

A Software Test Process Stochastic Control Model based on CMM Characterization



Research Section

João W. Cangussu*

Department of Computer Science, University of Texas at Dallas, PO Box 830668, M/S EC31, Richardson, TX 75083-0688, USA

The state variable model of the software test process has shown to be a promising approach for the control of the system test phase of the software test process. However, the presence of unforeseen perturbations and noise in the data collection process impacts the accuracy of the predictions of the model and motivates the application of a stochastic control approach.

This article explores a stochastic model construct upon the deterministic state variable model of the software test process. The Capability Maturity Model is used to identify levels of disturbance and noise that can be associated with the maturity level of processes. Simulation results are presented as an indication of the applicability of the stochastic state model. Copyright

© 2004 John Wiley & Sons, Ltd.

KEY WORDS: software test process; stochastic control; disturbance characterization; CMM

1. INTRODUCTION

The understanding and controlling of a process requires knowledge about the state of the process at a given time. The more information available regarding the states of the process, the better the controllability level, i.e. knowledge about the intermediate states of a process helps in determining required changes in the presence of perturbations. The software development process (SDP) has been modeled using different techniques, ranging from finite state machine, process language, and simulation-based models among others (Abdel-Hamid and Madnick 1991, Cugola and Ghezzi 1998,

Cugola 1998, Cass *et al.* 2000, Hansen 1996, Eickelmann 2001). Recently, a state variable approach has also been used to model and control the system test phase of the software test process (STP) (Cangussu *et al.* 2002b, Cangussu *et al.* 2001a). A state variable is a set of differential equations organized in a matrix format allowing the prediction and control of the states of a process. This approach distinguishes from the others by presenting, under a control theory perspective, a closed feedback control-loop solution.

Despite the difficulty in creating mathematical models, plausible accurate models have been derived for physical and nonphysical systems (Goodwin *et al.* 2001, Luenberger 1979). However, the difficulty level arises when modeling a non-physical system owing to the fact that observation/measurement of these processes is not accurate; more specifically, they are subject to disturbances and noisy data. Under these circumstances, a stochastic rather than a deterministic model appears to be an alternative solution to represent the process.

* Correspondence to: João W. Cangussu, Department of Computer Science, University of Texas at Dallas, PO Box 830668, M/S EC31, Richardson, TX 75083-0688, USA

†E-mail: cangussu@utdallas.edu



The software test process presents such characteristics and seems to be suitable for a stochastic approach.

Here, a stochastic model of the software test process is presented. The model is a variant of a previously specified deterministic control model, briefly described in Section 3. The new model accounts for foreseen and unforeseen perturbations as well as for noise in the data collection process. The level of disturbances can be adjusted according to the maturity level of the organization.

The remainder of this article is organized as follows. Section 2 presents background material related to the state model and the Capability Maturity Model (Paulk *et al.* 1993). A brief description of the deterministic model of the STP is presented in Section 3. The new stochastic model construct upon the deterministic version is presented in Section 4 along with a description of foreseen perturbations and the association of CMM levels to unforeseen disturbances and noise level. A result from the simulation runs is presented in Section 5. Section 6 presents a brief description of some related work and a comparison with the state variable approach. Finally, Section 7 presents the concluding remarks.

2. BACKGROUND

In this section, a brief description of two important topics related to the stochastic model of the software test process is presented. Section 2.1 presents a short description of state variable models, whereas Section 2.2 defines the maturity level of an organization based on the CMM levels. These levels are used later in Sections 4.3 and 4.4 to define disturbance and noise degrees expected in a specific process.

2.1. State Models

Linear state models have provided useful representations for a multitude of systems, ranging from engineering to biological and social processes (Luenberger 1979). The general format of a deterministic Linear Time Invariant (LTI) state model is presented in Eq. 1.

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t) \\ y(t) = Cx(t) + Du(t), \end{cases} \quad (1)$$

where $x(t)$ is the state vector, $y(t)$ is the output vector, $u(t)$ is the input, and A , B , C , and D are coefficient matrices (Goodwin *et al.* 2001, Luenberger

1979). The dominant variables that can properly characterize the states of a process compose the state vector $x(t)$. The availability of a state model capturing the dominant aspects of a process allows the application of control theory. The fundamental control problem, as defined by Goodwin (Goodwin *et al.* 2001), is stated below:

Definition (Goodwin *et al.* 2001): the central problem in control is to find a technically feasible way to act on a given process so that the process adheres, as closely as possible to some desired behavior. Furthermore, this approximated behavior should be achieved in the face of uncertainty of the process and in the presence of uncontrollable external disturbances acting on the process.

On the basis of this definition it can be seen that the SDP, more specifically the STP, presents all the characteristics for the application of control theory. The adherence to a desired behavior, the presence of uncertainty, and external disturbances are commonplace in any SDP and therefore justifies the application of the stochastic approach presented here.

2.2. Capability Maturity Model

The levels of the Capability Maturity Model are presented in Figure 1 (Paulk *et al.* 1993). A brief description (Paulk *et al.* 1993) of each level is presented next to allow the association, in Sections 4.3 and 4.4, of CMM levels to disturbance and noise levels in organizations/groups with distinct process maturity.

Initial Level (1): This level can be characterized by presenting an “ad hoc” process completely dependent on the skills of the project manager. In this level, crisis during the development process is likely to be addressed by increasing the effort on the coding and testing activities. The capability of organizations/groups at Level 1 is unpredictable, unstable, and dependent on individual rather than on organizational level.

Repeatable Level (2): This level presents an improvement by establishing policies and management procedures/controls at the organizational level. This allows successful practices, based on previous projects, to be repeated, leading to more realistic expectations. Projects at

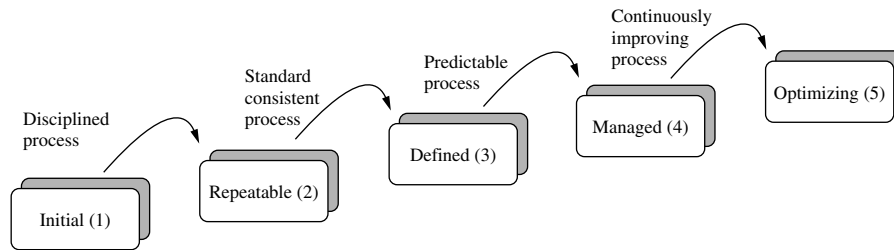


Figure 1. Levels of the Capability Maturity Model

Level 2 are more stable owing to the discipline ensured by the established management controls.

Defined Level (3): The major difference between Levels 2 and 3 is the presence of software process standards at the latter. Not only standards but also verification mechanisms and completeness criteria are in place at the organizational level. Stability and consistency are two major characteristics of processes in organizations at Level 3.

Managed Level (4): Qualitative and quantitative goals are established for both the software product and process. Measurement plays a central role in achieving these goals. Predictability represents the major improvement achieved by organizations at Level 4.

Optimizing Level (5): A continuous process improvement characterizes organizations at Level 5. Analysis of the results of projects and new technologies, combined with constant revision of the standards and control procedures, lead to process improvement and consequently better process-prediction capabilities.

The Managed Level (4) requires an environment where quality and productivity can be measured. With these two properties and others required at lower levels, a direct use of a deterministic model is foreseen. The same will apply to Level 5, since it represents an improvement over Level 4. Although Level 3 has a well-defined software development environment, the process characteristics are not completely appropriate to a deterministic approach due to the lack of or large inaccuracies on the measurement process. These measurement problems increase at levels 1 and 2, making the use of a deterministic model more difficult, though not impossible.

3. PREVIOUS RESULTS: DETERMINISTIC MODEL

The linear deterministic model of the STP is based upon three assumptions. The assumptions, presented below (Cangussu *et al.* 2002b), are based on an analogy of the STP with the physical process typified by a spring-mass-dashpot system and also in Volterra's predator-prey model (Luenberger 1979). A description and justification of this analogy, and the choice of a linear model is outside the scope of this paper (Cangussu *et al.* 2002b). The model has been validated using sets of data from testing projects (Cangussu *et al.* 2001a) and also by means of an extreme case and sensitivity analysis (Cangussu *et al.* 2003, Cangussu *et al.* 2001b).

Assumption 1: The rate at which the velocity of the remaining errors changes is directly proportional to the net applied effort (e_n) and inversely proportional to the complexity (s_c) of the program under test, i.e.

$$\ddot{r}(t) = \frac{e_n(t)}{s_c} \Rightarrow e_n(t) = \ddot{r}(t)s_c \quad (2)$$

Assumption 2: The effective test effort (e_r) is proportional to the product of the applied work force (w_f) and the number of remaining errors (r), i.e. for an appropriate $\zeta(s_c)$,

$$e_r(t) = \zeta(s_c) w_f r(t) \quad (3)$$

where $\zeta(s_c) = \frac{\zeta}{s_c^b}$ is a function of software complexity. Parameter b depends on the characteristics of the product under test.

Assumption 3: The error reduction resistance (e_r) opposes, is proportional to the error reduction velocity (\dot{r}), and is inversely proportional to the overall quality (γ) of the test phase, i.e. for an appropriate constant ξ ,

$$e_r(t) = -\xi \frac{1}{\gamma} \dot{r} \quad (4)$$



In the equations above, the first derivative (\dot{r}) of the number of remaining defects (r) represents the velocity, that is, the rate of change in r . Equivalently, \ddot{r} is the acceleration. Combining Equations 2, 3, and 4 in a force balance equation and organizing it in a State Variable (Goodwin *et al.* 2001, Luenberger 1979) format ($\dot{x} = Ax + Bu$) produces the following system of equations.

$$\begin{bmatrix} \dot{r}(t) \\ \ddot{r}(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\frac{\xi \omega_f}{s_c^{(1+b)}} & -\frac{\xi}{\gamma s_c} \end{bmatrix} \begin{bmatrix} r(t) \\ \dot{r}(t) \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{s_c} \end{bmatrix} F_d(t) \quad (5)$$

$$[r(t)] = [1 \quad 0] \begin{bmatrix} r(t) \\ \dot{r}(t) \end{bmatrix} \quad (6)$$

F_d above is included in the model to account for unforeseen disturbances such as hardware failures, personnel illness or any event that slows down or even interrupts the continuation of the test process.

Along with the model in Equation 5, an algorithm, based on system identification techniques (Ljung 1987, Juang 1993), has been developed to calibrate the parameters of the model (Cangussu *et al.* 2001c). Using data available and an estimation of \dot{r} (Cangussu *et al.* 2002b), the error difference for a specific period of time $D^i = [r(i) \quad \dot{r}(i)]^T - [r(i-1) \quad \dot{r}(i-1)]^T$ is computed. It can be shown that $D^i = M D^{i-1}$, where $M = e^{A T}$, T being the time increment between two consecutive measurements of data, and A the A-matrix of Eq. 1. Therefore, matrix M can be computed from Equation 7

$$R_1 = M R_2 \implies M = R_1 R_2^{-R} \quad (7)$$

where $R_1 = [D^n D^{n-1} D^{n-2} \dots D^4 D^3]$ and $R_2 = [D^{n-1} D^{n-2} D^{n-3} \dots D^3 D^2]$.

The Spectrum Mapping Theorem (DeCarlo 1989) shows that the eigenvalues of M , say λ_1^M and λ_2^M , have the following relation with the eigenvalues of matrix A : $\lambda_1^M = e^{\lambda_1 T}$ and $\lambda_2^M = e^{\lambda_2 T}$. Therefore, $\lambda_1 = \frac{1}{T} \ln(\lambda_1^M)$ and $\lambda_2 = \frac{1}{T} \ln(\lambda_2^M)$. The eigenvalues are the roots of the characteristic polynomial of the A-matrix, as in Equation 5, which is

$$\pi_A(\lambda) = \det[\lambda I - A] = \lambda^2 + \frac{\xi}{\hat{\gamma} s_c} \lambda + \frac{\zeta \hat{\omega}_f}{s_c^{(1+b)}} \quad (8)$$

Since ξ and ζ are the only unknowns at this time, they can be estimated by matching the roots of $\pi_A(\lambda)$ to λ_1 and λ_2 computed above.

The fast convergence presented by the algorithm increases the model applicability and accuracy (Cangussu 2003). Finally, a parametric control procedure is used to compute required changes in the model in order to converge to desired results according to time constraints (Cangussu *et al.* 2002b).

3.1. Model Validation

The state variable model of the STP has been validated using both static and dynamic techniques. An extreme case analysis has shown the model validity when the values of parameters such as the quality of the test process γ and the complexity of the software product s_c assume values ranging from very small to very large. A tensor product-based sensitivity analysis was another static technique used to validate the model (Cangussu *et al.* 2003). Initially, the sensitivity analysis has pointed out a flaw in the model that was subsequently correct. The final results were consistent with the expectations of a development process. Some of the results are: (i) consistency with Brooks' Law; (ii) the earlier the changes in the process, the more effective they are; and (iii) improvement of the quality of the process is a better alternative than increasing the size of the work force.

Regarding a dynamic validation, the deterministic model has already been successfully applied to many distinct projects. The model was first used on the error data reported by Knuth on the development of $T_E X$ (Knuth 1989). Though applied to a finished process, the experiment showed the model had produced results very similar to the ones presented by the error data. The results encourage the application of the model to an ongoing software test process in a large industrial project. The project was the translation of four million lines of Cobol to SAP/R3 code. Using data collected from the first 14 weeks, the model was used to predict a delay of 10 weeks to achieve the specified goals with no changes in the process. The feedback application predicted that an increase of 1.5 in the work force would be necessary to finish the project within the deadline. No actions were taken by the manager and the process finished with a delay of 12 weeks, showing an accuracy of 94.5% on the prediction of the total time to complete the project (Cangussu *et al.* 2002b).

After the case study at Razorfish, the model has been applied to eight distinct projects at Sun



Microsystems. The results of these case studies cannot yet be published because of proprietary reasons. However, the accuracy achieved with the application of the model ranges from 85 to 95% with regard to schedule predictions and control. The estimation of the initial number of defects has also provided encouraging results. The accuracy in those predictions range from 73 to 95% depending on how much data is used to make the predictions. As would be expected, the later the prediction (consequently more data is available), the better the accuracy (Cangussu 2003).

4. STOCHASTIC MODEL

The deterministic model described in Section 3 seems to be appropriated to control relatively well-defined test processes, where the level of unpredictability is not critical. However, in practice, test processes are subject to a variety of external disturbances. In addition, collected data is often noisy and prediction of the intermediate states of the process may be compromised depending on the level of inaccuracy of the data. Under such circumstances, one alternative would be the use of a stochastic model as presented in Equation 9. In this case, the prediction of the intermediate states of a system becomes an stochastic rather than a deterministic process.

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t) + G\eta(t) \\ y(t) = Cx(t) + D\varphi(t) \end{cases} \quad (9)$$

where $x(t)$ is the state vector, $y(t)$ is the output vector, $u(t)$ is the input, A , B , C , G , and D are coefficient matrices, and η and φ are two mutually independent noise sequences (Chen *et al.* 1995). Notice that the stochastic model is based on the same, already validated, fundamental assumptions as the deterministic model.

The inclusion of noise components represents the major difference between the deterministic and the stochastic model. The influence of randomly external disturbance is accounted for by the noise sequence $\eta(t)$ in Equation 9, whereas the inaccuracy of the collected/measured data is represented by the noise sequence $\varphi(t)$ in the output part of the same equation. These noise sequences represent the average expected perturbation and therefore do not account for specific events during the test process.

As addressed in Sections 4.3 and 4.4, the CMM level of the organization determines the level of disturbance to be inserted into the process.

4.1. Input Characterization

The input F_d in the deterministic model of Equation 5 was used to account for unforeseen perturbations. However, F_d was included, periodically, as the average perturbation from the previous time period(s) and therefore still characterizes a deterministic process. F_d was also used to model common disturbances in the test process, such as a training period, a migration of the product under test from the developers' to the users' environment, the replacement of an already tested component, etc. (Cangussu *et al.* 2002a). Since unforeseen perturbations are accounted for by the stochastic processes $\eta(t)$ and $\varphi(t)$, F_d is used here to account for foreseen perturbations as the ones mentioned above (Cangussu *et al.* 2002a). This appears to be a reasonable approach due to the fact that these disturbances can be modeled and they are, in many situations, anticipated/expected. As before, the system is not driven by the input $u(t)$ and a parametric control technique can still be applied.

For example, assume the test process cannot be conducted, from the beginning, at the user's environment. The process starts at the developer's environment and the migration will be allowed at a known time period. Since the time is known in advance, the effect of the migration on the outcome of the test process can be predicted. The prediction of the results, using the state variable model, at three different time periods can be seen in Figure 2. As expected, the later the migration is performed, the larger the effect of the disturbance (Cangussu *et al.* 2002a). More results on the characterization of the input can be found elsewhere (Cangussu *et al.* 2002a).

4.2. Probabilistic Distribution Choice

Experiments with distinct probabilistic distributions such as gamma, normal, exponential, and uniform were conducted in order to determine which distribution better fits the characterization of the two stochastic processes used in the model. Let Δ be the distance between the expected curve and the disturbed curve for a given process. For each probabilistic distribution and a disturbance

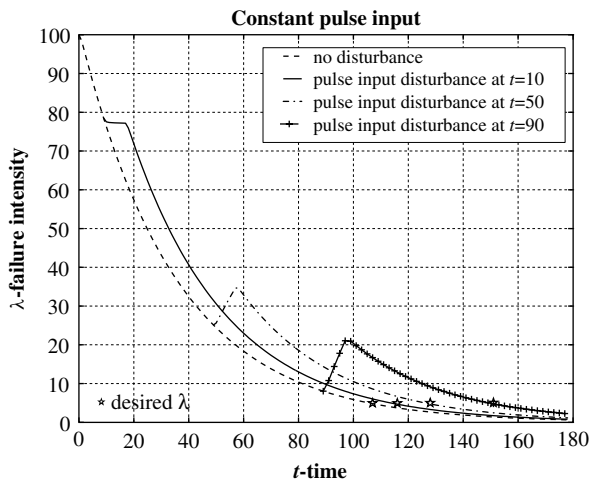


Figure 2. Results of the perturbation of the process due to the migration of the system from the development to the user's environment. A pulse input signal is used to represent the disturbance. (Cangussu *et al.* 2002a)

level ranging from 0 to 4, 50 simulation runs are executed and the average Δ_{avg} value computed.

With regard to the unforeseen perturbation represented by η , all the distributions resulted in an S-shaped format, starting from zero and stabilizing at $\Delta_{avg} \approx 70$, as the parameters from the distributions increased from 0 to 4. Obviously, an extra parameter related to the standard deviation is need for the normal and the gamma distribution. The difference between the results is how fast and smooth each distribution converges from zero to 70. Two points must be taken into consideration when analyzing the Δ_{avg} values. First, according to the experiments, values larger than 35 lead to unreasonable processes, even when considering organizations at CMM level 1. Second, the faster the Δ_{avg} change, the harder becomes to map disturbance intervals into the five CMM levels. The uniform distribution has presented a slower and smooth increase than all the other distributions used and therefore is selected here to represent the stochastic process η .

The same experiments are conducted to select the distribution to represent the stochastic process φ . Again, the uniform distribution presents the best alternative. The only difference in this case is that Δ_{avg} does not increase according to an S-shape function but rather as an exponential function. Owing to its small increase and saturation value, the uniform distribution appears to be a better

alternative to characterize the noise in the data collection process.

4.3. Disturbance Characterization

As stated above, $\eta(t)$ and $\varphi(t)$ account for unforeseen perturbations. Both of these stochastic processes are represented here by uniform and randomly distributed white noise sequences, as discussed in Section 4.2. The major concern becomes how to characterize the level of the disturbance in order to properly represent the ones found in different software test processes. The maturity of the organization/group where the process is being conducted is a good indication of the level of disturbance expected. The levels of the Capability Maturity Model (Paulk *et al.* 1993) presented in Section 2.2 can be used for this purpose.

Matrix G in Equation 9 determines the level of the disturbance inserted into the system. That is, $G = \begin{bmatrix} D_l \\ 0 \end{bmatrix}$, where D_l , ranging from zero to two, represents the disturbance level. The disturbance is inserted by perturbing the velocity component ($G_{1,1} = D_l$) of the model, which will, in a chain reaction, perturb the acceleration component. In this case, the direct perturbation of the acceleration component is not necessary and therefore $G_{2,1} = 0$. The stochastic version of the model is achieved by adding the term $G \times \eta(t)$ to Equation 5.

Now, let r_e be the expected decay of number of errors and r_d be the disturbed decay. The distance between these two curves is computed as $\Delta = |r_e - r_d|$. Figure 3 depicts how the average distance Δ changes as the disturbance level increases from 0 to 2 for 50 simulation runs. As can be observed in Figure 3, Δ increases exponentially as the disturbance level increases. It should be clear that Figure 3 is the initial part of an S-shaped plot that converges to $\Delta \approx 70$. In the case of Figure 3, disturbance values greater than 2 violate the assumption of an exponential decay and are disregarded here. One alternative would be to set the disturbance level at equal intervals of 0.4 for each CMM level, i.e. $(5 - L_i) \times 0.4 < D_l \leq (6 - L_i) \times 0.4$, where $L_i = 1, 2, \dots, 5$ represents the corresponding CMM levels. The distance Δ associated with each level under the equal interval assumption is observed in Figure 3.

At CMM Level 1, the ad hoc characteristics of the process does not allow a reasonable prediction

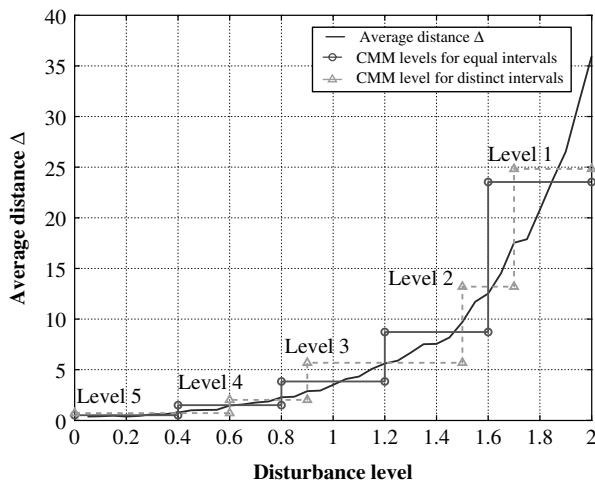


Figure 3. Average disturbance levels associated with each CMM level

of the results of a project and therefore the level of disturbance can be determined as high. A considerable improvement is expected as an organization moves from Level 1 to Level 2 and, therefore, a reasonable decrease in the disturbance level is expected. The exponential shape of the Δ curve captures this behavior. The improvement from Level 2 to Level 3, and consequently the decrease of the disturbance level, is supposed to be large but not as large as from Level 1 to 2. Again, the equal interval assumption appears

to be a good solution. The problem arises when considering the improvement from Level 3 to 4. Owing to the qualitative and quantitative goals, and the measurement scheme to achieve them, a reasonable decrease in the disturbance level is expected and, as observed from Figure 3, the equal interval assumption does not properly represent this behavior.

A larger disturbance gap between Levels 3 and 4 is desired. A definition of the intervals as presented in Table 1 appears to achieve the expected differences as an organization moves between levels. As observed from Figure 3, a large difference exists between Levels 1 and 2 and also a larger gap is observed between Levels 3 and 4. Though some decrease in the disturbance level is expected from Level 4 to 5, the decrease is not foreseen to be large. Improvement in performance due to a constantly optimization of the process is the major impact at Level 5 and a large decrease in disturbance is not a consequence of this.

Figure 4 shows the results of applying disturbance levels within the ranges specified in Table 1 for each of the CMM levels. It can be observed that the behavior of the process moves from chaotic at Level 1 to a reasonably stable and predictable process at Level 5. The increase of the number of remaining defects over time may be due to, for example, the insertion of more defects when fixing the detected ones. The test process in the plots

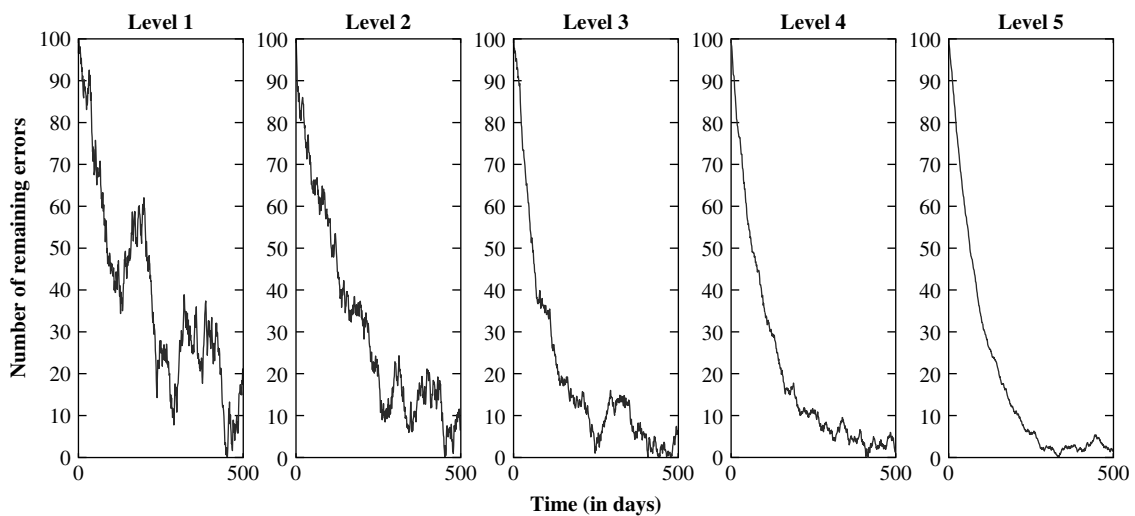


Figure 4. Unforeseen disturbance associated with each level of the Capability Maturity Model. The disturbance is generated by the randomly distributed white noise sequence η in Equation 9



Table 1. Disturbance and noise level associated with levels of the Capability Maturity Model used to generate the results in Figures 4 and 6. The values have a natural zero lower limit and the upper limits were determined to allow the number of remaining defects to converge, over time, to a small range

CMM level	Disturbance level	Noise level
5	$0 < D_i \leq 0.6$	$0 < N_i \leq 4$
4	$0.6 < D_i \leq 0.9$	$4 < N_i \leq 8$
3	$0.9 < D_i \leq 1.5$	$8 < N_i \leq 12$
2	$1.5 < D_i \leq 1.7$	$12 < N_i \leq 16$
1	$1.7 < D_i \leq 2$	$16 < N_i \leq 20$

is characterized by the following parameter values: $w_f = 2$, $s_c = 30$, $\gamma = 0.4$, $b = 1.12$, $\xi = 100$, and $\zeta = 60$. A sensitivity analysis of the model under the presence of disturbance can provide information of how changes in the parameters affect the behavior of the model. However, such analysis is outside the scope of this paper.

In addition, disturbances seem to have different levels, according to different time periods, for distinct projects. For example, the disturbance level for a test process where the test team is not familiar with the product under test will differ from a process, within the same organization, where the test team has previous experience with similar products. The interval associated with each CMM Level can be used to represent/adjust these differences.

4.4. Noise Characterization

The stochastic process $\varphi(t)$ in Equation 9 represents noise in the data collection process. Unreported errors, duplicated reports, and missing information such as date when the error was found are a few examples of problems/noise when collecting data. These are common problems that are likely to be present in any organization. However, the more mature the process and the organization, the less the likelihood and the frequency of these problems. Again, the CMM levels are used here in association with noise levels.

Matrix D in Equation 9 determines the level of the noise associated when measuring the output of the system. One output variable $r(t)$ is present in Equation 6 and therefore the dimension of matrix D is 1×1 . In this case, the single element of D , referred hereafter as N_i , accounts for the noise level. The addition of the term $D \times \varphi(t)$ to Equation 6 leads to its stochastic version.

Unlike the disturbance, the average distance Δ (computed as before) related to the noise level does not present an exponential increase. The increase, as observed from Figure 5, is linear for noise levels less than 20, although it presents an overall exponential behavior as previously discussed in Section 4.2.

Though it may not be true for organizations at CMM Level 1 or 2, let us assume that a data collection process is in place. As the maturity level of an organization or development group increases, the presence of standards, control mechanisms, quantitative measurements, and so on leads to a reduction on the noise in the data collection. However, the noise associated with reporting errors or defects is not expected to have a large impact on the behavior of the model. This can be noticed by the true nominal value of the average distance Δ in Figure 5. In addition, the characteristics of the CMM levels do not give an indication of a specific pattern in the increase of noise. Under these conditions, as shown in Figure 5 and Table 1, an equal interval for the noise associated with the CMM levels is assumed.

Figure 6 shows the results of inserting noise related to the data collection process in the output of the model. As can be observed, the perturbation of the output decreases linearly as the process moves from Level 1 up to Level 5. The process in the plot is characterized by the same parameters used in Figure 4. As before, a sensitivity analysis of the model can be used to collect information on

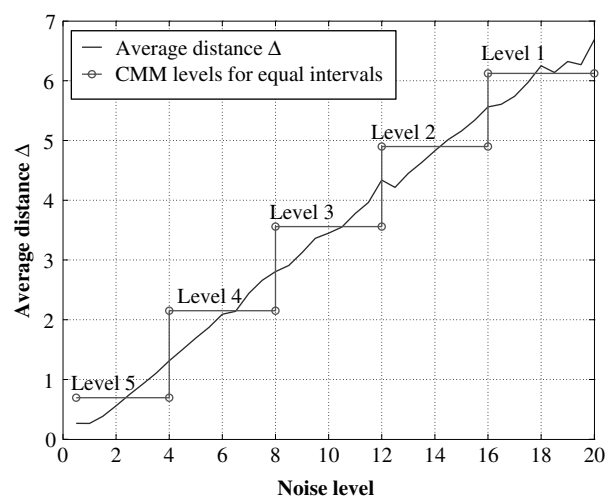


Figure 5. Average noise levels associated with each CMM level

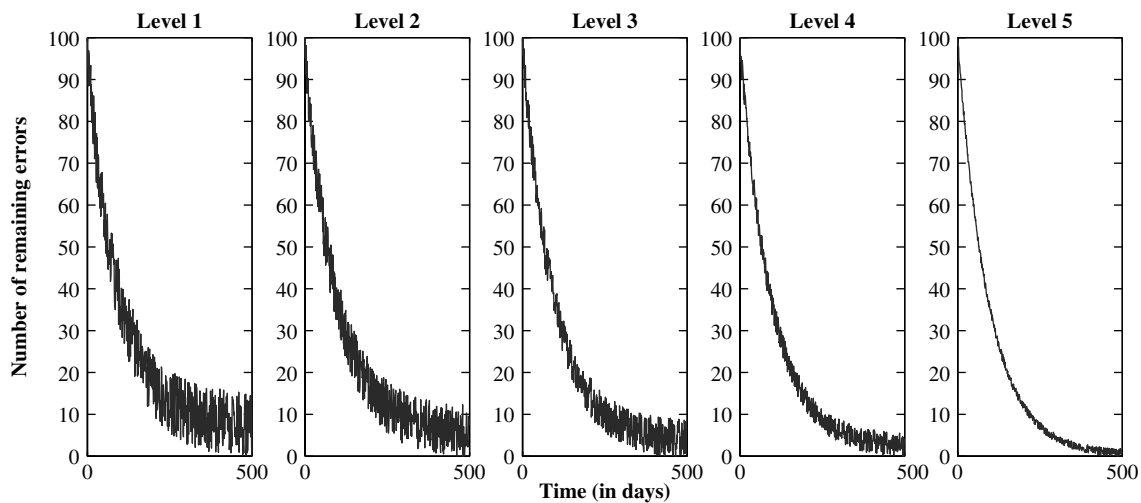


Figure 6. Noise in the data collection process associated with each level of the Capability Maturity Model. The noise is generated by a randomly distributed white noise sequence φ in Equation 9

the behavior of the model when noise is present. However, this issue is not addressed here.

The results of a combined insertion of data collection noise and unforeseen perturbations in a test process, as expected, produce results that appear as the merge of Figures 4 and 6.

Here, the noise sequences η and φ are represented by a uniform and randomly white noise sequence. However, under certain circumstances these sequences are likely to present not only a specific distribution but also specific trends. In certain cases, the effects of a disturbance are expected to increase as time progresses and such correlation needs to be taken into consideration when modeling the disturbance.

Another aspect to be considered is the distribution related to CMM levels. A uniform distribution may be appropriated to Levels 4 and 5 while a normal distribution may better represent the disturbance at Levels 1, 2, and 3. In the second case, the parameters μ and σ can be used to distinguish each CMM level. The study of probabilistic distributions that better represent the disturbance and noise sequences of the stochastic model in Equation 9 is a subject of future investigation.

4.5. Noise Filtering

The next step in improving the applicability of the model is the use of a Kalman filter. The filter allows a better prediction of the states of the system in the

presence of noise. Grewal and Andrews state "... it is not possible or desired to measure every variable you want to control, and the Kalman filter provides a mean for inferring the missing information from indirect (and noisy) measurements." (Grewal and Andrews 2001).

The result of the application of a Kalman Filter (Grewal and Andrews 2001) in the stochastic version of the state variable model of the software test process is presented in Figure 7. It can be inferred from Figure 7 that the Kalman filter has a great impact on the noise associated with the data

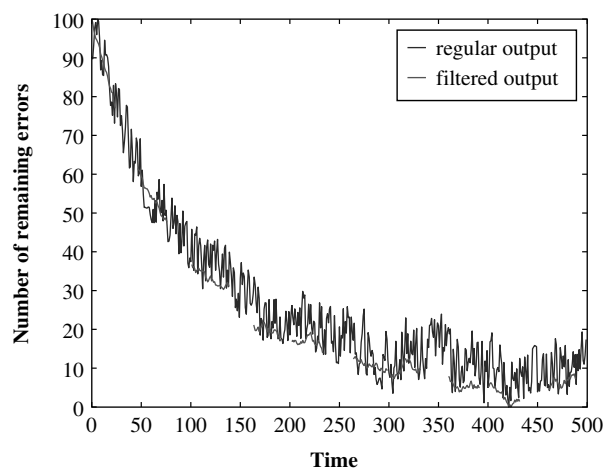


Figure 7. Application of a Kalman filter to a noisy test process



collection process but does not affect the unforeseen perturbations as much. This behavior of the filter properly represents the expected results, i.e. the noise is reduced but the effects of the perturbations are still considered. Changes in the overall structure of the feedback model of the STP are required for the application of the filter and a detailed explanation of these changes are beyond the scope of this paper.

5. SIMULATION RESULTS

One of the results of the simulation runs is depicted in Figure 8. The process in this Figure is typified by the same parameters defined in Section 4.3. In Figure 8, we can observe the results of the undisturbed deterministic model (dashed-dotted line) representing the expected decay of errors with a goal of reducing the number of remaining errors to 15% within 180 days. The result of the introduction of a disturbance sequence level $D_i = 1.3$ and a noise sequence level $N_i = 9$ is also shown in Figure 8. In this case, according to Table 1, a process at CMM Level 3 is characterized. As can be observed, the goal of the process cannot be achieved under the presence of disturbance.

Assuming that the first checkpoint is at time $t = 50$, it can be observed that the process is not proceeding according to the expected behavior. If we apply the control approach used for the deterministic model (Cangussu *et al.* 2002b), an increase of $\Delta w_f = 0.5$ (a half time tester) places

the eigenvalues of the system at the desired level. However, the disturbances are not accounted for and the process does not converge to the expected curve. A group/organization is unlikely to improve the CMM Level in the middle of a process and therefore disturbances are expected to keep on slowing down the process.

The parametric control approach used here is done at two levels. The first step is to calibrate the model to incorporate the disturbance into the parameters ζ and ξ . The result of the calibration is represented by the dashed line in Figure 8. Then, the changes needed to place the eigenvalues of the calibrated model and make the results converge to the expected curve are applied not to the calibrated but to the disturbed model. The changes in the calibrated model are already accounting for disturbances and therefore will have a similar impact in the disturbed model, as can be observed in Figure 8. An increase of $\Delta w_f = 5.5$ is required to achieve the desired goal within the deadline. As one could argue, side effects of increasing w_f (Brook's Law (Brooks 1995)) are expected and are analyzed elsewhere (Cangussu *et al.* 2003). Finally, the changes to drive the system to the desired behavior were presented here in terms of w_f . However, the same approach applies to changes in the quality of the test process γ or to combined changes of w_f and γ .

6. RELATED WORK

A variety of techniques has been proposed to model and/or control one or more phases of the SDP. Ghezzi's work provides a general view of its evolution (Cugola and Ghezzi 1998). However, no technique can individually fulfill all the desired characteristics to model and control a SDP. Here a brief discussion of a few techniques that are more related to the state variable approach (Cangussu *et al.* 2002b) is presented. Some of the techniques, such as Statistical Process Control (Florac and Carleton 1999), are more process control oriented. Others, such as Software Project Dynamics (Abdel-Hamid and Madnick 1991), are modeling and prediction techniques. Software Process Simulation (Hansen 1996) presents modeling and prediction with restricted control capabilities.

Software Project Dynamics, as presented by Abdel-Hamid and Madnick, provides a macro view

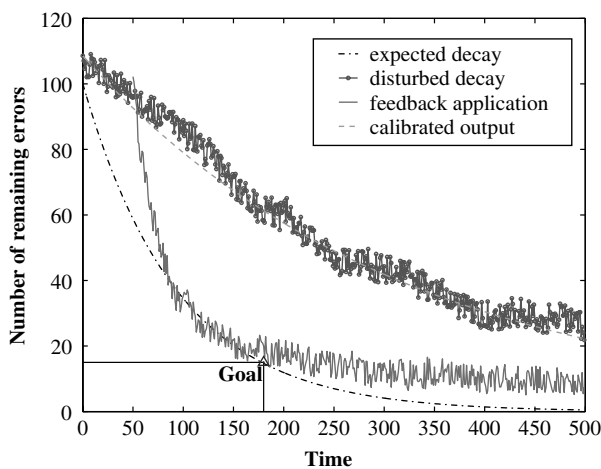


Figure 8. Result from feedback application of the Stochastic State Model for the Software Test Process



of the SDP through the integration of micro components (Abdel-Hamid and Madnick 1991). Consequently, predictions of the general behavior of the process are achieved by propagating local changes in the micro components through the system. This propagation does not have a self-regulation mechanism and therefore can be characterized as an open-loop feedback approach. The lack of a closed-loop feedback solution and a well-defined parameter calibration algorithm represents the main differences between Software Project Dynamics and state variable approach.

Statistical Process Control (Florac and Carleton 1999, Wheeler and Chambers 1992) is used to monitor an ongoing process. Data is collected to determine if the process is in a stable or unstable condition. In the case of an unstable process, the process manager has to make changes to stabilize it. If the process is stable, capability is analyzed to determine if the process will be capable of producing the desired results within a prespecified deadline. Again, Statistical Process Control lacks a quantitative feedback mechanism to suggest corrections to make the process stable or capable. This task relies on the process manager's experience and expertise.

Discrete event simulation techniques are commonly used to model and evaluate the SDP (Hansen 1996). The control capability of Software Process Simulation is restricted to answer 'what if' questions and therefore does not present a self-regulation mechanism. Optimization can be achieved using simulation, but the number of possible combinations of changes in the parameters of the model can reach high values, making simulation a very costly technique, although capable of achieving results similar to the state variable approach.

An argument could be made that the state variable approach would demand a reasonable level of specific training for a user without a control theory background. Indeed, it would be unreasonable to expect software managers to cope with the mathematical details of the model. However, the development of a tool hiding all the mathematical details of the technique and providing a friendly interface for data entry can significantly reduce the training period. Such period would be restricted to the time a user needs to familiarize with the interface.

7. CONCLUDING REMARKS

The introduction of noise sequences in the state variable model of the software test process extends its use to organizations in the lower CMM levels. Similarly, to the deterministic model, it allows the computation of changes in process parameters to correct for schedule deviations in the process. In addition, statistical information allowing inferences on the accuracy of the predictions related to the level of perturbation injected into the model can be acquired.

The results presented here are an indication of the applicability of the stochastic state model of the STP. However, actual data is needed to dynamically validate the model. This validation requires the identification of different types of disturbances and the quantification of their side effects on the progress of the process. Working closely with test managers is a requirement to obtain success in this task. The noise in the data collection process does not demand such individual analysis and can be done in a more generalized manner. Additionally, the application of a Kalman filter decreases the effects of the noise. Available data can be further used to analyze specific probabilistic distributions associated with the CMM levels.

REFERENCES

- Abdel-Hamid T, Madnick SE. 1991. *Software Project Dynamics: An Integrated Approach*. Prentice Hall Software Series: Upper Saddle River, NJ.
- Brooks FP. 1995. *The Mythical Man-month: Essays on Software Engineering*. Addison Wesley: Reading, MA.
- Cangussu JW. 2003. Convergence assessment of the calibration algorithm for the state variable model of the software test process. Proceeding of the IASTED International Conference on Software Engineering (SE'03), Innsbruck, Austria.
- Cangussu JW, DeCarlo RA, Mathur AP. 2001a. A formal model for the software test process. Technical Report SERC-TR-176-P, Purdue University-SERC, IN, USA.
- Cangussu JW, DeCarlo RA, Mathur AP. 2001b. Sensitivity analysis of a state variable model of the software test process. Proceedings of the 2001 IEEE Systems, Man, and Cybernetics Conference (SMC 2001), Tucson, AZ, 712–717.
- Cangussu JW, DeCarlo RA, Mathur AP. 2001c. A state model for the software test process with automated



parameter identification. Proceedings of the 2001 IEEE Systems, Man, and Cybernetics Conference (SMC 2001), Tucson, AZ, 706–711.

Cangussu JW, DeCarlo RA, Mathur AP. 2002a. Effect of disturbances on the convergence of failure intensity. Proceeding of 13th International Symposium on Software Reliability Engineering, Annapolis, MD.

Cangussu JW, DeCarlo RA, Mathur AP. 2002b. A formal model for the software test process. *IEEE Transaction on Software Engineering* **28**(8): 782–796.

Cangussu JW, DeCarlo RA, Mathur AP. 2003. Using sensitivity analysis to validate a state variable model of the software test process. *IEEE Transactions on Software Engineering* **29**(5): 430–443.

Cass AG, Lerner BS, McCall EK, Osterweil LJ, Sutton SM, Wise A. 2000. Little-jil/juliette: a process definition language and interpreter. *Proceedings of the 22nd International Conference on Software Engineering (ICSE 2000)*, IEEE: Limerick, Ireland, 754–757.

Chen G, Chen G, Hsu S-H. 1995. *Linear Stochastic Control Systems*. CRC Press, Inc.: Boca Raton, FL.

Cugola G. 1998. Tolerating deviations in process support systems via flexible enactment of process models. *IEEE Transactions on Software Engineering* **24**(11): 982–1001.

Cugola G, Ghezzi C. 1998. Software processes: a retrospective and a path to the future. *Software Process Improvement and Practice* **4**(2): 101–123.

DeCarlo RA. 1989. *Linear systems: A State Variable Approach with Numerical Implementation*. Prentice Hall: Upper Saddle River, NJ.

Eickelmann NS. 2001. Applying simulation technologies to manage test practices. Proceedings of the 18th International Conference on Testing Computer Software, Bethesda, MD.

Florac WA, Carleton AD. 1999. *Measuring the Software Process: Statistical Process Control for Software Process Improvement*. SEI Series in Software Engineering. Addison-Wesley: Reading, MA.

Goodwin GC, Graebe SF, Salgado ME. 2001. *Control System Design*. Prentice Hall: Upper Saddle River, NJ.

Grewal MS, Andrews AP. 2001. *Kalman Filtering: Theory and Practice Using MATLAB*, 2nd edn. John Wiley & Sons: New York.

Hansen GA. 1996. Simulating software development processes. *IEEE Computer* **29**(1): 73–77.

Juang J-N. 1993. *Applied System Identification*, 1st edn. Prentice Hall: Englewood Cliffs, NJ.

Knuth DE. 1989. The errors of $T_E X$. *Software-Practice and Experience* **19**(7): 607–685.

Ljung L. 1987. *System Identification: Theory for the User*. Prentice Hall: Englewood Cliffs, NJ.

Luenberger DG. 1979. *Introduction to Dynamic Systems: Theory, Models and Applications*. John Wiley & Sons: New York.

Paulk MC, Curtis B, Chrissis M, Weber C. 1993. Capability maturity model for software. Technical report: CMU/SEI-93-TR-024, Pittsburgh, PA, USA.

Wheeler DJ, Chambers DS. 1992. *Understanding Statistical Process Control*. SPC Press: Knoxville, TN.