The Tester's Dashboard: Release Decision Support

Robert V. Binder
System Verification Associates, LLC.
Chicago, Illinois, USA
rbinder@ieee.org

Peter B. Lakey
Cognitive Concepts, Inc.
Webster Groves, Missouri, USA
peterlakey@sbcglobal.net

***Abstract*— A perennial and critical question for software testing is: When can we stop? Apart from "when time is up," many test coverage measures have been proposed and used. Automated test generation from a Markov model can track model coverage. The reliability demonstration chart uses sequential sampling to evaluate when the failure intensity observed in test is, within a certain confidence level, an acceptable estimate of post-release failure intensity. Embedded in a decision support model, this produces risk-adjusted guidance for whether to continue testing or to reject or accept the system under test. We introduce the relative proximity metric to indicate the extent to which test cases generated from a Markov model approximate complete coverage of steady state. Test suites generated from two case study models show that presentation of these metrics as a dashboard can better substantiate a release decision.**

Keywords-reliability, model-based testing, Kullback Distance, Markov model coverage

I. INTRODUCTION

In software reliability engineering (SRE), two criteria can be used to stop testing: (1) meeting a failure intensity objective (FIO) and (2) meeting a minimum statistical confidence in the extent of testing [10].

Demonstrating achievement of a specified reliability target is relatively straightforward. A calculation of observed results is compared against the established target. If the observation is better than the established threshold, then the goal is achieved. For instance, the goal may be set as 99% probability of success for a given operational scenario. If one hundred randomly generated scenario-based tests results in a single system failure then the measured reliability is 0.99, which just meets the FIO. The question is, can we conclude that it's safe or appropriate to release the system under test (SUT) based on this observation?

Our confidence in a reliability estimate is the crux of the problem. The key determining factor in helping to answer the question "when can we stop?" is how closely the sample represents the population. Predictive SRE models require test suites composed of test cases that occur in the same relative frequency and interleaving as the operational profile. A quantitative comparison of reliability-oriented stop-test models appears in [8]. An approach using the RDC in concert with other metrics is presented in [4].

We should not stop testing when the risk of inadequate field reliability is too high. At the same time, we should stop testing when we have sufficient confidence that adequate observed reliability will translate to adequate field reliability. We believe that using three established quantitative measurements together can lead to better release decisions than applying any of them in isolation.

1) Model Coverage. With the use of model-based testing to develop and generate an operational profile, production and analysis of coverage metrics that take advantage of the mathematical formalisms of Markov models are feasible and meaningful. For example, as test generation covers more model states and arcs, correspondingly more unique SUT inputs and responses are exercised.
2) Test Progress. The well-known reliability demonstration chart indicates when reliability observed during testing indicates that the system under test will meet (or not meet) a field reliability goal, within a certain confidence level.
3) Relative Proximity. For test suites that have well-defined and quantifiable behavior, we introduce the *relative proximity* metric. This can provide an indication of the extent to which the sequences in a test suite have achieved the steady state behavior of the Markov model.

A. Model-based Testing for Reliability Estimation

The idea of testing based on a model of system behavior has gained considerable traction over the past dozen years or so [7] [13]. In particular, the finite state machine has been adopted by many as a mechanism for storing a representation of complex system behavior, particular applications that depend heavily on state.

To support software reliability evaluation, model-based statistical testing has been proposed [12]. Such a construct enables us to approximate the operational profile relatively closely compared to previous methods that employed an essentially flat profile [9]. With this approach the operational profile is stored in the form of Markov chain usage model (MCUM), a finite state machine that has probability values assigned to each transition. The states represent unique states of use; transitions represent stimuli to the system. In this paper, we limit our discussion to tests generated from an operational profile represented as an MCUM.

A test suite generated from a representative operational profile is a precondition for reliability analysis. This may be met when some form of Monte Carlo simulation uses an operational profile to determine the frequency and sequence

of operation test cases. With simulation, more likely operations will be generated more often than less likely ones; some model states will be reached rarely, in proportion to the modeled profile. Further, representation of an operational profile as a MCUM allows analysis of the profile and testing results using the well-understood mathematics of Markov systems.

B. Model Coverage Metrics

The main structural metrics for a MCUM are state coverage and transition coverage. State coverage is the ratio of the total number of model states to those reached in least one test case. Transition coverage is the ratio of the number of unique transition arcs in the model exercised in at least one test case. For example, a test suite that reaches 100% of model states and 100% of transitions between states would be considered highly covered. Achieving high structural coverage reduces the risk that rare input/state combinations are not exercised during test. However, owing to the stochastic nature of the model and the simulation that produces the tests, even 100% structural coverage of a model does not necessarily result in high relative proximity and an "accept."

C. Reliability Demonstration Chart [*]

A reliability demonstration chart (RDC) shows when cumulative failure observations indicate that a failure intensity objective has or has not been met within a specified level of risk [9] [10]. It is assumed that success and failure data is produced by a test suite consistent with an operational profile [10]. An RDC graph plots cumulative number of failures versus normalized time of failure. Figure 1 shows an example chart produced by RDC.xls [5]. The graph has three regions that indicate (1) that the tests are strong evidence that the system under test (SUT) will achieve its failure intensity objective (green, "accept"), (2) that more testing is needed to make a determination (yellow, "continue"), or (3) that the SUT is unlikely to achieve its failure intensity objective (red, "reject").

The model's three decision outputs (accept, continue, reject) directly reflect the producer's tolerance for error in estimating the actual reliability of the SUT. The user's tolerance for risk of an inappropriate decision is expressed with three input parameters: the producer's risk threshold, α: the highest probability the producer is willing to accept that the model will incorrectly indicate "reject" when the SUT would meet or surpass its failure intensity objective; the customer's risk threshold, β: the highest probability the producer is willing to accept that the model will incorrectly indicate "accept" when the SUT would not meet or surpass its failure intensity objective; and the discrimination ratio γ: the error in estimating failure intensity the developer is willing to accept. "The discrimination ratio is the ratio of the upper test MTBF to the lower test MTBF and is a measure of the power of the test to reach a decision quickly and, together with the decision risks, define a sequential test's accept-reject criteria. In general, the higher the discrimination ratio, the shorter the test." [1]

These parameters determine the graph region boundaries. They represent the producer's expectation of both opportunity (time to market) and failure (rework and remediation) costs. For example, lowering risk parameter values expands the continue region towards the northwest and southeast corners. This typically means more testing will be needed to cross a boundary while reducing the chance that a satisfactory SUT is rejected or an unsatisfactory SUT is accepted. In a time-to-market race, higher risk tolerances could be used, narrowing the continue region. In either case, decision makers can weigh the consequences and make a release decision using explicit risk quantification.

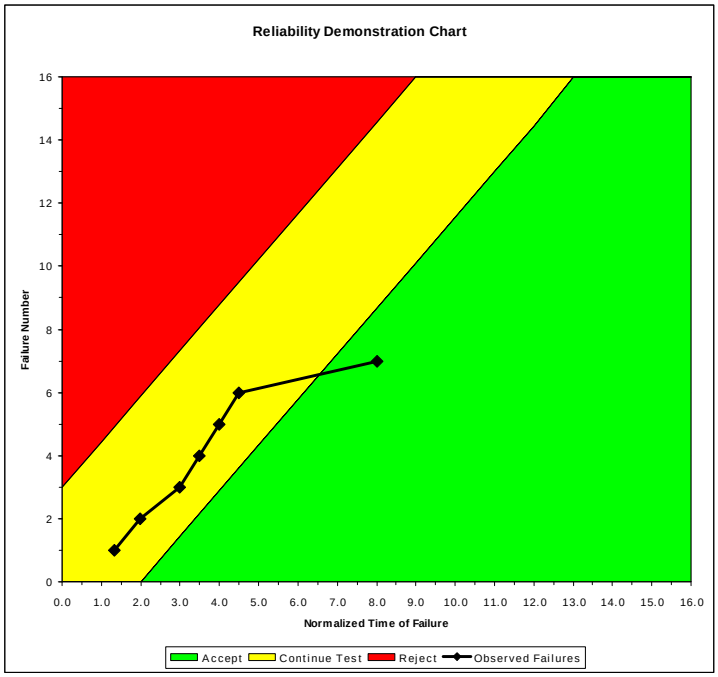


Figure 1 Typical Reliability Demonstration Chart

D. Relative Proximity

The Kullback-Leibler (KL) distance is an information-theoretic measurement of the extent to which two observations (samples) of the same event space (population) are different [6]. The KL distance has been expressed in a discrete form, useful for comparing two data sets, and a continuous form, useful for comparing sample parameters. When the KL distance is zero, the samples are statistically equivalent, although the actual content may be quite different.

> "…KL-divergence of "model from reality" is also useful even if the only clues we have about reality are some experimental measurements. … it tells you about

[*] This section adapted from [5], with permission.

surprises that reality has up its sleeve or, in other words, how much the model has yet to learn." [14]

The KL distance was originally defined between expected and observed discrete occurrence frequencies defining the entropy of an abstract communication channel, and is also known as "relative entropy." We use it here to provide additional insight into the statistical closeness (proximity) of test suites. To distinguish this from the many extant variations on the KL distance, we call our model *relative proximity*.

For software testing, both forms of the KL distance can provide an indication of the difference between the distribution expressed by a sample of tests and the distribution inherent in a model of the complete population of possible tests. It is relevant because a complex software system may require many thousands of tests before a test suite reaches all the individual behavior variations.

To illustrate how relative proximity complements coverage and the RDC chart, we use the TGAM [3] tool which computes the Kullback discriminant, the "expected value of the log-likelihood ratio of two stochastic processes" [11]. This formulation of the KL distance allows direct comparison of the steady state probabilities for states and transitions (the expected usage model) and their evolving counterparts as a simulation produces a test suite.

II. CASE STUDIES

A. Stochastic Models

The two application models described here were developed under government-funded research and development [3]. The purpose of each was to demonstrate the feasibility (but not the application) of model-based software testing and that meaningful software reliability estimation can be supported with stochastic model-based testing.

MCUMs were developed for each system. Test cases were generated using a Monte Carlo simulation of input events to cause the model's states and transitions produce a stream of inputs and expected resultant states. As each model state was reached, a pseudo-random simulation algorithm selected the next transition (user action.) As the number of generated test cases increases, the distribution of these sequences approaches the model distribution.

We have used the actual numbers for tests and coverage to illustrate how different analyses and representations provide complementary information.

B. Assumed Failure Rates

None of the tests were executed, however, because that was beyond the scope of the research effort. Therefore, no actual test results, including failures, were produced. To demonstrate the dashboard, we synthesized failures for the generated test suites. We assumed that certain types of stimuli would be more likely to result in failure. An analysis of 25 samples of 100 tests generated from the GMD model described below conveyed that one of the relatively rare operations was observed as many as 12 times in one sample, while only a single time in another. Assuming that system failures are uniformly distributed among certain rare events then we believe it's reasonable to assume that experienced failures may vary similarly across test samples of equivalent size.

The failure rates used in these examples are higher than would be acceptable in practice and were selected to simplify this notional discussion of the decision support metrics.

C. Word Processing Application

This model represents a simplification of a user's interaction with a word processing (WP) application. The model contains a set of stimuli that represent actions that can be taken by the user such as clicking on various icons, selecting options from pull-down menus and, of course, entering and editing text. It also contains a set of state variables that dictate when the user can apply a particular stimulus and what the expected system response is when the input is applied. These variables include the number of open documents, the size of the documents, the edit status of the documents, the current user view, etc. As a whole, the model is represented in a finite state machine where all the transitions (inputs) and states (unique state variable combinations) are contained. Each walk through the model represents a single usage session, where the user launches the application, performs a series of actions, and then exits the application. An example test is:

1. Start the WP app
2. Click New Document Icon
3. Enter Text
4. Click Search Icon
5. Enter Text
6. Click Save Icon
7. Select File/Close
8. Click New Document Icon
9. Enter Text
10. Click Save Icon
11. Exit the WP app

This particular sample model contains 25 unique stimuli and 127 usage states. It is a small, simplified model. An application like Microsoft Word 2007 would typically have hundreds of unique input types and the model would encompass perhaps tens of thousands of usage states.

D. Ground-Based Midcourse Missile Defense

The GMD model is a very small subset of a system model developed under a research contract for the Missile Defense Agency [3]. Representing usage of a Ground-based Midcourse Defense missile defense system, the model consisted of four independent and linked sub-models. The sub-models correspond to the four main operational modes of the system: No Threat, Booster Stage, Midcourse Stage and Terminal Stage.

The GMD No Threat sub-model is used here. This covers GMD system behaviors starting with No Threat present state, and then assumes a long range missile is detected from some source. It exits the sub-model once the missile ascends to the Booster Stage. In it, the system detects missiles, identifies where they are coming from, determines the altitude of the missile, and allows a user sitting in a mission control center to switch between display screens. Typical stimuli include the following: SS18 Launched from Enemy A, DF5A Launched from Enemy B, ICBM Directed Toward Continental US, ICBM Directed Toward Hawaii, ICBM Detected Early, ICBM Altitude Increases, Tracker Selects ICBM View, Tracker Selects Global View, Exit to Booster Stage Model.

The state variables in the model include Number of Missiles, Type of Missile, Range of Missile, Direction of Missile, Altitude of Missile, Current Display Screen. Each state in the sub-model is a unique combination of values for this set of usage variables. This particular sub-model consists of 18 stimuli (unique inputs) and 366 states. Here is a typical test case generated from the model:

1. Start Detection
2. SS24 Launched from Enemy A
3. ICBM Detected Early
4. ICBM Directed Away from US
5. Threat Ignored
6. DF5A Launched from Enemy B
7. ICBM Detected Early
8. Tracker Selects ICBM View
9. Tracker Selects Global View
10. ICBM Directed Toward US Hawaii
11. Exit to Booster Stage Model

III. Application and Analysis

A. The Dashboard

A dashboard presents summary information of critical measurements. The Tester's Dashboard has three gauges: Model Coverage, Test Progress, and Proximity.

1) Model Coverage is shown in a bar graph. The state and transition coverage data was generated along with the test cases using a software tool called the Test Generation and Analysis Module (TGAM) [3], and represents the coverage if all generated tests are executed. An Excel spreadsheet generated the bar graphs. This indicates how much modeled behavior has been explored in a test suite.

2) Test Progress is shown with a reliability demonstration chart. The charts were generated using the open source RDC program [5]. It takes as input the risk parameters α, β, and γ, and failure observations. This indicates the sufficiency of the test suite in predicting the failure intensity objective.

3) Proximity is shown with the Kullback distance graphing applet [2]. The graphed values were obtained from the TGAM tool, using the Kullback discriminant [11]. This indicates the extent to which the test suite has achieved the all of the variation allowed in the model.

B. WP Dashboard

Figure 2 shows successive dashboard read-outs for the WP model. Each corresponds to a certain number of generated tests. When 10 random tests were produced model state coverage is 60% and transition coverage is 40%. The RDC plot indicates no failures and no guidance after the first 10 tests. The proximity of 184 is assumed to be minimal.

After 100 tests were generated the model coverage metrics increased to 78% and 81%, respectively. Here a single failure occurred (in the fabricated failure data), but this is not yet visible on the RDC plot. Proximity has increased to 1.0, which indicates that the test suite has very nearly achieved all the variation allowed in the model. However, with 6 failures, the RDC indication is to continue testing. Finally, at 3,000 tests the proximity is almost zero and the RDC indicates acceptance.

Several samples of test data were generated from the Word example and proximity data was obtained for each sample. Figure 3 plots the proximity for additional samples. The results are fairly consistent across samples. As the number of tests reaches 1,000, proximity approaches zero.

C. GMD Dashboard

Figure 4 shows successive dashboard read-outs for the GMD model. The information summarized in the charts and data for the GMD example are significantly different from the WP example. After 10 test cases, relatively little has been accomplished in terms of coverage and proximity is assumed to be at a minimum. Even after 100 tests model coverage is low and the proximity is increasing. Not until 1,000 test cases do we start to get an indication that progress is being made toward model coverage and the sample is starting to approximate the population through the proximity measure. Also note a single failure there.

After 5,000 tests have been generated and executed we finally start to see a picture coming into focus. The state coverage achieved is now 76% and transition coverage 58%. The proximity has increased, consistent with higher confidence that the sample is representative. The RDC chart here is interesting in that it shows the failure rate is very near the accept region.

After 10,000 tests the picture changes regarding whether to accept or reject the system. Model coverage creeps up just a little with the 5,000 additional tests. Although the model is relatively small (366 states), after generating 10,000 tests we still haven't reached 66 of the model states, achieving 82% state coverage. This stems from the fact that some state/event combinations are very unlikely.

The proximity of 6 is much closer. Note, however, that the failure rate has jumped significantly, moving the RDC indicator from near accept to near reject. The proximity trend is shown for the GMD system in Figure 5.

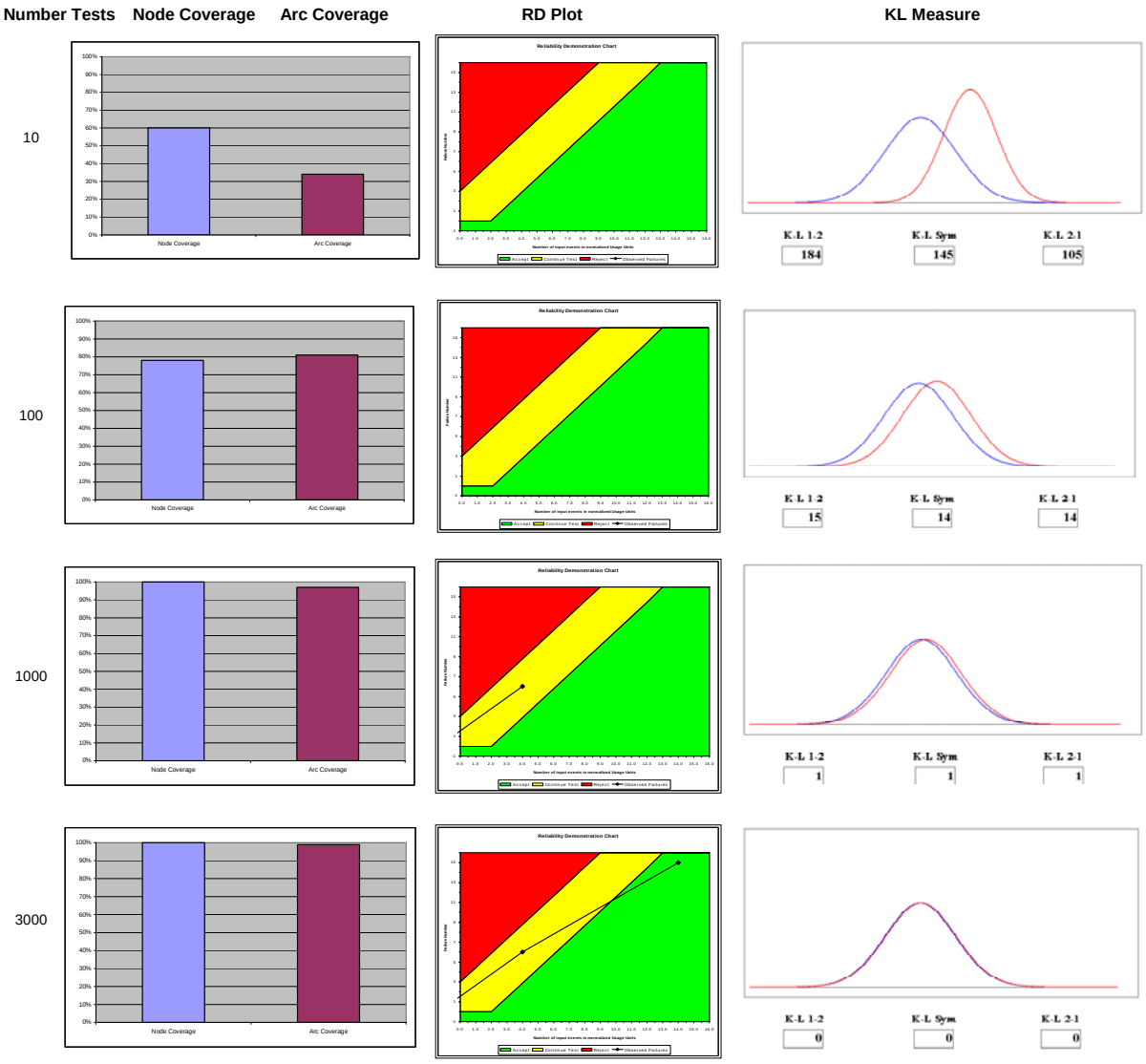


Figure 2 WP Model Dashboard

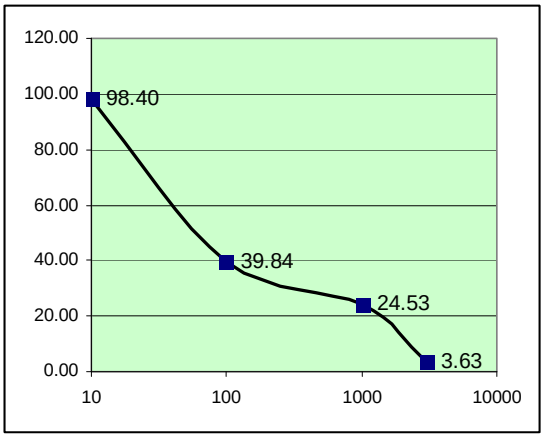


Figure 3 Proximity Trend for WP Model

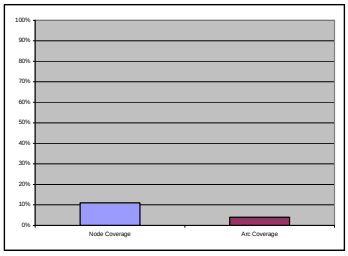
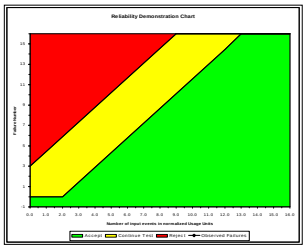
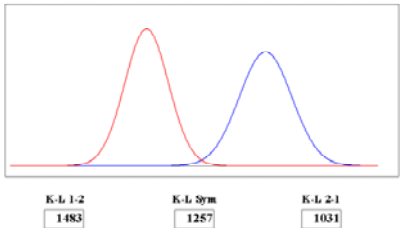
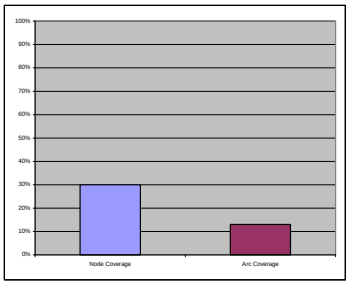
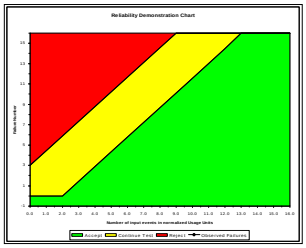
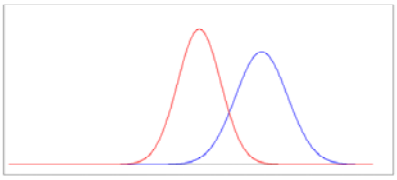
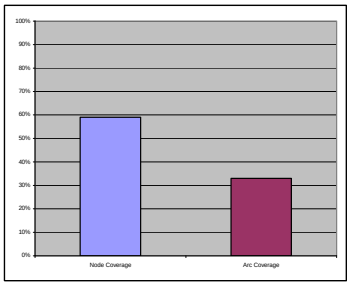
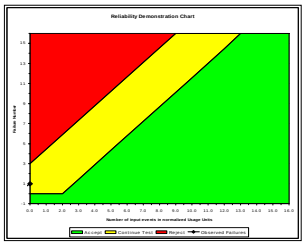
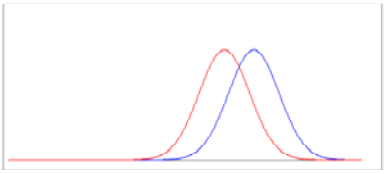
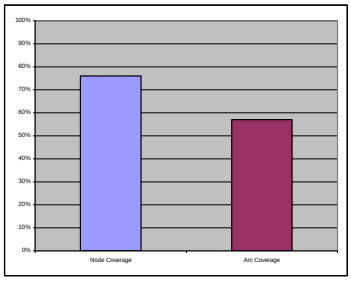
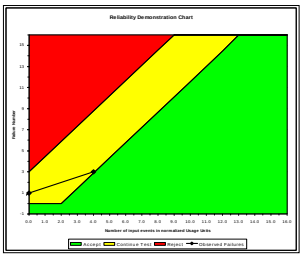
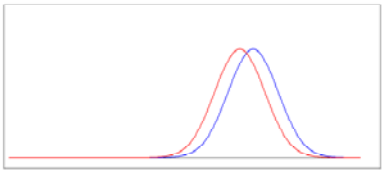
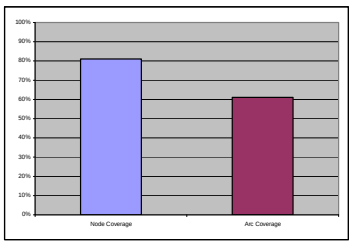
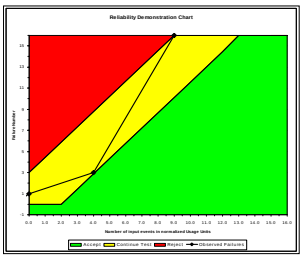
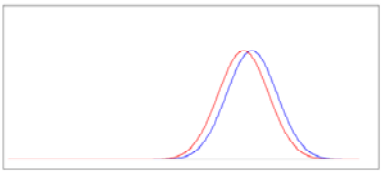
Number Tests	Node Coverage	Arc Coverage	RD Plot	KL Measure
10				
100				
1000				
5000				
10000				

Figure 4 GMD Model Dashboard

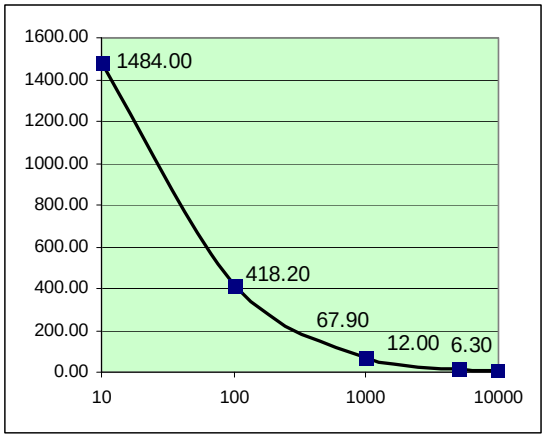


Figure 5 Proximity Trend for GMD Model

IV. DISCUSSION

A. *Complementary Views Reduce Risk*

As structural model coverage goes up, proximity increases. For the WP model, as coverage approaches 100%, proximity approaches zero. This result is expected because with an increase in the number of tests the states and transitions covered by those tests should increase. As these numbers increase the probability distribution of the sample looks more and more like the distribution inherent in the model. The proximity should increase (indicated with decreasing KL discriminant values).

Note, however, the different rates of change in GMD model coverage and proximity. Many more tests are needed to achieve high model coverage and low proximity. The word model has 127 states compared to 366 for the GMD model; so one could expect at least three times as many tests would be needed to achieve higher coverage. The primary explanation, however, is in the relative frequency values assigned to transitions in the GMD model. Some transitions are assigned values of 0.001 or lower, which means states that are reachable through those transitions will be rarely visited. The GMD model contains many states that would be reached only after hundreds of thousands of input sequences were simulated.

For both of the figures, the RDC chart does not become relevant until the number of tests is relatively large. Note how the plotted results in the two examples differ. The WP model plot has a continue point and an accept point. Because the coverage is high and proximity nearly maximal, we can be more confident in the RDC model's indication that the WP SUT is acceptably reliable. Absent the coverage and proximity information, one might wonder if the test suite was truly representative of the model.

For the GMD example, however, following the RDC guidance alone could lead to prematurely accepting the system when in fact it may not meet target reliability objectives. We can interpret the jump from near accept to near reject as a reflection that the second set of 5,000 generated tests reached parts of the usage model that were sufficiently different from the first 5,000, revealing a substantially different failure modes. Such a result is not uncommon in models that are complex and that include rare transitions [10].

B. *Conclusions*

This study raises additional questions. For example, we see that the relative proximity can also take account of expected and actual failure rates, providing an indication of the extent to which an actual failure distribution is consistent with the event distribution implied by a failure intensity objective. This could provide information to assess failures in the same way that the model-based relative proximity helps to assess structural model coverage.

Our motivation in preparing the paper was to investigate methods for helping us to answer the question, "when is it OK to stop testing?" We harness two model-based testing measures with the familiar reliability demonstration chart to provide a more complete picture. Our basic conclusion is that employing a dashboard that includes the three methods used in tandem enables a more complete perspective than utilizing the RDC chart alone.

REFERENCES

[1] —. *Handbook for Reliability Test Methods, Plans, and Environments for Engineering, Development Qualification, And Production.* MIL-HDBK-781A. U. S. Department Of Defense, 1996.

[2] —. "Kullback-Leibler Distance". AI ACCESS Glossary of Data Modeling. http://www.aiaccess.net/English/Glossaries/GlosMod/e_gm_kullbak.htm. (accessed August 2, 2010).

[3] —. *Test Generation and Analysis Module*, Army Phase II Small Business Innovative Research Contract with Picatinny Arsenal Advanced Research Development and Engineering Center (ARDEC), Contract No. W15QKN-06-C-0110. February 2008.

[4] Pankaj Bhawnani, Behrouz H. Far, and Guenther Ruhe. "Explorative study to provide decision support for software release decisions," *21st IEEE International Conference on Software Maintenance*, 200, 617-620.

[5] Robert V. Binder. *RDC: The Reliability Demo Chart*. http://sourceforge.net/projects/rdc/ (accessed August 2, 2010).

[6] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory*, New York: Wiley, 1991.

[7] Wolfgang Grieskamp, Nico Kicillof, and Keith Stobie. "Model-based quality assurance of protocol documentation: tools and methodology," *Journal of Software Testing, Verification and Reliability.* Forthcoming.

[8] Amjad Hajjar, T. Chen, I. Munn, A. Andrews, M. Bjorkman. "High quality behavioral verification using statistical stopping criteria," *Proceedings of Design, Automation, and Test in Europe* (DATE 01), 1530-1591.

[9] John D. Musa,, A. Iannino, and K. Okumoto. *Software Reliability: Measurement, Prediction, Application*, McGraw-Hill, 1987.

[10] John D. Musa, *Software Reliability Engineering: More Reliable Software Faster and Cheaper*, 2nd. ed., AuthorHouse, 2004.

[11] Kirk Sayre and Jessie H. Poore. Stopping criteria for statistical testing. *Information and Software Technology*. 42(12): 851-857, September 2000.

[12] James A. Whittaker and Michael G. Thomason. A Markov chain model for statistical software testing. *IEEE Transactions on Software Engineering*, 20(10):812–824, October 1994.

[13] Utting, Mark and Bruno Legeard. *Practical Model-Based Testing: A Tools Approach*. Morgan-Kaufmann, 2006.

[14] Wikipedia contributors, "Kullback–Leibler divergence," Wikipedia, The Free Encyclopedia, http://en.wikipedia.org/wiki/Kullback%E2%80%93Leibler_divergence (accessed August 2, 2010).