and Markov model, individual reliability of components directly affects the reliability of the system they are used in.

- **Support:** The open source project should have a remarkable and active support community in order to assess the quality evaluation. Again the frequency of bug reports and response times, number of contributors, developer profiles and documentation can be the evaluation criteria for open source software while creating a reliability model.

- **Support:** Similar to the community support, the reliability and quality of an OFBiz project can be assessed by the availability of commercial support. However, commercial support is usually available for more widely used products by the bigger companies. This kind of support can be an extra or meta-evaluation criterion in an OFBiz reliability model.

- **Version:** The release of a new version usually comes up with solutions to existing bugs in the previous version. Extra feature requests can be handled, existing features can be enhanced and code quality can be increased. Hence, the overall quality and reliability of the OFBiz increases. In order to be used for reliability metrics or evaluation criteria of an OFBiz reliability model, relevant data about the releases such as number of versions and bug fixing frequencies are fundamental.

- **Documentation:** For all the current developers, further contributors and end users, documentation is a central resource to assess the usability and quality of the software. Furthermore, from the documentation the history of bug fixes, implemented features and open issues can be traced. In addition, the overall characteristics of the end product can be derived from the documentation and the evaluation of how much it satisfies the user needs and how reliable it is can be concluded. For OFBiz reliability model the quantity of available documentation and the depth of functional information can be used as evaluation criteria.

- **Coverage:** The resulting product of an OFBiz project must be specialized before use. This means narrowing down the range of tasks the software can handle for a specific purpose or user need. As a result of specialization the effort for constructing and deploying it reduces and the efficiency increases
and therefore the overall performance and reliability of the software increase. Maintainability and modularity of the code will also be affected in a positive way, increasing the user satisfaction levels. In an OFBiz reliability model, relevant data for coverage, maintainability and modularity can be used as evaluation criteria.

6.4 Future scope of work

This section presents the salient aspects of some the future work that can be taken up in the field of Software Reliability Engineering.

- **Improving Parameter Estimation**: Maximum likelihood estimation of parameters based on failure data taken during execution yields reasonable results, but there appears to be considerable room for improvement.

- **Fault Density Prediction**: Accurate means of predicting fault density are needed if we are to predict the parameters of the exponential model so that it can be used prior to program execution. At the present time investigators have identified some of the factors that appear to affect fault density, based on a moderate number of projects.

- **Fault Exposure Ratio**: The fault exposure ratio is the ratio of the initial failure intensity at the start of system test to the product of the linear execution frequency and the number of inherent faults. The linear execution frequency is the average instruction execution rate divided by the object program size. It relates reliability to fault density.

  Fault exposure ration may be constant or close to it. This must be verified over a larger sample of projects. If it is not constant, then the factors that influence it need to be identified and relationship determined.

- **Fault Reduction Factor**: The fault reduction factor is the ration of net fault reduction to failures experienced as time of execution approaches infinity. We need to determine its value over a wide variety of projects and determine factors that affect it.

  The main research requirement is data on net faults removed and failures experienced. If factors that affect fault reduction factor are identified, we need to determine their values. There is no need for software tools.
• **Resource Usage Parameters:** - Information on resource usage parameters is needed on a wide variety of projects. Either they will be constant, or they will vary with factors which must be determined. The requirements for research here are data on resource usage (failure identification effort, failure resolution effort, computer time) as a function of execution time and failures experienced. Data will also be required on the values of any variables that may affect resource usage.

• **SRE and Unit Test:** - there is a good chance that software reliability estimation could be extended to unit test. There are problem that must be addressed. First, the size of the sample of failures may be solved in grouping the failures of a number of units in some way. Second, the operational profile for the unit must be related to the system operational profile in some way or one must compensate for the difference.

• **Homogeneity of Failure Severity Classification:** - Some evidence indicates that the proportion of failures in each failure classification on a given project remains approximately constant over the life of the project. Checking this hypothesis will require failure data from a variety of projects, with the execution time and severity classification of failure recorded.

• **Relationship between Reliability and Problems Found During Inspection:** - Reliability estimation and prediction during the coding phase or earlier is an open research question. Some suggestions can however put forth. All of these suggestions will not guarantee good results if followed. These are only recommendations based upon the experience of software developers.

Based on the findings of the use cases, suggestions for improvements on the software reliability definition and ideas for a model for OFBiz reliability, further research and development can be performed. It is really hard to create a unified software reliability model for different open source software development paradigms. However it is possible to improve current approaches for evaluating OFBiz reliability, providing more and accurate options for evaluating a wide range of OFBiz products.