Volume 4, No. 8, August 2013

Journal of Global Research in Computer Science

ISSN-2229-371X

REVIEW ARTICLE

Available Online at www.jgrcs.info

Operational Profile: A Critical Review

Pawan Kumar Chaurasia[1] and Jahid Ali[2]
Assistant Professor[1]
Department of Information Technology[1],
BBAUniversity, Lucknow-226025, India[1]
pkc.gkp@gmail.com
Professor[2]
SSICMT, Pathankot[2]
Punjab -145001, India[2]

Abstract—Technology is changed with the requirement of the user. There is a tremendous change in size, speed and structure of computer hardware and software. With these changes, complexity of computer system also increases with the time. As a result, different types of faults, bugs and failures has also increased in due coverage of time. Therefore, plenty of work has been done to test the failure using operational profile and numerous analysis software reliability models for estimating software reliability growth has been proposed. In order to assess their usability and importance, this paper does a critical review on the need and significance of a specifying operational and usage based modeling.

Keywords - Meant Time Between Failure [MTBF], Mean Time to Failure [MTTF], Mean Time to Repair [MTTR, Software Operational Profile [SOP], Markov Chain Model, State Hierarchy Model [SHY]

INTRODUCTION

Software testing is very costly for any organization, if it is not in a planned and systematic way to optimize the number of tests. So the first requirement is to find the number of test cases which are executed in various function/modules. The reliability of a software product is to estimate how the computer and other elements of the devices will used by user/developer. In 1993 Musa proposed a dynamic and innovative approach which is called Operational Profile(OP) which allocates test cases and estimate the the reliability of the software product. Software reliability is defined as "The probability for failure free operations of a program for a specified time under set of operating conditions in specific environment"[1]. Reliability of a system is to be accessed by test cases by testing the software product under simulated conditions. It covers methods, models and metrics of how to estimate and predict software reliability. There are different methods which are used to measure the reliability. It is measure by Mean Time Between Failures [MTBF] and defined as the Mean Time to Failure [MTTF] and Mean Time to Repair [MTTR][2]. Failure is the condition in which system fails to perform its required function. Reliability is the amount of time that software is available for use. The major problem in the field of software reliability estimation is the accuracy of the Operational Profile[3].
Software Operational Profile (SOP) is a quantitative description of software field usage. SOP consists of a set of software operations together with their occurrence probability [4]. An OP, guides testing, ensures that if testing is finished and the software is shipped because of imperative schedule constraints, the most used operations will have perform the most testing and the reliability level will be the utmost. It establish the communication between customers and developers. It also discuss about the features they would like to use. On the base of usage and usage specification models are defined. It helps to organize the work processes which are related to user processes and help the customer's training efforts towards the most-used operations. By using the operational profile and software reliability, reorganize the test and test conditions and improve the reliability and reduced the cost of the software.
This paper is organized in four sections. Section I, depicts about the introduction part of the software reliability and operational profile. In section II different types of profiles are discussed and describe in step-by-step. Different Usage models are defined in section III and conclusion of the paper with the future scope of the paper is presented.

PROFILE

When we develop any OP, several other profiles are also developed as required by the profile. A profile is simply a set of disjoint options with the probability that each will occur [5]. Profile is explained with the help of example by using the two variables X and Y, if X comes 30 percent of the time and Y comes 70 percent of the time, then the operational profile of X, 0.30 and Y is 0.70. It is the set of independent operations that a software system performs and their associated probabilities. To develop any OP, five steps are to be processed in a consecutive mode.

- ➢ Find the Customer Profile.
- ➢ Establish the User Profile
- ➢ Define the System-Mode Profile
- ➢ Find Functional Profile.
- ➢ Implement Operational Profile

In figure 1 the first four profiles (Customer, User, System-Mode, and Functional) are started on the design level of a system while the last profile (Operational) is on the implementation level and works on the operations of a

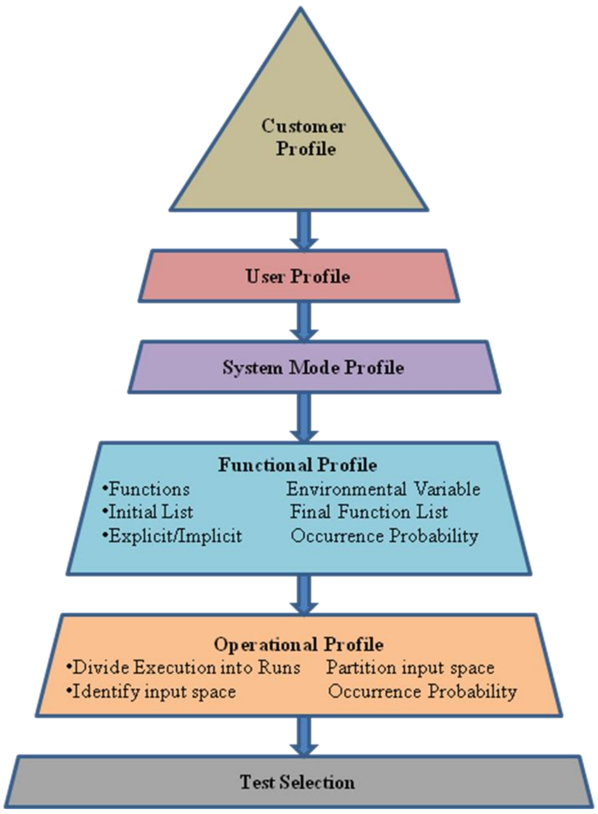

Figure 1. Operational Profile

system. All the first four profiles are not essential for every system. Customer profile is not used if there is a single customer or if the entire customer used the system in a same manner. Figure 1 Operational Profile is adopted from [6] to implement the OP to guide test selection.

CUSTOMER PROFILE

It is a set of customer groups with corresponding occurrence probabilities makes the customer profile. A customer is the person, group, institutions or organization that acquires the system. Customers in a customer group use the system in a same manner, or in a different manner from other customer's types which categorize the user in homogeneous or heterogeneous users. If the entire user used in a same manner, then it is called homogeneous and if the entire user execute the different operations in different manner then it is called heterogeneous. In table 1, there are two different types of customer groups in Institution, Small Institution and Large Institution. Small institutions execute the required operations with 40 percent of the use and the large institutions execute 60 percent. So, the customer profile with occurrence of probability is 0.40 and 0.60 which is calculated on the occurrence of the group.

Table 1: Customer Profile

Customer Group	**Percent**	**Occurrence Probability**
Small Institution	40%	0.40
Large Institution	60%	0.60

USER PROFILE

A user is a group, person or institution that employs, in the system not acquire the system. System users are different types of users which are not necessarily identical to its customers. Different user groups can divide the task of developing the operational profile among different analysts of the system. The user profile can be defined on the experience of customer profile and determining the different user groups for each customer group.

Different user groups like homogeneous/heterogeneous employ of the system separately work in the system. User groups are system administrators, maintenance users, regular users, part-time/full-time users etc. The overall occurrence probability for user groups can be obtained by multiplying the probability of user group and customer group with the occurrence probability of that customer group.

In table 2, User Groups are divided into two parts, Regular Employee and Part-Time Employee. If User Groups are combined over different customer groups, then their probabilities will have to be added and calculate the total User Group probability. Suppose if the input customer profile is (40% Small Institute and 60% Large Institute) and the user group use (30% Regular Employee and 70% Part-Time Employee). In each customer group in a User Profile of 0.12 (40% * 30%), used the Small Institute of Regular Employee and 0.28(40%*30%) for Part-Time Employee, 0.18(60%*30) and 0.42(60%*70%) for Large Institution. Total use by Regular Employee is 0.30 of system and 0.70 by Part-Time Employee of a system.

Table 2: User Profile

	Small Institute probability=0.4		Large Institute Probability=0.6		Total User group Prob.
User Group	User with cust. Prob. Group	Overall user group prob. for customer group	User with cust Prob. Group	Overall user group prob. for customer group	
Reg. Emp	0.30	0.12	0.30	0.18	0.30
Part-Time Emp	0.70	0.28	0.70	0.42	0.70

SYSTEM MODE PROFILE

System mode is a set of function and operations which is helpful in analyzing the behavior of the user and the system. Functions are used at the design level and operations are used at the implementation level. A system mode profile is the set of system modes and their associated occurrence probabilities. It is possible to have system modes that can only be used if no other system modes are used, but it is also possible to have multiple system (operational and functional) simultaneous system modes used. The same function or operation can occur in different system modes. There is no limit to establish to establish the system mode but make balance between effort and cost to determine their associated operational profile. In system mode, there is a single user or different types of users which can be used the system by the administrator, user or guest.

FUNCTIONAL PROFILE

After implementing a system mode profile, then we evaluate the system mode for the functions performed during that mode, and then assigning probabilities to each of the functions. Functional profiles are usually designed during the requirement phases or during early design phases and it should be kept updated when changes occur. Basically, functions worked as external entity set that user can execute with the system. To create functional profile the system modes have to be broken down into the single functions and classified as in figure 1(Operational Profile):

- Number of Functions
- Initial Function List
- Explicit/Implicit Function List
- Environmental Variables
- Final Functional List
- Occurrence Probabilities

(a) Number of Functions: The number of operations or functions in a functional profile is not fixed. It will vary based on the project size, number of system modes, environmental conditions and functions breadth. For developing a system, task into two functions, are the possibilities to develop them with different priorities and the frequency of use.

(b) Initial Function List: The initial function list highlight features, which are function capabilities of interest and values to users. This list can be designed by functions which are relevant to each key input variable. Features should be taken from the customer or user and may be from requirement specification. To identify the environmental input variables and their values or value ranges that will require separate development efforts. It defines the conditions that affect the program runs, but do not relate to the features. Traffic load and hardware configuration are examples of environmental variables.

(c) Explicit/Implicit Function List: A functional profile can be either explicit or implicit, depending on the key input variables. A key input variable is an external parameter which affects the execution path a software system traversed on different values of the parameter. Implicit profile can be used only when the input variables are independent with each other and consider on the occurrence probability of their value while explicit profile consist of enumerated set of all variables with their associated occurrence probabilities. An explicit profile includes a cross product of all key input variables with the respective occurrence probabilities.

(d) Environmental Variables: The environmental input variables can be identified in different conditions that affect the way the program runs. These parameters variables can cause the range of variables and different operations to be performed.

(e) Final Function List: To create final function list, first examine the dependencies among the key input variables and its feature. If the variable is fully depending on another, then it can eliminate from the final list.

Final Function List = No. of Environmental Variable Values (No. of functions in Initial list - Combination of Initial Function)

(f) Occurrence Probabilities: It can be measured by the usage taken on the log of the system, latest use or automation of the manual function. Occurrence Probability can be calculated on Operation. When new versions are released then the combination of old functions and new functions are measured. So estimation of combined function is less accurate than measure of the function.

Functional profile implemented at the design level of the system. After implementing functional profile, it is divided into number of functions and the probability of function is defined. The initial function list highlight the function of the list which is further either explicit/implicit of the function which depends on the input variables. How the program runs in different environmental conditions are defined on the base of environmental variables and final functional list identify the dependencies among the variable. On the base of usage, probability of the occurrence of the function is defined.

OPERATIONAL PROFILE

Operations are used at implementation level of a system while functions are task of a system which is used for design. The number of operations is higher than the number of functions. A single function may be implemented by multiple operations in the system. It is also possible to set of functions to set of operations.

Operational Profile process is divided into three different stages. First operations are associated with runs. To develop the operational profile, runs which divide the execution time of a program. A run is a quantity of work or a set of task initiated by some specific user intervention or input state and represent the activity. The input space is the set of input states that can occur during the system executions. The required input space and the design input space is different, which required conditions to be tested to execute the program. A list of input state is defined with the corresponding probabilities for an input state profile.

As with the functional profile, there are two ways to determine the occurrence probabilities are by recording the input space or by estimating the occurrence probabilities of the functional profile. After each input variable is partition into ranges, with the probability of each variable must be identified. The initial estimation of the system should be performed by the expert who has knowledge of the system and the user. The number of operations is too long, it is essentially to minimize the number of operations by applying three methods.

- Reduced run types.
- Run types are executed in a group.
- Avoid the run types expected to have total occurrence probability considerable less than the failure intensity objective of the system.

It is beneficial to reduce the number of run types in reducing the testing effort, design and implementation costs. We can reduce the number of run types either by reducing the number of input variables or the number of values for each input variables. There are different ways to reduce the number of input variables.

- Minimize operations.
- Minimize hardware configuration, if possible.
- Environment conditions are limited to execute the operation.
- Dependency between successive runs is reduced.
- The system must tolerate the faults like human, hardware and software.

Operational profile can change when new features added with the profile and measured data regular on the base of number of runs of each run types. This measurement is used to identify the failures detection and recording functions or other performance measurement system.

TEST SELECTION

Test cases are derived from the various possible taken runs in each operation which define different states in a function. Test cases can be selected efficiently on the based of usage and the most used operation will be tested the most. Testing, execute by an operational profile, is very effective for identifying failures and their occurrence probability. It is difficult to test all the input state. Selection should be based on operation and run types, which is replaced if the failure occurs. Selection must be perform without replacement in which runs can be chosen only. Thus test are organized from the incomplete design input state, because environment are changed over time, repeating the same operation.

USAGE BASED MODELING

Modeling the usage in a usage specification, defines the intended usage of the system. Specification defines both how the users can use the system and the probabilities for different use of the system. From the usage specification test cases are generated according to the usage profile. If the probability distribution is same as the system is used during operation, we can estimate reliability of the system used. There are several techniques that have been proposed for the usage specification. The most used usage specification models are introduced.

A. Markov Chain Model

WHITTAKER et. al. [7] proposed Markov-chains for sequence of inputs of modeling sequences to software systems. Musa describe usage for the purpose of generating test cases and to guide software testing statically. Events are executed in a consecutive sequence and represent as a stochastic process. These sequences represent test cases and can be used for statistical software testing. Construction of Markov Chain model is divided into two phases, structural phase and the statistical phase. During structured phase, a state is created for every input of the system which is able to receive. Arcs connect consecutive actions of the events of the system which define the initial and final state of the system. After establish the Markov chain, probabilities of the arcs are assigned to the arcs during the execution of the statistical phase.

In Figure 2. Markov Chain defines finite state with distinct parameters are used to model the sequences. The states of the chain represent the inputs of the software, while the arcs represent the sequences of states and are glossed with probabilities. Each arc is independent from previous state and represents the present state of the model.

The advantage of the Markov chain model is to generate the execution of sequences of the usage and capture the operational behavior of the system. It also helps to analysis of the process which is based on random process. The disadvantage of the system is that for a large system the number of states acquire very large.

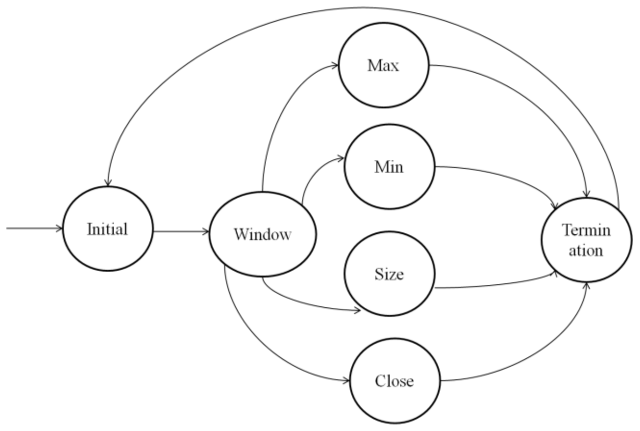

Figure 2. Markov Chain Model

B. State Hierarchy Model

This model is used for complex systems and several users' types and heterogeneous types of users. The aim of this model is to divide the usage problem into different levels. Markov chain is used by the Wohlin and Runsen[8] for usage modeling and reliability engineering of software components.

Arora, Mishra and Kumre[9] worked on the issue of allocation of test cases to infrequent operations. For usage modeling of software components for probabilistic state-charts [10] describe usage structure and profile. The representation in a hierarchical form of Markov model is also called as State Hierarchy Model (SHY) is used for the representation of the usage model. In figure 3 State Hierarchy models [7] is divided into different complex systems with several user types and numerous different users.

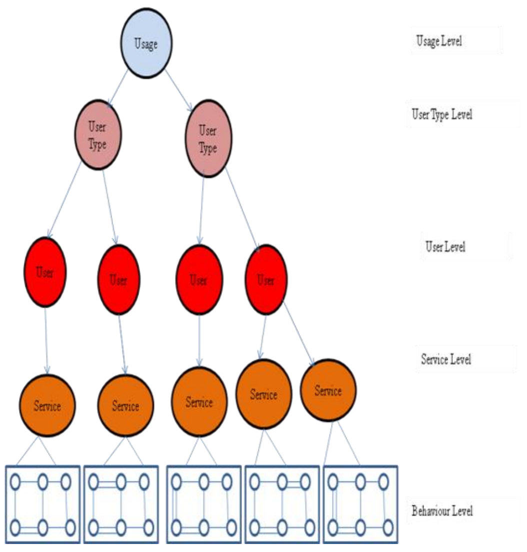

Figure 3. State Hierarchy Model

The function of this model is to divide the usage model into different levels for concentrating on one expression at the time. The number of levels can easily be added in the modeling when required. In figure 3 the usage level represent all the different usage of the system, the user type level represent heterogeneous/homogeneous types of user like (client user and administrative user) is assumed from [11] to define the state hierarchy model. User level represent the users of the system and the service level describes about the services which service a particular user can avail. The behavior level describes about the structural description of a service. On the base of service transaction is made and an event is added to the test case. Test cases can be generated by system to top-down approach through the SHY model by selecting different User types, User level, Services and the corresponding Markov Models. All users of a specific user type have the same individual profile. This profile behaves as transactions probabilities. The choice of specific user to generate the next event depends on the actual state of the user and the state of its services. The weight of the state depends on the behavior level to capture the probability of the next event.

The advantage of this model is to allow for the dynamic probabilities of the state. The disadvantage of this model is that, it is difficult to find a desirable list of the state and the second problem is both the services and users are dependent of each other.

CONCLUSION

From the above approaches of developing different profiles and usage modeling for software reliability. It is found that Operational profile plays an important role in reliability estimation. When the developer have limited time to test the test cases and to select the test cases for testing the most used functions ensure to increase the software reliability.

On the base of usage, Markov Chain and State Hierarchy Model is proposed for measure the reliability of the software. To calculate the reliability, failure data can be collected on the based of usage or samples from the intended usage and representation of the operation or test cases. In future, novel model can be proposed to make relation in extended levels. It is accommodating to calculate the reliability of the handheld devices in medical and signal control system in railway.

REFERENCES

[1] IEEE standard “Glossary of Software Engineerng Terminology”, ANSI/IEEE standard 729, 1991.

[2] R.E. Mullen, “ The Lognormal Distribution of Software Failure Rates: Application to Software Relaibility Growth Modelling”, Cisco Systems, IEEE, 1998, pp. 134-142.

[3] R. A Khan, K. Mustafa and S.I Ahson, “Operational Profile-Factor for Reliability Estimation”, University Press, Gautam Das & V P Gulati, CIT, 2004, pp. 347-354.

[4] J.D Musa, A. Iannio and Okumoto, “Software Relaibility: Measurement, Prediction, Application”, Professional Edition, Mc Graw Hill Publishers, New York 1990.

[5] J.D Musa, “The Operational Profile in Software Reliability Engineering: An Overview”, Proceeding: 3rd International Symposium on Software Reliability Engineering, 7-10 Oct 1992, pp. 140-154.

[6] J.D Musa, “Operational Profile in Software Reliability Engineering”, IEEE Software, Vol. 10, No2, pp 14-32, March 1993, ISSN 0740-7459.

[7] J.A.Whittaker, J.H. Poore, “Markov Analysis of Software Specification”, ACM Transaction. Software Engineering Methodology (1993), pp 93-106, ISSN 1049-331X.

[8] C. Wohlin, P. Runeson, “Certification of Software Components”, IEEE Trans.Softw. Eng. 20 (1994), pp. 494–499, ISSN 0098-5589.

[9] S.Arora, RB Mishra,V.M Kumre,“Software Reliability Improvement through Operational Profile Driven Testing”. In Proceedings of Annual IEEE Conference on Reliability and Maintainability Symphosium, Virginia, 2005, pp, 621-627.

[10] R.Shukla, D. Carrington, P. Strooper, “Systematic Operational Profile Development from Software Components”, in APSEC ’04: Proceedings of the 11th Asia Pacific Software Engineering Conference, Washington, DC USA: IEEE Computer Science, 2004, ISBN 0-7695-9, pp 528-537.

[11] Heiko Koziolek, “Operational Profiles for Software Reliability”, Seminar “Dependability Engineering”, 6th July 2005, pp 1-17.