

# Software Test Process Control: Status and Future Directions

Ghaffari Abu Leeny Chacko João W. Cangussu  
Department of Computer Science  
University of Texas at Dallas  
{gabu,leeny\_chacko,cangussu}@utdallas.edu

## Abstract

*The modeling of the software test process using a state variable approach and the use of control theory concepts to regulate the process and calibrate the model has proven to be a valuable alternative for software process control. In this paper a description of the current status of the technique along with recent model improvements are presented. The architecture of a web-based tool to hide the complexity of the model and algorithms and increase the applicability of the technique is also presented. Finally, the directions of future research toward improvement in the accuracy of the model and in the calibration algorithm are described.*

## 1 Introduction

The state variable model of the Software Test Process (STP) combined with the parametric control approach and the least square based calibration algorithm [1] has brought the management of a test process to a more predictable, manageable, and quantitative level. In addition to static analysis [2], the application of these techniques to several projects at a large industrial corporation has provided enough evidence of the capability and accuracy of the model.

In spite of the good results achieved with the first model, further improvement in the model and the calibration algorithm are foreseen. The recent addition of learning aspects to the model has shown to increase its accuracy. The development of a web-based interface to host the model and algorithms are under its way to increase its applicability. The results have also shown that more aspects related to the test process need to be taken into account and that the estimation of the initial number of defects need to be improved.

The state variable model approach [1] applies concepts of control theory to regulate the testing phase of the software development process. Similarly, the work of Cai [3] also applies control concepts to regulate the testing process. While the first approach achieves its goals by changing pa-

rameters associated with the development process the second focus on improving the process by optimizing test case selection based on a Controlled Markov Chain. Both approaches are of special interest to the area of Software Cybernetics [4] that encompass the interplay of many different aspects of software and control theory.

The goal of this paper is to present the current status of the modeling of the STP (Section 2) along with recently developed improvements (Section 3) and the future directions (Section 4) for the software test process control.

## 2 Current Status

A second order model of the STP is used to regulate the process. Three main aspects are considered in the model: (i) the complexity of the product under test, (ii) the size of the work force, and (iii) the quality of the test process in place. Assumptions based on each of these aspects provide the mathematical description of the relationship between them [1].

The complexity of the product is taken into consideration by the observation that a more complex product demands more effort to reduce the number of defects at a specific rate. Increase in the rate of defect reduction can be achieved by increasing the effort and/or decreasing the friction<sup>1</sup> created by the process itself. The increase in effort means an increase in the size of the work force with a non-linear relation. That is, doubling the work force does not lead to a double increase in the rate of defect detection. This relation is analogous to the predator-prey model where the number of possible encounters between a predator (tester) and a prey (defect) is the product of their population. Each encounter decreases the prey population and consequently decreases the number of possible encounters which, in general, determines the exponential decay of the number of defects. The complexity also plays a role in this aspect. A very complex product represents an environment that reduces the possibility of an encounter between a predator (tester) and a prey

---

<sup>1</sup>resistance to defect removal

(defect). However, a low complex product represents an environment that facilitates such encounters. The friction aspect of the model is accounted by the inverse of the quality of the process and the rate of defect detection. A high quality process has a lower friction than a low quality process. Trying to detect defects faster than the test process allows also leads to increases in the friction. The combination of these three aspects and the corresponding equations lead to second order state variable model [1].

The regulation of the process is achieved by using a parametric control approach. Two parameters are under the control of a test manager: the size of the test team and the quality of the process. When a process is behind schedule and the desired defect reduction is specified, a pole placement approach is used to compute the required changes in the work force and/or the quality of the process to achieve desired results. The calibration algorithm is based on a least square approach to compute the state transition matrix and on the use of the spectrum mapping theorem to relate the eigenvalues of this matrix with the coefficient matrix for the model [5]. The model has been validated using techniques such as extremal case and sensitivity analysis [2] and has recently been successfully applied to several projects at a large corporation.

### 3 Recently Developed Improvements

Learning has been identified as an important factor in the STP. Its insertion into the STP model has shown to increase the model accuracy and a brief description of its modeling is presented in Section 3.1. The applicability of the model will have a major improvement once the development of a web-based interface for the STP control is finished. Section 3.2 addresses some design issues in the development of the interface.

#### 3.1 Learning

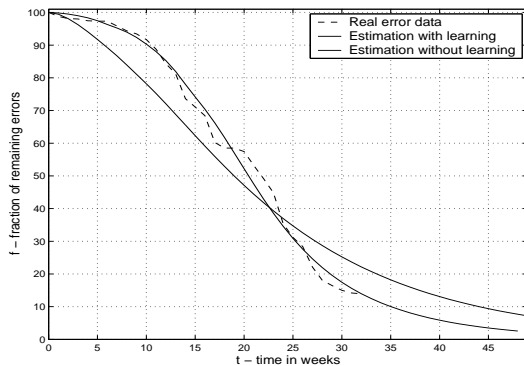


Figure 1. Effect of Learning model on STP.

Learning during Software Test Process is dependent on the initial knowledge, skills and experience of the testers. Acquired knowledge during software test process includes enhanced domain knowledge, better understanding of the requirements, and improved testing skills. Such enhancement in knowledge leads to increase in efficiency of the testing process and improved quality of the software for the next release/project. However, unlimited potential of such growth is restricted by the time of the project. Software life cycle ultimately limits the growth in learning and in some conditions may pragmatically lead to a saturation value.

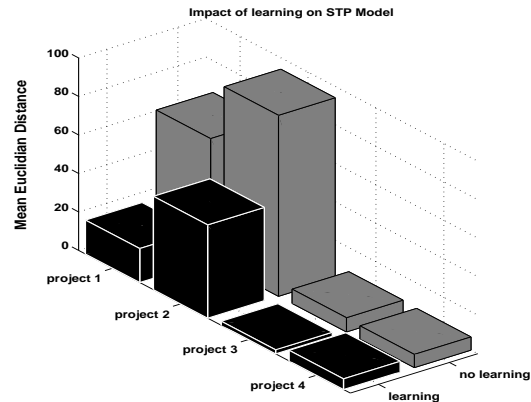


Figure 2. Average euclidean distance between the application of the State Model with and without learning. Projects 1, 2, and 3 show improvements around 50-70% and Project 4 shows a 20% improvement when using the model accounting for learning.

Figure 1 shows the effect of Learning on the STP according to our proposed model. The learning rate is proportional to the product of initial knowledge of the tester and remaining knowledge to be gained to reach the saturation value <sup>2</sup>. In a typical scenario, the initial learning is slow and gradual, which is governed by the initial knowledge and rate of learning of the tester. Later a sharp growth in learning with time occurs eventually reaching a certain saturation level during the software life cycle. Such typical behavior is captured into the model by taking into account various abstract qualities of each tester as well as the entire team. Existing time invariant software test process model when combined with time dependent learning behavior converts into a time variant state model. Figure 2 compares the improvement in accuracy of the proposed model with the existing model using data from large industrial projects<sup>3</sup>. The figure shows the average euclidean distance between actual data from the projects and the approximations from the previous and ex-

<sup>2</sup>saturation value is assumed to be 1 for normalized data

<sup>3</sup>source of data is not disclosed for proprietary reasons.

tended model. As can be seen, the results support the expected improvements due to the incorporation of learning into the model.

### 3.2 Web-Based Interface

The major barrier to the application of the STP state model has been the lack of a user friendly interface hiding all the mathematical details and presenting only the outcomes of the model. This would allow a manager with no knowledge on control theory to benefit from the use of the STP state model.

The STP state model and the corresponding parametric control and parameter calibration procedures have been implemented using MATLAB, more specifically, using functions from the control and the system identification tool boxes. A MATLAB GUI could be easily developed to provide a friendly interface for the users. However, this alternative requires the availability of MATLAB to the users. In reality, most managers lack such working environments. Furthermore, some knowledge of MATLAB would also be required. A better alternative is the implementation of a web-based interface that does not present any of these drawbacks. Two solutions are possible: (i) an in-house implementation of the MATLAB functions, and (ii) implementation of a communication between MATLAB and a Web-server. Due to its complexity, the first solution was promptly disregarded. The second solution has been adopted and a brief description of its architecture is presented below.



**Figure 3. Communication mechanism used in the implementation of the web-based interface for the STP state model.**

We were unable to find a direct way to invoke a MATLAB script from a web-browser. However, the PyMat module in the Python [6] language can be used to achieve this goal and make the communication mechanism transparent to the user. When a user selects one of the options of the interface, for example “Calibrate Model”, the Python implementation collects any data entered by the user and send it to an active MATLAB section that in turns execute the specified script and returns the results to the browser through Python. An overview of this solution is depicted in Figure 3. Though the entire system has not been completely developed, a prototype of the communication mechanism has already been implemented.

The availability of the approach becomes almost unlimited with the use of a web-based interface. However, com-

panies are very sensitive to defect data and security mechanisms need to be developed to guarantee the confidentiality of the data.

## 4 Future Directions

Two major aspects need to be addressed to increase the accuracy and the applicability of the state variable model of the STP. The first is the inclusion of features of the test process that are not accounted for in the current model. The second is the improvement of the calibration algorithm with regard to the estimation of the initial number of defects. These issues are addressed next.

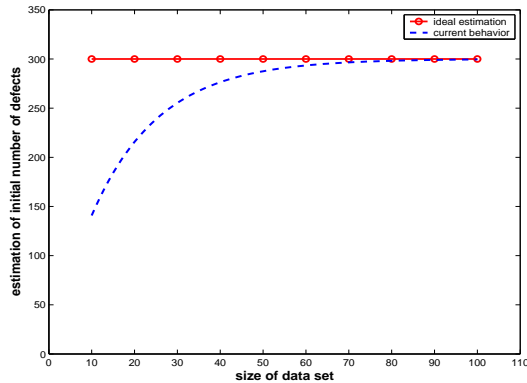
### 4.1 Improvement of the State Model

The inclusion of learning in the state model has increased its accuracy as can be observed from Figure 2. However, the complexity of the model has also increased. The attempt to linearize the model did not produce reasonable results and a time varying approach was used instead. As delineated by a sensitivity analysis, communication overhead is another aspect of the STP that may have a large impact on the outcomes of a testing process. Therefore, the next step to further improve the accuracy of the STP state model is the incorporation of communication overhead. In the case of learning, a large body of quantitative models were available and used as foundation for the development of the extended model. However, this does not appear to be the case for communication overhead where quantification has been marginally considered and the majority of the research has focused on qualitative aspects. This fact may impose additional difficulties in the modeling and in the validation process. In this case, collaboration with test managers may play a crucial role not only in acquiring data but mainly on assuring that the behavior of communication has been properly addressed.

The increase in the model complexity that the inclusion of communication will incur is justified by its importance to the process. Many other aspects such as risk evaluation, saturation effect, and maturity of the organization could also be included in the model. However, at this point, we believe that the trade-off between accuracy and model complexity may not justify such extensions. Further investigation need to be conducted to support this conjecture. Additionally, models for integration test and other phases of the development process could be construct but this is beyond the scope of this paper.

## 4.2 Improvement of the Parameter Calibration Algorithm

The calibration algorithm has two main purposes. The first is the estimation of the proportionality constants of the coefficient matrix  $A$  of the model and the second is the estimation of the initial number of defects. The algorithm has shown good results even when the calibration is done using a small sample of data points for the estimation of the constants. On average, the estimation converges to 85% of accuracy using approximately what would be 10% of the entire data set for a project [7].



**Figure 4. Ideal and current behavior of the calibration algorithm with respect to the estimation of the initial number of defects.**

With respect to the estimation of the initial number of defects  $R_0$ , the calibration algorithm does not provide the same outstanding results as in the case above. Figure 4 shows the general behavior of the algorithm and what would be the ideal result for an hypothetical process. For this process, let us assume that the actual number of defects is 300 and that the size of the data set at the end of the project would be 100 points. The ideal behavior of the calibration algorithm would be an accurate prediction of the initial number of defects with a small sample of the data set. However, due to many uncertainties in the test process such behavior is not realistic. The current implementation of the calibration algorithm produce results with a similar behavior as presented in Figure 4. It can be noticed that the accuracy of the estimation increases exponentially as more data points become available. It can also be notice that a gap between the actual number of defects and the initial estimations is large. Therefore, the calibration algorithm needs to be improved to decrease the size of the area between the two curves in Figure 4. Alternatives such as the use of a time dependent weighted least square approach along with additional process and project dependent information such as defect density are under investigation to achieve this reduc-

tion. However, it is still premature to determine the magnitude of the improvement.

## 5 Concluding Remarks

The results observed on the application of the STP state model to several projects in a large corporation are a strong evidence of its applicability and accuracy. However, the model can be further extended to incorporate additional aspects of the test process. As pointed out in Section 3.1, the inclusion of learning into the model has considerably increased its accuracy and other aspects of the process are also being considered. The applicability issue due to the complexity of the model is also being significantly improved through the implementation of a web-based interface. Finally, improvements in the algorithm for the estimation of the initial number of defects are being considered. These enhancements will further corroborate the STP state model as a powerful alternative for the control of a software test process.

## References

- [1] 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.
- [2] 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. 29, pp. 430–443, May 2003.
- [3] K.-Y. Cai, "Optimal software testing and adaptive software testing in the context of software cybernetics," *Information and Software Technology*, no. 44, pp. 841–855, 2002.
- [4] K.-Y. Cai, T. Y. Chen, and T. H. Tse, "Towards research on software cybernetics," in *7th IEEE International Symposium on High Assurance Systems Engineering*, pp. 240–241, IEEE, 2002.
- [5] R. A. DeCarlo, *Linear systems : A state variable approach with numerical implementation*. Upper Saddle River, New Jersey: Prentice-Hall, 1989.
- [6] M. Lutz, *Programming Python*. O'Reilly & Associates, second ed., March 2001.
- [7] 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.