Monitoring the Software Test Process Using Statistical Process Control: A Logarithmic Approach

João W. Cangussu
Department of Computer Science
University of Texas at Dallas
Richardson, TX - USA
75083-0688
cangussu@utdallas.edu

Raymond A. DeCarlo
Department of Electrical and Computer Engineering
Purdue University
West Lafayette, IN - USA
47907-1285
decarlo@ecn.purdue.edu

Aditya P. Mathur*
Department of Computer Science
Purdue University
West Lafayette, IN - USA
47907-7823
apm@cs.purdue.edu

ABSTRACT

Statistical Process Control (SPC) is a powerful tool to control the quality of processes. It assists management personnel in the identification of problems and actions to be taken to bring a process into a stable state. SPC has been applied in various fields, including the Software Development Process. However, some processes are better characterized by factors that exhibit an exponential behavior. The use of such factors for process control limits the application of traditional SPC techniques. The Software Test Process (STP) characterized by the decay in the number of remaining errors, failure intensity, and an increase in code coverage, is one such process.

A variant of the traditional SPC technique is proposed. This variant uses logarithmic transformation to allow the statistical control of processes whose dominant behavior is best described by an exponential. An evaluation of the proposed transformation carried out using simulation and a case study from an industrial project, encourages the application of the proposed variant to the STP.

Categories and Subject Descriptors

H.1 [**Information Systems Applications**]: Models and Principles; D.2.9 [**Software Engineering**]: Management—*productivity, software quality assurance*

General Terms

Management, Measurement

Keywords

Statistical process control, software test process, logarithmic approach

*Financial support in part by SERC and NSF.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

ESEC/FSE'03, September 1–5, 2003, Helsinki, Finland.

Copyright 2003 ACM 1-58113-743-5/03/0009 ...$5.00.

1. INTRODUCTION

Statistical Process Control is a quality control method based on the monitoring of an undergoing process. SPC was idealized for the quality control of production lines [1]. For example, when multiple instances of a product, say a rivet, are to be produced within specified size limits, SPC can be used to check if the number of instances within these limits is acceptable and if the entire process will achieve the desired production within the control limits for a specified duration. The success of SPC to control the quality in production lines led to its application in other areas such as chemical, electronics, food, packaging, and software development [2, 3, 4, 5, 6]. The Six-σ approach for quality improvement of businesses and software processes is another example of a methodology based on SPC [7, 8, 9].

The application of SPC and its derivatives is constrained to processes whose desired quality measures do not vary over time. However, the quality and progress of some processes is best analyzed by monitoring and measuring factors that present an exponential decay or rise. The Software Test Process (STP) [10, 11, 12] and the decay of the effectiveness of some filters used in a renal dialysis machine are two examples of such processes. As explained later, traditional techniques of SPC do not allow the control of processes that present such behavior. The application of weighted moving averages [13] also does not address the statistical control of processes with characteristic features that are exponential.

In this paper we focus on the control of the STP using key quality factors. However, the proposed technique can be used to monitor any process with factors that exhibit exponential behavior. Code coverage, number of remaining errors, and the failure intensity are examples of three quality factors in the STP that exhibit exponential behavior [12, 14]. For an analysis of the approach presented here, we focus on the number of errors found per time unit. A variant of the SPC based on a logarithmic transformation is proposed to allow for the monitoring and the statistical control of the STP using the decay in the number of remaining errors in the product under test.

Since the focus of this paper is on the STP, one could argue why not use existing Reliability growth models, instead of SPC, to monitor the process. Reliability growth models focus on prediction and not on monitoring. That is, they do not address the issues of the time to make changes in

the process. Suppose a 1% deviation from the expected reliability is observed. Should the manager make changes in the process or is the variation too small to be taken into consideration? This is some other questions, not addressed by Reliability growth models, justify the application of the SPC variation described here.

The remainder of this paper is organized as follows. For the sake of completeness, a brief introduction to SPC is provided in Section 2. The difficulties in applying SPC to the STP, and a variant of SPC to make its application possible, are presented in Section 3. Section 4 describes a method used to evaluate the proposed variant. Section 5 presents an analysis of the results obtained using the evaluation method. A summary of this work and conclusions thereof are in Section 6.

2. STATISTICAL PROCESS CONTROL

Statistical Process Control, as described by Florac and Carleton [15], is an analysis tool to improve the controllability of the Software Development Process (SDP). The two concepts of stability and capability in SPC apply directly to the control of SDP.

Stability [15]: *A process is stable if it varies within predictable limits. If the variation is beyond these limits, the process is considered unstable (and not under control).*

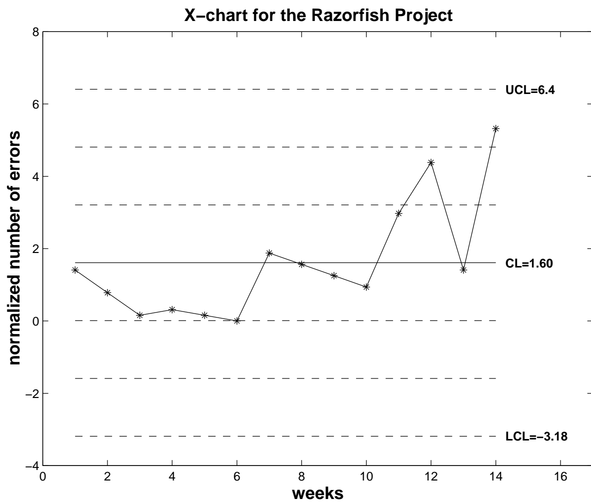

Figure 1: Control Chart for a commercial project.

Capability [15]: *A process is capable if it is able to achieve the expected results within a specified time. Capability is based on the probability that the process can achieve the results within the specified time. Clearly this presupposes that the process is stable.*

A Control Chart is used commonly to determine whether or not a process is under control, i.e., stable. Several different types of Control Charts exist; X-Bar, Range Charts, U Charts, and Z Charts are a few examples. Each chart is designed to be used based on specific conditions related to the available data. We exemplify the use of a Control Chart by describing an X-Bar chart [16].

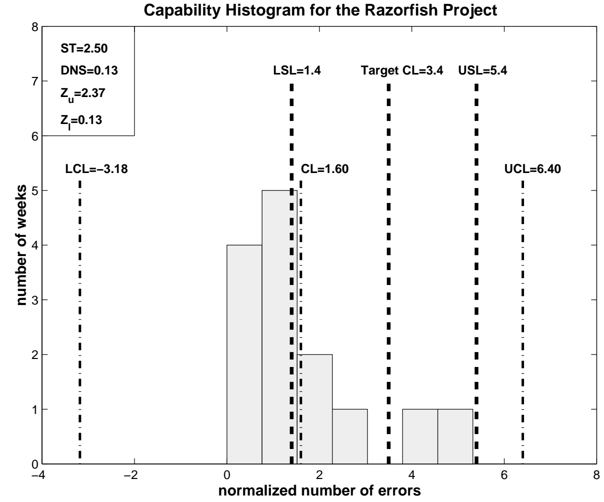

Figure 2: Process Capability Histogram for an industrial project

An X-bar chart is based on averages of grouped data. It requires that data be grouped into subgroups of size at least two. The data for each subgroup i is averaged ($\overline{X}_i$) and the standard deviation for the subgroup σ_i is computed. Then, the average of the subgroup averages ($\overline{\overline{X}}$) is computed along with the average standard deviation ($\overline{\sigma}$). Based on these values, three lines are presented in an X-Bar chart: the (CL) center line which is $\overline{\overline{X}}$, the (UCL) upper control limit $\overline{\overline{X}} + 3\overline{\sigma}$, and (LCL) the lower control limit $\overline{\overline{X}} - 3\overline{\sigma}$. An example X-Bar chart is shown in Figure 1. In this work we assume the standard control limit ($\overline{\overline{X}} \pm k\overline{\sigma}$) for k = 3. However, as pointed out by Jalote and Saxena [17], an optimized control limit can be established to better match the specific characteristics of a software process. This change in the control limit will not affect the SPC variant presented here.

The analysis of an X-Bar chart determines process stability. A process is said to be out of control (unstable) if any one of the following tests fail [15].

- Test 1 : A single point is outside the limits of LCL and UPL.
- Test 2 : At least two out of three successive values fall on the same side of, and more than 2σ units away from, the center line.
- Test 3 : At least four out of five successive values fall on the same side of, and more than 1σ unit away from, the center line.
- Test 4 : At least eight successive points fall on the same side of the center line.

We observe in Figure 1 that according to the tests defined above the process under observation is under control, that is, the process is stable. Although the process is stable, the average number of errors removed is about 1.8% below its expected value. The data from the Razorfish project is presented here normalized to a 100 units. Therefore, each unit represents 1%. The Target Center Line, determined by the

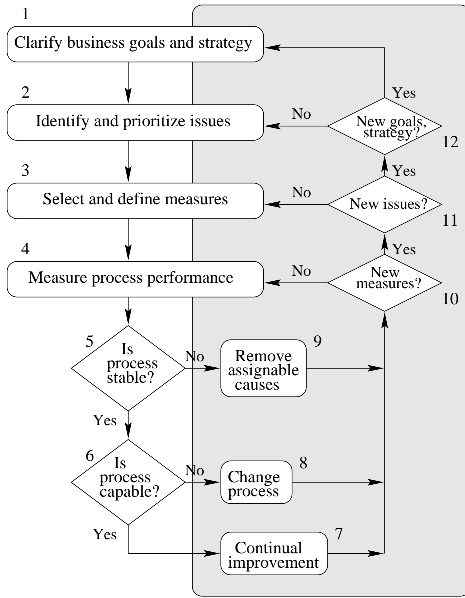

Figure 3: Sequence of steps for Statistical Process Control [15].

manager, is $TCL = 3.4$ leading to the 1.8% deviation when compared to the center line in Figure 1. Thus, we need to analyze whether or not the process is capable of accomplishing the planned task when restricted to a predetermined cost and schedule. The deviation from the center line does not provide enough evidence that a process is not capable. A Capability Histogram such as the one in Figure 2 can be used to accomplish this task by determining if a process is outside the expected limits. Being outside expected limits represents a higher probability of generating non-conforming results and indicates that the process should be adjusted in order to achieve the expected results.

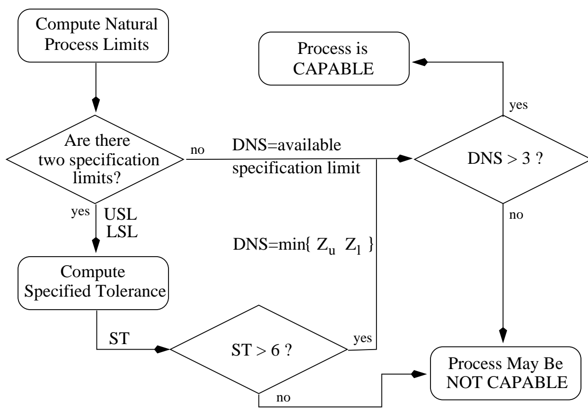

Figure 4: Capability verification for Statistical Process Control [15].

The equations used to determine process capability [15] are presented below.

$$ST = \frac{USL - LSL}{\sigma} \quad (1)$$

$$DNS = min\{Z_u, Z_l\} \quad (2)$$

$$Z_u = \frac{USL - \overline{\overline{X}}}{\sigma} \quad (3)$$

$$Z_l = \frac{\overline{\overline{X}} - LSL}{\sigma} \quad (4)$$

where $\overline{\overline{X}}$ is the average of the observed values during the period under consideration, ST is the Specification Tolerance, USL is the Upper Specification Limit, LSL is the Lower Specification Limit[1], DNS is Distance to the Nearest Specification, Z_u is the distance in σ units between the process average and USL; and Z_l is the distance in σ units between the process average and LSL. Wheeler states "... sigma units express the number of measurement units which correspond to one standard unit of dispersion" [1]. Sigma units are used to characterize how much of the data is within a given α distance from the observed average [1].

In Figure 3 we observe the steps followed to monitor a process when SPC [15] is used. Step 5 represents one of the stability tests listed earlier. The capability test is represented by Step 6. The details of how SPC determines if a process may or may not be capable are provided in Figure 4.

3. APPLYING SPC TO THE STP

We shall now explain why an STP is characterized by an exponential behavior and then derive reasons why traditional SPC is not applicable to control the STP. Following this discussion we introduce in Section 3.3 a variant of SPC based on a logarithmic transformation, hereafter referred to as SPC_{log}, that makes it possible to control the STP and any other process presenting an exponential behavior.

3.1 Does the STP Actually Present an Exponential Behavior?

SPC_{log} as described in Section 3.3 can be applied to any process presenting an exponential behavior. Here, we use the STP to exemplify the applicability of this technique.

It is widely experienced that defects are relatively easier to find during the early portions of the system test phase. Perhaps this is because there are many of them and not difficult to trigger [18]. Finding defects becomes increasing difficult with the progress of time perhaps because there are fewer of them and often require a specific combination of events to be triggered. This behavior implies an exponential shape for the decay in the number of accumulated defects and, as defects are removed, failure intensity and reliability also present an exponential behavior [12]. However, in some situations the STP presents a S rather than an exponential shape. This behavior is perhaps due to factors such as learning and adaptation present at the beginning of the STP [19]. In both cases, the assumption here is that the STP is a well defined, not ad-hoc, process. Our concern is with processes where the learning and adaptation effects are minimized and the dominant behavior can be characterized as exponential.

[1]Both USL and LSL are limits determined by the manager according to his/her expectations.

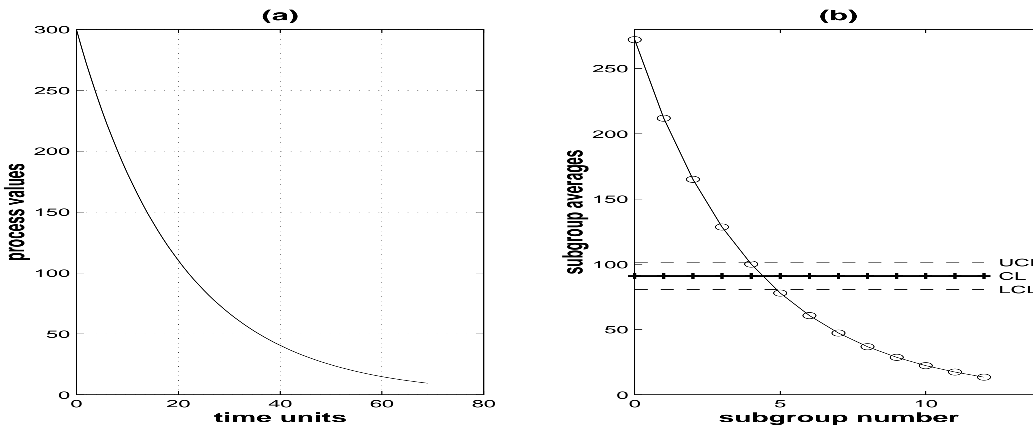

Figure 5: Example of a process presenting an exponential decay.

Researchers have provided evidence of test processes presenting an exponential behavior. The error log reported by Knuth has a clear exponential decay during the test phase [20]. The accumulated number of faults plotted in Dalal's work [21] presents an exponential behavior which is also observed in the Razorfish project [18]. Jacoby and Tohma's reliability growth model [22] presents cases of both exponential and S shaped curves. Additionally, Ehlrlich [23] presents examples of exponentially accumulated failure and failure intensity decay.

3.2 Difficulties in applying traditional SPC Techniques to the quality control of the STP

The use of statistical process control does not assume any specific distribution of values of the factor being observed for process control and is applicable to a variety of distributions such as uniform, exponential, normal, triangular, and chi-square [1]. This fact is justified by experiments that reveal at least 95% of the observed values lie between the 6σ limit [1], i.e. between the Lower and Upper control limits. The assumption in all the cases is that values of the observed factor are randomly distributed. This assumption raises the following question: "What happens if a process produces correlated values that are not randomly distributed?"

As an example of correlated values consider a process presenting an expected exponential decay as shown in Figure 5a. By "expected" we refer to the values (v_j) that must be produced by an almost perfect process. Assuming subgroups of size δ ($\delta = 5$ in Figure 5) and computing the average $(\overline{X}_i)$ and the standard deviation (σ_i) for each subgroup produces the results shown in Figure 5b. From the analysis of each subgroup it is observed that the standard deviation is smaller than the difference between the first (largest) and last (smallest) value of each subgroup ($\sigma_i < v_{i\times\delta} - v_{((i-1)\times\delta)+1}$). The largest standard deviation will then be smaller than the largest difference between δ consecutive points ($\sigma_{max} < Max(v_{i\times\delta} - v_{((i-1)\times\delta)+1})$, for $i = 1, \ldots, n_s$), where n_s is the number of subgroups. Thus, the average standard deviation will be smaller than the maximum value, $\overline{\sigma} < \sigma_{max}$. The control limits for the process are $\overline{\overline{X}} \pm 3 \times \overline{\sigma}$. It can be shown that if the number of subgroups is greater than 2 ($n_s > 2$), many points will fall outside of these limits as observed in Figure 5b. In this case, according to the X-Bar chart in Figure 5b, the process is not stable while indeed it presents a perfect expected exponential decay. Even the application of an exponential weighted moving average technique [13] to compute the control limits does not solve the problem.

Assume now that the values (v_j) from Figure 5a are randomly distributed as seen in Figure 6a. Constructing the X-Bar chart with the same configuration as above leads to results shown in Figure 6b. It can be observed from this figure that all tests defined in Section 2 are successful and therefore the process is stable.

The conclusion we draw from the above discussion is that the assumption of a random distribution of values prohibits the application of SPC to a process presenting an exponential behavior. The STP can be characterized and/or monitored by analyzing the number of errors found per time unit or the increase in code coverage. In both these cases, an exponential behavior is observed, a decay in the case of the number of errors found per unit time and an increase in the case of code coverage. Though the STP will most likely not present a perfect exponential decay, the behavior will be well approximated by an exponential and will present characteristics that prevent the use of SPC.

3.3 SPC_{log}

As mentioned earlier, SPC expects the variation of the process to occur over a horizontal line. Let $v_i, i = 1 \ldots n$ be the values observed for a process. A logarithmic transformation described next allows the application of SPC to control a process with exponential behavior.

The first step is to compute the natural logarithm of the observed data for each value, $log(v_i), i = 1 \ldots n$. This transformation generates an overall linear decay rather than an exponential decay. The second step is to compute a linear least square fit of $log(v)$ producing $LSF_i, i = 1 \ldots n$. This approximation captures an average linear decay of the data. The variation can now be observed over a line with a negative slope but not a zero slope (horizontal line) as expected by SPC. We now ask whether or not the values of $log(v)$ are under control according to the tests defined in Section 2. To obtain an answer we use the Euclidean Distance (δ) in Eq. 5 to compute the distance between each point of the $log(v_i)$ and the respective point in the least square fit LSF_i as in Eq. 6. The two points under consideration here are always

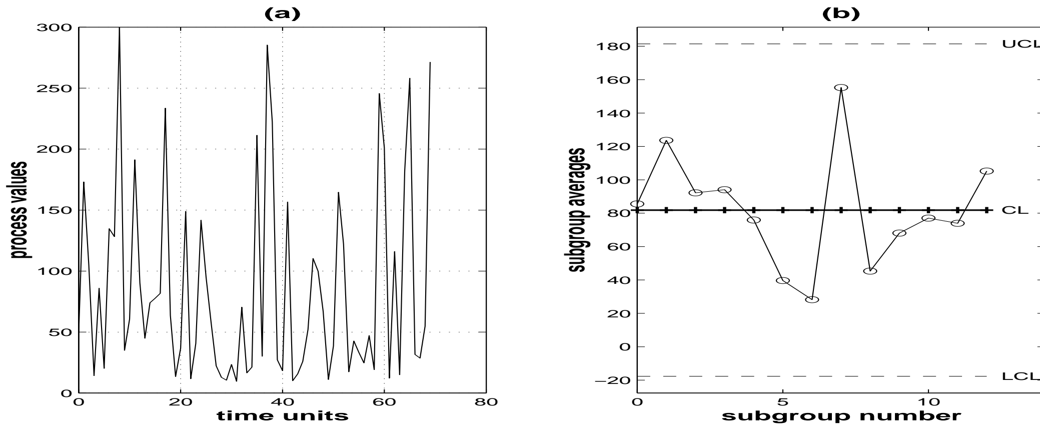

Figure 6: Random permutation of the process values from Figure 5.

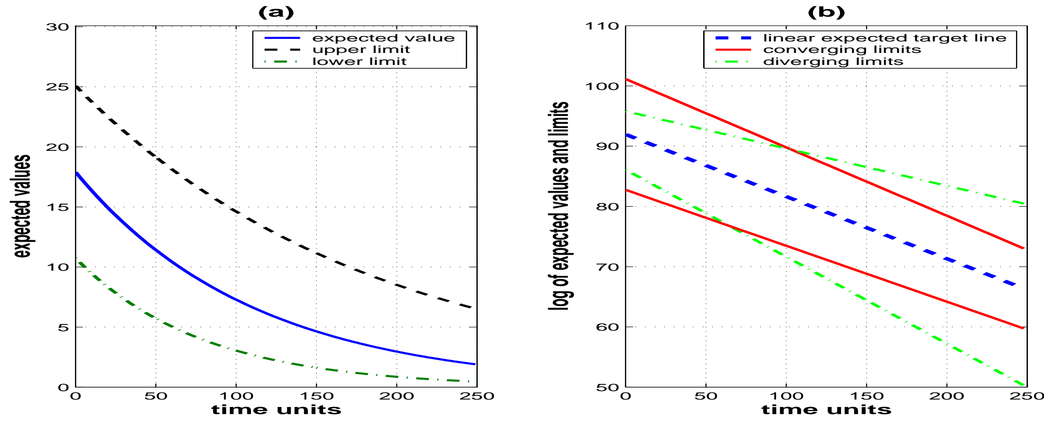

Figure 7: Example of expected behavior, upper, and lower limits and the corresponding logarithmic transformation.

in the same position on the time axis and the distance is therefore measured considering only the vertical axis as in Eq. 5.

$$\delta_i = \sqrt{(x(i)_1 - x(i)_2)^2 + (y(i)_1 - y(i)_2)^2} \xrightarrow{x(i)_1 = x(i)_2} \delta_i = \sqrt{(y(i)_1 - y(i)_2)^2} \quad (5)$$

Computing the Euclidean distance between the log values and the least square fit, as in Eq. 6, generates a linear variation with a zero slope decay allowing the application of standard SPC techniques.

$$x_i = \sqrt{(LSF_i - log(v_i))^2}\ , i = 1 \dots n \quad (6)$$

Assuming subgroups of size m, the average and standard deviation can now be computed for each subgroup.

$$\overline{x_i} = \frac{\sum_{j=i}^{i+m-1} x_j}{m} \text{ for } i = 1, 1+m, 1+2m, \dots, n \quad (7)$$

$$\sigma_i = \sqrt{\frac{\sum_{j=i}^{i+m-1} (x_j - \overline{x_i})^2}{m}} \text{ for } i = 1, 1+m, 1+2m, \dots, n \quad (8)$$

Finally, the center line ($\overline{\overline{x}}$) and the control limits are computed as in SPC and the process can be checked for stability.

A compact listing of the steps to check stability for a process presenting an exponential decay is provided in Table 1.

The steps above determine whether a process is stable or not. In the case stability is observed, the process needs to be analyzed for capability. A capability histogram can be plotted from the subgroup averages $\overline{x_i}$ together with the computed control limits.

Under the assumption the process has an expected exponential behavior, the expected decay (Target Center Line) and an initial value should be provided by the process manager. The upper and lower limits, USL and LSL, respectively, should also be provided by the manager. Let us assume the expected values are defined by $e_v(t) = v_0 e^{-\lambda \times t}$. Thus the initial value v_0 and the expected decay parameter λ is to be provided by the process manager.

A first attempt to generate the upper and lower specification limitis was to derive them directly from the expected exponential decay. Assume that the manager accepts a Δ variation for the decay parameter and the initial value. Under these conditions, the upper and lower limits can be computed as in Eqs. 9 and 10.

$$e_v^{up}(t) = log((1 + \Delta) \times v_0 e^{-((1-\Delta)\times\lambda)\times t}) \quad (9)$$

$$e_v^{low}(t) = log((1 - \Delta) \times v_0 e^{-((1+\Delta)\times\lambda)\times t}) \quad (10)$$

Table 1: Steps in applying the SPC_{log} approach

Step	Stability	Capability
1	$ld_i = log(v_i),\ i = 1,\ldots,n$	compute the expected decay ($e_v = v_0 e^{-\lambda \times t}$) .
2	compute least square fit (lsf_i) of ld_i	compute the log of the expected value ($log(e_v)$) and the upper ($e_v^{up} = (1+\Delta) \times log(e_v)$) and lower ($e_v^{low} = (1-\Delta) \times log(e_v)$) specification limits for a given Δ variation.
3	compute $x_i = \sqrt{(lsf_i - ld_i)^2}$	compute TCL, the average Euclidean distance between the expected and observed averages ($TCL = \overline{\sqrt{(e_{ld}(i) - log(v_i))^2}}$).
4	using $x_i, i = 1,\ldots,n$, compute the control limits as in SPC	compute $USL = TCL + \overline{\sqrt{(e_v^{up}(i) - e_{ld}(i))^2}}$ and $LSL = TCL - \overline{\sqrt{(e_{ld}(i) - e_v^{low}(i))^2}}$.
5	check stability using SPC tests	check capability using SPC tests

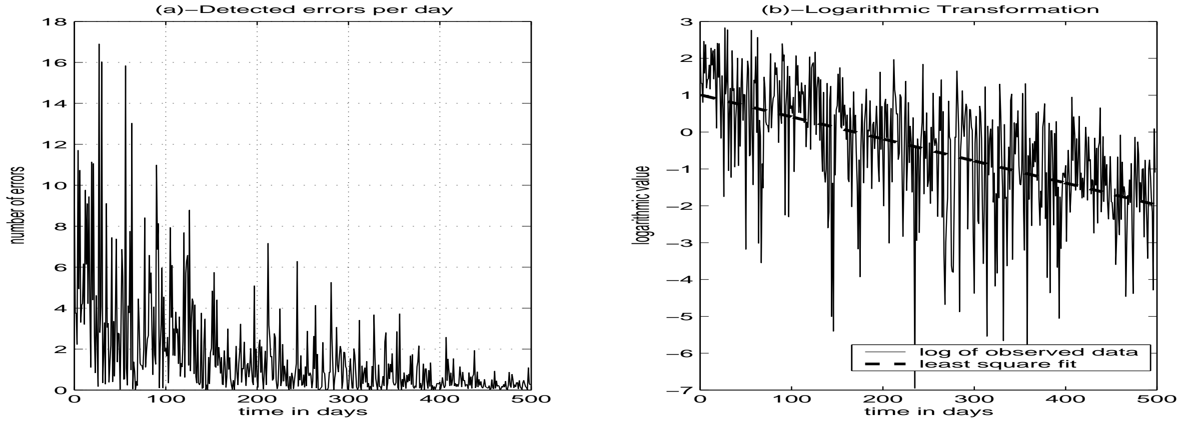

Figure 8: Results from simulation Scenario I. Figure (a) depicts the generated data and Figure (b) shows the corresponding logarithmic transformation along with the least square fit.

As shown in Figure 7 this approach produces diverging specification limits. The computation of DNS in Eq. 2 and ST in Eq. 1 becomes more sensitivity to the use of this approach as the value of t increases. That is, due to the diverging specification limits the average euclidean distance increases with time and so does the values of DNS and ST.

An alternative way to define the specification limits is to apply the Δ variation to the log of the expected decay instead of apply it to the exponential decay. Eqs. 11 and 12 are used to achieve this goal. This results in converging rather than diverging limits that appear to be more appropriated for the computation of DNS and ST. Figure 7 depicts the two converging limits.

$$e_v^{up}(t) = (1+\Delta) \times log(e_v) \tag{11}$$
$$e_v^{low}(t) = (1-\Delta) \times log(e_v) \tag{12}$$

Computing $log(e_v)$ produces a linear expected decay (e_{ld}) that can be compared with the observed average linear decay derived from the least squares fit. That is, we are interested in computing how distant the average observed values (determined by the least squares fit) are from the expected values from the process manager's estimation. This distance represents the Target Center Line (TCL). The center line $\overline{\overline{x}}$ in the capability histogram is computed using the average distance from $\sqrt{(log(v) - LSF)^2}$. To check how far the observed line is from the expected line, we need to compute the average based on the same set of points. Thus we use the euclidean distance between e_{ld} and $log(v)$ and not between e_{ld} and LSF leading to the computation of the TCL as in Eq. 13.

$$TCL = \overline{\delta}, \quad \text{for } \delta_i = \sqrt{(e_{ld}(i) - log(v_i))^2} \tag{13}$$

The computation of e_v^{up} and e_v^{low} produces the upper and lower linear limits for the process as observed in Figure 7. The average Euclidean distance between these limits and the expected decay (e_{ld}) is used to compute the USL and LSL in Eqs. 14 and 15, respectively.

$$USL = TCL + \overline{\delta^{up}}, \quad \text{for } \delta_i^{up} = \sqrt{(e_v^{up}(i) - e_{ld}(i))^2} \tag{14}$$
$$LSL = TCL - \overline{\delta^{low}}, \quad \text{for } \delta_i^{low} = \sqrt{(e_{ld}(i) - e_v^{low}(i))^2} \tag{15}$$

The USL and LSL are used to compute ST in Eq. 1 and DNS in Eq. 2. Finally, the capability of the process can be checked as shown in Figure 4. A summary of the steps to check process capability is presented in Table 1.

4. EVALUATION OF SPC_{LOG}

The SPC_{log} technique presented here is evaluated using simulation and data collected from a commercial project.

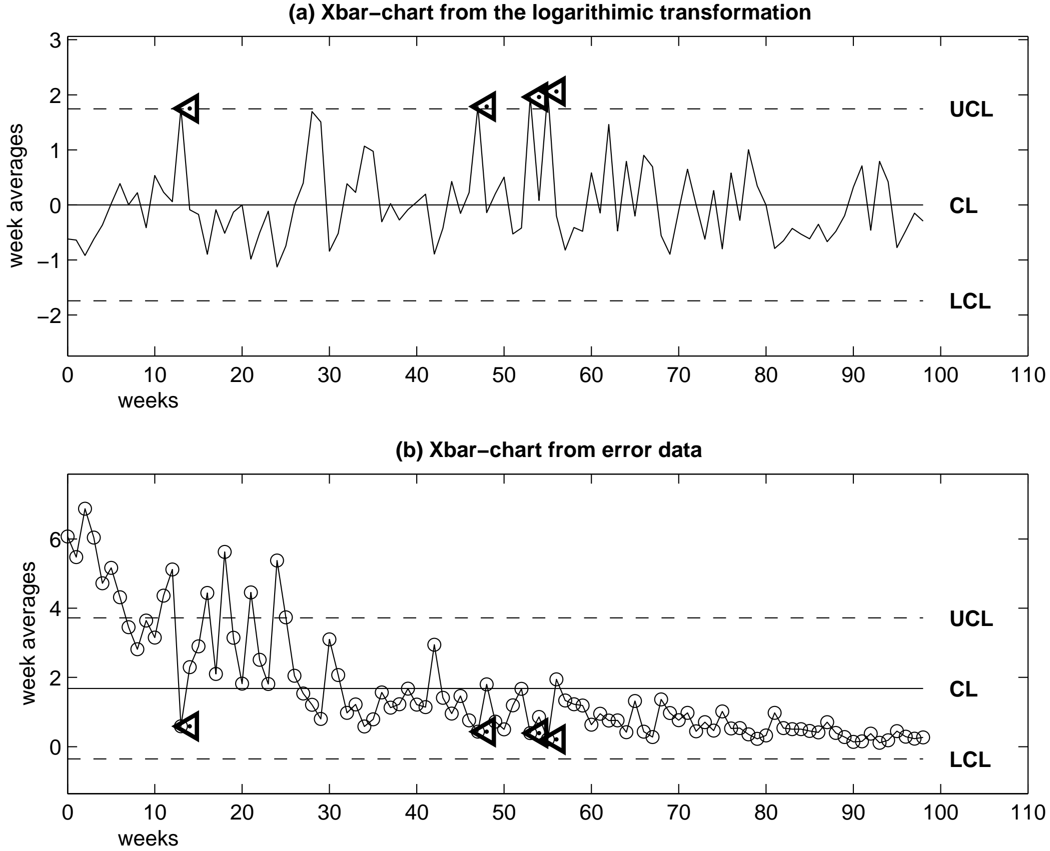

Figure 9: Results for simulation Scenario I. (a) the Xbar chart for the euclidean distance between the $log(errors)$ and the least squares fit. (b) the X-bar chart for the actual number of errors.

The simulation runs exercise SPC_{log} in different ways resulting in three different scenarios, two of which are described in Section 4.1. The second part of the evaluation process has shown the applicability of SPC_{log} to an actual project but its analysis is beyond the scope of this paper.

4.1 Simulation scenarios

Three scenarios are of interest (i) an unstable process, (ii) a stable and capable process, and (iii) a stable but not-capable process. Due to limitations of space only the first two of the three scenarios are presented here. These two scenarios are based on random perturbations of an exponential decay process. Having an expected decay determined by $v_e(t) = v_o e^{-\lambda \times t}$, the perturbations are inserted by computing $v_i = \alpha_i \times v_e$, where $0 < \alpha_i < 2$. Therefore, the process slows when $\alpha_i < 1$ and speeds when $1 < \alpha_i < 2$. The probabilities of these two cases are also controlled to decelerate or accelerate the process and produce the desired scenarios. The subgroup size is defined to be 5 to account for weekly average. However, the technique presented here does not impose any restriction on subgroup sizes and different values can be used.

Scenario I

The first scenario represents an unstable software test process. In this case, capability is not analyzed as the process must be in a stable condition before capability is considered. Figures 8 and 9 show the results of this scenario that are analyzed in Section 5.1. The graph in Figure 8(a) was generated according to the random perturbation of an exponential decay process described above whereas Figure 8(b) depicts corresponding logarithmic values along with a least square fit.

Scenario II

Simulation Scenario II illustrates a stable and capable STP. Capability for this scenario was achieved by using a normal distribution to generate equal probability of $0 < \alpha_i <= 1$ and $1 < \alpha_i < 2$, i.e. equal probability of decelerating or accelerating the process.

5. ANALYSIS OF THE SPC_{LOG} RESULTS

5.1 Scenario I

One result of the simulation for Scenario I is depicted in Figures 8 and 9. As observed from Figure 9a, the process is not stable because four points fall outside the 3σ limit, i.e. outside the Upper Control Limit (UCL). The points are outside the limits because of a sequence of low values of the number of detected errors $v_i, \ldots, v_{i+4}$ within the same week (subgroup) generating points in the logarithmic transformation (Figure 8b) that are far from the least squares fit. Let $lsf_i, \ldots, lsf_{i+4}$ be the least squares fit of the logarithmic transformation for week i. When the computation of

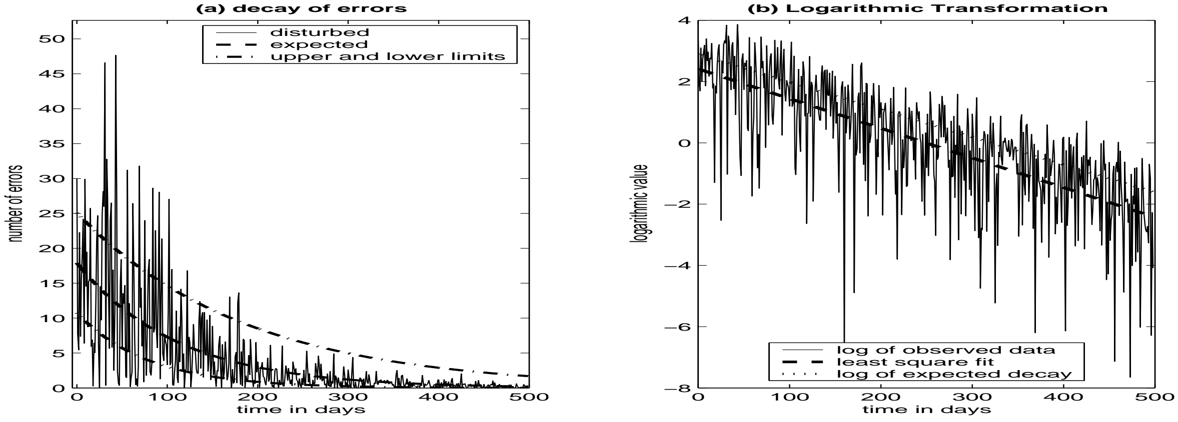

Figure 10: Simulation Scenario II: (a) the generated data with the manager's expectations represented by the expected decay with upper and lower limits. (b) the corresponding logarithmic transformation along with the least square fit and the linear expected decay.

$\sqrt{(lsf_j - log(v_j))^2}$, for $j = i, \ldots, i+4$ leads to a sequence of small or large values (the points are distant from the least squares fit) it increases the average computed for the X-Bar chart for SPC_{log} generating points over the 3σ limit. The relation of the actual number of errors with the points outside the limits can be observed by comparing the "triangles" from Figures 9a and 9b. A sequence of low values for the error detection increases the average $\overline{\sqrt{(lsf - log(v))^2}}$ and generates a point above the Upper Control Limit (LCL) identifying an unstable process.

It is also noticed from Figure 9a that an X-bar chart for the actual number of errors will most likely produce points outside the limits even though the process may be under control. The initial weekly averages will in general be larger than the UCL characterizing an unstable process and the last week's averages will be smaller than the UCL which according to Test 4 in Section 2 also characterizes an unstable process. Although the process in Figure 9a and 9b is detected to be unstable by both techniques (SPC and SPC_{log}), as pointed out in Section 3.2 it may not be the case for a stable process.

5.2 Scenario II

For the simulation Scenario II, the expected behavior is generated by $r(t) = r_0 e^{-\lambda \times t}$ and the upper and lower specified limits (USL and LSL) are defined respectively by $e_v^{up}(t) = (1+\Delta) \times log(r(t))$ and $e_v^{low}(t) = (1-\Delta) \times log(r(t))$, where $\delta = 0.15$. These results are observed from Figure 10a while Figure 10b shows the log of the data points, the least squares fit, and the log of the expected decay. From the X-bar chart in Figure 11a we can see that the process is stable. Notice that the use of standard SPC techniques would have identified the process as unstable as can be observed from Figure 11b.

As the process is under control, capability should be analyzed. Using the method in Section 3.3, we computed TCL= 1.049, USL= 2.268, and LSL= -0.357 as shown in the capability histogram in Figure 12a. The value of σ for this scenario was 0.337.

Since two specified limits are available, the first step to verify capability is to check if $ST > 6$, which is observed to be true for this case: $ST = \dfrac{USL - LSL}{\sigma} = 7.79$. The next step is to verify if the distance to the nearest specification (DNS) is greater than three. For Scenario II, $Z_u = \dfrac{USL - \overline{\overline{X}}}{\sigma} = 3.62$ and $Z_l = \dfrac{\overline{\overline{X}} - LSL}{\sigma} = 4.17$. Therefore, $DNS = min\{Z_u, Z_l\} = 3.62 > 3$ resulting in a capable process as observed from Figure 12b.

6. SUMMARY AND DISCUSSION

Factors that characterize a process and present an exponential shape prevent the application of SPC to monitor and control such processes. The STP is characterized by several quality factors that exhibit an exponential behavior over time. Examples of such factors include code coverage, number of remaining errors, and reliability of the product under test. A logarithmic transformation named SPC_{log} is presented in this paper. This transformation allows the control of processes with such factors and improves the applicability of SPC to the statistical control of the STP.

The analysis of the applicability of SPC_{log} presented here is based on simulation runs for two scenarios. Results obtained using our analysis demonstrate the applicability of the proposed approach, and its correctness, when applied to the STP. The results of a third scenario and a case study using data from a large industrial project also led to the same conclusion but a description of the same is beyond the scope of this paper. Though the analysis of the approach was based only on the behavior of the number of errors found per time unit, the same transformation applies also to the other process factors mentioned above. Furthermore, we believe that there is no restriction in the application of SPC_{log} to any process presenting an exponential behavior.

7. REFERENCES

[1] D. J. Wheeler and D. S. Chambers, *Understanding Statistical Process Control*. SPC Press, 1992.

[2] J. F. Manji, "SPC inspires global quality culture for multinational giant," *Managing Automation*, November 1999.

[3] J. L. Cawley, "Improving yields with statistical process control," *Circuits Assembly*, March 1999.

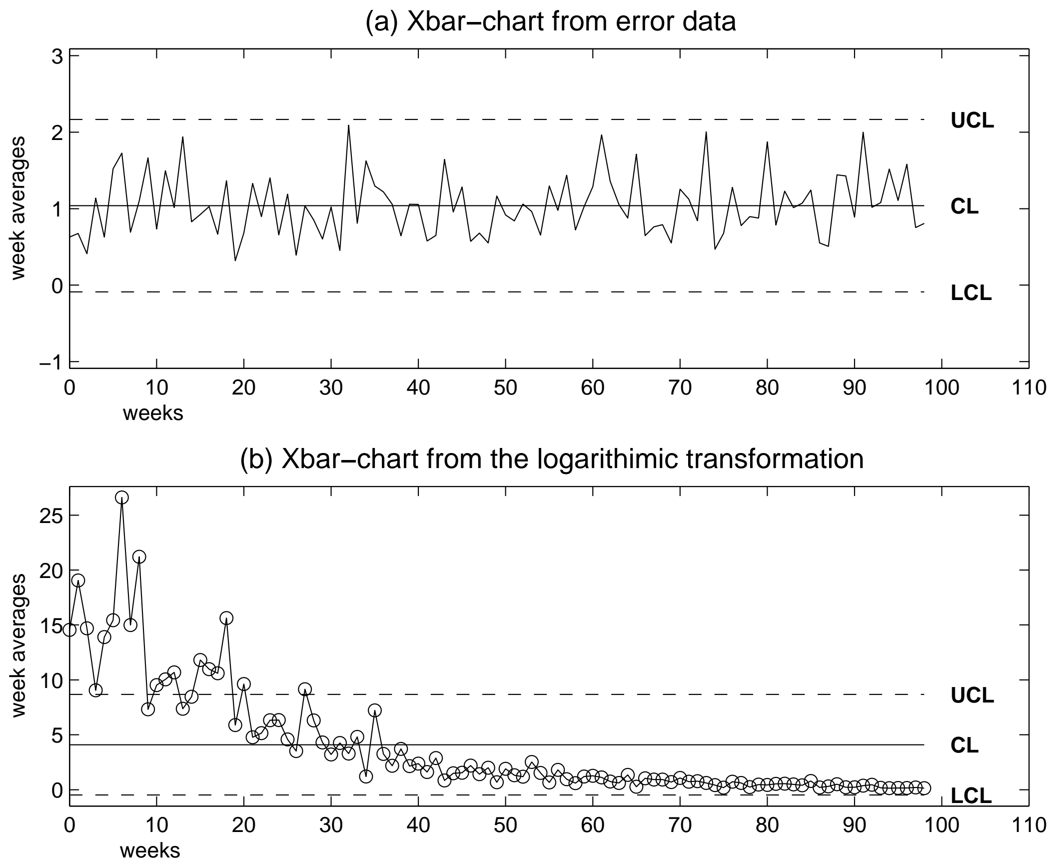

Figure 11: Results for simulation Scenario II. (a) the Xbar chart for the euclidean distance between the $log(errors)$ and the least square fit. Figure (b) the X-bar chart for the actual number of errors.

[4] J. G. Surak, J. L. Cawley, and S. A. Hussain, "Integrating haccp and spc," *Food Quality*, May 1998.

[5] M. A. Lantzy, "Application of statistical process control to the software process," in *Proceedings of the* 9th *Washington Ada symposium on Empowering software users and developers*, (New York, NY, USA), pp. 113–123, ACM Press, 1992.

[6] E. F. Weller, "Practical applications of statistical process control," *IEEE Software*, vol. 17, pp. 48–55, May-June 2000.

[7] P. S. Pande, R. P. Neuman, and R. R. Cavanagh, *The Six Sigma Way: How GE, Motorola, and Other Top Companies are Honing Their Performance*. McGraw-Hill, 2000.

[8] D. N., "Sorting out six sigma and the cmm," *IEEE Software*, vol. 17, pp. 11–13, May-June 2000.

[9] R. V. Binder, "Can a manufacturing quality model work for software?," *IEEE Software*, September-October 1997.

[10] A. P. Mathur, F. D. Frate, P. Garg, and A. Pasquini, "On the correlation between code coverage and software reliability," in *Proceedings of the Sixth International Symposium on Software Reliability Engineering*, (Toulouse, France), pp. 124–132, IEEE Press, October 24-27 1995.

[11] S. Biffl, "Using inspection data for defect estimation," *IEEE Software*, vol. 17, pp. 36–43, November/December 2000.

[12] J. W. Cangussu, R. A. DeCarlo, and A. P. Mathur, "Feedback control of the software test process through measurements of software reliability," in *Prooceding of* 12th *International Symposium on Software Reliability Engineering*, (Hong Kong), pp. 232–241, November 2001.

[13] J. S. Oakland, *Statistical Process Control*. Butterworth Heinemann, fourth ed., 1999.

[14] W. K. Ehrlich, J. P. Stampfel, and J. R. Wu, "Application of software reliability modeling to product quality and test process," in *Proceedings of ICSE*, pp. 108–116, 1990.

[15] W. A. Florac and A. D. Carleton, *Measuring the Software Process: Statistical Process Control for Software Process Improvement*. SEI Series in Software Engineering, Addison-Wesley, 1999.

[16] W. A. Florac, A. D. Carleton, and J. Barnard, "Statistical process control: Analyzing a space shuttle onboard software process," *IEEE Software*, vol. 17, pp. 97–106, July/August 2000.

[17] P. Jalote and A. Saxena, "Optimum control limits for employing statistical process control in software process," *IEEE Transactions on Software Engineering*, vol. 28, no. 12, pp. 1126–1134, 2002.

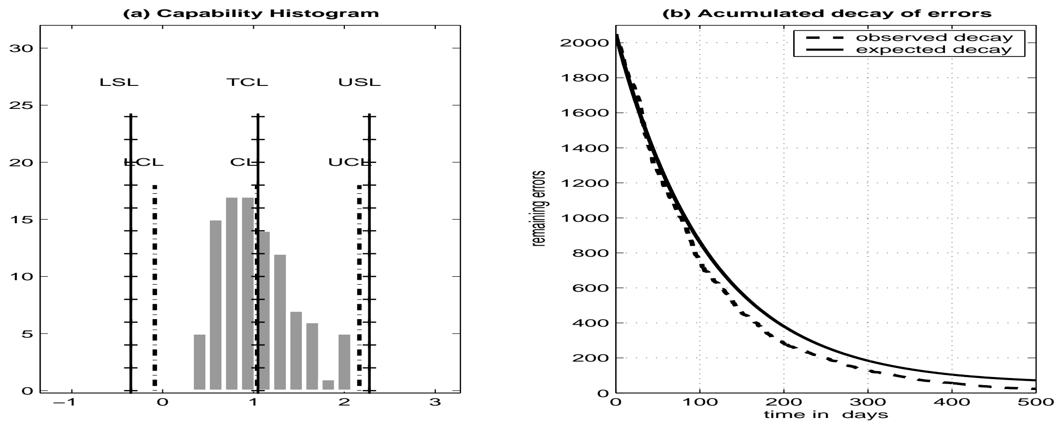

Figure 12: Simulation Scenario II: (a) the capability histogram for data from Figure 11 and (b) the accumulated decay of errors confirming a capable process.

[18] J. W. Cangussu, R. A. DeCarlo, and A. P. Mathur, "A formal model for the software test process," *IEEE Transaction on Software Engineering*, vol. 28, pp. 782–796, August 2002.

[19] J. W. Cangussu, R. A. DeCarlo, and A. P. Mathur, "Using sensitivity analysis to validate a state variable model of the software test process," *IEEE Transactions on Software Engineering*, vol. 5, May 2003.

[20] D. E. Knuth, "The errors of T_EX," *Software-Practice and Experience*, vol. 19, no. 7, pp. 607–685, 1989.

[21] S. R. Dalal and C. L. Mallows, "Some graphical aids for deciding when to stop testing software," *IEEE Journal on Selected Areas in Communications*, vol. 8, pp. 169–175, February 1990.

[22] R. Jacoby and Y. Tohma, "The hyper-geometric distribution software reliability growth model (hgdm): Precise formulation and applicability," in *14th Conference on Computer Software and Applications (COMPSAC'90)*, (Chicago, IL), pp. 13–19, IEEE, 31 October - 2 November 1990.

[23] W. K. Ehrlich, J. P. Stampfel, and J. R. Wu, "Application of software reliability modeling to product quality and test process," in *12th International Conference on Software Engineering (ICSE'90)*, (Nice, France), pp. 108–116, IEEE, 26-30 March 1990.