CHAPTER 2
LITERATURE SURVEY AND PROPOSED STUDY

This chapter presents a detailed survey of the literature that lead to carry out this research work grouped under the topics such as, Software reliability and their growth models, regression, Statistical Process Control and Sequential Probability Ratio Test (SPRT). This survey helped me to trace out some research problems for further investigation that formed my research work.

2.1 BASIC TERMS AND CONCEPTS

Software Reliability Modelling plays an important role in developing software systems and enhancing computer software. Software reliability theory deals with probabilistic methods applied to the analysis of random occurrence of failures in a given software system.

Software reliability is defined as the probability that a software system operates with no failure occurring for a specified time on specified operating conditions.

A failure is an observed departure of the external result of software operation from software requirements or user expectations. When a system in operation does not deliver its intended functionality and quality, it is said to fail.

A fault is a defective, missing, or extra instruction, which is the cause of one or more actual or potential failures. Faults that are introduced through fault correction, or design changes, form a separate class of modification faults.

A metric that is commonly used to describe software reliability is failure intensity. It is a good measure for reflecting the user perspective of software quality. When reliability growth is being experienced, failure intensity will decrease over time.

If let \( N(t) \) denote the overall number of failures of a software system by time \( t \), then the counting process \( \{ N(t); t \geq 0 \} \) can be seen as a Non-homogeneous Poisson Process. In the existing stochastic software reliability models, a class is constructed based on the theory of NHPP. For a software
reliability growth model, the mean value function \( m(t) \), representing the expectation of the number of failures by time \( t \) as estimated by the model, is given by \( m(t) = E[N(t)] \).

2.2 APPLICATION OF CONTROL CHARTS IN SOFTWARE ENGINEERING

2.2.1 Utilization of SPC in Software

The interest to apply SPC techniques in the software industry has been growing during the last decade as more organizations advance in maturity levels of process improvement models such as Capability Maturity Model (CMM) (Paulk et al., 1993), Capability Maturity Model Integration (CMMI, 2001) and SPICE (ISO/IEC TR 15504-5, 1998). These models implicitly direct software companies to implement SPC as a crucial step for achieving higher process maturity levels. They suggest control charts for both project level process control and organizational level process improvement purposes. In the literature, there are several resources on the usage of statistical techniques in software development (Card, 1994; Kan, 1995; Lantzy, 1992). Some researchers contribute to this trend by providing approaches to utilize SPC techniques for software industry. Moreover, most of the examples exhibited in the studies refer to defect and inspection data (Paulk and Chrissis, 2002).

2.2.2 Advantages of SPC

SPC is a simple, effective approach to problem solving and process improvement. Training in the use of the basic tools should be available for everyone in the organization.

The SPC approach, correctly introduced, will lead to decisions based on facts, an increase in quality awareness at all levels, a systematic approach to problem solving, release of valuable experience, and all round improvements.

Where SPC is used properly, quality costs are lower; low usage is associated with lack of knowledge of variation and its importance, especially in senior management.
Successful users of SPC have, typically, committed knowledgeable senior management, people involvement and understanding, training followed by clear management systems, a systematic approach to SPC introduction, and a dedicated well-informed internal champion.

The benefits of SPC include: improved and continued reputation for consistent quality products/service, healthy market share or improved efficiency/effectiveness, and reduction in failure costs (internal and external) and appraisal costs.

A stepwise approach to SPC implementation should include the phases: review management systems, review requirements/design specification, emphasize the need for process understanding and control, plan for education and training (with follow-up), tackle one process or problem at a time, record detailed observed data, measure process capabilities and make use of data on the process.

A good management system provides a foundation for successful application of SPC techniques. These together will bring a much better understanding of the nature and causes of process variation to deliver improved performance (Okland, 2007).

Statistical process control appears to have several major strengths in software development activities. First and foremost, SPC provides a formal framework for doing the right thing. Many bitter experiences have taught the software industry that standards are needed early in the development process; that design walkthroughs, code walkthroughs, and unit test activities are cost-effective; and that the cost to correct a defect found late in the development process may be one hundred times or more than the cost to detect and correct the problem when the defect was born. Although many development efforts have intended to do it right the first time, attitude itself is not enough. SPC provides the formal tools needed to identify the ingredients for success and the tools to monitor process execution. Process monitoring has two major advantages compared to the detailed inspection of completed software units.

First, errors may be detected earlier or prevented altogether. With process monitoring, error causing situations e.g. inadequate standards, insufficient training, and incompatible hardware may be detected before defects are created
(Demmy and Petrini, 1989). Second, less effort may be required for successful applications to ensure that processes are operating discipline. They require correctly than is required to perform detailed checks on all the outputs of that process. Thus, higher quality may be achieved at a lower development expense.

Finally, statistical process control provides quantitative measures of progress and of problems. Less intuition is required; rather, measurements of errors and process characteristics quantify the scope of the problem and identify trends in process improvement.

2.2.3 Reliability and Control Charts

The control chart is considered as the formal beginning of the statistical quality control. Statistical process control (SPC) can be defined as a collection of tools, which track the statistical behavior of production processes, in order to maintain and improve product quality, thereby reliability. The ideology behind SPC is similar to that of other quality and reliability philosophies like Total Quality Management (TQM) and Six Sigma. Therefore, SPC is regarded as an important component of Total Quality Management (Cheng and Dawson, 1998) and other quality philosophies.

Of all the tools of Statistical Process Control, Control charts are, perhaps, most technically sophisticated. The basic idea behind any control chart is to monitor a process and to identify any unusual causes of variation from the chance causes of variation to assess reliability. The technique of control chart has been used in the software engineering so as to improve the quality of software products.

The Shewhart charts have been applied during each stage of the development of software system and used to monitor the number of defects per 1000 lines of codes (Scott, 1991).

Luke et al. (1993) managed to apply the tool of statistical quality control for maintenance by monitoring the response time, defect rate, fraction defective rate in the CMS-2 software project of a naval military fleet. A subgroup of response time consisted of five observations and subgroups were plotted on a Shewhart $\overline{X} - R$ chart with $3-\sigma$ control limits.
One of the recent works was done by Haworth (1996), which studied the usability of regression control charts for software maintenance. But it is questioned whether this regression technique is successful or not (Lewis, 1999).

Statistical quality control (SQC) technique has been used in software code inspection. A variation of the standard SQC chart (Christenson et al., 1990) is applied to monitor the software development process.

Gana and Huang (1997) stated that SQC has been used in the development of 5ESS-2000-switch project and gained substantial improvement of the product quality and reliability.

A work applying statistical quality control in measuring software reliability is by Stieber (1997). The metrics of Wald’s sequential probability ratio test developed by Wald (1945) is used, and the work assumes that software failure follows a homogenous Poisson process. This approach is used in the work to roughly evaluate the reliability of a software system, but the failure process is assumed to be a homogenous Poisson process, while most cases faced in reality are non-homogenous. Under this consideration, the point estimation technique is used to construct charts to monitor failure process.

2.3 REVIEW OF RELEVANT LITERATURE

Trivedi and Shooman (1975) suggested a many state Markov model for the purpose of estimation and prediction of software performance parameters. The availability and reliability of the software system are also presented for the purpose of predictions. Based on the notion of the distribution function of random variable Forman and Singpurwalla (1978) developed a probabilistic model describing the software failure phenomenon to suggest estimates of the parameters in the model and termination procedure for debugging the software.

Schick and Wolverton (1978) describe the most commonly used software reliability growth models as divided into two groups of time domain and data domain. They present a comparative picture of the model as applicable to various actual data sets. Goel and Okumoto (1979) are two of the foremost researchers that have considered the probabilistic nature of software failure phenomenon based on an NHPP. They have analyzed the failure process to develop a suitable
mean value function which in turn is used to get software performance measures like distribution of the cumulative number of software errors, the residual number of errors, software reliability, maximum likelihood estimators of the parameters of the model and joint confidence region of the parameters. The mean value function of this model turns out to be based on the cumulative distribution function of the well-known exponential distribution. Musa (1980) presented the need for potential use of software reliability measurement and made a comparison of software and hardware reliabilities.

Lai (1981) has shown us a large sample result for the SPRT, when the observations satisfy a “slowly changing sequence” condition. Goel (1982) presented an overview of the key modelling approaches provided a critical analysis of the underlying assumptions, and assessed the limitations and applicability of software reliability models during the software development life cycle. In Reckase (1983) and Spray (1993), they have done some numerical studies of the performance of the SPRT under tailored test set-up for both the mastery and the multiple category criterion-referenced tests.

Iannino et al. (1984) gave a descriptive narration of various criteria for the comparison of software reliability models based on predictive validity, capability, quality of assumptions, applicability, and simplicity. Kazuhira Okumoto (1985) proposed a statistical method that can be used to monitor, control and predict the quality of a software system being tested. Singpurwalla and Soyer (1985) suggested random coefficients auto regressive process to assess software reliability growth and applied their approach to a real life time data.

Humphrey (1989) can be regarded as a reflection of quality management on software engineering discipline. He describes a framework for software process management, outlines the actions to provide higher maturity levels and acts as a basic guide to improve processes in a software organization. In this book, Statistical Process Control appears as a means of data analysis technique for level 4 organizations. Demmy and Petrini (1989) shows how SPC technologies may be used to improve the quality and productivity of large-scale software development. They also described the steps required to apply SPC to software development activities and identified activities that appear to be particularly good SPC
candidates. They also summarized major advantages and disadvantages of the SPC approach to software development.

Fault density and failure intensity of an SRGM are used as two metrics to monitor software development capabilities and to measure customer satisfaction with the developed product in the research investigation of Huensch et al. (1990). Ehrlich et al. (1990) used the software reliability data collected during the testing of a system to measure the software quality in terms of experienced software failures. The notions of exponential NHPP are made use of for goodness of fit of the data. The time to repair, mean logistic delay time variants at the system level are modelled by Crow (1990).

Tohma et al. (1991) investigated six ways of the estimation of parameters in a hyper geometric distribution to get an estimate of the number of initial faults in a program at the beginning of its testing/debugging, along with their relative accuracies. Sofer and Miller (1991) use a nonparametric method of estimating the software failure rate in completely monotone models which can be compared with parametric approaches. Vallee and Ragot (1991) demonstrated the application of NHPP approach to the industrial world generating accurate predictions with specific applications in space research.

Lyu and Nikora (1991a) proposed linear combinations of software reliability models for the purpose of automating the procedures of software reliability analysis. The concept of equally weighted linear combination model resulting from linear combination of three popular software reliability models is suggested by Lyu and Nikora (1991a) for applying to software failure data sets. Tohma et al. (1991) made use of hyper geometric distribution to estimate by least squares sum method the number of faults initially present in a program at the beginning of test/debug process arriving at reduction in the time needed for calculating the estimates.

Jacoby and Tohma (1991) explained the idea of capture-recapture process for software faults in the context of a proposed testing environment. Estimation of the degree of unavailability of the software system is also given for hyper geometric distribution. Lantzy (1992) proposed SPC based solutions which can be applied, ranging from the creation of the informal environments of a garage shop
to the implementation of the formalized processes of software factories. He also mentioned that IBM used SPC methods in the development of OS/2 version 2.0. He stressed the need for adopting SPC techniques as increasing demand for software continues to out-pace software production capabilities in terms of both quantity and quality.

Khoshgoftaer et al. (1992) suggested an SRGM with the help of curve fitting techniques, predicting the number of faults in a system through fitting non-linear regression models to the number of faults in a program module. Scott A. Vander Wiel et al. (1992) explained the goal of algorithmic statistical process control to reduce predictable quality variations. Lantzy (1992) presents one of the earliest studies on the debate of applying Statistical Process Control to software processes. He summarizes the concept of SPC and gives some practical examples from manufacturing industry. Then he offers a set of transformations on these principles via software quality characteristics revealing the uniqueness of software products. After giving the process-product relationship, he outlines a seven-step guideline for successful SPC implementation in a software organization.

If testing and operation are the two phases of developed software, taking into considerations the notion of software reliability during testing phase an SRGM is suggested by Yamada et al. (1993). The concept of data aging is made use of to identify the starting failure count of a software failure data in an optimum way to choose an SRGM for failure data analysis by Schneidewind (1993). This model suggests four optimal criteria to determine the start of a failure count. Necessary and sufficient conditions for the existence and finiteness of the MLEs of the parameters of SRGMs are derived by Hossain and Dahiya (1993) and applied to various standard models. They presented a necessary and sufficient condition for the likelihood estimates to be finite, positive and unique and suggested a modification to Goel and Okumoto model.

Eric Martinus Mathiesen (1994) studied and presented that by combining accurate real time data collection, SPC methods, and a reliable simulation program, system engineers will be better able to realize real savings in monetary terms, increased operational availability and decreased mission time. As Card (1994) pointed out, SPC and other measurement based techniques are based on the assumption that the organization has defined processes. He discussed the
utilization of SPC for software by also considering some of the objections and mentioning possible implementation problems. He gave an example control chart for testing effectiveness measure and concludes that SPC principles can be beneficial for a software organization although formal statistical control techniques may not be used.

Ronald P Anjard (1995) presented a sampled method, a fool-proof, for selection of the appropriate SPC Chart. Kan (1995) provides detailed information about software metrics, software reliability models, and models and analysis of program complexity. Burr and Owen (1996) describe the statistical techniques currently available for managing and controlling the quality of software during specification, design, production and maintenance. The main focus is given to control charts as beneficial SPC tools and guidelines are provided for measurement, process improvement and process management within software domain.

Haworth (1996) investigates the use of regression control charts for software maintenance in a medium sized manufacturing firm. Software reliability prediction in a nonparametric scenario through Bayesian approach is studied by Aroui and Soler (1996). The optimal testing rule and hence the optimal release policy for a model based on hyper geometric distribution are investigated by Hou et al. (1996). Shima et al. (1997) examined the failure intensity distribution of an SRGM from an empirical failure data and show that the procedure predicts a number of detected faults more accurately than the conventional ones. Safety critical analysis applied in reliability modeling in order to increase the trust in the reliability of such products is the contribution of Schneidewind (1997) by integrating software safety criteria, risk analysis, reliability prediction and stopping rule for testing. Fenton and Pfleeger (1997) provide valuable information by considering the whole life cycle of measurement from planning to data analysis in a software organization. They also define the basics of some mostly used metrics and provide empirical information in software organizations.

Radice (1998) described SPC techniques constrained within software domain and gives a detailed tutorial by supporting his theoretical knowledge with practical experiences. He states that all SPC techniques may not be applicable for software processes and gives XmR and u charts as possible techniques. He also
explains the relevance of SPC for CMM Level 4 and regards back-off of control charts in Level 4 as a mistake. He states five problems with control charts: too much variation; unnecessary use of control charts; lack of enough data; lack of specification limits from the clients; the idea that control charts cannot be used with software processes. Carleton and Florac (1999) represent CMM understanding on the utilization of Statistical Process Control for software process improvement. The book includes detailed technical information about SPC and provides a roadmap for SPC implementation. It mostly focuses on the benefits of control charts depending on Shewhart’s principles.

Evanco (1999) used the notion of proportional hazard model in the form of a software development by identifying the three explanatory variables namely software complexity, software development volatility and cumulative execution time as variables influencing software reliability. He developed the regression approach to affect the reliability of software by these three variables. Relaxing the nature of statistical independence that is generally assumed among successive software failures, Goseva and Trivedi (2000) suggested software reliability modeling framework based on Markov renewal process. This modeling becomes a general case of many models that assume failure independence. Keiller and Mazzuchi (2000) investigated the improvement in the performance of Goel-Okumoto model with various smoothing techniques including Laplace trend test.

Weller (2000) provides a distinct study by presenting the SPC implementation details from a software organization. In order to regard defect density as an indicator of product quality, he first wants to be sure that inspection process is stable in the organization. He uses X and moving range charts for the lines of code inspected per hour for each inspection and achieves a stable inspection process after removing the outliers from the dataset. Then he draws u-chart for the defect density data for each inspection. By using these findings, he makes reliable estimations for inspection effectiveness and gains an insight on when to stop testing. This study carries the idea of SPC beyond theory and shows its benefits in providing control of process data and in creating a performance base for making predictions in real application environment.

Pham and Pham (2001) studied the predictive performance of a software reliability model inserting pseudo failures based on Bayes approach. Combination
of failure detection and fault correction in a single model is done by Schneidewind (2002) in order to have a prediction support. Homogenizing failure data with respect to failure intensities and adopting different SRGMs for different failure intensities, a combined SRGM for a better reliability assessment is suggested by Tian (2002). Reliability assessment of software divided into a finite number of models with the help of a Markovian model and that of statistical test of hypothesis is studied by Rajgopal and Mazumdar (2002). Keiller and Mazzuchi (2002) studied the performance of a set of SRGMs using smoothing techniques including Laplace trend test. Jakolte and Saxena(2002) move ahead on the idea of 3 sigma control limits and propose a model for the calculation of control limits to minimize the cost of type1 and type2 errors. The foundation of this study is a pioneering one as it questions an accepted practice for control charts and the results of the example studies are encouraging. Rahul Singh, Glenn Gilbreath (2002) described the design and implementation of a real time multi variable process control system that features a graphical user interface.

Irad Ben-Gal and Gail Morag (2003) introduced the context based SPC (CSPC) methodology for state-dependent data generated by a finite-memory source. Carina Anderson (2003) focuses on exploring the state of practice of the verification and validation process and presented methods for achieving efficient fault detection during the software development. Spiegelhalter et al. (2003) investigated the use of the risk-adjusted sequential probability ratio test in monitoring the cumulative occurrence of adverse clinical outcomes. Huang and Kuo (2003) describe a unified scheme of estimation in an SRGM using weighted arithmetic mean, weighted geometric mean and weighted harmonic mean. Their unified approach was verified to cover many of the well known NHPP models under different sets of conditions. Boland and Singh (2003) adopted a birth process approach to Moranda geometric software reliability model and derived the mean value function, intensity function, reliability function, probability distribution of relevant point process and optimum release time for the software. Combining the notions of fault removal efficiency, failure rate and fault introduction rate a unified model that integrates these three is proposed by Zhang et al. (2003).
The notions of faults dependency and time dependent delay are introduced by Huang et al. (2004) and described an SRGM with imperfect debugging. A unified mathematical modeling for explaining the imperfect debugging in a software fault process by dividing the original failure process of the model into two different NHPPs is introduced by Huang (2004). Groen et al. (2004) presented reliability data collection and analysis system – a software tool, in short- ReDCAS, employing Bayesian techniques to estimate reliability measures. The concept of infinite time to the next failure is the general consideration of many SRGMs indicating that the software product is very good. Mullen and Gokhale (2005) have considered a Poisson process with log normal distribution as the mean value function and presented goodness of fit tests for software failure data. The estimates of software failure rates adjusted in tune with the test data for the suitability of appropriate reliability models is considered by Jeske et al. (2005).

Cheong Wee Tat (2005) focused on the study of SPC techniques for high yield processes, and included some topics on high reliability systems. It deals with the statistical aspects of establishing SPC in high yield processing, and provide insight and promising opportunity for future research on high reliability systems. Danilo caivano (2005) proposed on SPC-based approach that reinterprets SPC, and applies it from a software process point of view. He focused on primary differences between manufacturing and software process characteristics. An efficient software reliability assessment model for the actual open source system development to compare between S-shaped and other models is investigated by Tamura and Yamada (2005). Crow (2005) gave a detailed note about the methods improving the effectiveness of the reliability tasks. The concept of multiple change points and imperfect debugging are integrated in a single model to study the software reliability by Huang and Lin (2005).

Diane Manlove & Steve Kan (2006) presented IBM system i software development, quality, and in-process metrics. Also challenges in applying SPC in software development and SPC application in system software development. Sargut and Demirors (2006) performed a case study of the application of SPC techniques using existing measurement data in an emergent software organization. They also analyzed defect density; rework percentage and inspection performance
inertias. And provided a practical insight as the usability of SPC for the selected metrics in the specific process and described their observations on the difficulties and the benefits of applying SPC to an emergent software organization. Mustsumi Komuro (2006) described experiences of applying SPC techniques to software development processes. They gave several real examples to apply SPC in Hitachi software. They also described characteristics of software development processes, their influence on SPC, and lessons learned when applying SPC to software process. In particular, they discussed the importance of self-directed and proactive improvements. Liu (2006) made study to disadvantages of Shewart attributes chart and Time Between Events Charts. And also presented the methods for improving performance of control charts and thus make the monitoring of high-quality processes more effective and economical.

Koutras et al. (2007) presented the basic principles and recent advances in the area of statistical process control charting with the aid of run rules. Statist Software, Inc (2007) defined the fundamental concepts of SPC and Control Charts that are useful for the construction of a chart.

Naikan (2008) discussed about the basic principles of development, design and application of various types of control charts. Boffoli et al. (2008) the experiences in using SPC in industrial contexts, points out the main issues concerning software process monitoring and highlights how the technique addresses them. Their main contribution was to formalize and put a set of guidelines together in a disciplined process for guiding practitioners in correctly using SPC during process monitoring.

Hairulliza Mohammad Judi et al. (2009) presented the implementation of quality control in three Malaysian companies and identified the factors that influence the selection of quality control techniques in these companies. And also identified that the initial role of SPC is to prevent product or process deterioration rather than identify product or process deterioration. SamulE. Buttrey, (2009) presented a free add-in for excel that draws the most common sorts of control charts.

Liu (2011) proposed a function based nonlinear least squares estimation method which extends the potential fitting functions of traditional least square
estimation in estimating parameters of Jelinski-morando reliability model. Kulldorff et al. (2011) proposed a maximized sequential probability ratio test (MaxSPRT) based on a composite alternative hypothesis, which works well across a range of relative risks. They illustrated the use of this method on vaccine safety surveillance and compare it with the classical SPRT.

2.4 PROPOSED STUDY

After thoroughly scanning the literature presented in this section, we are motivated to study the following research problems related to software reliability. We are motivated to study the following research problems as these seem to be unattempted / unavailable in published form.

(i) A mechanism to calculate the reliability of the software based on the cumulative observations of time domain failure data using mean value function of HLSRGM using two step approach.

(ii) A process control mechanism based on the cumulative observations of time domain failure data using mean value function of HLSRGM using two step approach.

(iii) An attempt is made to adopt the methodology of Stieber to an NHPP with a general mean value function and specialize it for our specific NHPP based Software Reliability Growth Model (HLSRG) in time domain using two step approach. The work of Stieber (1997) dealt with detection of unreliable software components using the theory of Wald’s sequential probability ratio test procedure as a statistical quality control tool. It is about a homogenous Poisson process.

Our attempts and findings in these directions are presented in the chapters that follow with an aim of proposing analytical techniques for quality software. Reprints of some of our papers published in standard journals are appended towards the end of the thesis.