International Journal of Applied Engineering Research ISSN 0973-4562 Volume 12, Number 22 (2017) pp. 11865-11873
© Research India Publications. http://www.ripublication.com

Development of Software Operational Profile

***Amrita[1], Dilip Kumar Yadav[2]**
[1,2]Department of Computer Applications
National Institute of Technology, Jamshedpur- 831 014, Jharkhand, India.

1Orcid: 0000-0001-7685-4799

Abstract

Operational profile of the software is an important feature of software reliability engineering. Operational profile is used to guide the software testing process. Software testing takes near about sixty percent of the total software development cost. For software testing, test cases are generated for all operations and functions used in the software. However, all operations do not have the same usage priority. Therefore, operational profile based test case generation is very effective for software testing. Operational profiles are a quantification of usage patterns for a software application. These profiles are used to measure software reliability by testing the software in a manner that represents actual use. Based on these profiles optimal number of test cases can be allocated to the operations which required higher priority. In this way, it helps to improve software development process, quality of software product and also user satisfaction. The operational profile based testing is effective and practical approach to software testing since it improves the reliability and speeds up the development process. This paper outlines the step by step methodology of software operational profile development. The proposed methodology is illustrated with a case study. The validation result is satisfactory.

Keywords: Software reliability, Operational profile, Usage probability, Occurrence probability, Functions, Criticality.

INTRODUCTION

Now-a-days software systems play a very significant role in our daily lives. Software systems are the product used by end users. A quality software product should have many characteristics such as reliability, maintainability, availability, modifiability, performance, security, usability and testability etc. Among all, reliability can be considered as the most important one and hence it is very necessary to make a reliable software product. The reliability issues in software based systems are a well-known fact [1]. Reliability is ruined by having faults in software that causes failure of the system. Faults in software can occur due to incorrect input received from environment and therefore reliability of software should be calculated based on how software would work in actual field environment. To maintain high quality and low cost product, appropriate information regarding software usage needs to be collected to guide quality assurance activities.

Software Operational Profile (SOP) reflects the actual usage of software product in field environment [2, 3]. It shows about user's view of software product. Previously there were no expectations from end users, but now it has been changed from developer's viewpoint to user's view point [4]. SOP can be defined as "a quantitative characterization of how a system will be used" [5]. Due to some schedule constraint, software operational profile can be very useful in terms of assuring that most used functions will be tested first and lesser used will be in next phases [6].Along with all this, SOP provides other benefits also such as making testing faster, captures user's need more precisely, reduces cost with reduced operation, reduces the system risk by more realistic testing and improves productivity by allocating the resources in relation to use and criticality. SOP is used in wide applications such as performance prediction, detection of redundant code, web usage mining, aircraft applications, reliability monitoring, test case planning and can also be used for testing purposes of GUI interfaces.

SOP has some limitations also. A software in a system comprises of many functions, of which some are most used and some functions are lesser used, whose occurrence probability is relatively low, but it is not always true that low frequency functions are not critical. For example, in railway reservation system, if the system fails and is not been able to reserve seat will not be as critical as allotting the same seat to different passengers. It is assumed that all users are of equal importance but it is not true in all cases. For example, a higher authority checks records few number times instead of lower level employees, but higher authority checking will be critical for the system.

It is required to discuss some important terms related to SOP development before discussing about it. Profile is defined as set of disjoint alternatives, with their occurrence probability. Occurrence probability can be obtained by dividing transactions per hour by total transactions. Operational profile is a profile for operations at implementation phase while functional profile is a profile for functions at design stage. Operations are defined based on runs. A run is defined as sub-division of the time period of execution of a program. These are concepts used in SOP development. However, we have to find out different classes associated with operational profile (OP) development. Therefore, different classes of OP are described in order to develop an effective operational profile.

Categorization of different OP classes is given in [7]. OP classes are classified into common features, software boundary, dependency and development and further sub-classified for each class. Common features are divided into profiles, structures, abstraction level, scenario, mode, originator, configuration and critical operations. Software boundary is divided into executive scope, external error and input data. Dependency class is divided into code dependency and field of interest and finally development class is divided into life cycle phase and tool support. All SOP model falls into these categories. Operational profile can be developed based on tree structure, state and set.

Previously Musa [6] had given a 5 step process for operational profile development includes customer profile, user profile, system mode profile, functional profile and operational profile. He had used multiple profile, tree based structure, different modes (system modes), configuration class (like environment), and input data class. Musa had also specified the originator for OP model as human, which is not mentioned in Whittaker [8]. There is not any procedure for handling uncertainty. Also, there can be different users with different usage having different environment, in which they operate. However, no such step has been given for the inclusion of usage and configuration profile in Musa's procedure.

There are a lot of work left for the development of software operational profile like usage modeling, environment setup, measurement of quality for operational profile, rules for profile and structure, originators, handling critical operations, rules for defining code dependency etc. While concerning for operational profile, it is all related to its occurrence probability, and the occurrence probability is defined by usage of element. Also it is important to bring attention towards environment and hardware configuration for effective software operational profile development. The major goal of this paper is to construct a software operational profile in effective way with some modifications in Musa's approach. Efficient development of software operational profile provides efficient way of testing a system.

This paper is structured as follows: section 2 presents literature survey related to software operational profile development. Section 3 describes the overall development procedure for software operational profile. Section 4 and 5 describes case study for the overall software operational profile development and validation for the proposed methodology and section 6 describes conclusion and future scope.

LITERATURE SURVEY

Operational profile was first introduced by Musa in [6]. In this paper, five steps for the development of an operational profile are given. A case study of telecommunication switching system has been given for the development of operational profile. B.D. Juhlin [9] identified a new way to develop operational profile where the approach involves decomposing an operational profile into two components: a configuration profile and a usage profile. These components are then recombined using appropriate selection and/or sampling strategies. It is most useful when the product's customers, usage, and configurations are varied and complex.

Usage profile is defined as how customers or users use it. A usage profile is described by the list of users that together define usage profile. According to [9, 20, 21], usage profile is defined as descriptions of each user's product usage including the probability that reflects the relative frequency of that usage. Usage profile can be obtained based on the type of users and their usage. All users can not have same usage for a software system. It depends on the type of user or customer and their frequency of usage. Per Runeson [13] described usage profile as statistical usage profile, in which usage is described by states in state machine and arc between states show transition, which can understood in terms of occurrence probability.

Configuration profile is defined as how customers set up their system. According to Juhlin, a configuration profile defines the physical and logical configuration that makes up an operational Profile.

In [5], Musa had given some basic concepts related to operational profile development. Himanshu Pant et. al [10] had given a structured approach for improving software reliability through operational profile. In this paper a method for development of an operational profile for interactive users has been explained. Leung [11] has given various testing strategies. Out of which, operational profile is found out to be more effective. In [8], Whittaker et. al has given two problems that has not been considered in previous profiles. Moreover it does not consider the environment in which program is executing. Also the final output depends on how the system is used because it is not necessary that user will use the system in the same way as developer has developed to use and on the basis of these problems a new extended operational profile has been developed by Gittens et.al [12].The extended version consist of three profiles in addition to original operational profile. These are process profile, structure profile and data profile. Process profile is similar to usage profile. Structure profile is described as the structure of the system on which application is running .Data profile is described in terms of actual data.

Further the dependencies among software modules are also not provided explicitly among the inputs of a software system. Some authors have used markov chain for modeling dependency among different inputs. In [8], Whittaker et.al used finite state discrete parameter markov chain, which is used to model the input sequences. It will be advantageous to use markov chain based model because it provides certain information for test cases. Wohlin et.al [13] has used markov chain for modeling dependency and for certification of components. In [14], Sebastian et.al has given a methodology for operational profile refinement. According to author, a single operational profile is used to represent that only one author is going to use the system. However, if the customer base is heterogeneous, It will not be appropriate to use single SOP. Clustering analysis was used to support the refinement methodology for identifying groups of customers with similar characteristics.

In [15], Rakesh shukla has given a systematic operational profile development for software components. The method uses both actual usage data and intended usage assumptions to derive a usage structure, usage distribution and characteristics of parameters. In [16], Sham Arora has identified one problem of fewer test cases or no test cases for infrequent operation. In 2007, Hany EL Yamany [17] has given a multi-agent framework for building an automatic operational profile. The research described in this paper investigates a novel multi-agent framework for automatically creating an operational profile for generic distributed systems after their release into the market. J. P. Fu, M. Y. Lu [18], has developed operational profile with uniform design and operation number effectively. It reduces no. of operations by partitioning based on certain attributes.

Another approach for the development of an operational profile has been given by Sravana et.al [19]. As operational profile can be developed either by step by step approach or some data can be available from the previous data or some expert opinion, survey, previous release. It is known that development of operational profile is useful if it is developed during early stage of software development lifecycle. A fuzzy software operational profile has been proposed, which is taking input data in form of linguistic variables.

In [7], a review has been presented for operational profile. Though various papers are found regarding operational profile, this paper categorized all classes of SOP in which it can fall. OP classes have been categorized as common feature classes, dependency and development. All researches related OP falls into these categories. From this survey it is clear that effective development of software operational profile needs a lot of work and there is less certainty that it will cover all aspects. However, based on previous work we can enhance the development model to cover useful aspects. As described by Musa, five basic elements are used to develop operational profile. Based on the survey, we can add two more profiles in the existing model to make efficient operational profile and thereby improved reliability. Next section will present the proposed methodology for the development of software operational profile, which is enhancement of Musa's model.

PROPOSED METHODOLOGY FOR THE DEVELOPMENT OF SOFTWARE OPERATIONAL PROFILE

The proposed methodology has seven steps for software operational profile development. Previously Musa had given five step methodologies for operational profile development. Two steps have been added in the previous model to cover more aspects as well as making overall process effective.

The procedure discussed here transforms usage from user level to system level [10]. Inputs are processed and the output is the set of operations along their occurrence probability. Following are the key steps for the software operational profile development:

Step 1 Defining customer profile

Step 2 Identifying user profile

Step 3 Identifying usage profile

Step 4 Identifying configuration profile

Step 5 Define system mode profile

Step 6 Determine the functional profile

Step 7 Determine the operational profile itself

In first 6 steps, system use is breaking down in further detail. First step is defining customer profile, which breaks down in user profile. Now, user profile breaks down in usage profile, which depicts the usage categorization of users. Further, it is required to identify configuration profile. It is needed because different users can run their application in different configuration. Now, a single user group or multiple user groups can call several system modes. In turn, each system mode has different functions. In the last step, Functions are categorized into operations [6].Based on operational profile test cases are allocated to operations.

Some steps may not be necessary for the development. For example, if customer and users are same for the organization, then both steps can be merged. However, profiles can also vary as if single and multiple. If there will be a single customer or a group of customers having same usage & acquire the system, then there will be single profile for all customer. However, customers with different usage or different type of customers make multiple profiles. The same procedure will be followed for all elements like user profile, usage profile, configuration profile, system mode profile, functional profile & operational profile. Each element at different level carries its occurrence probability. However, software operational profile is constructed by multiplying its occurrence probability with the occurrence probability of its precedence element. It shows their logical relationship with each other. There may be the case, where it is possible to directly develop operational profile, without considering immediate stages of functional profile. It is possible when there is clear description of requirements. However, for developing software operational profile, you can use the procedure outlined here.

Step 1 Defining customer profile

Customer is anyone that will acquire the system. It can be any individual, group of individuals or some organization. It is the first step in developing the operational profile as it is required to know that who is going to acquire the system. Customers can be homogenous or heterogeneous in their usage. If there are multiple customers with different purposes then, a single operational profile will not be useful for all type of customers.

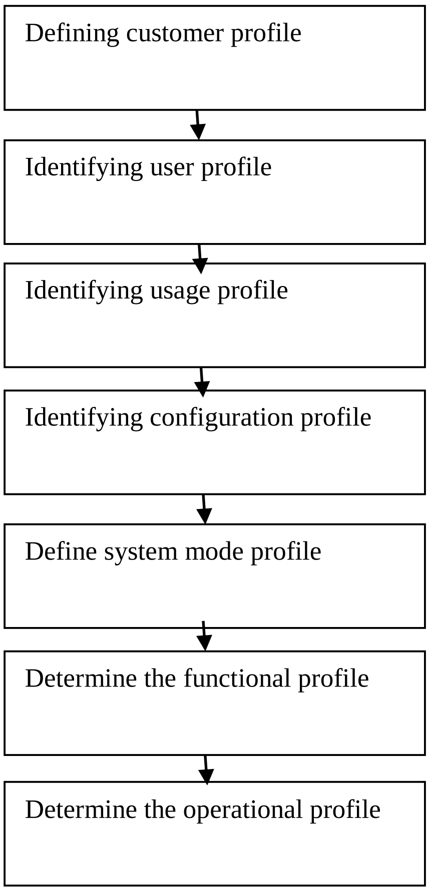

Defining customer profile
Identifying user profile
Identifying usage profile
Identifying configuration profile
Define system mode profile
Determine the functional profile
Determine the operational profile

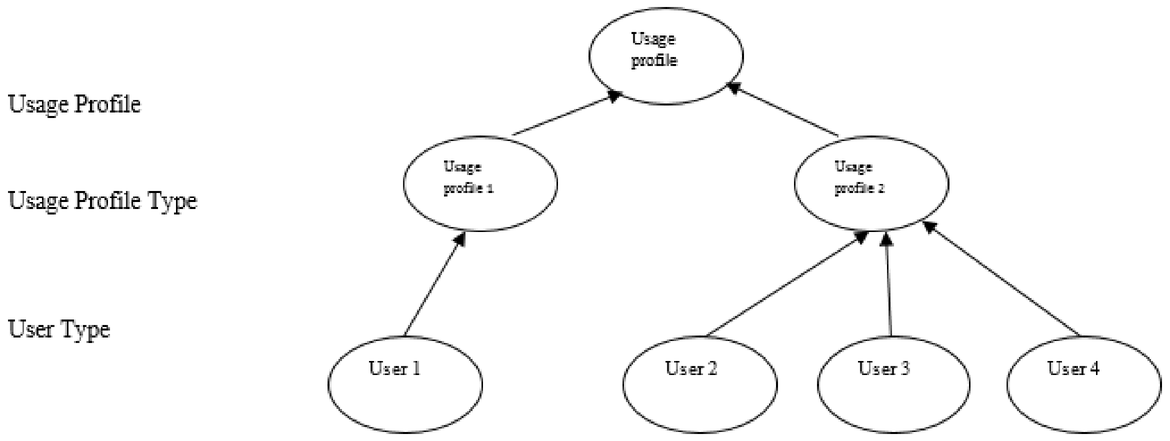
Figure 1: Proposed methodology for software operational profile development

Step 2 Identifying user profile

User profile may or may not be same as customer profile. Users are the individuals that will use the system. User profile is the set of users with their set of occurrence probability. It is required to break customer groups into users. If there will be only one user then no need to define it separately, otherwise user profile is created after breaking down the customer profile. To find out the weighting factor for user group, it is required to find out the proportion of customer usage it represents. If there are multiple customers, then user's probability should be multiplied by the customer's group probability. for ex there are two customers A1 & B1 with their occurrence probability 0.4 and 0.6 and their users are C1& D1 with occurrence probability 0.7 and 0.5 w.r.t A1 and 0.6 and 0.4 w.r.t B1 then user group C1 has 0.28 and 0.30 occurrence probability w.r.t to A1 & B1 whereas user group D1 has 0.24 and 0.24 occurrence probability w.r.t to A1& B1. Thus the overall occurrence probability for C1 will be 0.58 whereas for D1 it will be 0.48. Otherwise user weighting factor can be calculated by the user's usage by total users. For ex. there are 1000 users in an organization out of which first user group consist of 800 users, second consist of 90 and the third group consist of 110 users, then the occurrence probability for group 1 is 0.80, group 2 is 0.09 and the third group will have 0.11 occurrence probability.

Step 3 Identifying usage profile

It is new step that has been added to Musa's approach. It can be noted with the previous step that users are not same in their usage proportion. Some users have high occurrence probability than the rest of users. So it is much needed to focus on usage profile as we want to improve testing by considering most used part in first phase and lesser used in next releases. For example, considering two different scenarios, web and transportation control application. There are group of users that are assigned for handling the operations separately. However, usage can be different in both scenarios. In web excavating system, it will not be useful to assign group of users to handle criticality as in such kind of scenario users are concerned about the operations performed whereas in transportation control application, it is required to assign a group of users to handle critical operations as it is related to safety issues. Therefore, user group of web, to handle critical operation is not of much concern. They can be assigned less probability comparing to transportation control application users. In these situations, it will be beneficial if we prioritize user group based on their usage. For prioritization, usage is divided into different usage profile, which will be helpful for deciding in, which group is needed for prior testing. However, there may be users of having high usage probability than the rest of users. Usage probability will be calculated by dividing user's usage by total usage. Usage profile is the proportion of user's profile in respective customer profile. Based on that usage probability, usage profile will be created.



Step 4 Identifying configuration profile

It is also a new step added in Musa's approach. Configuration profile is defined as the way customer set up their system. This is the environment in which one or more usage profile can take place.

Step 5 Determine system mode profile

System mode is defined as different ways of system usage or environment. It further breaks in functions and operations. System mode is necessary to consider as there may be different environment that needed separate testing. Musa has defined some patterns in which system mode can fall. These are user group, significant environment condition, operational structure, severity, user's experience and hardware configuration.

Step 6 Determine Functional profiles

Functional profile is defined as set of functions with their occurrence probability. After breaking down system modes, set of function can be obtained. It is different from operations as it is useful in design while operations are at implementation level. Musa has given a procedure to define functional profile. We have to first find out all the functions that are part of the system. This will create the initial function list. It is also necessary to find out several environmental conditions that may affect functions further during their implementation. So, environmental conditions are necessary to determine. With the help of initial function list and environment condition, we can find out final function list with their associated probabilities. These are the functions defined for certain operational mode. Now we will define, how to develop functional profile in brief:

Initial function list (IF)

Here, we have to first identify the basic functions. If software requirement specification is complete, we can get the initial set of functions easily. However, we cannot get the complete list if SRS is incomplete. There are some other sources for identifying the list of functions. For ex. prototype of some product. However, it will not work for new released software. It can be beneficial to involve users for identifying the list. Users can better predict for functions of the system. Initial function list can be predicted for different system modes.

Environment variables (EV)

These are the condition that affects the execution of a program. Hardware configuration and traffic loads are example of environment variables.

Final function list (FF)

Final function list is calculated by multiplication of initial function list and environment variables.

$$FF = IF * EV$$

Step 7 Determine Operational profiles

Operational profile is defined as set of operations with their occurrence probability. Set of functions further breaks down into set of operations.

Occurrence probability of an operation = Ri / ∑Ri= 1, 2,

Where Ri is percentage of occurrence of a single operation

∑Ri is sum of percentage of occurrence of all operations

However, there is difference between a function and an operation. A function breaks down into operation, which is more refined. For ex. a function of administrative mode wants to relocate telephone. This function breaks down into two operations: removal and installation. Operations are more refined than functions .Now for determining operational profile, it is needed to first divide the execution into runs, identifying input space, partitioning input space and finding the occurrence probability of each operations.

Dividing execution into runs

Operations are related to runs. By considering some environmental condition, a run can be obtained by dividing the execution of operation. It is advisable to categorize runs having same input variable. Run is explained as an instruction. It will execute, unless or until there is some interruption from outside. Input state is used here as an interruption that is external

Identifying Input spaces

Input space is set of all input states that occur during operation. Operational profile aims to provide testing the system in a way, it is in actual field. It will not be sufficient to take only those input variables, that program is taking. It is necessary to list all possible input variables that can be taken for program. We have to consider those inputs also, that results in failure. type.

Partitioning the input space into operations

It is required to partition input space into operations, for non-uniform selection of test cases that should match actual SOP.

Record the occurrence of operations, so that occurrence probability can be calculated easily. Now the proposed methodology will be illustrated with the help of a numerical example.

CASE STUDY AND RESULTS

A case study of telecommunication switching system has been taken for consideration of proposed methodology. SOP is constructed step by step by including all necessary profiles.

Step1 Define customer profile:

Assume there are two customer groups, Retail stores and hospitals with their occurrence probability 0.6 & 0.4 respectively.

Table 1: No. of customers with their occurrence probability

Number of customers	Occurrence probability
Retail stores	0.6
Hospital	0.4

Step2 Identifying user profile

Table 2: No. of users with their occurrence probability

	Large retail stores (0.60) customer group		Hospital (0.40) customer group		
	User group weighting factor for customer group	Overall User group weighting factor for customer group	User group weighting factor for customer group	Overall User group weighting factor for customer group	Overall user group weighting factor
Telecommunication user (0.90)	0.90	0.54	0.90	0.36	0.90
Attendants (0.05)	0.07	0.042	0.05	0.02	0.062
System administrator (0.035)	0.01	0.006	0.035	0.014	0.02
Maintenance personal (0.015)	0.02	0.012	0.015	0.006	0.018

Step3 Identifying usage profile

This step is added to provide extra benefit of dividing users on the basis of their usage. We found two different usage profiles. Let the probability of first usage profile is 0.90 and second usage profile is 0.10. Based on the usage of customers and users these profiles are identified.

Table 3: No. of usage profile with their occurrence probability

	Usage profile 1 (0.90) for Large retail stores (0.60) customer group		Usage profile 2 (0.10) for Hospital (0.40) customer group		
	User group weighting factor for customer group	Overall Usage group weighting factor for user group	User group weighting factor for customer group	Overall Usage group weighting factor for user group	Overall usage group weighting factor based on user's profile
Telecommunication user	0.54	0.486	0.36	0.036	0.522
Attendant	0.042	0.0378	0.02	0.002	0.0398
System administrator (0.035)	0.006	0.0054	0.014	0.0014	0.0068
Maintenance personal	0.012	0.0108	0.006	0.0006	0.0114

Step4 Identifying configuration profile

There is no sufficient information available for the system setup. So we will omit this step.

Step5 Define system mode profile

System mode profile is set of functions and operations, grouped together in order to analyze the whole system. System mode is categorized based on user experience, user type, criticality of the operation etc. Therefore, we have to divide system mode based on customer types only. However, now users will have probability based on usage profile.

It has been assumed that the entire large retail store is business use while hospital is 60% business use and 40% personal use. So, business use can be calculated by adding the business use probability of large retail stores and hospital then multiplying it with business use probability. Personal use probability can be calculated by multiplying personal use probability to personal use proportion of hospital.

Table 4: No. of system modes for different profiles with their occurrence probability

Business use mode	0.3132
Personal use mode	0.036
Attendants	0.0398
System administrator	0.0068
Maintenance personal	0.0114

Step6 Determine functional profile

In this section functional profile is defined for administration mode. This telecommunication switching system has initially 4 functions: adding a new telephone to the exchange, removing a telephone, relocating a telephone and updating online directory from changes. Environmental variables that are considered here is analog and digital. In this example, there are 80 telephone additions, 70 removals and 800 relocations have been performed. Directory update is five percent of total use in administrative mode of usage profile 2. We can now easily calculate function's occurrence probability.

So the function list is shown below:

Table 5: Initial function list with their occurrence probability

	Administration mode (0.0068)	
Functions	Occurrence probability	Overall occurrence probability
relocation	0.80	0.00544
addition	0.08	0.00054
Removal	0.07	0.00047
Directory updating	0.05	0.00034

After considering the environmental factors, now the final function list is:

Table 6: Final function list of administration mode (0.0068)

Function	environment	Occurrence probability
Relocation	Analog (0.8)	0.00435
	Digital (0.2)	0.00108
Addition	Analog (0.8)	0.00043
	Digital (0.2)	0.00010
Removal	Analog (0.8)	0.00037
	Digital (0.2)	0.00009
Dir update		0.00034

In the same fashion, function list can be created for other system modes

Step7 Determining operational profile

From previous step, we have identified 4 functions: relocation, removal, addition and update. These functions are performed based on 2 parameters i.e. users and location. There is no effect of dividing the location. However, user type can be different in their processing, therefore this operation is divided into 3 runs i.e. staff, secretary and manager. One operation 'add' has 3 runs. Suppose there are 780 relocations, 80 additions which in turn 70 staff, 5 secretary, 5 manager and 90 removals. Following table will show the operational profile for administrative mode:

Table7: Operational Profile without normalization

Operations	Transactions	Occurrence probability	Occurrence probability for administration mode (0.0068)
Relocation	780	0.8041	0.00546
add staff	70	0.0721	0.00049
add secretary	5	0.0051	0.00003
add manager	25	0.0257	0.00017
Removal	90	0.0927	0.00063
Update			0.00034

Operations are normalized because test cases will be generated based on these operations given in [16, 19, 22]. After normalization resultant operational profile is given below:

Table8: Resultant Operational Profile

Operations	Transactions	Occurrence probability for proposed model	Occurrence probability for proposed model (Normalized)
Relocation	780	0.00546	0.766853933
add staff	70	0.00049	0.068820225
add secretary	5	0.00003	0.004213483
add manager	25	0.00017	0.023876404
Removal	90	0.00063	0.088483146
Update		0.00034	0.047752809
$\sum$			1

VALIDATION

Model validation After operational profile development, now we have set of operations with their occurrence probability. Now, to validate this model, the results are compared with Musa's model.

Table9: Comparison between proposed model and Musa's model

Operations	Transactions	Occurrence probability for musa's model	Occurrence probability for proposed model
Relocation	780	0.765	0.766853933
add staff	70	0.07	0.068820225
add secretary	5	0.005	0.004213483
add manager	25	0.025	0.023876404
Removal	90	0.085	0.088483146
Update		0.05	0.047752809

Evaluation measure

For the purpose of evaluation we used the Root Mean Square Error (RMSE) method, which is used to measure the difference between the actual obtained value from the calculation that is being modeled and the value predicted by the model. RMSE is defined as the square root of the mean squared error as shown in equation.

$$\text{RMSE}=\sqrt{\frac{\sum_{i=1}^{n}(Xobt-Xprev)^2}{n}}$$

RMSE = $[1/6\{(0.7668-0.765)^2+(0.068-0.07)^2+(0.0042-0.005)^2+(0.0238-0.025)^2+(0.088-0.085)^2+(0.04-0.05)^2\}]^{1/2}$

$= 0.00182$

This equals to 0.0000182% error i.e. acceptable.

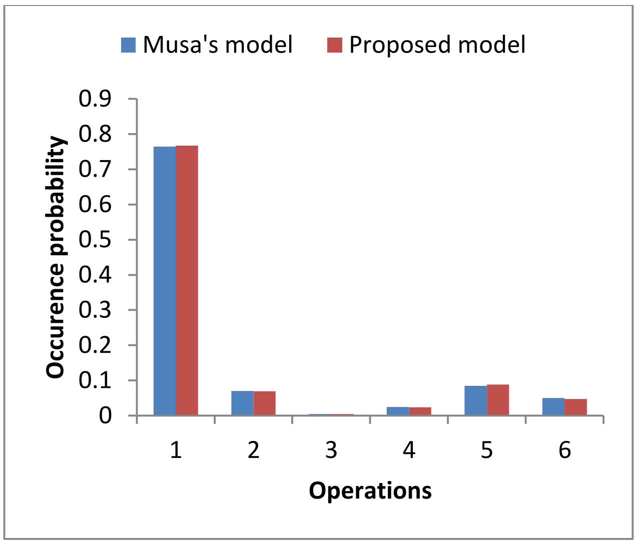


Figure 2: Comparison between Musa's model and proposed model

Validation results

From figure 2, it is clear that proposed model is very similar to Musa's model. There can be many operations for a software system. However, we have selected only most used and prominent operations for the development of operational profile. Error percentage calculated is 0.0000182%.It is clear that adding profiles into existing model is similar to musa's model. Therefore, incorporating usage profile and configuration profile provides wider aspect to cover all essential elements for developing an effective software operational profile. Moreover, the proposed model added benefit of making effective testing by providing two profiles that needed to test. As now, we have more accurate profiles that are needed prior testing. Using the improved software operational profile development model, we can effectively model its usage that meets the principle of software operational profile and demonstrates the feasibility of the method.

CONCLUSION AND FUTURE SCOPE

A new method to develop a software operational profile has been introduced in this paper. It is a systematic approach which considers all the factors that will contribute to make an effective software operational profile. Analysis shows this method is feasible and it will improve the software reliability. It delivers the quality product and therefore increasing the user's satisfaction. This paper can be considered as a step towards making more reliable product through an effective software operational profile. Also dependency among key elements is presented to make efficient development. Our proposed model will help for software development, where we should have knowledge about its users and how they are going to use this software. This paper considers human as an originator but originator can be hardware or software also. So it will be the future scope that how can get inputs from other sources and make it automated as well as validate it. Also we are presenting elements like customer, user manually. Although there can be choice to select these elements based on their dependency and different criteria. Our future scope will be based on these issues.

REFERENCES

[1] D. K. Yadav, S. K. Chaturvedi, R. B. Misra, Early Software Defects Prediction Using Fuzzy Logic, International Journal of Performability Engineering ; vol. 8, no. 4, pp. 399–408, 2012.

[2] Amrita, D. K. Yadav. A Novel Approach for Early Software Operational Profile Estimation. In International Journal of Applied Engineering Research, vol. 12, 15 pp. 5131-5136, 2017.

[3] H. Koziolek, “Operational Profiles for Software Reliability,” in *Seminar “*Dependability Engineering,” pp. 1–17, 2005.

[4] I. Sommerville, *Software Engineering*, Addison-Wesley, 7th Edition, Chapter 24, 2004.

[5] J. D. Musa. Operational profiles in software-reliability engineering. IEEE Software 10, 2, pp. 14–32, 1993.

[6] J. D. Musa. The operational profile in software reliability engineering: An overview. In Proc. of the 3rd International Symposium on Software Reliability Engineering (ISSRE’92). pp. 140–154, 1992.

[7] C .Smidth, J. Mutha, Gerber. Software Testing with an Operational Profile: OP Definition. In journal ACM Computing Surveys (CSUR), Vol. 46, 2014.

[8] J. A. Whittaker and J. Voas. Toward a more reliable theory of software reliability. IEEE Computer 33,12, pp. 36–42,2000.

[9] B. D. Juhlin. Implementing operational profiles to measure system reliability. In Proc. of the 3rd International Symposium on Software Reliability Engineering (ISSRE’02*).*pp. 286–295, 1992.

[10] P. Himanshu, P. Franklin and W. Everett, A structured approaches to improving software-reliability using operational profiles, IEEE Proc. RAMS, pp. 142–146, 1994.

[11] H. K. N. Leung, P. W. L. Wong. A Study of User Acceptance Tests. Software Quality Journal, vol. 6, no. 2, pp. 137-149, 1997.

[12] M. Gittens, H. Lutfiyya, and M. Bauer. An extended operational profile model. In Proc. of the 15th International Symposium on Software Reliability Engineering (ISSRE’04). pp. 314–325, 2007.

[13] C. Wohlin, P. Runeson. Certification of Software Components. In: IEEE Trans. Software. Eng., pp. 494–499, 1994.

[14] S. Narla Elbaum. Methodology for Operational Profile Refinement In Reliability and Maintainability Symposium, 2001. Proceedings. Annual, pp. 142-149, 2001.

[15] R. Shukla, D. Carrington, and P. Strooper. Systematic operational profile development for software components. In Proc. of the 11th Asia-Pacific Software Engineering Conference (APSEC’04). pp. 528–537, 2004.

[16] S. Arora, R. B. Misra, and V. M. Kumre. Software reliability improvement through operational profile driven testing. In *Proc.* of the 53rd Annual Reliability and Maintainability Symposium .pp. 621–627, 2005.

[17] H. EL. Yamany, A.M Miriam .A Multi-Agent Framework for Building an Automatic Operational Profile. In Advances and Innovations in Systems, Computing Sciences and Software Engineering ,pp.161-166, 2007.

[18] J. P. Fu and M. Y. Lu, Develop software operational profile with uniform design, in 2009 IEEE International Conference on Industrial Engineering and Engineering Management, pp. 842–846, 2009.

[19] K. S. Kumar., R. B. Misra., Software Operational Profile Based Test Case Allocation Using Fuzzy Logic In International Journal of Automation and Computing, pp. 388-395, 2007.

[20] J. D. Musa, A. Iannino. K. Okumoto, Software Reliability Measurement. Prediction, Application, McGraw-Hill, 1987.

[21] S.K.Abramson, B.D.Jcnsen, B.D.Juhlin, C.L.Spudic,"Customer Satisfaction Based Product Development", in the *Proceedings of the XIV International Switching Symposium,* 1992.

[22] F. Urem, Ž. Mikulić. Developing operational profile for ERP software module reliability prediction MIPRO, 2010 Proceedings of the 33rd International Convention, 2010.