CHAPTER 1
INTRODUCTION

1.1 Background

Software reliability is one of the major software quality attributes, which quantitatively expresses the continuity of correct service delivery. Reliability models are typically measurement-based models, and mostly employed in isolation at the later stage of the software development process. However, the current requirement of software reliability calls for highly efficient techniques that have multi-dimensional potential to address various ranges of dynamic issues in software development. Therefore, the existing software reliability models are highly insufficient for the purpose of designing prediction frameworks as well as for analysis [1]. Although, there has been abundant literatures that has addressed the issues of software reliability, but still there is an emergent need for highly optimal models especially in the area of component based development and classification frameworks. Targeting to address such issues, the proposed study contributes a novel reliability technique that leverages reliability analysis in early stages of software development by taking into account (1) the complexities of component structure surfaced in the requirements phase and (2) the concurrent nature of component-based systems. The study also introduces a mean to accomplish reliability design and analysis for model driven engineering following the Model Driven Architecture standards. Hence, the proposed thesis contributes to the task of systematically integrating reliability modeling from the early to the late stages of software engineering and thus semantically integrating analysis, design and deployment models for reliability into one uniform environment.

Open Source refers to any software or program whose source code is made available to other developers for use, modification and enhancement. Open Source does not mean only access to the source code, the distribution terms of Open Source Software must comply with several criteria related to distribution, derived works, integrity of the author’s source code, distribution of license, and license terms among others. The Open Source business model relies on shifting the revenue stream from the product itself to the product auxiliaries such as support, training, documentation, etc.
Companies are increasingly looking for flexibility, cost savings and efficiency in the business applications they acquire. The increased competition has driven companies to look for new ways to succeed and remain competitive. In this IT-led business environment, staying on the cutting edge of new technology acts as an enabler for companies to gain effectiveness and efficiency to face the competition. Enterprise Resource Planning (ERP) [2] is one of those technologies that are at the heart of the IT strategy of a company. Flexible ERPs bring adaptable processes that support diversification strategy and ensure quick responses to the continuously changing business needs. In addition Open Source ERP systems bring cost savings compared to proprietary ERP systems and this makes them to be more popular and increasingly solicited.

Proprietary ERP systems [2] such as SAP and Oracle have long been leading the ERP industry, but new competitors have emerged and they seem to offer similar capabilities as their more established competitors and most importantly they are Open Source. Major proprietary ERP vendors include: SAP¹, Oracle Applications², Microsoft Dynamics³, the SAGE group⁴, PeopleSoft⁵ and many more.

Early 2000s saw a major software vendor consolidation with PeopleSoft merging with J.D. Edwards in 2003, then Oracle acquiring PeopleSoft in December 2004. Infor Global Solutions acquired SSA Global Technologies in 2006 which had acquired Baan in 2003. SAP and Oracle Applications alone control about 40% of the ERP software market. In the Open Source ERP software market major competitors are: Adempiere, Compiere, ERP5, GNU Enterprise, OpenERP, Openbravo, Opentaps, WebERP and BlueERP.

Open Source Software adoption in large companies is considered to be a relatively recent movement. Open Source Software is gaining terrain in large organizations; some see it as just another development alternative, others see in it a strategic competitive advantage.

¹ http://www.sap.com/pc/bp/erp.html
³ http://www.microsoftbusinesshub.com/products/Dynamics_ERP
⁴ http://na.sage.com/us/erp
The use of Open Source has proved itself in companies of the developed world. A study carried out by Forrester research in September 2008 revealed that Open Source is used on average by 1 in every 5 companies in Europe and North America, and 22% of those companies have chosen a 100% Open Source model. The study revealed also that, in general, companies start by introducing Open Source through the use of an Apache web server or an Open Source operating system such as GNU/Linux. Then, development tools and programming languages such as Eclipse, PHP and Perl usually follow. The third phase is usually adopting Open Source solutions for database management, such as MySQL and Postgresql.

When it comes to application areas, the study revealed that the major areas of applications where Open Source solutions are used are office applications such as OpenOffice and StarOffice⁶ (used by 62% of the Open Source companies surveyed) and messaging such as Open-Xchange⁷ (used by 50% of the companies surveyed). Where is ERP from all this? The study showed that Open Source ERP is used by 38% of the companies surveyed. It is a well known fact that large companies still have a preference for proprietary ERPs to run and manage their business. This market of proprietary ERP has been dominated for a long time by major competitors such as Oracle, SAP, Microsoft, PeopleSoft and Sage. Those companies offer a variety of products. But these products are complex, bloated with features, and too expensive for most mid-sized companies. Also, many of the traditional mid-market software vendors have disappeared in the consolidation trend that this mature market has been witnessing for quite some time. Customers continue to pay maintenance fees to the new owners for proprietary products that receive little new investment, and moreover, they have very few options by not having access to the source code.

Evidence shows that the big vendor dominated market of ERP applications is also ready to start leaning towards nonproprietary technology. Flexibility, cost savings, vendor independence and efficiency have been driving companies away from proprietary technology to Open Source. Moreover, the mid-market ERP segment is not well served–small to mid-size enterprises (SMEs) simply cannot afford the expensive proprietary products that the aforementioned ERP giants put on the market.

⁶ http://www.openoffice.org/tools/releases/q-concept.html
⁷ http://www.open-xchange.com/home
SMEs are companies whose headcount or turnover falls below certain limits. This abbreviation is commonly used in the European Union and international organizations such as the World Trade Organization, the World Bank and the United Nations. The European Union definition of an SME is “The category of micro, small and medium-sized enterprises (SMEs) is made up of enterprises which employ fewer than 250 persons and which have an annual turnover not exceeding 50 million euro, and/or an annual balance sheet total not exceeding 43 million euro [142].” Traditionally, ERP systems were reserved to large organizations. SMEs could not afford or have access to them and they were obliged somehow to content themselves with some modest accounting and sales management software, and not with a real ERP capable of adapting to the management and production processes specific to each company. It became clear that there is room for Open Source ERP systems and Open Source ERP competitors. In recent years, open source ERP software companies, such as Compiere, Openbravo, and Open for Business, with better product features and with radically different business models than those of proprietary ERP vendors, have emerged with hopes of changing the ERP software industry.

1.2 Problem Description

In spite of those interests and efforts, techniques available to validate a design against nonfunctional properties, particularly reliability, often require significant expertise unrelated to the usual business of engineering software. As reliability measures quantitatively the quality of correct service delivery, it is probably the most important characteristic for the software engineering discipline. However, as pointed out by Michael R. Lyu [3], Software Reliability Engineering (SRE) is not yet fully delivering its promise. Major reasons behind it are that reliability models are typically measurement-based models, and therefore employed in isolation at the later stage of the software development process. On the other hand, early software reliability prediction models are often insufficiently formal to be analyzable and not usually connected to the target system. Therefore, as mentioned by Lyu [3]:

There is currently a need for a creditable end-to-end software reliability model that can be directly linked to reliability prediction from the very beginning (i.e. software design), so as to establish a systematic SRE procedure that can be certified, generalized and refined.
In the pursuit of modeling software reliability of component-based development, there are some key questions that need to be addressed during the design of the system architecture, such as:

- Identifying the reliability each component needs to have in order to meet the intended system reliability. Likewise, identifying components more significantly impacts on the system reliability. Usually, software analysts have to exploit complex analytical functions describing the system behavior in order to find out how sensitive the system reliability is to the components’ failure probability.
- Stakeholders want to know the effect that different system features and execution scenarios have on system reliability. Therefore, it may be desirable to know if or how different designs affect the system reliability before the decisions are made and costs committed.
- If one wants to investigate component-based software reliability from designs, it is also plausible to investigate through software analysis if the concurrent nature of the software reveals intended or unintended component interactions and if those interactions affect the system reliability.

However, the means to answer those questions are few and inadequate. The vast majority of techniques for reliability prediction are tailored to be used during the late phase and not during the requirements elicitation or design phases of the software lifecycle.

Although model driven approach promises to overcome important unsolved problems in software engineering, it has not specified ways to comprehensively represent software reliability yet. The object management group in the QoS profile also denotes the QoS characteristics. The support for quantification of reliability in terms of the ability of a system to keep operating correctly over a period of time is provided by QoS dimensions. However, it is believed that those properties are not comprehensive enough to support reliability engineering, in particular analysis. The prime reason behind this is that it does not address the modeling of dynamic aspects of a system, such as scenarios, component interactions or information regarding transitions between states of a modeled system, often required in modeling and analyzing
component-based software systems. A system consists of a set of interacting components such that the interactions can reveal faults [4]. The modeling and annotating those interactions appropriately can assist us in predicting software reliability. On the other hand, dynamic aspects have been defined in the UML Profile for Schedulability, Performance and Time Specification [5] (henceforth referred to as the SPT Profile), but they were not incorporated into the QoS Profile. In fact, this gap may result from lack of adequate techniques to predict reliability using simple modeling elements, e.g. Message Sequence Charts (MSCs) to describe system requirements.

1.3 Motivation

The SRE is a part of the software engineering that concentrates on the quality property named reliability. It implies all the steps from the development process in order to identify the demand regarding the reliability and in order to control if one can find them in the final product. The operational capacity of a functional system represents its capacity to fulfill certain operational demands, in a period, in certain conditions. Software reliability can be defined as the running probability without failure of a software system, for a certain period and in a certain environment. Another way to express reliability is to consider reliability as the mean time between the defects. As software products have become increasingly complex, software reliability is a growing concern, which is defined as the probability of failure free operation of a computer program in a specified environment for a specified period of time [6-7]. Reliability growth modeling has been one approach to address software reliability concern, which dates back to early 1970’s [8-10].

Reliability modeling enables the measurement and prediction of software behaviors such as Mean Time to Failure, future product reliability, testing period, and planning for product release time. Software reliability growth models generally fall into two major classes: time between failures and fault count models. The main input parameter to the “time between failures” models is the intervals of successful operations. As the failures occur and are fixed, it is expected that these intervals have to be increased. The pattern of these intervals is reflected by a probability distribution model whose parameters are estimated from simulation, testing, and operation
profiles. Open source ERP systems are often targeted to enterprises whose requirements are not covered by standard software. Similar circumstances apply to organizations that need continuous adaption of the software to changing processes and needs. In this work, the suitability of current open source ERP systems for these enterprises will be examined. It provides sufficient information for a small or medium enterprise to choose a flexible and adaptable open source ERP system. Therefore adoption of open source ERP system pave better opportunities for any organization with better process management and flexibility in systems. Besides the focus on flexibility, open source specific criteria for support, continuity and maturity are worked out. Then selected open source ERP projects are reviewed and classified according to these criteria. The results are a criteria catalog and a classification of selected open source ERP systems.

1.4 Research Gap

There is currently a need for a creditable end-to-end software reliability model that can be directly linked to reliability prediction from the very beginning (i.e. software design), so as to establish a systematic SRE procedure that can be certified, generalized and refined. The cost of correcting a software error generally increases by magnitudes for every phase of the life cycle. Ideally most of the errors are detected by the end of the unit testing phase. This can be associated with the burn-in phase for hardware. Ideally, the errors found during the integration phase are those due to interfaces that could not have been easily or possibly found during previous phases. Ideally the number of errors detected levels off by the acceptance test phase. If it is known what the average cost of fixing a bug is during each phase of the life cycle, it can be estimated what the cost of repair is and also what it could be. If the average cost is not known, then the relative cost may be found by comparing real errors detected over time against the ideal. Although, there has been extensive research work being conducted in the area of software reliability, but there is far less work being focused on open source software, especially of ERP type, which for the current business has high demands. The Majority of the prior research work is either a mathematical model using simulation or not real time. Hence, this poses as high research gap in the area of open source ERP applications in terms of software reliability.
1.5 Research Objectives

Based on the research gap, the following objectives are framed for research work:

1. To understand various types of open source software and their current commercial or research usage scenario from global perspective.
2. To analyze open source software reliability using the OFBiz ERP package written in the Java programming language.
3. To perform extensive testing of OFBiz and extract the errors formulated during real-time usage of this application.
4. To design an application using Java on the Enterprise edition (J2EE) platform for extracting all the error data or bugs after usage of OFBiz.
5. To propose and design an optimal software reliability model based on an evolutionary algorithm and the Monte Carlo method by deploying real time data.
6. To perform an analysis for evaluating the efficiency of the proposed mathematical model.

1.6 Methodology:

This part describes the methodology used to assess the chosen Open Source ERP systems. The intention from the beginning was to carry out the evaluation in the most objective way possible. The study is a mixture of theoretical and empirical research. The theoretical research focuses on carrying out a comprehensive review of relevant research work to be able to build the model that will be used as a basis for the empirical study i.e. the evaluation of the chosen Open Source ERP systems. The model includes the evaluation criteria to be referred to when considering or assessing an Open Source ERP package to be used by an SME or a large organization. SMEs and large organizations may have different as well as common needs from an ERP. The reviewed literature focuses mainly on discussing ERP systems selection criteria of SMEs and large organizations. The literature review aims to put forth a list of “dimensions” which represent one of the two components of the evaluation model. The other component is the set of features which were identified by looking at the feature offerings of the ERP systems. Once the theoretical study is completed, the model for evaluating the Open Source ERP systems is built based on the “dimensions” and “features” identified through the literature study and the study of
the ERP systems themselves. The model serves as the guiding principle when examining the ERP systems and collecting the empirical data. The evaluation of the systems based on the “dimensions” is performed in a qualitative way, and was fed by searching the documentation published on the vendors’ websites and also by evaluating the systems themselves after downloading them and installing them.

1.7 Scope of Thesis

The scope of this thesis is the design and implementation of a prototype for a Web-based application using the open-source approach. This thesis proposes specific open source software and methods that can be used to implement a Web-based application. The application will focus on personal information management for military organizations. The prototype will be an entire Web application system, including an operating system using Windows / Linux, a Web server using Apache Tomcat, a database server using MySQL, and server-side programming using Java Servlets and Java Server Pages. The prototype will manage simulated personal information in the Thai military organization.

1.8 Thesis Contribution

This research work starts from the client organization’s perspective with assumption that the open source ERP system is not completed software in early design stage. It is usually a product that has already been delivered to many customers in the past but some adaptation and configuration must be done for a specific client. The main problem is how to select an appropriate software reliability model with optimal parameter choice. In sections of the thesis, the fundamental definitions are reviewed for parameters which affect the reliability of software and define what types of defects
are reported in observed ERP system using OFBiz. All defects are tracked and categorized by the proposed scheme from users and vendor in a separate repository.

Measurement-based analysis represents a useful foundation for the current research work in modeling of open source ERP system reliability. This research work furnishes an example of how to use existing user-generated ERP software defect reports in choosing an appropriate software reliability model in reliability modeling and later in reliability forecasting. It is important to emphasize that this work has analyzed data for only one installation of described ERP product (OFBiz), collected during system usage after software acquisition. A relevant approach for the situation when user wants to validate product maintenance and predict the future trends during product usage. Another important point is the analysis of a maintenance log from the moment when the product was in full production so the number of reported defects is relatively small.

In this thesis, different software reliability metrics, related models based on these metrics and different open source software paradigms are analyzed targeting to accomplish reliability in open source software. All these themes are developed in order to determine whether an open source software application being developed is ready for use and how it can be used alongside with other ERP related open source software packages. A power exponential model is introduced that will be used to evaluate the software reliability of ERP package ie of OFBiz.
1.9 Thesis Summary

**Reliability Study and Analysis of Opensource Enterprise Resource Planning Software Package**

**Chapter 1**
Introduction
- Problems
- Research Gap
- Objectives

**Chapter 2**
Literature Survey
Review on Software Reliability
Past Research Work

**Chapter 3**
Software Reliability & Open Source
- SR-Models
- OFBiz

**Chapter 4**
Research Methodology
- Research Design
- Proposed Algorithm
- Proposed Schema of Software Reliability
- Proposed Model

**Chapter 5**
Data Analysis
Comparative Analysis

**Conclusion**
- Conclusion
- Recommendation
- Limitations

Fig. 1.2: Thesis Summary