CHAPTER 2

LITERATURE SURVEY

2.1 INTRODUCTION

Software reliability Engineering is a well developed discipline that can help organizations improve the reliability of their products and processes. As the cost of software application failures grows and as these failures increasingly impact business performance, software reliability will become progressively more important. Employing effective software reliability engineering techniques to improve product and process reliability would be the industry’s best interests as well as major challenges (Rattikorn Hewetti et al 2006). Koji Ohishi et al (2005) implements Gompertz non Homogeneous model with well-known statistical theory of extreme value and the quite similar asymptotic property to the classical Gompertz curve. An S-shaped reliability growth curve is used to describe a reliability growth trend with a lower rate of debugging and growth at the early stage and a higher rate later on better fixes are implemented (Dimitri Kececioglu et al 1994). They have also presented the modified Gompertz model with four parameters for fitting data using S-Shaped model.

Software reliability is affected by many factors such as general factors, stability factors, compliance of process, development of document, coding test cases, human resource factor etc. Hong-Li Wu (2010) calculated number of defects by using number of software code line, software process capability indicator. A software reliability early prediction model is proposed to make use of project measurement history companying with the benchmarking. Kai-Yuan Cai (1997) developed censored software reliability
models to account for non-failure stops and directly dealt with censored data of software reliability. He observed that censored software reliability models reduce to non–censored software reliability models if no non-failure stop occurs. Mei - Hwa Chen et al (2001) observed that the existing software reliability growth models often over – estimate the reliability of a given program, this over estimations exist because the models do not account for the nature of the testing. Every testing technique has a limit to its ability to reveal faults in a given system. Thus as testing continues in its region of saturation, no more faults are discovered and in accurate reliability growth phenomena are predicted from the models. They have used both time and code coverage measures for the prediction of software failures.

Shinji Inoue and Yamada (2007) discussed a unified framework for discrete software reliability growth modeling in which the software failure- occurrence times follow discrete – time probability distribution. They have practiced maximum likelihood parameter estimation to predict software reliability. Daisuke Satoh and Shigeru Yamada (2001) proposed a criterion to determine a more appropriate software reliability growth model in the early development phase. The values of the proposed criterion of a discrete software reliability growth model are smaller to that of another software reliability growth model in all periods during the test phase with actual data set. They concluded that the models yielded accurate parameter estimates in spite of a small amount of input data in an actual software testing.

Software reliability is also concerned with how well the software functions to meet customer requirements. Reliability becomes an integral part of the requirement development process rather than an afterthought of testing at the end of the development process (John Bowels 2006). Hence the work justified that the requirements analysis is also a part of reliability. A software reliability modeling frame work has been proposed, based on Markov renewal
Approach. It is observed that the software reliability depends on the software runs. Jack M. Davis (1997), also contributed towards realistic modeling of software reliability and integrated the phenomena of failure correlation.

Li Wu et al (2010) proposed an idea that to predict the software reliability making use of the similar projects measurement data based on software process benchmark measurement and software process data before software test. Harald Stieber (2007) described a family of SRGM’s which depends continuously on a parameter. All of the models of this family allow the prediction of $\lambda_0$, the failure rate at the beginning of the test and $N_0$, the overall number of faults at the beginning of the test. A new model was developed by the author which belongs to J. Musa’s model family and described how this family of models can be used for feedback control of the software test process.

Shaik Mohammad Rafi and Shaheda Akthar (2010) have made an analysis on software growth model with Gompertz test effect function. They concluded that the Gompertz TEF in SRM can fit for any kind of software failure data. By incorporating both TEF and test efficiency one can reduce the total testing cost and increase the reliability.

Generally a testing-effort is better described by number of persons involved, number of test cases used and calendar time. Software reliability growth model incorporates the Gompertz testing-effort function and an analysis is made on optimal release (bev Littlewood 1979). They have examined the analysis with Yamadha delayed S-Shaped model and Gompertz testing—effort function and observed that Gompertz curve reaches its peak value very quickly. It is also observed that the Gompertz model gave the realistic value in software. In that way the total testing cost can be reduced with increased reliability.
Andy Ozment (2005), the author has used the growth model to examine vulnerabilities. Three of the seven reliability models tested were found to have acceptable one step ahead predictive accuracy for the set of independent data points. They have concluded that these estimates could serve as both a useful relative and absolute measure of the security of the product.

Ohishi K et al (2008) proposed a new approach to estimate parameters and concluded that the Gompertz SRM has a stronger non-linearity than the existing NHPP based SRM, the computational procedure of maximum likelihood estimates is more complex also classical iterative algorithms are possibly used to solve the likelihood equations numerically. So it does not often function well to compute the parameters of Gompertz SRMs.

Mei-Hwa Chen and Michael Lyu (2001) suggested a new approach to test effectiveness and code coverage for the prediction of software failures and operation coverage information collected during testing is used only to consider the effective portion of the test data. Execution time between test cases, which neither increases code coverage nor causes a failure, is reduced by a parameterized factor. The parameters used in software reliability model play an important role in predicting the account reliability. Parameter estimation is achieved by applying techniques like Maximum Likelihood estimation and Least Mean Square estimation. Tholon et al (2006) discussed the Gompertz curve and analyzed the variance in parameters in the Gompertz would result well. The authors have concluded that Gompertz model had an excellent adjust and the predicted mean error.

Daisuke Satoh and Shigeru Yamada (2001) described a software reliability growth model that yields accurate parameter estimates even with a small amount of input data. The model is based on discrete analog of a Gompertz equation that has an exact solution. The difference equation tends to a differential equation on which the Gompertz curve model is defined, when
the time interval tends to zero. It is observed that the model provided accurate parameter estimates, making it possible to predict in the early test phase when software can be released.

Garg (2010) developed a deterministic quantitative model based on a Distance Based Approach (DBA) method and applied it for evaluation, optimal selection and ranking of SRGMs. DBA recognized the need for relative importance of criteria for a given application, without which inter-criterion comparison could not be accomplished. It requires a set of model selection criteria along with a set of SRGMs and their level of criteria for optimal selection. They have used two distinct real data sets for demonstration of the DBA method. The result of this study was the ranking of SRGMs based on the Euclidean composite distance of each alternative to the designed optimal SRGM.

Sy-Yen Kuo and Yi-Ping Chang (1997) investigated two optimal resource allocation (OPT/RA) problems in software module testing. The work is based on the hyper geometric distribution software reliability growth model. Experimental results shown that the OPT/RA method can improve the quality and reliability of the software system much more than the simple allocation methods. It is observed that OPT/RA method is very efficient for solving the testing resource allocation problem.

Yamada and Osaki (1985) summarized existing software reliability growth models designed by a non homogeneous Poisson Processes. The SRGM’s are discussed for software reliability data analysis and software reliability evaluation. Using actual software error data observed by software testing, application examples of the existing SRGM’s are illustrated.

Hemminger and Mei-Huei Tang (2007) developed a moving average non-homogeneous Poisson Process reliability model which includes the
benefits of the time domain and structure based approaches. This method overcomes the deficiency of existing NHPP techniques that fall short of addressing repair and internal system structures simultaneously. Several experiments on different system scenarios and circumstances are discussed, indicating the usefulness of the proposed approach.

A stochastic model G-O for the software failure phenomenon based on a NHPP was suggested by Goel and Okumoto. This model has been widely used but some important work remains undone on estimating the parameters. They have presented necessary and sufficient conditions for the likelihood estimates to be finite, positive and unique. A modification of the G-O model is also suggested. The performance measures and parametric inferences of the new model are discussed.

Spereij and Peter (1985) studied the problem of maximum likelihood estimation in the Jeliniski-Moranda software reliability model. The distribution of the stochastic variable that completely determines the maximum likelihood estimate is obtained. S-confidence intervals for the parameter of interest are constructed by using the same stochastic variable.

There are many software reliability models that are based on the times of occurrences of errors in the debugging of software. Joe (1989) shown that it is possible to do asymptotic likelihood inference for software reliability models based on order statistics or non homogeneous Poisson process, with asymptotic confidence levels for interval estimates of parameters.

Ohba, Mitusuru (1984) discussed improvements to conventional software reliability analysis models by making the assumptions on which they are based more realistic. In an actual project environment Sometimes no more information is available than reliability data obtained from a test report. The models described are designed to resolve the problems caused by this constraint
on the availability of reliability of data. By utilizing the technical knowledge about a program, a test and test data we can select an appropriate software reliability analysis model for the accurate quality assessment.

Ohba Mitusuru and Osaki, Shunji (1983) investigated a stochastic model for a software error detection process in which the growth curve of the number of detected software errors for the observed data is S-Shaped. The model was a non homogeneous Poisson process where the mean-value function has an S-Shaped growth curve. The model was applied to actual software error data. Statistical inference of the unknown parameters was discussed. It was observed that the proposed model fitted the observed data better than other models in the literature.

Many existing software models are based on the assumption of statistical independence among successive software failures. But in reality this assumption could be satisfied easily. Parameter estimation is an important issue that is crucial software reliability model Bo Yang et al (2008). They have proposed a relationship function among model parameter which could reduce number of parameters to be used in prediction. Two parameter estimation methods have been used and the simulation results show that the Least Mean Square estimation would give better result than the Maximum Likelihood Estimation (MLE). Changyou Zheng et al (2011) proposed Ant Colony algorithm to estimate SRM parameters.

Several combinational methods of SRGMs have been proposed to improve the reliability estimation and prediction accurately. In Bo Yang et al (2008), a new approach namely AdaBoosting algorithm has proposed, comparing individual SRGMs, and ACM on the basis of time complexity. Gompertz model can be practically used for predicting life expectancy. Adam Lenart, (2010) proposed that Gompertz propositional hazard models provide
scenarios for morality reduction in accordance with the proportionality assumption of Bongaarts and fiency.

Chatterjee et al (2009), the authors developed a new procedure for estimating software reliability based on Bayesian sequential model and illustrated the new method with some system data. They concluded that this proposed method was computationally very simple and efficient for software engineers. Ann Marie Neufelder (2006) developed a multi-parameter method to predict defects long before the testing begins. They have used this method with new software technology and data from a variety of industries and application types. This method is easy to estimate the project cost and schedule. How a software engineering organization can predict defects and manage them before even writing a single line of code is discussed.

The concept of S-shaped model came into being in the early eighties of the last century, where S-shaped model was considered a novel mechanism for predicting and solving software reliability issue. This model differs from the Musa’s basic model in that it reflects the view that the earlier discovered failures have a greater impact on reducing the failures intensity function than in other models. Modeling scheme based on the concept of Gompertz curve has been utilized commonly in Japan based software companies to estimate the number of residual faults in testing phase of software development (Khaled Faqih 2009).

OhiShi et al (1991) concluded that the Gompertz software growth model was rather attractive comparing with the existing growth models, to improve the performance of software reliability models, since they have been under continuous fire from people everywhere as they are unable to deliver the right solutions for achieving error-free software system.
Rattikorn Hewett et al (2006) proposed an effective use of reliability models and defect data to help managers to make decision about software release by applying a strategy for selecting a suitable model, which best fitted the customer defect data as testing progresses. They also experimented the usage of G-O model, Delayed S-Shaped model and Gompertz model with customer defect data and concluded that the Gompertz model would give stable result. The Gompertz curve has been used to analyze the noise in an avascular tumor growth. This model was used to predict the growth of tumor cells. They have concluded that the Gompertz model could give accurate prediction result.

Chien – Chia Chen et al (2006) authors presented a computer aided reliability assessment tools for software based on object oriented analysis and design. It can be used as a analysis on both SRGMs and neural network methods to assess software reliability. The authors concluded that this tool can analyze the software reliability easily due to its object oriented design.

Vladimir Zeljkovic et al (2011) calculated reliability based on the failure data collected during the maintenance phase. It also showed the importance of reliability estimation during the development/ testing, thus providing the product readiness to be delivered to the customer as well as during the operational phase for validating the product reliability.

Harald Stieber (2007) observed that Gompertz models give approximates based on the failure data collected during the maintenance phase, that match the real data. It is also shown the importance of reliability estimation during the development / testing, the Gompertz will play an important role in reliability estimation and also a new model was developed and tested.

2.2 PROBLEM DEFINITION
In literature, there was no method proposed to estimate the parameter in Gompertz reliability model so far. Most of the works concentrated on estimating the parameter using Maximum likelihood or Least Mean Square estimation with only number of failure and time interval as input data. Further the literature review showed that the reliability prediction has been calculated using various model like Okumot model, Yamadha model etc. None of the work has been done on the specified parameter estimation method with Gompertz model. The main aim of this work is to proposed an improved method for estimation of parameters in Gompertz software reliability model. We considered the number of failures, time interval and number of test cases for the specified module as the input data. We propose to apply the model on a large diversity of projects’ testing data. We expect that the predicting capability of Gompertz model with proposed approach of parameter estimation would bring out the efficiency of this model over other models. We apply the proposed parameter estimation method on two parameter models such as G-O model and Delayed S-Shaped model and compare the result with existing LMS parameter estimation method.

The implementation will be done on 50 real time projects which are collected from heterogeneous group of companies by using dot Net.

2.3 SUMMARY

A detailed survey of the research work carried out in the area of software reliability model is elaborated. The format definition of the proposed work is given with possible directions. The next chapter deals with the Gompertz software reliability model.