

# A Stochastic Control Model of the Software Test Process

João W. Cangussu  
Department of Computer Science  
University of Texas at Dallas  
Richardson - TX  
75083-0688, USA  
(972)883-2193  
cangussu@utdallas.edu

## 1 Introduction

The understanding and control 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. The software development process has been modeled using different techniques ranging from finite state machine, process language, and simulation based models among others [1, 2, 3, 4, 5, 6]. Recently, a state variable approach has also been used to model and control the system test phase of the software development process [7, 8]. 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.

Despite the difficult in creating mathematical models, plausible accurate models have been derived for physical and non-physical systems [9, 10]. However, the difficult level arises when modeling non-physical system due to the fact that observation/measurement of these processes is not accurate, more specifically, they are subject to disturbances and noise data. Under these circumstances, a stochastic rather than a deterministic model appears to be a better solution to represent the process. 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 2. 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.

## 2 Previous Results: Deterministic Model

The linear deterministic model of the STP is based upon three assumptions. These assumptions and the corresponding equations are presented below [7]. They 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 [10]. A description and justification of this analogy and the choice of a linear model is outside the scope of this paper [7]. The model has been validated using sets of data from testing projects [11] and also by means of a extremal case and sensitivity analysis [12, 13].

**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.,

$$\ddot{r}(t) = \frac{e_n(t)}{s_c} \Rightarrow e_n(t) = \ddot{r}(t) s_c \quad (1)$$

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

$$e_f(t) = \zeta(s_c) w_f r(t) \quad (2)$$

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 ( $\gamma$ ) of the test phase, i.e., for an appropriate constant  $\xi$ ,

$$e_r(t) = -\xi \frac{1}{\gamma} \dot{r} \quad (3)$$

Combining Eqs. 1, 2, and 3 in a force balance equation and organizing it in a State Variable format ( $\dot{x} = Ax + Bu$ ) [9, 10] produces the following system of equations.

$$\begin{bmatrix} \dot{r}(t) \\ \ddot{r}(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ \frac{-\zeta w_f}{s_c^{(1+b)}} & \frac{-\xi}{\gamma s_c} \end{bmatrix} \begin{bmatrix} r(t) \\ \dot{r}(t) \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{s_c} \end{bmatrix} F_d(t) \quad (4)$$

$F_d$  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 Eq. 4 an algorithm has been developed to calibrate the parameters of the model [14]. The fast convergence presented by the algorithm increases the model applicability and accuracy [15]. 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 [7].

### 3 Stochastic Model

The deterministic model in Eq. 4 has the general format of Eq. 5.

#### Deterministic Model

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t) \\ y(t) = Cx(t) \end{cases} \quad (5)$$

where  $x(t)$  is the state vector,  $u(t)$  is the input, and  $A$ ,  $B$ , and  $C$  are coefficient matrices [9, 10].

#### Stochastic Model

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t) + \eta(t) \\ y(t) = Cx(t) + \varphi(t) \end{cases} \quad (6)$$

where  $x(t)$  is the state vector,  $u(t)$  is the input,  $A$ ,  $B$ , and  $C$  are coefficient matrices, and  $\eta$  and  $\varphi$  are two mutually independent noise sequences [16].

The input  $u(t)$  is, in general, used to drive the system. However, the procedure used for the control of the system from Eq. 4 uses a parametric approach and the input is used to account for unforeseen perturbations.

The deterministic model seems to be appropriated to control 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 stochastic model as represented by Eq. 6. In this case, the prediction of the intermediate states of a system becomes an stochastic rather than a deterministic process.

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 Eq. 6 whereas the inaccuracy of the collected/measured data is represented by the noise sequence  $\varphi(t)$  in the output part of the same equation.

The input  $F_d$  in the deterministic model of Eq. 4 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 characterize 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. [17]. For the stochastic model,  $F_d$  will account for foreseen perturbations as the ones just described. 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 state above,  $\eta(t)$  and  $\varphi(t)$  will account for the unforeseen perturbations.

The major concern becomes how to characterize the disturbance to proper represent the ones find in different software test processes. The maturity of the process is a good indication of the level of disturbance expected in a specific organization. The Capability Maturity Model [18] can be used for this purpose. In addition, disturbances seem to have different levels, according to different time periods, for distinct project. For example, the disturbance level for a test process where the test team is not familiar with the product under test will differ from the a process, within the same organization, where the test team has previous experience with similar products.

The introduction of noise sequences in the state variable model of the software test process extends its applicability 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, it provides statistical information allowing inferences on the accuracy of the predictions related to the level of perturbation injected into the model.

## References

- [1] T. Abdel-Hamid and S. E. Madnick, *Software Project Dynamics: An Integrated Approach*. Prentice Hall Software Series, New Jersey, 1991.
- [2] G. Cugola and C. Ghezzi, "Software processes: A retrospective and a path to the future," *Software Process Improvement and Practice*, 1999.
- [3] G. Cugola, "Tolerating deviations in process support systems via flexible enactment of process models," *IEEE Transactions on Software Engineering*, vol. 24, November 1998.
- [4] A. G. Cass, B. S. Lerner, E. K. McCall, L. J. Osterweil, S. M. Sutton, and A. Wise, "Little-jil/juliette: A process definition language and interpreter," in *Proceedings of the 22nd International Conference on Software Engineering (ICSE 2000)*, (Limerick, Ireland), pp. 754–757, IEEE, June 2000.
- [5] G. A. Hansen, "Simulating software development processes," *IEEE Computer*, vol. 29, pp. 73–77, January 1996.
- [6] N. S. Eickelmann, "Applying simulation technologies to manage test practices," in *Proceedings of the 18th International Conference on Testing Computer Software*, (Bethesda, MD), June 18-22 2001.

- [7] 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.
- [8] J. W. Cangussu, R. A. DeCarlo, and A. P. Mathur, "A formal model for the software test process," Tech. Rep. SERC-TR-176-P, Purdue University-SERC, March 2001.
- [9] G. C. Goodwin, S. F. Graebe, and M. E. Salgado., *Control system design*. Upper Saddle River, New Jersey: Prentice Hall, 2001.
- [10] D. G. Luenberger, *Introduction to Dynamic Systems: Theory, models and applications*. John Wiley & Sons, 1979.
- [11] J. W. Cangussu, *A Mathematical Foundation for Software Process Control*. PhD thesis, Purdue University, April 2002.
- [12] 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. (to appear).
- [13] J. W. Cangussu, R. A. DeCarlo, and A. P. Mathur, "Sensitivity analysis of a state variable model of the software test process," in *Proceedings of the 2001 IEEE Systems, Man, and Cybernetics Conference (SMC 2001)*, (Tucson, Arizona), pp. 712–717, October 2001.
- [14] J. W. Cangussu, R. A. DeCarlo, and A. P. Mathur, "A state model for the software test process with automated parameter identification," in *Proceedings of the 2001 IEEE Systems, Man, and Cybernetics Conference (SMC 2001)*, (Tucson, Arizona), pp. 706–711, October 2001.
- [15] J. W. Cangussu, "Convergence assessment of the calibration algorithm for the state variable model of the software test process," in *Proceeding of the IASTED International Conference on Software Engineering (SE'03)*, (Innsbruck, Austria), February, 10-13 2003.
- [16] G. Chen, G. Chen, and S.-H. Hsu, *Linear Stochastic Control Systems*. CRC Press, Inc., 1995.
- [17] J. W. Cangussu, R. A. DeCarlo, and A. P. Mathur, "Effect of disturbances on the convergence of failure intensity," in *Proceeding of 13<sup>th</sup> International Symposium on Software Reliability Engineering*, (Annapolis, MD, USA), November 12-15 2002.
- [18] M. C. Paulk and et. al., "Capability maturity model for software," tech. rep., Software Engineering Institute, Carnegie Mellon University, 1993.