CHAPTER 3

GOMPertz SOFTWARE RELIABILITY MODEL

3.1 INTRODUCTION

For about 25 years, software reliability modeling has been an active process arena in the software engineering environment and an attractive area for technical publishers. The professional has identified more than seven hundred models all that can generally be classified as a “Software Reliability” model. In the past few years a number of different software reliability models have been introduced to find the reliability of a software product. Selection of a particular model is a challenging problem for software reliability prediction. The selection of release time and the value of resource allocation decision are the two main reason (Sulthan Aljahdali et al 2003).

The model will be selected for implementation based on some criteria, namely;

- **Estimation Validity**: The ability to estimate the future state of the software reliability based on the current data available for use in the model.

- **Quality of Assumption**: How logical the method assumptions are in comparison to the software engineering environment and the likelihood that the model assumptions can be satisfied in the collection of data and for application of the model.
- **Parameter Estimation:** What method be used to estimate the parameters of the model and how efficient it is and number of parameters used in the model?

- **Simplicity:** The implementation of the model in computer language like C, java and ability to technically and inexpensively collect the data to use the model.

The most common approach to develop software reliability model is the probabilistic approach. The probabilistic model represents the failure occurrences and the fault removals as probabilistic events. There are numerous software reliability models available for use according to probabilistic assumption. They are classified into various groups including error seeding models, failure rate models, curve fitting models, reliability growth models, Markov structure model and non homogenous Poisson processes models. The NHPP-based models are the most important models because of their simplicity, convenience and compatibility (Khaled Faqihl 2009).

A common used approach for measuring software reliability via and analytical model whose parameters are generally estimated from available data on software failures. There are number of views as: what software reliability is and how it should be quantified. Some people believe that this measure should be binary in nature, others feel that software reliability should be defined as the relative frequency of the time that the program works as intended by the user (Jack Davis 1997).

A number of analytical models have been proposed to address the problem of software reliability measurement. These approaches are mainly based on the failure history of the software, among all the models Gompertz model is a superior in estimating validity and stability (Koji Ohishi et al 2008).
and Sulthan Aljahdali et al 2003). The Gompertz model is widely used in IT companies which are having Japan based clients.

3.2 MODEL DESCRIPTION

The Gompertz model is introduced by Gompertz. This model is used to assess the reliability growth. The Gompertz model is of particular value because it is robust without many substantial limitations caused by assumption. In that model is that the testing/debugging effort throughout the test phase effort and that fixes are not accumulated in batches before being implemented. A good test plan is helpful, if not absolutely required, in achieving homogeneous test and debug effort. Gompertz model is also one of the S-Shaped software reliability growth model. This model will give good approximation to a cumulative number of software faults observed in testing software. The Gompertz model is an Non Homogeneous Poisson Process (NHPP) model. It takes the number of faults per unit of time as independent Poisson random variables. The model was first proposed in 1979 by Amrit Goel and Kazuokumoto and has formed the basis for the models using the observed number of faults per unit time group. Recently an enhanced model of Gompertz known as discrete Gompertz curve by discretizing the usual Gompertz equation is proposed and applies it to predict the number of defected software faults. Since the Gompertz curve is deterministic curve, one cannot access quantitatively the software reliability and its related software dependability measures. To over come this, the NHPP model with the mean value function with Gompertz curve has been used in our work. In addition, the most severe problem is that the method of maximum likelihood cannot be easily applied to estimate the parameter in this model, because of its strong non-linearity. An alternative way for the development of an NHPP model based on the Gompertz curve is to allow a discontinuous mean value function (Dimitri Kececioglu et al 1994).
The Gompertz Model is of particular value because it is robust without many substantial limitations caused by assumptions. The only substantial assumption for the Gompertz model is that the testing/debugging effort is homogeneous throughout the entire test phase effort and that fixes are not accumulated in batches before being implemented. This homogeneous effort entails consistent application of test and debugging resources, methods and procedures (Jack Davis 1997).

The Gompertz model is of additional usefulness because of the robustness of the type of information that can be modeled. This model classifies into two classes;

i. time between failure models

ii. Failure count model.

The Gompertz model can successfully fit the following testing period series;

- Reliability as a probability (% or decimal)
- Mean time to failure
- Cumulative failure count

The basic assumptions which are followed during the implementation of this model are;

- The rate of fault detection is proportional to the current fault content of the software.
- The fault detection rate remains constant over the intervals between fault occurrences.
A fault is corrected instantaneously without introducing new faults into the software.

The software is operated in a similar manner as that in which reliability predictions are to be made.

Every fault has the same chance of being encountered within a severity class as any other fault in that class.

The failures, when the faults are detected are independent.

The cumulative number of failures by time \( t \)

The Gompertz software reliability model can be expressed by:

\[
\mu(t) = ab^c
\]  

\text{(3.1)}

where \( \mu(t) \) is the software reliability at time \( t \) and is dimensionless percent. ‘a’ is a constant that provides an upper bound to \( \mu(t) \), ‘b’ is a constant and ‘c’ is also a Gompertz constant, which provides a shape parameter to the Gompertz model equation. A relatively small value of ‘c’ promulgates rapid early reliability growth while large values of ‘c’ indicate slower reliability. The approach to estimate values of ‘a’, ‘b’ and ‘c’ is discussed in chapter 4.

In recent years, researchers have proposed several software reliability models to estimate the number of faults during the software testing process. These parametric models have a set of unknown parameters. These parameters must be estimated using observed historic failure data. For example, Least Mean Square, Maximum Likelihood, Instrumental Variable method are few methods used to solve the parameter estimation problem.

The first requirement to use any model is to collect input data. The software testing effort is composed of a testing and debugging activity. The test period usually comprises time, which is classified as; execution time, clock
time and calendar time. Execution time is called CPU time. Clock time is defined as the time that the software program is executed on the computer. Calendar time is just straight elapsed time from beginning to end of the period. Calendar time is mostly used by the entire program manager to measure and manage the programmatic schedule. In our work we have considered calendar time to calculate the reliability of the software. Second step is to estimate the parameters used in this model equation. There are two popular parameter estimation methods exist in reliability world. The method of parameter estimation has been discussed in chapter 4. The third step is to calculate the reliability of each test period by Gompertz equation by using values of parameters. The model is able to calculate a list of statistical features describing the Gompertz equation as related to the input reliability data.

In this work the researcher has applied a new approach for parameter estimation method in G-O model and Delayed S-Shaped model and observed the results. The Gompertz model is a three parameter model and the remaining two models are two parameter model. The parameter estimation method could give better result for all types of software reliability models.

3.3 SUMMARY

Selection of particular model is a challenging problem for software reliability prediction. The criteria for selecting a software reliability models are analysed. A detailed description of Gompertz model is given. The next chapter deals with the various parameter estimation methods and the proposed Least Mean Square estimation.