ISSN: 2321-7782 (Online)
e-ISJN: A4372-3114
Impact Factor: 6.047

Volume 5, Issue 7, July 2017

International Journal of Advance Research in Computer Science and Management Studies

Research Article / Survey Paper / Case Study

Available online at: www.ijarcsms.com

A Unified Approach for Classification of Software Reliability Models

Kuldeep Singh Kaswan[1]
Department of Computer Science
PDM College of Engg., Bahadurgarh
Haryana – India

Sunita Choudhary[2]
Department of Computer Science
Banasthali University, Banasthali
Rajasthan – India

Kapil Sharma[3]
Department of Computer Science & Engineering
Delhi Technological University
Delhi – India

Abstract: ***There are a number of classification schemes for the software reliability models. These classification schemes follow different parameters for classification as SDLC (Software Development Life Cycle) based, failure count based, nature based, software architecture based, randomness based and data requirement based etc. Emphasis of this paper is on the Software Development Life Cycle based classification and then tries to attach these existing classification schemes with the KS Kaswan purposed classification scheme. This study will provide us a new classification approach, which would help to achieve better confidence level in model selection process.***

Keywords: SDLC, software reliability models classification, architecture based, input domain based models.

I. INTRODUCTION

A software reliability model is the form of a random process that describes the behavior of software failure with respect to time [1].

When we develop software, a number of faults are generated and these faults degrade the efficiency of the software. So to remove and estimate such type of errors in the software development, software reliability models help us. In the past few years a lot of work has been carried out in this field of software reliability models, as the consequences, lots of models are in the market today.

In the initial phase of development of software reliability models, these were viewed as a Birth and Death process by Hudson (1967).

But the first influential model was developed by Jelinski and Moranda (JM Model) in 1972. Later on there were many variations of the JM model in market. Goel and Okumoto (1979), Meinhold and Singpurwalla (1983), Karunanithi (1992), Popstojanova et al. (2000), Teng and Pham (2002), Aljahdali (2008), Sharma et al. (2010), Chatterjee et al. (2012), Inoue and Yamada (2013) and Zheng et al. (2013) and Tyagi and Sharma (2014) have their large contribute in this field.

This paper is organized as follows: section II includes existing classifications based on Software development cycle. In section III classification based on other parameters has discussed. Software Development Cycle Phases with Existing Reliability Models Classifications is given in section IV. Summarization of the whole work is given in section V. Finally, the paper concludes with section VI.

II. Classification based on Software Development Life Cycle (SDLC)

In the past few decades a number of software reliability models have been purposed, designed and modified. These models are very useful but some of these have also been suffered from a lot of critiques. To make it easy to select the best one among the existing models, many researchers' categorized these models based on different parameters. In this section, we have reviewed the existing software reliability classification based on SDLC phases.

Ramamoorthy and Bastani [2] classified the existing software reliability models based on the phases of software development life cycle during which the model is applicable. These models are applicable during the testing, debugging, validation or operational phase.

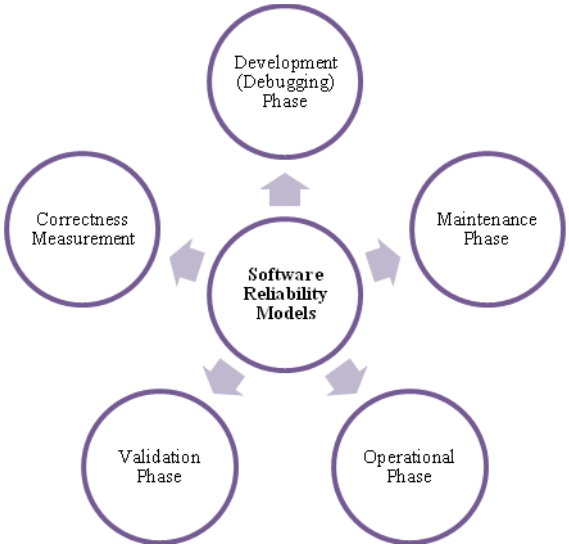

Fig. 1: Classification Based on SDLC

K Sharma [3] discussed about the SDLC based classification of exiting software reliability models. In this classification, the models are categorized as the Requirement phase models, Design phase models, Implementation phase, Testing phase and Validation Phase models.

Zhanwei Hui [4] designed the classification scheme for software reliability models according to the Software Development cycle phases which includes the Requirement phase models, design phase models, Realization models, test and Confirm Phase models.

KS Kaswan [5] classified the models as the same as K Sharma [3] classification but this classification scheme also includes a new family of models as the User Domain models in the maintenance and Operation phase.

III. Classification based on other Parameters

In this section we have reviewed the existing software reliability classification based on different parameters.

a. Based on Nature of Debugging Strategy

Ramamoorthy & Bastani (1986) classified the software reliability models in three categories as [6]

1. Reliability Growth Model

2. Seeding Model

3. Sampling Method Model

Reliability growth models measures and predicts the improvement of reliability through the debugging process. A growth function is used to represent the progress. There are two types of variables, dependent and independent, Reliability, failure rate or cumulative number of errors detected is dependent and time, number of test cases or testing stages are independent variables.

The error seeding models estimates the number of errors in a program by using the capture-recapture sampling technique. These errors can be divided into two categories as existing errors and seeding errors. Existing errors are estimated from the number of seeded errors.

b. Based on Nature of Failure Process

AL Goel (1985) divides models into four classes as given below [7]

1. Time between Failure Models

2. Failure Count Model

3. Fault Seeding Model

4. Input Domain Models

Time between failures models consider the time between failures for the process. Time between two consecutive failures follows a distribution which is depended on the number of remaining faults in the interval.

Failure count model consider the number of faults in specified time intervals rather than in times between failures. Failure rate parameters can be estimated from the observed values of failure counts or from failure counts.

In fault seeding models, a predefined number of artificially generated errors are seeded in the program code. Now by testing we detect the errors and find out the ratio between actual and artificial errors based on the total number of detected errors. This is used to assess software reliability and other relevant measures.

Input domain based models is based on the set of test cases from an input distribution which is assumed to be representative of the operational usage of the program. In this approach we divide the input domain into a set of equivalence classes. From the execution of these set of test cases we obtained the failures and on the behalf of these failures estimate the reliability.

c. Based on Unified SDLC

KS Kaswan (2015) classified the software reliability models according to the unified software development life cycle which includes the four phases as given below [8]

1. Inception

2. Elaboration

3. Construction

4. Transition

In the first phase, inception, of unified software development model, we set the scope of project with its boundary settings, estimate the cost.

In the second phase, Elaboration, continue work on the use-case model, sketch out application architecture and recognize the risk, deal with recognized risk factors to begin validating the system architecture. The final Elaboration phase deliverable is a sketch including expenditure and schedule estimates used for the Construction phase.

Third phase is the construction phase, Unified Modeling Language diagrams are used throughout this phase and encompass Collaboration, State Transition, Activity, Sequence and Interaction Overview diagrams. The implementation is the most significant body of work enhanced during construction.

In the final, transition phase, the system is hand over to the intended users and a complete system testing is being take place.

d. Based on Testing Process

PB Kysetti (2004) emphasized on Black Box and White Box models for software reliability models as given below[9].

1. Single System (White Box)

2. Multi Components System (Black Box)

This model considers the whole software as a single monolithic system and the structure of the model is not considered in the process of reliability estimation of the software system. These models not considered for the structure of the software. Its failure behavior is usually modeled using software reliability growth model.

Multi components based system uses the software architecture of the system. It's a components based software modeling. These models fit in the testing method and the structure of the software being tested.

e. Based on White Box Modeling

White box models are those that model modular software systems considering the architecture of the system. Popstajanova and Trivedi (2001) classified the models as the given below [10]

1. State Based Models

2. Path Based Models

3. Additive Models

State based models uses the program flow graph to represent the architecture of the system taking into the consideration that modules transfer control has a Markov property. The system has the properties of discrete time Markov chain, continuous time Markov chain or semi Markov process.

Path based models has the same common steps as the state based models, consider software architecture with components and interfaces. Initially the different paths in system are obtained so that we can find these reliabilities along the path. The system reliability is the average of all the path reliabilities.

Additive based models do not consider explicitly the architecture of the software. In this class of models the emphasis is on estimating the overall application reliability by components failure data. Additive based models consider software testing phase and each component reliability is modeled by NHPP.

f. Based on Failure Counting Classification

J. D. Musa and K. Okumoto (1984) classified the software reliability models based on the failure counting in software as given below [11]

1. Finite Failure

2. Infinite Failure

All Exponential, Weibull and Pareto are the finite failures categories models.

Software is not completely error free when mean value function of a particular model tends to infinity. Models come under the category of Infinite Failure Time Model.

g. Classification Based on Analytical Modeling

Pham (1999) [12] categories the models as given below:

1. Reliability Growth Models

2. Sampling Method Models

Reliability growth models measures and predicts the improvement of reliability through the debugging process. A growth function is used to represent the progress. There are two types of variables, dependent and independent, Reliability, failure rate or cumulative number of errors detected is dependent and time, number of test cases or testing stages are independent variables.

IV. New Classification Approach for Existing Reliability Model

The integration of the existing classification will create a new structure that will be helpful in understanding the concept of reliability models applicability.

(A) There are two classification scheme, first one is purposed a new classification scheme based on applicability of software reliability models according to the software development life cycle phases including the maintenance phase, which plays a critical role in software development now a days in the competitive world. Second classification scheme based on Unified Software development life cycle, which is applicable on the same class of models.

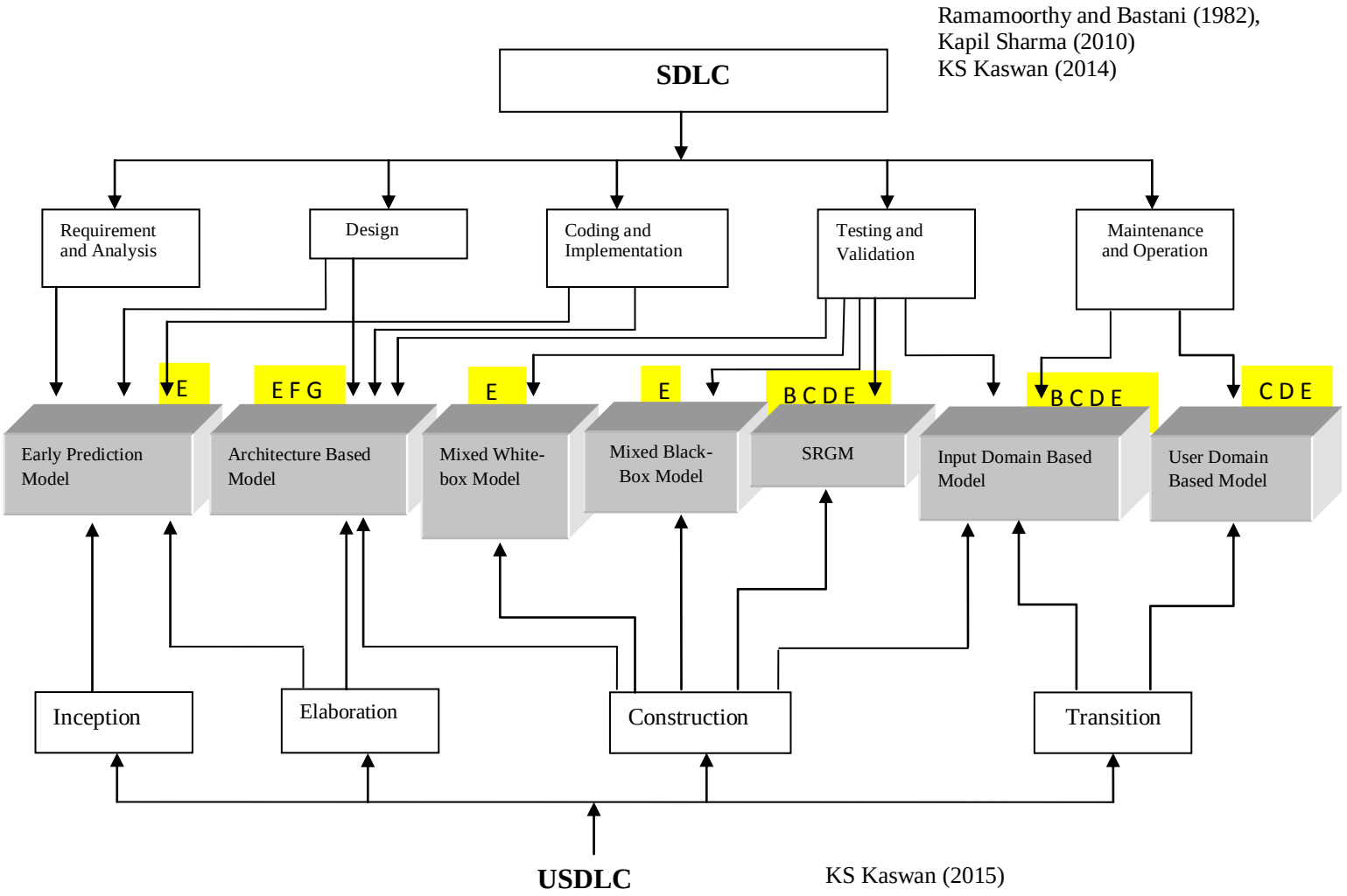

Fig 2: Classification Based on SDLC and USDLC

(B) The number of failures that can be experienced in infinite time is finite or infinite.

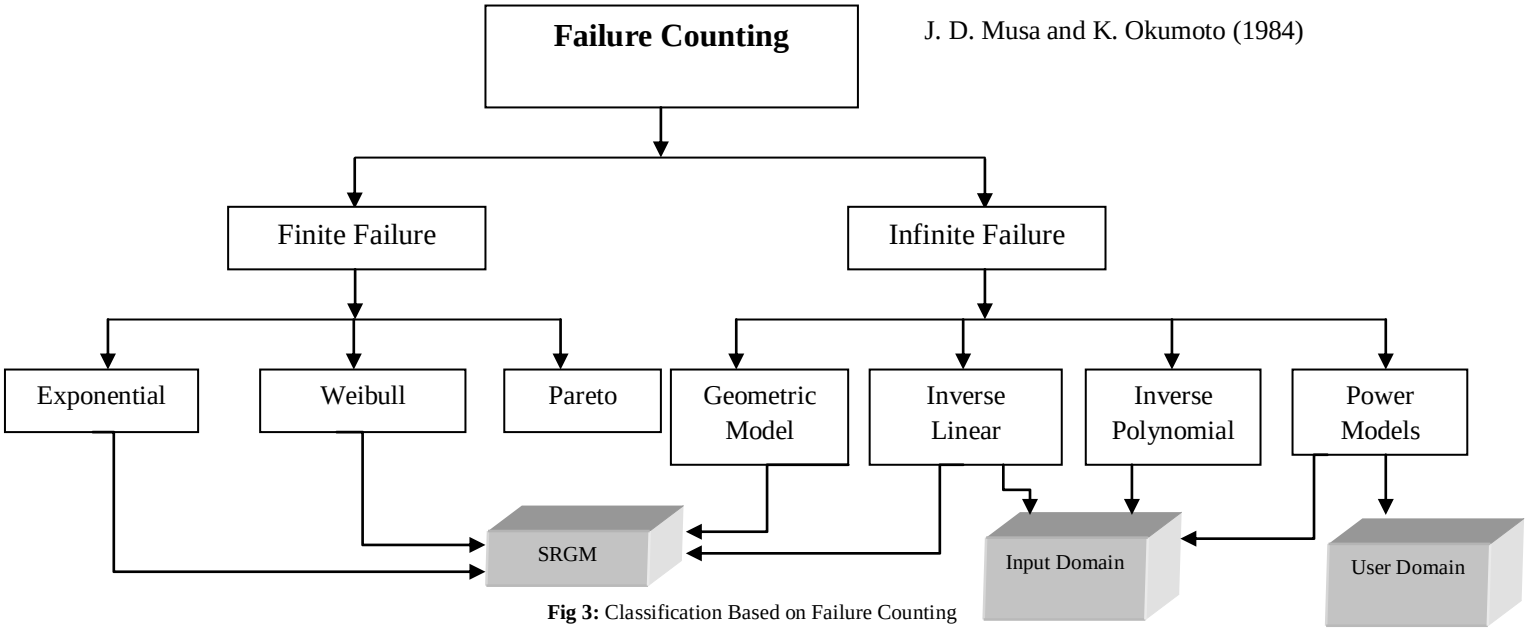

Fig 3: Classification Based on Failure Counting

(C) Software reliability models can be classified based on the nature of the failure process as given below:

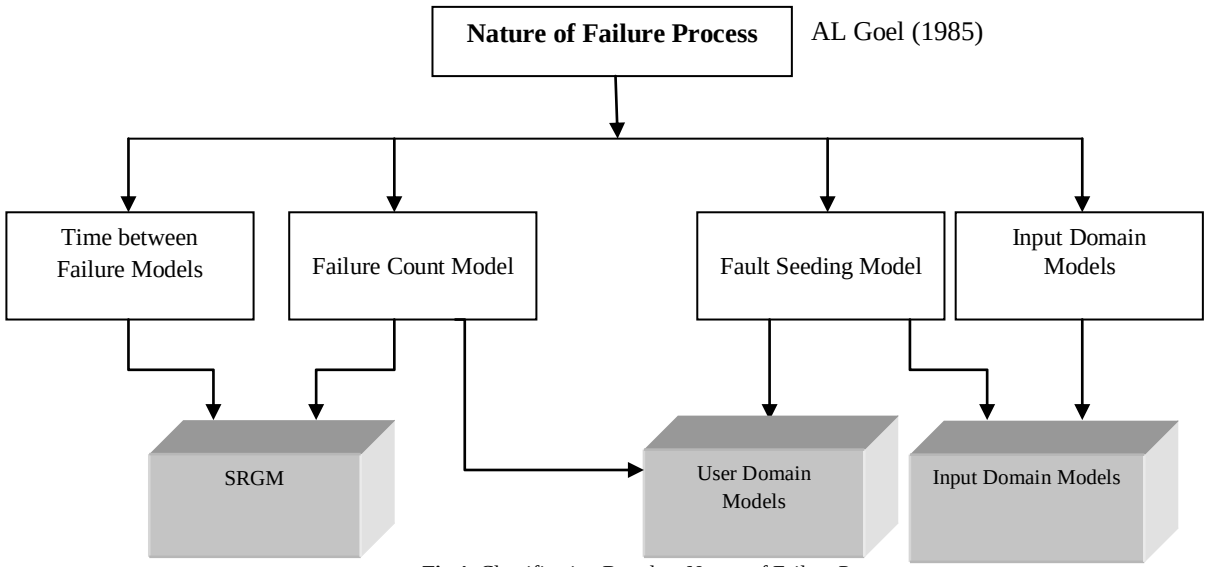

Fig 4: Classification Based on Nature of Failure Process

(D) There are different debugging strategies for the software testing based on these strategies software reliability models can be classified into three different categories as given below:

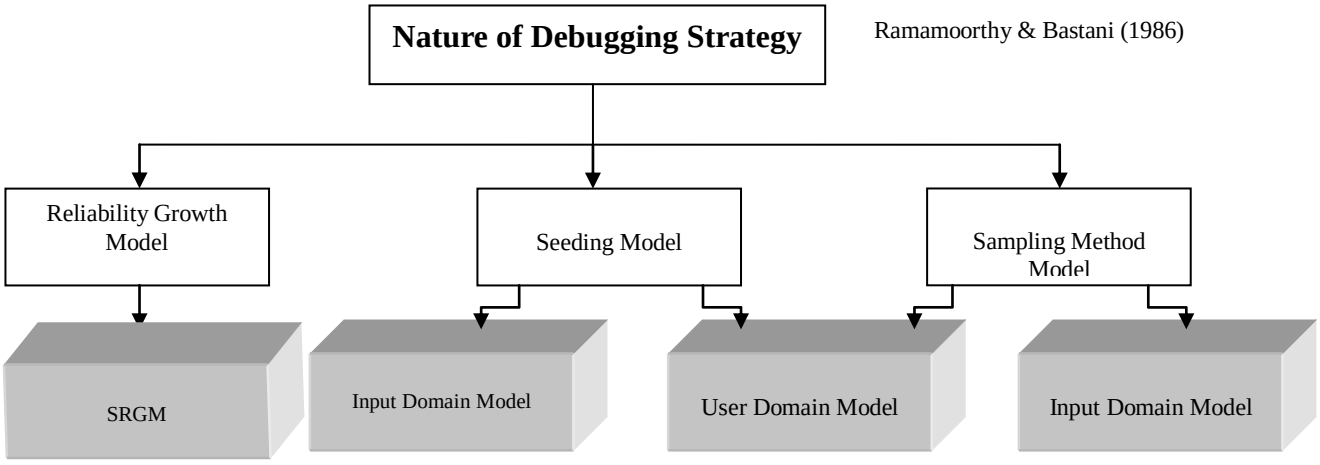

Fig 5: Classification Based on Nature of Debugging Strategy

(E) An analytical model requires some form of data gathered from software failures during software testing and prediction of SWR parameters from the fitted distribution. Analytical models can be further subdivided into Static Models and Dynamic Models based on time dependent behavior of collected data. [12]

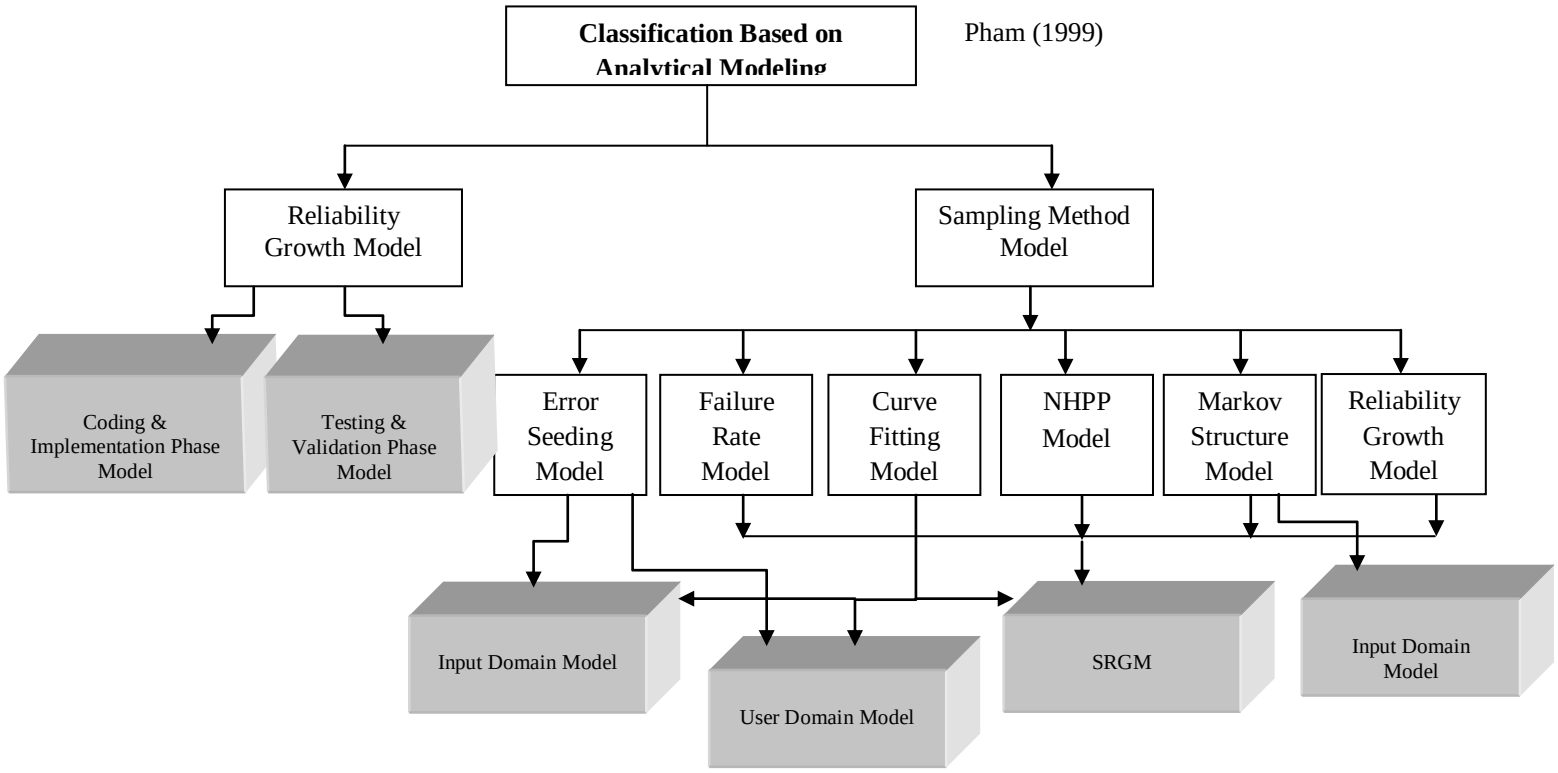

Fig 6: Classification Based on Analytical Modeling

(F) White box modeling approach to software reliability that takes into account the information about the architecture of the software made out of components.

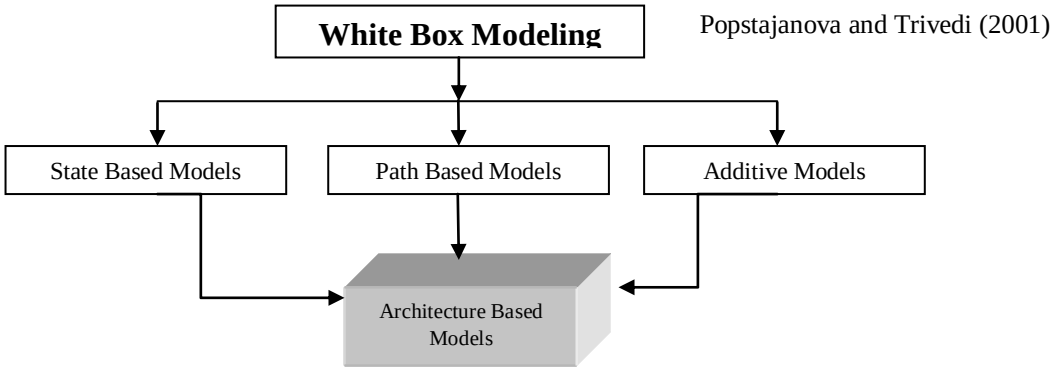

Fig 7: Classification Based on White Box Modeling

(G) In this classification Software Reliability Models are divided into white box and black box models

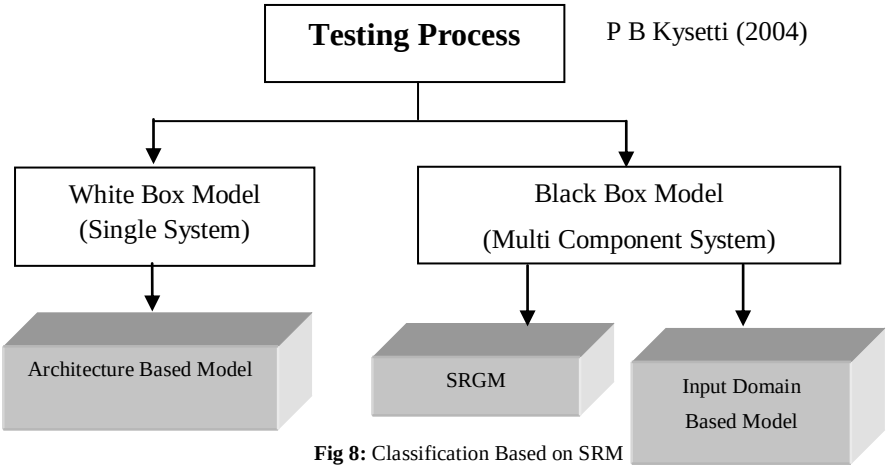

Fig 8: Classification Based on SRM

V. SUMMARIZATION

In this section we summarize the work done by the different authors in the classification of the software reliability models.

Table 1: Summary of Software Reliability Models Classifications

Sr. No.	Author(s)	Year	Criteria for Classification	Proposed Phases by various author(s)	Examples	References
1.	Ramamoorthy & Bastani	1982	SDLC Phases	• Development Phase • Correctness Measurement • Validation Phase • Operational Phase • Maintenance Phase	➢ All existing phases are SDLC phases	[2]
2.	J. D. Musa and K. Okumoto	1984	Failure Counting	• Finite Failure • Infinite Failure	➢ SRGM ➢ Input Domain Based Model ➢ User Domain Based Model	[11]
3.	AL Goel	1985	Nature of Failure Process	• Time between Failure Models • Failure Count Model • Fault Seeding Model • Input Domain Models	➢ SRGM ➢ Input Domain Based Model ➢ User Domain Based Model	[7]
4.	Ramamoorthy & Bastani	1986	Nature of Debugging Strategy	• Reliability Growth Model • Seeding Model • Sampling Method Model	➢ SRGM ➢ Input Domain Model ➢ User Domain Model	[6]
5.	Pham	1999	Analytical Modeling	• Reliability Growth Models • Sampling Method Models	➢ Reliability Growth Model ➢ SRGM ➢ Input Domain Model ➢ User Domain Model	[12]
6.	Popstajanova and Trivedi	2001	White Box Modeling	• State Based Models • Path Based Models • Additive Models	➢ Architecture Based Models	[10]
7.	PB Kysetti	2004	Testing Process	• Single System (White Box) • Multi Components System (Black Box)	➢ Architecture Based Model ➢ SRGM ➢ Input Domain Model Based Model	[9]
8.	K Sharma	2010	SDLC Phases	• Requirement Phase • Design Phase • Implementation Phase • Testing Phase • Validation Phase.	➢ All existing phases are SDLC phases	[3]
9.	Zhanwei Hui	2011	SDLC Phases	• Requirement Phase • Design Phase • Realization • Test • Confirm Phase	➢ All existing phases are SDLC phases	[4]
10.	KS Kaswan	2014	SDLC Phases	• Requirement and Analysis • Design • Coding and • Implementation • Testing and Validation • Maintenance and Operation	➢ All existing phases are SDLC phases	[6]

11.	KS Kaswan	2015	Unified SDLC	• Inception • Elaboration • Construction • Transition	➢ Early Prediction Model ➢ Architecture Based Model ➢ Mixed White-box Model ➢ Mixed Black-Box Model ➢ SRGM ➢ Input Domain Based Model ➢ User Domain Based Model	[8]

This table revealed that the existing software reliability models classifications can be attached with the different phases of the software development life cycle.

VI. CONCLUSION

This paper present an overview on the different existing classifications of software reliability models and showed how these classifications can be attached with the classification purposed by KS Kaswan. This paper will provide user an insight into the classification structure of such models that would be helpful in determining which model to use in a given software development environment. We can easily reject 80% of the models which is not useful for us as per requirement and choose the particular model out of the remaining which is not so difficult task.

References

1. Michael R. Lyu, "Software Reliability Engineering: A Roadmap", IEEE Computer Society, Future of software Engineering, pp: 153-170, 2007.
2. Ramamoorthy&Bastani82 C. V. Ramamoorthy, F. B. Bastani, Software reliability — status and perspectives, IEEE, Trans. Soft. Eng., SE-8, No. 4, July 1982, pp. 354 - 371.
3. Kapil Sharma, Rakesh Garg, C. K. Nagpal, and R. K. Garg, "Selection of Optimal Software Reliability Growth Models Using a Distance Based Approach", IEEE Transactions on Reliability, Vol. 59, no. 2, June 2010
4. Zhanwei Hui, Xiaoming Liu and Song Huang, "Software Reliability Model Metrics: Precision and Robustness", International Journal of Advanced Computer Science, Vol. 1, No. 6, Pp. 250-256, 2011.
5. K. S. Kaswan, Sunita Choudhary and Kapil Sharma, "A New Classification and Applicability of Software Reliability Models", International Journal of Advance Research in Computer Science and Management Studies, Vol. 2, Issue 7, pp. 99-104, 2014
6. F.B. Bastani, C.V. Ramamoorthy, "Input-domain based models for estimating the correctness of process control programs", Reliability Theory (Serra & Barlow, Eds), 1986, pp 321-378; North Holland.
7. Goel85 A. L. Goel, Software Reliability Models: assumptions, limitations, and applicability, IEEE, Trans. Soft. Eng., SE-11, No. 12, Dec. 1985, pp. 1411 - 1423.
8. Kuldeep Singh Kaswan, Sunita Choudhary and Kapil Sharma, "Unified Software Development Process Based Classification of Software Reliability Models", pp. 2078-2083, 2015 IEEE.
9. Praveen Babu Kysetti, "Optimal Test Case Selection for Multi-Component Software System", the Department of Industrial & Manufacturing Systems Engineering, B.S., Bangalore University, India, 2004.
10. Popstojanova, K.G., and Trivedi, K.S., (2001), "Architecture-Based approach to reliability assessment of Software Systems," Preprint submitted to Elsevier 2001, Citseer.
11. J. D. Musa and K. Okumoto, "A Logarithmic Poisson Execution Time Model for Software Reliability Measurement", pp. 230-238, 1984 IEEE.
12. Pham, H. and Zhang, X., "A software cost model with warranty and risk costs", IEEE Transactions on Computers, 48: 71-75, 1999.

Author(s) Profile

Kuldeep Singh Kaswan, was born in Haryana, India in 1982. He received Doctors Degree in Computer Science under the faculty of Computer Science at Banasthali Vidyapith, Rajasthan. He has obtained his Master Degree in Computer Science and Engineering from Choudhary Devi Lal University, Sirsa (Haryana) and Bachelor Degree from Kurukshetra University, Kurukshetra (Haryana). His area of interests includes software engineering and reliability.

Sunita Choudhary, was born in Rajasthan, India in 1978. She received Doctors Degree in Computer Science under the Faculty of Computer Science at the Banasthali University (Rajasthan), India. She has obtained his Bachelor of Science in Computer Science and Master of Computer Application Degrees from Banasthali University in 1999, and 2001 respectively. Presently, she is working as an associate professor in Department of Computer Science at Banasthali University, Rajasthan, India. Her research interests include Algorithms and Distributed Computing.

Kapil Sharma, was born in Haryana, India in 1977. He received Doctors Degree in Computer Science and Engineering under the Faculty of Engineering and Technology at the M. D. University, Rohtak (Haryana), India. He has obtained his Bachelor of Engineering and Master of Technology Degrees in Computer Science & Engineering and Information Technology from M. D. University, Rohtak, and IASE, Rajasthan, India in 2000, and 2005 respectively. Presently, he is working as an associate professor in Department of Computer Science at Delhi Technological University, Delhi, India. His research interests include software engineering, and reliability.

