22

# Mutation testing applied to validate specifications based on Petri Nets (*)

*S.C.P.F. Fabbri*[1]
*Universidade Federal de São Carlos-Departamento de Computação*
*S.Carlos-SP,Brazil,PB.676,13565-905,sfabbri@icmsc.sc.usp.br*
*J.C.Maldonado*
*Inst. de Ciências Matem. de S.Carlos-USP - Depto. de Computação*
*S.Carlos-SP,Brazil,PB.668,13560-970,jcmaldon@icmsc.sc.usp.br*
*P.C.Masiero*
*Inst. de Ciências Matem. de S.Carlos-USP - Depto. de Computação*
*S.Carlos-SP,Brazil,PB.668,13560-970,masiero@icmsc.sc.usp.br*
*M.E.Delamaro*
*(1) Ph.D. Candidate at Instituto de Física de São Carlos - USP*
*S.Carlos-SP,Brazil,13560-970,med@icmsc.sc.usp.br*
*E.Wong*
*Bell Communications Research*
*445 South Street,Morristown,NJ 07960,USA,ewong@bellcore.com*

**Abstract**

Testing is one of the fundamental software development life cycle activities. Considering Reactive Systems such as: metro control, patient hospital monitoring and communication protocols, the testing activity becomes more relevant as errors in these systems can promote severe economical and social losses. The objective of this work is to evaluate the adequacy of applying the Mutation Analysis criterion to validate Petri Net based specifications. A set of mutation operators for Petri Nets, a key point for using Mutation Analysis, as well as the results of applying manually these operators to a Petri Net modeling a level 3 protocol are presented. We also examine the ideas of constrained and randomly selected mutation in this context.

**Keywords**

Petri Nets, Mutation Analysis, Test and Validation.

(*) This work was partially financed by a grant from CNPq and CAPES

G. V. Bochmann et al. (eds.), *Formal Description Techniques VIII*

# 1 INTRODUCTION

The most used graphical techniques for the specification of Reactive Systems - a class of systems whose main feature is to interact with the environment reacting to stimuli - are: Finite State Machines (FSM) (Gill,1962), Statecharts (Harel,1987) and Petri Nets (Peterson,1981). Testing and Validation of the behavioral aspect of systems described by these techniques is done, in general, by reachability analysis, many types of simulation and, for FSMs, by several test sequences generation methods, for example, the W method (Chow,1978).

We are interested in exploring the adequacy, the cost and how testing techniques can be applied and complement each other to validate system specifications. In recent work we investigated the use of the Mutation Analysis criterion to test and validate Finite State Machines (Fabbri,1994). At program level it is well known that Mutation Testing is complementary to other testing techniques and some empirical studies provide evidences of its effectiveness (Wong,1994b). In our manually applied experiment we also obtained evidences that the FSM testing activity can be improved using such a criterion.

The objective of this work is to explore the Mutation Analysis criterion application to validate the behavioral aspect of reactive systems that is specified through Petri Nets, aiming at giving insights for theoretical and empirical studies within a broader objective of comparing several techniques for validation of the dynamic of reactive systems. In this work, the Mutation Analysis criterion is manually applied, based on a preliminary mutation operator set, to a Level 3 Protocol specification, extracted from (Tanenbaum,1989). We also examine the ideas of constrained and randomly selected mutation (Wong,1994b).

This paper is organized as follows: in Sections 2 and 3 the Mutation Analysis criterion and Petri Nets are briefly introduced; in Section 4 the results obtained from the application of the Mutation Analysis, randomly selected and constrained mutation criteria to validate Petri Nets Models are presented. Concluding remarks are presented in Section 5.

# 2 MUTATION ANALYSIS

Mutation Analysis aims at creating the confidence that a program P is correct producing, through small syntactic changes, a set of mutant programs that are similar to P, and generating test cases that are capable of causing behavioral differences between P and each one of its mutants. These changes are based on an operator set called *mutation operators*, a key factor for the success of Mutation Analysis. To each operator it is associated an error class that we want to reveal in the program. Operators are usually defined based on the *competent programmer* hypothesis, which affirms that a program produced by a competent programmer is either correct or near correct. Another hypothesis considered by Mutation Analysis is the *coupling effect* that, according to DeMillo (1978), can be described as: 'test data that distinguishes all programs differing from a correct one by only simple errors is so sensitive that it would also implicitly distinguish more complex errors'.

Mutation Analysis consists of four steps: mutant generation, P execution based on a defined test case set T, mutant execution based on T and adequacy analysis. If a mutant $P_i$ presents different results from P, it is said to be dead (distinguished); otherwise, it is said to be alive. This fact can occur due to two reasons: either there are no test cases in T that are capable to

distinguish $P_i$ from P or $P_i$ and P are equivalent; this information is obtained interactively as an entry from the tester, who plays a relevant role in this decision, as the equivalence question is, in general, undecidable. In the first case, a test case must be inserted in T aiming at killing this mutant. Our objective must be to find a test case set able to kill all non-equivalent mutants; such a test case is considered adequate to test P.

The *Mutation Score*, computed from the number of mutants generated, the number of equivalent mutants and the number of mutants killed, provides an objective measure for the confidence level of the adequacy of a test case set T.

## 3 PETRI NETS

Petri Net (PN) (Peterson,1981) is a modeling technique; it has two basic components: a set of *places*, *P*, and a set of *transitions*, *T*, related by two functions connecting transitions to places: the *input* function *I* that defines, for each transition $t_j$ , the set $I(t_j)$ of *input places* and the *output* function *O* that defines, for each transition $t_j$, the set $O(t_j)$ of *output places*. Formally, a Petri Net C is defined as a four-tuple C, were:

$C = (P,T,I,O)$ $\qquad I = T \rightarrow P^{\infty}$

$P = \{ p_1, ..., p_n\}$ $n \geq 0$ (finite) $\qquad O = T \rightarrow P^{\infty}$

$T = \{ t_1, ..., t_m\}$ $m \geq 0$ (finite) $\qquad P \cap T = \phi$

A Petri Net can also be represented graphically: circles represent places and bars represent transitions. The input and output functions are represented by direct arcs from the places to the transitions and from the transitions to the places, respectively.

The execution of a Petri Net is controlled by the position and movement of marks, named tokens, that are represented by small solid dots inside the places; the tokens are moved by *firing* transitions of the net. Transitions must be enabled to fire and this is true if each of its input places has at least one token in it. A transition fires by removing one token from each of its input places, and putting one token in each of its output places. The distribution of tokens in the Petri Net defines the net state which is called marking and is represented by a vector $\mu = (\mu_1, \mu_2,...,\mu_n)$, that gives the number of tokens in each place, i. e., $\mu(p_i) = \mu_i$. In Figure 1a, a Petri Net specifying a level 3 protocol, to illustrate the ideas explored herein, is presented (Tanenbaum,1989).

## 4 MUTATION TESTING AND PETRI NETS

Originally, the Mutation Analysis criterion was used for program testing; in the context of this research, it is 'mapped' to another application level, that is, validation of behavioral aspect of reactive system specifications. With this purpose in mind, we revisited the basic hypothesis of the Mutation Analysis criterion at the specification level (Fabbri,1994). To design the mutant operators we considered the *designer competent hypothesis* and at first, to apply them the *coupling effect* that, in fact, must be validated in this context. So, given a specification S, a mutant set of S is generated, $\phi(S)$, and we consider that a test set T is adequate for S with relation to $\phi$ if for each specification Z of $\phi$, either Z is equivalent to S or Z differs from S at least on a test point.

Although there is not a direct link between FSM errors and Petri Nets errors, to design the Petri Net mutation operators we inspired ourselves on Chow's FSM error classification: operator, transfer and extra/missing state errors (Chow,1978). For example, a transfer error in a Petri Net could be associated to an output function or to an input function. Considering the protocol of Figure 1a, we could produce the mutant shown in Figure 1b, that contains a specification error caused by eliminating a place from the input set of a transition and including it in the input place of another transition (the input-shifted operator). The initial set of mutation operators for Petri Nets includes: input-missing, input-extra, input-shifted, input-exchanged, output-missing, output-extra, output-shifted, output-exchanged, wrong-initial-marking (Fabbri,1995). Following we give the definition of some of these operators.

*Op.1 - Input-missing* is defined by:

for each $t_j$, $0 \le j \le m$, r mutants are defined, $0 \le r \le |\, I(t_j)\,|$ such that
$|\, I(t_j)_r\,| = (\,|\, I(t_j)\,| - 1\,)$ where
$I(t_j)_r = I(t_j) - \{p_i\}$, for each $p_i \in I(t_j)$

*Op.2 - Input-extra* is defined by:

for each $t_j$, $0 \le j \le m$, r mutants are defined, $0 \le r \le (\, n - |\, I(t_j)\,|\,)$ such that
$|\, I(t_j)_r\,| = (\,|\, I(t_j)\,| + 1\,)$ where
$I(t_j)_r = I(t_j) \cup \{p_i\}$, for each $p_i$ such that $p_i \in P$ and $p_i \notin I(t_j)$

*Op.3 - Input-shifted* is defined by:

for each $t_j$, $0 \le j \le m$, r mutants are defined, $0 \le r \le (\,|\, I(t_j)\,| * (\, m - 1\,)\,)$ such that
$|\, I(t_i)_r\,| = |\, I(t_j) - 1|$, where $I(t_j)_r = I(t_j) - \{p_k\}$
and $|\, I(t_i)_r\,| = |\, I(t_i) + 1|$, with
$I(t_i)_r = I(t_i) \cup \{p_k\}$, for all $i \neq j$ such that $p_k \notin I(t_j)$

*Op.4 - Input-exchanged* is defined by:

for each $t_j$, $0 \le j \le m$, r mutants are defined, $0 \le r \le |\, I(t_j)\,| * (\, n - |\, I(t_j)\,|\,)$ such that
$|\, I(t_j)_r\,| = |\, I(t_j)\,|$ where
$I(t_j)_r = I(t_j) \cup \{p_k\} - \{p_i\}$,
for each $p_i \in I(t_j)$ and
for each $p_k \in P$ and $p_k \notin I(t_j)$

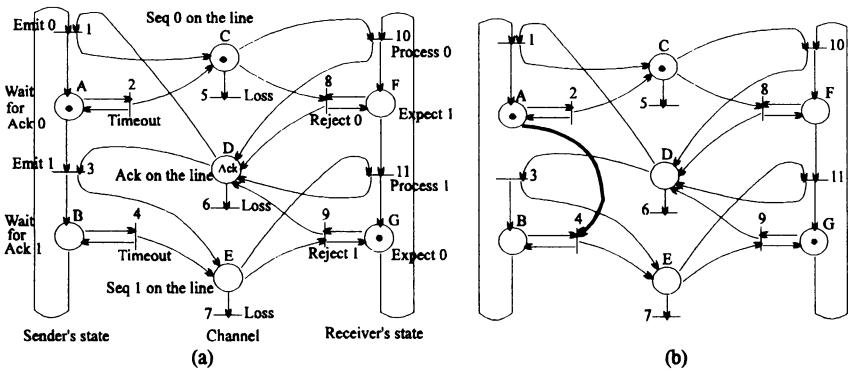


**Figure 1** (a) A Level 3 Protocol specified by a Petri Net (Tanenbaum,1989)
(b) A mutant of (a) generated by input-shifted operator.

Considering the initial mutation operator set 643 Petri Nets mutants were generated for the protocol of Figure 1a, shown in Table 1. Mutants which could be classified as "generalized Petri Nets" were included in the analysis (Peterson,1981); if we have decided otherwise, the cost of applying Mutation Analysis could be reduced as the number of mutants would have decreased.

Consider initially the set T-inic ={ (10, 3, 11, 1), (10, 3, 11, 1, 2 ,5, 10, 2, 6, 8, 3, 7, 4, 11, 4, 9)} of test sequences that represents a typical transmission without losses of messages and retransmission of messages and ack frames. The set T-inic is all-transition-adequate, i. e., guarantees that all transitions are fired at least once. The mutation score of T-inic according to the operators set defined is 0.8616. We have that 13.84% of the mutants are still alive. If one of the non-equivalent mutants is taken as the proposed specification, the sequences used would not be sensible enough to reveal this type of error. Thus, based on the set of alive mutants the test case set can be improved. Consider, for example, the mutant of Figure 1b. To kill this mutant a test sequence like ( 2, 5, 2, 10, 6, 2, 8, 3, 7, 4, 11, 6, 4, 9, 1) needs to be included in the test case T-inic.

To decide if a mutant is dead or not we compared the markings $\mu$ and $\mu'$ of the original and of the mutant Petri Net, respectively, after each test sequence used, without distinguishing the number of tokens present in each place. A mutant is dead if $\mu \neq \mu'$ and:

$$\mu \neq \mu' \Leftrightarrow \exists\, p_i \mid (\mu(p_i) = 0 \wedge \mu'(p_i) \neq 0) \vee (\mu(p_i) \neq 0 \wedge \mu'(p_i) = 0)$$

We then improved T-inic trying to follow the semantics aspects of the example identifying sequences of most typical situations: transmission sequences with retransmission of package 10; transmission sequences with retransmission of package 1; and transmission sequences with noise in the transmission and reception channels, causing package acknowledgment retransmissions. Our final set of sequences was then: T-end = { (10, 3 11, 1), ( 10, 3, 11, 1, 2, 5, 10, 2, 6, 8, 3, 7, 4, 11, 4, 9), ( 2, 5, 10, 3, 11, 1), (10, 6, 2, 8, 3, 11, 1), ( 10, 3, 4, 7, 11, 1), ( 10, 3, 11, 6, 4, 9, 1), ( 2, 5, 10, 3, 4, 7, 11, 1), ( 10, 6, 2, 8, 3, 11, 6, 4, 9, 1), ( 2, 5, 2, 10, 6, 2, 8, 3, 7, 4, 11, 6, 4, 9, 1)}.

Now, with T-end we obtained a mutation score of 0.9466; only 35 mutants remained alive, as shown in Table 1. This could go on extending T-end aiming at killing the alive mutants, if possible, due to the equivalence problem (Reachability Problem (Peterson,1981)). Equivalence of models is an important issue to the effective application and automatization of Mutation Analysis. We do not dwell further on this subject which is outside of the scope of this paper.

**Table 1** Synthesis of the results obtained for the test case set T-end

| *Operator* | *Killed* | *Alive* | *Total* |
|---|---|---|---|
| Op.1 Input-missing | 15 | 2 | 17 |
| Op.2 Input-extra | 60 | 0 | 60 |
| Op.3 Input-shifted | 127 | 13 | 140 |
| Op.4 Input-exchanged | 90 | 0 | 90 |
| Op.5 Output-missing | 16 | 0 | 16 |
| Op.6 Output-extra | 52 | 9 | 61 |
| Op.7 Output-shifted | 149 | 11 | 160 |
| Op.8 Output-exchanged | 80 | 0 | 80 |
| Op.9 Wrong-initial-marking | 19 | 0 | 19 |
| Total | 608 | 35 | 643 |

In our example there is no equivalent mutant as all the remaining mutants can be killed. In the process of killing the alive mutants some atypical situations, that would not be considered in a typical test, came out. For example, one of the mutants generated by the input-missing mutation operator require a test sequence that reflects a situation where the same package is processed twice. In this way, the Mutation Analysis criterion leads us to an improvement in the testing activity.

### *Alternate mutation – constrained and randomly selected mutation*

It could be argued that the cost of Mutation Analysis is impractical in real situations as it has been done in the context of program testing. Although some applications would justify a high cost of VV&T activities, we explore alternatives of mutation aiming at reducing its cost and making feasible its use in applications where time and budget are more restrict.

Constrained mutation and randomly selected mutation are ways to deal with this matter reducing the number of mutants. Wong et al (Wong,1994b) gave evidences at program level that examining only a small percentage of the mutants may be a useful heuristic for evaluating and constructing test sets in practice. Following we define the alternative mutation criteria:

- **Randomly Selected Mutation Criterion**: examines a small percentage of randomly selected mutants of each mutant type and ignore the others. In this paper we use 10% of each mutant type. Hereafter, we refer to this criterion as 10% mutation.
- **Constrained Mutation Criterion**: examines only a few specified types of mutants and ignores the others. In this paper we use two constrained mutation criteria: input/output missing and input/output missing/exchanged.

To compare these criteria we use the same metrics proposed by Wong et al (1994b). Two Cost Metrics: i) $C_1$ – the percentage of size reduction of the test set with respect to mutation analysis; ii) $C_2$ – the percentage of decreasing in the number of mutants examined with respect to mutation analysis; and one Strength Metric: $C_3$ – the percentage of the relation between the number of mutants distinguished by a test set and the total number of non-equivalent mutants.

In our case, we used only one adequate test set for each criteria proposed in this paper. Initially we analyzed the adequacy of T-inic and T-end with respect to the alternate mutants and completed these sets in order to get the adequate test sets: T-inic-10% and T-end-10% for 10% mutation; T-inic-2op and T-end-2op for input/output missing mutation; and T-inic-4op and T-end-4op for input/output missing/exchanged mutation.

Following we analyzed the adequacy of each alternate adequate test set with respect to the mutation criterion and to the other alternate criteria. The results obtained are summarized in Table 2. From this table we can compute the metrics $C_1$, $C_2$ and $C_3$ for each of the defined alternate mutation criteria, given in Table 3. The results obtained give evidences that the alternate mutation criteria provide significant cost reduction in terms of the number of test sequences required and the number of mutants examined.

**Table 2** Strength Analysis Data

| *sequence* | *length* | *10% mutation (65 mutants)* | *input/output missing (33 mutants)* | *input/output missing/ exchanged (203 mutants)* | *mutation (643 mutants)* |
|---|---|---|---|---|---|
| T-inic | 2 | 0.9077 | 0.8485 | 0.9656 | 0.8616 |
| T-end | 9 | 0.9693 | 0.9394 | 0.9902 | 0.9456 |
| T-inic-4op | 7 | 0.9693 | 1 | 1 | 0.9332 |
| T-end-4op | 11 | 0.9847 | 1 | 1 | 0.9643 |
| T-inic-2op | 7 | 0.9693 | 1 | 1 | 0.9332 |
| T-end-2op | 11 | 0.9847 | 1 | 1 | 0.9643 |
| T-inic-10% | 7 | 1 | 0.9394 | 0.9902 | 0.9145 |
| T-end-10% | 11 | 1 | 0.9697 | 0.9951 | 0.9643 |

**Table 3** Cost and Strength Data

| *metric* | | *10% mutation* | *input/output missing* | *input/output missing/exchanged* |
|---|---|---|---|---|
| cost | $C_1$ | 78.13% | 78.13% | 78.13% |
| | $C_2$ | 89.89% | 94.87% | 68.43% |
| strength | $C_3$ | 91.45% | 93.32% | 93.32% |

## 5 CONCLUDING REMARKS

The results presented in this paper give evidences, agreeing with the results reported in (Fabbri,1994b), that using Mutation Analysis to validate the behavioral aspect of systems, in this case Petri Net based specifications, contributes to the improvement of this activity; it also motivates the development of a supporting tool.

Currently, a tool is being developed, named Proteum/FSM, whose aim is to support the application of Mutation Analysis to validate specifications based on FSM. Based on the experience of developing Proteum/FSM is simple to devise a tool to support the validation of Petri Nets based on the Mutation Analysis criterion.

We explored two alternative criteria to apply Mutation Analysis: randomly selected mutation and constrained mutation. Evidences were obtained that the same benefits of using these alternative criteria in the context of program testing can be obtained in the context of Petri Nets: i) both 10% randomly and constrained mutation provide significant cost reduction in terms of the number of test sequences required and the number of mutants examined; and ii) the benefits in cost reduction is accompanied by a small strength loss in the ability to distinguish non-equivalent mutants.

Although these studies are presented in a limited scope, they provide insights that motivate a more systematic study that could lead to conclusive remarks. The development of a tool such as the one discussed above would enable to conduct a more significant study, including effectiveness analysis (Wong,1994b).

We plan to continue this research reviewing and complementing the set of mutation operators defined in this paper; a deeper study of typical errors made by a specifier using Petri Nets should be conducted aiming at refining and expanding the initial operator set. Specific classes or extensions to Petri Nets may lead to the definition of new operators or even elimination of some of them. The same study can be done considering specific properties of Petri Nets as well as the criteria to characterize a mutant as alive, dead, anomalous or equivalent. Furthermore, the coupling hypothesis will be explored and validated in the context of Petri Nets, as in this paper we used only first-order mutants (Budd,1980). Finally, two other issues will be addressed: investigation of heuristics for determining equivalence of Petri Nets and realization of empirical studies to evaluate alternative strategies to apply the Mutation Analysis criterion, in the same vein of the ideas discussed in this work.

## 6 REFERENCES

Budd, T.A.; DeMillo, R.A.; Lipton, R.J.; Sayward, F.G.(1980) Theoretical and Empirical Studies on Using Prog Mutation to Test the Functional Correctness of Prog., *7th ACM Symposium on Principles of Programming Languages*, Jan.

Chow, T.S.(1978) Testing Software Design Modeled by Finite-State Machines. *IEEE Transactions on Software Engineering*, SE(4(3)), 178-87.

DeMillo, R.A.; Lipton, R.J.; Sayward, F.G.(1978) Hints on Test Data Selection: Help for the Practicing Programmer, *Computer*, Vol. 11(4), 34-41.

Fabbri, S.C.P.F.; Maldonado, J.C.; Masiero, P.C.; Delamaro, M.E.(1994) Mutation Analysis Testing for Finite State Machines, *in Proceedings of the Fifth International Symposium on Software Reliability Engineering*, California, Nov.

Fabbri, S.C.P.F.; Maldonado, J.C.(1995) Mutation Analysis Applied on Petri Nets, Technical Report, ICMSC-USP.

Gill, A.(1962) *Introduction to the Theory of Finite-State Machines.* New York, McGraw-Hill.

Harel, D.(1987) Statecharts: A Visual Formalism for Complex Systems, *Science of Computer Programming*.

Peterson, J.L.(1981) *Petri Net Theory and the Modeling of Systems*, Prentice-Hall, Inc., Englewood Cliffs, NJ.

Tanenbaum, A.S.(1989) *Computer Networks*, (2nd. edition), Prentice Hall.

Wong, W.E.; Maldonado, J.C.; Delamaro, M.E.; Mathur, A.P.(1994a) Constrained Mutation in C Programs, *in Proceedings of the VIII SBES - Simpósio Brasileiro de Engenharia de Software*, Curitiba, Oct.

Wong, W.E.; Maldonado, J.C.; Mathur, A.P.(1994b) Mutation versus All-uses: An Empirical Evaluation of Cost, Strength and Effectiveness, *in Proceedings of the First IFIP/SQI International Conference on Software Quality and Productivity*, Hong Kong, Nov.

## 7 BIOGRAPHY

*Sandra C.P.F.Fabbri* received her M.S. in Computer Science from ICMSC-University of São Paulo. She is a PhD Candidate at IFSC-University of São Paulo. She is an assistant professor at Federal University of São Carlos. Her research interests include systems analysis and specification, focusing on testing and validation.

*José Carlos Maldonado* received his B.S. in Electronic Engineering from University of São Paulo; the M.S. degree in Electronic and Telecommunications from INPE (Spacial Research Institute); and his D.Sc. in Electric Engineering/Computation and Automation from University of Campinas (UNICAMP). He has worked from 1979 to 1985 at INPE. Currently he is an associate professor at University of São Paulo (USP) and a Visiting Professor at Purdue University, USA. His interests include software engineering environments, software testing and validation, reliability and software quality evaluation. He is a member of the SBC (Brazilian Computer Society).

*Paulo C. Masiero* is a Full Professor at the University of São Paulo. He has worked in the field of computing in both industry and academia since 1978. His interests include validation of specifications, software engineering environments, formal methods, object-oriented analysis and design methods and hypertext modeling. He holds a D. Sc. degree in Management Information Systems and an M.Sc. in Computer Science both from University of São Paulo. He is a member of the IEEE Computer Society and the SBC (Brazilian Computer Society).

*Marcio Delamaro* is a PhD Candidate at IFSC-University of São Paulo, graduated in Computer Science in 1985. He has received his M.S. from ICMSC-University of São Paulo in 1993. He worked for 5 years as Software Analyst. His research interests include systems analysis and specification, focusing on testing and validation. Currently he is a Visiting Scholar at Purdue University.

*W. Eric Wong* received his B.S. in Computer Science from Eastern Michigan University; and his M.S. and Ph.D. in Computer Science from Purdue University in 1988, 1991, and 1993, respectively. From January 1994 to February 1995 he worked at Hughes Network Systems. Currently, he is a research scientist at Bellcore. Dr. Wong's research interests are in software testing, reliability, maintenance, and reusability. He is a member of ACM and IEEE Computer and Reliability Society.