International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048

Volume 5, Issue 8, pp 94-101, August-2021

https://doi.org/10.47001/IRJIET/2021.508016

A Newly Proposed Ambidextrous Software Testing Model Based on Conventional Black Box Testing Strategy Using the Applications of Gaussian Distribution

[1]Shivankur Thapliyal, [2]Renu Bahuguna

[1]Assistant Professor, Computer Science and Engineering, Doon Institute of Engineering and Technology, Rishikesh, Uttarakhand, India

[2]Head of Department, Computer Science and Engineering, Doon Institute of Engineering and Technology, Rishikesh, Uttarakhand, India

***Abstract* - To Tests Software Systems with all of its aspects and areas are really very typical and challenging tasks for today's Software Engineering issues. Bug free and fully tested software systems are very typical to build or manufacture and one of the most challenging tasks for recent software industries issues in today's information age, because many software systems crash out or failure due to the reason of bug presence, or partial testing approach or not fully tested the software systems and missing the examine of the behavior of the software systems with all of its input values, but it's a very complex and typical tasks to tests any software systems with all of its possible input tests values. So the need of the hour is we have some potential and robust software system testing model which are responsible and also capable to tests any software system product with all of its functionalities with including all areas and investigate or to examine the behavior of software system for all of its possible input tests values with also achieve greater reliability. Some Black Box testing strategies such as Equivalent Partitioning and Boundary Value Analysis (BVA), which are responsible to provide range of input vales among multiple values, but we don't have any strict mechanisms to filter or to select some input tests values among the set of multiple input tests, because the major reason behind the software crisis and software failures are that we don't check software systems with all of its input values or missing to notify the behavior of software systems for some particular input, Here we proposed a robust software testing model, which adopts the mechanisms of Gaussian Distribution for selection of input data and work at the module level of software systems and this model also applies with each module of software systems.**

Keywords: Software Testing model, Black Box testing strategies, Gaussian based Software testing model, Software testing strategies.

I. INTRODUCTION

Software Testing are one of the most challenging issues in today's Software Engineering Concepts [1]. To Test software with all its functionalities and its major tasks orientation specifications are really a very typical and challenging issues for today's software development industries. In the Real environment scenarios, most of the software crises [2] and software failures [3] are to done through some bugs which reported on the software systems and strictly related to the functionality of software systems. So the need of the hour is that we have some strong potential software testing methodology [4] or mechanisms which strictly navigate or to discover bugs or errors and software failures [6] misconceptions in a very easily manner, because the main reason which is responsible for behind the software failures are that we have not an appropriate or adequate mechanisms, which tests software at all aspects corresponding to it's all branching conditions and looping or jumping statements, and also a one major reason behind that, we doesn't tests software for all of its input values or not to draw all test cases related to all input values. In this model we proposed a new software testing model which tests values for all of its input cases and also defines the major importance of its input values. In the previous developed software testing methodology to draw or design test cases, we generally adopt some traditional mechanisms such as equivalent partitioning [7,11,14] and Boundary Value Analysis (BVA) [8,9,11,14] to design tests cases for all of its inputs, but In this previously developed traditional approach based model also contains many misconceptions that it's doesn't tests software for all of its input cases and very slow to perform, but In this newly proposed software testing model we adopts or to use Statistical Distribution [10]. There are many distributions are available for mapping any value with its probability density function [10] with its importance, which based upon the nature of the variable, which is discrete and continuous in nature. Here with the use of probability density function [10] we find out the importance of each values to look through the probabilistic

International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048

Volume 5, Issue 8, pp 94-101, August-2021

https://doi.org/10.47001/IRJIET/2021.508016

value and their occurring patterns, so for using this methodology, we tests software with all of its aspects with all of its input values, so we examine or tests software in a very efficiently manner. Here we proposed or adopted a modular designing approach [11] of software system architecture [12,13], which work with all of its module in a very similar manner, but the working criteria of all modules are must be different. To tests a software with all of its aspects such as: code coverage and path coverage or branching and looping are really a very challenging and typical concepts. But In this newly proposed model we don't require or examine any WHITE BOX TESTING [14] methodology, here we adopt some BLACK BOX [14] testing methodology which works in a very fast efficient manner, but here we only checked the functionality of the software with all of its aspect of the software system in a new manner, we don't check any non-functional requirements such as the code coverage, branch coverage, looping and some jumping statements of the software systems. So that's why we will design all of its test cases with all of its input values using this newly proposed methodology for software systems architecture. Because Normal Distribution [10] or Gaussian distribution [10] are bell shaped curve, which describe or to define all of its values with probability density functions. To pick up any values we find out the importance factor of each values using the Gaussian Probability Density function [10]. We will design test cases for all inputs, which contains a low probabilistic value or those which contains a high probabilistic value. We also design test cases for invalid inputs. We will check any software systems with all of its functional area using this model in a very efficient and reliable manner. Its Computational Cost are very low and gives an accurate result and have capability to easily detect some error prone areas inside the software system architecture. Now the working mechanisms of this model are detailed define in the upcoming section of this research paper.

II. SOME PREVIOUSLY KNOWN METHODOLOGY

Equivalence Partitioning: In this methodology we partitioned some input values into some range of values, In detailed we can say that if we have multiple inputs varieties then In the mechanisms of Equivalent Partitioning we partitioned or range the values of inputs which gives similar types of output or which behaviors are same. In general we can say that Equivalence partitioning or equivalence class partitioning (ECP) is a software testing technique or strategy that divides or partitioned the input data of a software unit into partitions of equivalent or similar data from which test cases can be derived. In this mechanisms of equivalent partitioning a very reliable and accurate test cases have to generate such as test cases are easy to generate for all input values using the mechanisms of Equivalent Partitioning. The concepts of Equivalent Partitioning come from the Equivalence Relation. Equivalent Partitioning are the strategy of Black Box testing technique, where the number of inputs are to be divided into some similar classes. In equivalence portioning, equivalence classes are evaluated for given input conditions. Whenever any input is given, then type of input condition is checked, then for this input conditions, Equivalence class represents or describes set of valid or invalid states.

Guidelines for Equivalence Partitioning:

- If we gives the range condition as an input then one valid and two invalid equivalences classes are to be defined.
- If a specific values are gives as an input, then also one valid and two invalid inputs are to be defined.
- If we gives some member of a set as an input, then one valid and one invalid equivalence classes are to be defined.
- If we gives Boolean number as an input, then also one valid and one invalid equivalences classes are to be defined.

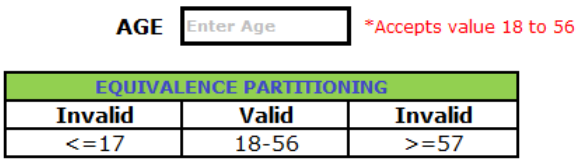

Fig. a: Example of Equivalent Partitioning

Boundary Value Analysis (BVA): In the concept of Boundary Value Analysis (BVA) we test any software systems at the boundaries such as if we take a range of input values such as 0-100, then we check software systems at -1 and 101 for invalid cases and 0 and 100 for valid cases corresponding to range, because a software systems provides some abnormal results at the boundaries of the range of the input values. Boundary Value Analysis (BVA) are testing strategy of Black Box Testing where any software product have to be tested at extremely ends because a software can give some abnormal and unsatisfactory results at the boundary of the input partitioning. These extremely ends like Start-End, Upper-Lower, Maximum-Minimum, these are called the boundary values of any input partition.

The basic idea in normal boundary value testing is to select input variable values at their:

- Minimum
- Just above the minimum
- A nominal value
- Just below the maximum
- Maximum

International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048

Volume 5, Issue 8, pp 94-101, August-2021

https://doi.org/10.47001/IRJIET/2021.508016

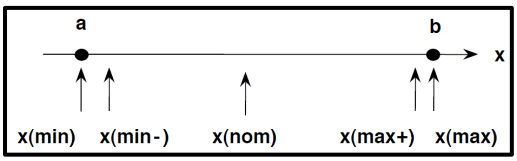

Fig. b: Diagrammatic representation of Boundary Value Analysis

In Boundary Value Analysis, Equivalence Partitioning play's a good role. Boundary Value Analysis comes after the Equivalent Partitioning.

III. A NEWLY PROPOSED MODEL FOR TESTING

The diagrammatic representation of this Software Testing model is representing in fig c:

This software testing model applies on each modules of the while software product, generally at the broad level this model comes under the category of Black Box Testing, because these model checks the functionalities of the software systems rather the code coverage and branch coverage or some looping or jumping. Because to test overall functionalities of the software with tracing or investigate and examine each path or branching coverage are really a very typical tasks. So In this Paper we modeled some Black Box testing strategies for conduct a software testing in a very easy and reliable manner. Now we describe these models by using some phases or steps:

First phase: In the first phase we categorized data into various categories which based on their types, behaviors and their tasks to perform. We categorized data into three types which is:

- Numeric Data
- Textual Data
- File Data

Further Numeric data also subdivided into some types such as the floating point numbers, or some integers numbers, similarly textual data further subdivided into some categories and file data are further subdivided into image, audio, video, documents or some pdf files. After categorized data into some categories, the next step is applying some interval arithmetic and initializes the range of values, which will be the tests by the software module. But this type of range are easy to developed with Numeric Data, but data like textual or file type it's very complex to define the range especially textual and image or some multimedia data, so here for textual and file data we assign some random numeric identity, this process are called the Randomization after Randomization of textual data and some file data, we applying the interval arithmetic to define the range.

Second Phase: In the second phase we are applying the Gaussian Distribution on that ranges of data. Here we applies Gaussian Distribution behind the reason is that Gaussian Distribution defines the probability of each number which existing in the range of values and it's a continuous probabilistic distribution where each numbers we assign a unique probability for their occurrence, so with the use of probability, we better tests each values which placed in our test case and better observe or explore the behavior of software system on each value. Gaussian Distribution also called the Normal Distribution or Laplacian Distribution. Because in the Gaussian Distribution values which are near far the means contains a high probability relatively to those which are far away from mean. Generally Gaussian Distribution works on the theorem of Central Tendency, Here the middle line represents the mean of the sample size or ranges, and other values which puts between the ranges are equally distributed both sides of the means, that's why it's contains a bath tub curve. It's a continuous probabilistic distribution, so using the mechanisms of probability we better tests software systems with each values and select or filter values for tests through their respective probabilities and it's gives a crisp clearance about values that, which values have to select for testing. The detailed about Gaussian Distribution are as follows:

Gaussian Distribution [16] are also known as the Normal Distribution [16] or the Laplace Distribution [16] . It is a continuous probability distribution which based on some probabilistic approach and works on real valued numbered variables. In this distribution we have a sample size of various random numbers and we find out the occurring probability of each numbers using some mathematical and statistical techniques. Gaussian Distribution [16] contains some probability density function. The formula of this probability density function are as follows:

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}(\frac{x-\mu}{\sigma})^2}$$

The parameter μ is the mean or expectation of the distribution (also it's median and mode), while the parameter σ is it's standard deviation. The Variance of the distribution is σ^2. A random variable with a Gaussian distribution is said to be normally distributed, and is called a normal deviate.

Normal distributions play's a much significant role in statistics and are frequently used in the physics and mathematical modelling to represent real-valued random variables whose distributions are not known. It's works on Central Limit Theorem. It states that, under some conditions, the average of various samples (observations) of a random variable with finite mean and variance is itself a random variable—whose distribution converges to a normal distribution as the number of samples increases. Therefore,

International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048

Volume 5, Issue 8, pp 94-101, August-2021

https://doi.org/10.47001/IRJIET/2021.508016

physical quantities that are expected to be the sum of many independent processes, such as measurement errors, extremely have distributions that are nearly normal.

Moreover, Gaussian distributions [16] contains some unique properties and features that are valuable in analytic studies and mathematical modelling. For instance, any linear combination of a fixed collection of normal deviates is a normal deviate. Many results and methods, such as propagation of uncertainty and least squares parameter fitting, can be derived analytically in explicit form when the relevant variables are normally distributed.

A normal distribution [16] is sometimes informally called a bell curve. However, many other distributions are bell-shaped (such as the Cauchy, Student's *t*, and logistic distributions).

Normal distribution, which is frequently known as the Gaussian distribution [16] or Laplace distribution [16], is a probability distribution that is symmetric about the mean, showing that data near about the mean are more frequent in occurrence than data far from the mean. In graph form, normal distribution will appear as a bell curve.

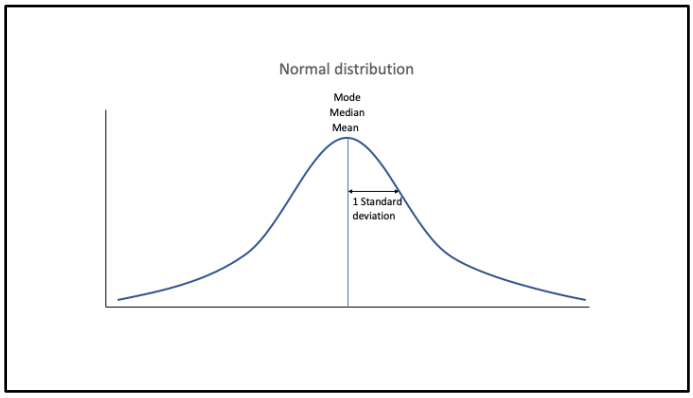

Fig. c: Graph of Normal Distribution

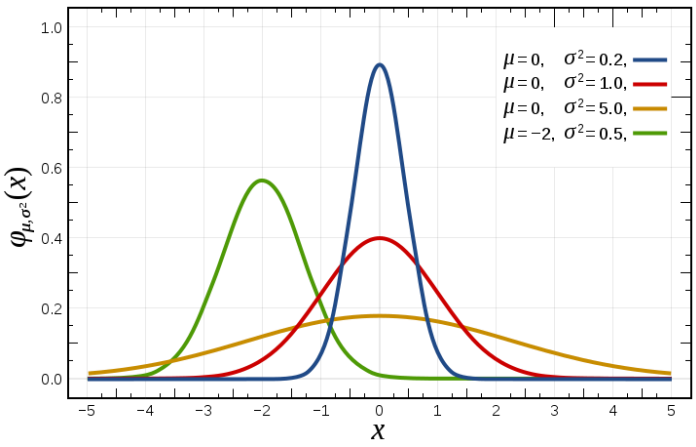

Fig. d: Probability Density Function of Normal Distribution

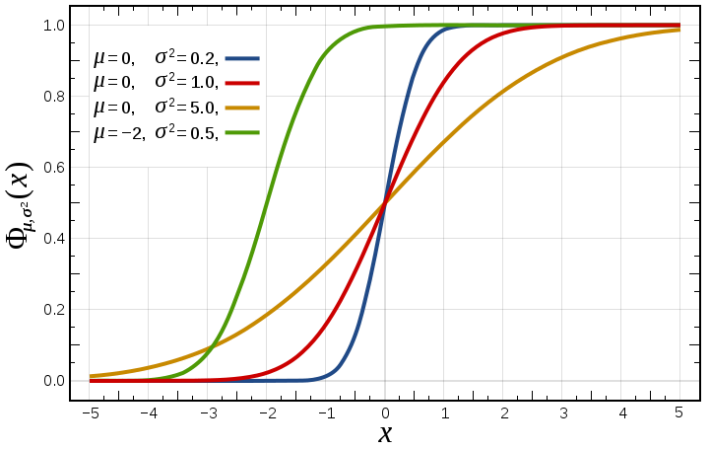

Fig. e: Commulative Distribution Function of Normal Distribution

Table 1: Significant expressions of Gaussian Distributions

Notation	$N(\mu, \sigma^2)$
PDF	$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}(\frac{x-\mu}{\sigma})^2}$
CDF	$\frac{1}{2}\left[1 + \text{erf}(\frac{x-\mu}{\sigma\sqrt{2}})\right]$
Quantile	$\mu + \sigma\sqrt{2}\, erf^{-1}(2p-1)$
Mean	μ
Median	μ
Mode	μ
Variance	σ^2
MAD	$\sigma\sqrt{2/\pi}$
Entropy	$\frac{1}{2}\log(2\pi\sigma^2) + \frac{1}{2}$
MGF	$\exp(\mu t + \sigma^2 t^2/2)$
CF	$\exp(i\mu t - \sigma^2 t^2/2)$
Fisher Information	$I(\mu, \sigma) = \begin{pmatrix} 1/\sigma^2 & 0 \\ 0 & 2/\sigma^2 \end{pmatrix}$ $I(\mu, \sigma^2) = \begin{pmatrix} 1/\sigma^2 & 0 \\ 0 & 1/(2\sigma^4) \end{pmatrix}$
Kullback-Leibler divergence	$\frac{1}{2}\left\{\left(\frac{\sigma_0}{\sigma_1}\right)^2 + \frac{(\mu_1-\mu_0)^2}{\sigma_1^2} - 1 + 2In\frac{\sigma_1}{\sigma_0}\right\}$

International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048

Volume 5, Issue 8, pp 94-101, August-2021

https://doi.org/10.47001/IRJIET/2021.508016

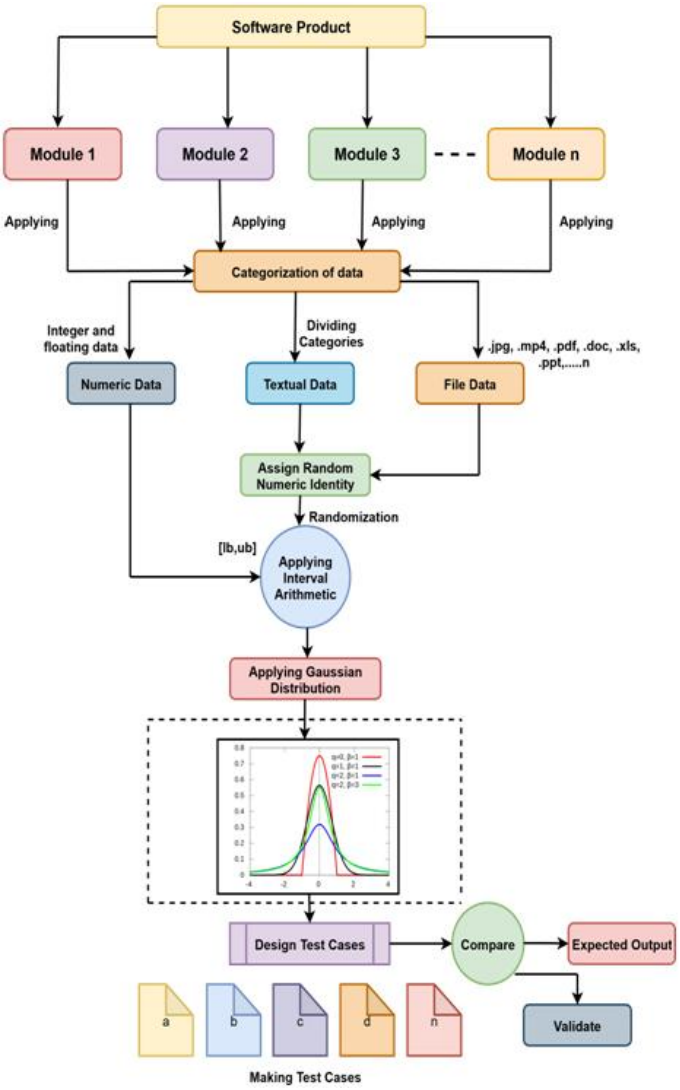

Fig. f: A newly proposed Gaussian distribution based Software testing model

Third Phase: In the third phase we designed test cases for each module and write the actual results on test case. Now after designing the test cases of each module, at the end we compare the resultant data with expected data and find out the error rate or success percentage through comparing and after do the detailed analysis we validate the test data. We also design test cases for invalid inputs or those inputs which occur rarely, because to decide that which inputs occur frequently or rarely are clearly decided through the probability of that number. This model gives much accuracy and clearance about the selection of input tests data. Because to tests or validate any software systems with all path coverage and branching coverage are really a very typical and complex tasks, so with the use of Gaussian Distribution we easy this tasks and also take concern the accuracy and clearance of tests data inputs. There are many distributions are available such as Binomial Distribution, chi-square Distribution , Discrete unit distribution, Cauchy Distribution, but here we elicit the Gaussian or Normal Distribution, because this distribution contains the continuous probabilistic distribution, that's why it's distribute each numbers which is within the range based on their probability. So these are the three phases of this model. This model equally applies on each of the modules of the software product, because In the previous models of Black Box testing are Equivalence Partitioning and Boundary Value Analysis (BVA), but in this model we also make some range of input values, but we don't have any mechanisms that we decided which input gives best results or which gives worst result before tests any software systems, but In this newly proposed model, we easily decided that which inputs occurrence pattern frequently or not, and which inputs occur rarely based on the probability of that number, so that's why we easily take judgments for occurring patterns of the inputs.

IV. FUTURISTIC SCOPE AND IT'S APPLICATIONS

Software testing is one of the most challenging issues for today's information age and software industry issues. Software Testing contains a very typical and complex procedure to draw out an exact outcomes corresponding to particular input in the form of test cases and test suits. Because a most significant activity, which is Software Quality Assurance are totally based on the designing or modelling of the test cases, but in today's software engineering scenario it's very hard to draw or to find out test cases for each of its values, which placed on the interval set. So how we draw out potential mechanisms to assured that this system works perfectly or adequately and also contains reliability features without reporting or to examine some bugs or errors are to be placed inside the code coverage of software modules are major issues for today's Software Engineering issues. Black Box testing are the most significant testing methodology, which tests software with whole aspects which based on their functionalities and task oriented or task perform features of software systems, but again a one major questions have to arise that can we test software as a whole with all of its input values, unfortunately the answer of this question are No, because when we making some interval set of the values then it's very hard to tests each value which placed inside the interval, so here we elicit value based on the Random manner, that's why to judge or to examine or notify behavior of the some elements are have to be missing. But In this newly proposed model, we use some statistical techniques and distribution theory with the interval arithmetic.

Here we adopt the Gaussian or Normal distribution, because this distribution are based on some probability features and contains continuous distributions approach, that's why we elicit values based on their probability. Using the probability features of values we judge that which value occurring utmost during task performing and we also judge or

ISSN (online): 2581-3048
Volume 5, Issue 8, pp 94-101, August-2021
https://doi.org/10.47001/IRJIET/2021.508016

examine those elements behavior which contains lower probability, because Gaussian Distribution works on the concepts of the theorem of Central of Tendency, here a middle line represents the means and left or right side along the means the elements are equally distributed based on their probabilities. Using the probabilistic features we optimize the error rate and increase the bug detection. Using this newly proposed model the failures rates of software in the near future are strictly optimum. This model also suitable for following major applications of Software Engineering Issues such as:

- Real time and Simulation based Software Systems tests.
- Some Automated framework for complex software projects.
- API testing methodology.
- Checked Quality Assurance and Quality Control of Software Management issues.

There are various automated software testing software such as Selenium, HPE Unified Functional Testing, Test Complete etc. which pick value from in the interval set in the Randomized manner, so using or to applying Gaussian Distribution gives a better results for to design tests cases and tests suits.

To design test cases and tests suits for any software systems are really a very complex and typical tasks, because tests cases or tests suits defined the overall mechanisms and functionality of the software systems, and black box testing play's a major role for designing tests cases, but various previously developed traditional black box testing strategies such as: Equivalent Partitioning, and Boundary Value analysis divide or partitioned some set of numbers into similar classes or to do check software behavior at extreme ends respectively, but here both these model picks value in the Randomized manner, but in this newly proposed model we use some distributions mechanisms such as Gaussian or normal distribution whereas each number defined based on their probability and we select particular input based on their probability, because these probability defines the occurring patterns of particular input, which provides reliability to select input and tests the behavior of software systems on that input. So there are many improvisations have to be done in this model in the upcoming future and this model is very significant to release a bug free software product.

V. CONCLUSION

This newly proposed model comes under the category of Black Box Testing methodology, because this newly proposed model examine and compile the functionalities of the software systems rather path coverage or code coverage or some branching and lopping, but In the previously developed Black Box testing strategies such as Equivalent Partitioning and Boundary Value Analysis (BVA), are also defining the range of the values, but it's not give a crisp clearance and reliability about the selection of the inputs, that In other words we have to say that some previously developed Black box testing strategies also provides the range of input tests values, but it's not give the reliability to select or filter the input values among multiple values which placed within the range that which input values have to be selected for tests data, because it's impossible to tests software systems with all of its input values so we have not any mechanisms to select a desired input values for tests which behavior are either positive (Best) or negative (worst). But In this newly proposed model we use the Gaussian Distribution to select or filter input values based on their probability, because distribution is the perfect methodology for selecting and filtering input values among the set of multiple values. Now a one reason behind the use of Gaussian distribution is that it's a continuous probabilistic distribution that works on the theorem of the Central of Tendency because it's equally distributed values along the means according to their respective probability values. So using the mechanisms of probability we easily select or filter inputs for tests data among the set of multiple input values.

This model works with the modular approach of software system architecture because this model equally applies on each module of the software systems. Some previously Black Box testing methodologies such as Equivalent Partitioning and Boundary Value Analysis (BVA) provides the range of the input values so we filter or to select value from these range through Randomization or Randomly because it's very typical and complex to tests all input values and impossible to tests a whole software systems with each nook and corner so we don't achieve desired output or we don't easily examine the behavior of the software systems with it's input tests values, because we take tests data as a input through the mechanisms of randomization, but In this newly proposed model we take tests data with the mechanisms of Gaussian Distribution.

Gaussian Distribution is a continuous probabilistic approach based distribution, which equally distributes values based on their probabilities, so for filtering or to selection of any input values through their occurrence probability gives better results as compare to Randomization mechanisms. So with the use of Gaussian Distribution this model becomes a fundamental Black Box testing method, which tests or to explore each aspects of software systems in a very reliable manner. This model also categorized data into different categories such as Numeric data, Textual data or some other files data generally called the multimedia data such as image, audio, video or some document files etc. so for numeric data we easily finding range using the Interval Arithmetic, but for some file data such as image, or some textual data we assign a

ISSN (online): 2581-3048
Volume 5, Issue 8, pp 94-101, August-2021
https://doi.org/10.47001/IRJIET/2021.508016

unique integer value for each data item using the mechanisms of Randomization, then further we apply Interval Arithmetic on these values. So here categorization of data also a biggest milestone for successfully tests software systems with whole aspects. We design test cases also for invalid inputs, so at the end we also compare our actual resultant data set with the expected data set and compare the accuracy and correctness of our tests. After done the detailed analysis of data sets at the end we validate the test cases. So this mechanisms applying on each of the module of the software systems, so it's also work with module level. Using this model for tests any software systems functionalities we tests software product in a very reliable and accurate manner.

ACKNOWLEDGEMENT

We are very much grateful to all respected professors of DIET Rishikesh for their kind help, lasting encouragement, valuable suggestion throughout the entire period of our project work. We are highly indebted to his astute guidance, sincere support and boosting confidence to make this Research successful. The acknowledgment will be incomplete if we fail to express our obligation and reverence to our family members and friends whose moral support is great factor in doing this research.

FUNDING

This study was not funded by any other profitable or non-profitable organisations.

CONFLICT OF INTEREST

The authors declare that they have no conflict of interest.

REFERENCES

[1] P. Ron. Software testing. Vol. 2. Indianapolis: Sam's, 2001.

[2] S. Amland, "Risk-based testing:" Journal of Systems and Software, vol. 53, no. 3, pp. 287–295, Sep. 2000.

[3] Redmill and Felix, "Theory and Practice of Risk-based Testing", Software Testing, Verification and Reliability, Vol. 15, No. 1, March 2005.

[4] B. Agarwal et al., "Software engineering and testing". Jones & Bartlett Learning, 2010.

[5] K. Bogdan. "Automated software test data generation". Software Engineering, IEEE Transactions on 16.8 (1990): 870-879.

[6] Jacobson et al. The unified software development process. Vol. 1. Reading: Addison-Wesley, 1999.

[7] Everett et al., "Software testing: testing across the entire software development life cycle". John Wiley & Sons, 2007.

[8] J.Irena. "Software Testing Methods and Techniques", 2008, pp. 30-35.

[9] Guide to the Software Engineering Body of Knowledge, Swebok, A project of the IEEE Computer Society Professional Practices Committee, 2004.

[10] E. F. Miller, "Introduction to Software Testing Technology", Software Testing & Validation Techniques, IEEE, 1981, pp. 4-16

[11] M. Shaw, "Prospects for an engineering discipline of software," IEEE Software, November 1990, pp.15-24

[12] D. Nicola et al. "A grey-box approach to the functional testing of complex automatic train protection systems." Dependable Computing-EDCC 5. Springer Berlin Heidelberg, 2005. 305-317. 181

[13] J. A. Whittaker, "What is Software Testing? And Why Is It So Hard?" IEEE Software, 2000, pp. 70-79.

[14] N. Jenkins, "A Software Testing Primer", 2008, pp.3-15.

[15] Luo, Lu, and Carnegie, "Software Testing TechniquesTechnology Maturation and Research Strategies', Institute for Software Research International-Carnegie Mellon University, Pittsburgh, Technical Report, 2010.

[16] Krithikadatta J, Valarmathi S. Research Methodology in dentistry: Part II — The relevance of statistics in research. J Conserv Dent 2012;15:206-213.

[17] M. S. Sharmila and E. Ramadevi. "Analysis of performance testing on web application." International Journal of Advanced Research in Computer and Communication Engineering, 2014.

[18] S.Sampath and R. Bryce, Improving the effectiveness of Test Suite Reduction for User-Session-Based Testing of Web Applications, Elsevier Information and Software Technology Journal, 2012.

[19] B. Pedersen and S. Manchester, Test Suite Prioritization by Cost based Combinatorial Interaction Coverage International Journal of Systems Assurance Engineering and Management, SPRINGER, 2011.

[20] S. Sprenkle et al., "Applying Concept Analysis to User-sessionbased Testing of Web Applications", IEEE Transactions on Software Engineering, Vol. 33, No. 10, 2007, pp. 643 – 658

[21] C. Michael, "Generating software test data by evolution, Software Engineering", IEEE Transaction, Volume: 27, 2001.

[22] A.Memon, "A Uniform Representation of Hybrid Criteria for Regression Testing", Transactions on Software Engineering (TSE), 2013.

[23] R. W. Miller, "Acceptance testing", 2001, Data retrieved from (http://www.dsc.ufcg.edu.br/~jacques/cursos/map/recursos/Testin g05.pdf)

ISSN (online): 2581-3048
Volume 5, Issue 8, pp 94-101, August-2021
https://doi.org/10.47001/IRJIET/2021.508016

[24] Infosys, "Metric model", white paper, 2012. Data retrieved from (http://www.infosys.com/engineering-services/whitepapers/Documents/comprehensive-metrics-model.pdf)

[25] B. Boehm, "Some Future Trends and Implications for Systems and Software Engineering Processes", 2005, pp.1-11.

[26] R. Bryce, "Test Suite Prioritisation and Reduction by Combinational based Criteria", IEEE Computer Society", 2014, pp.21-22.

[27] M. I. Babar, "Software Quality Enhancement for value based systems through Stakeholders Quantification", 2005, pp.359-360. Data retrieved from (http://www.jatit.org/volumes/Vol55No3/10Vol55No3.pdf)

[28] R. Ramler, S. Biffl, and P. Grünbacher, "Value-based management of software testing," in Value-Based Software Engineering. Springer Science Business Media, 2006, pp. 225– 244.

[29] D. Graham, "Requirements and testing: Seven missing-link myths," Software, IEEE, vol. 19, 2002, pp. 15-17.

Citation of this Article:

Shivankur Thapliyal, Renu Bahuguna, "A Newly Proposed Ambidextrous Software Testing Model Based on Conventional Black Box Testing Strategy Using the Applications of Gaussian Distribution" Published in *International Research Journal of Innovations in Engineering and Technology - IRJIET*, Volume 5, Issue 8, pp 94-101, August 2021. Article DOI https://doi.org/10.47001/IRJIET/2021.508016

****

