

# Marrying Software Fault Injection Technology Results with Software Reliability Growth Models

Jeffrey Voas, Cigital    Norman Schneidewind, Naval Postgraduate School

**Abstract -- Combining fault injection technology with software reliability growth models promises to improve both the reliability of software and accuracy of software reliability predictions. This result is achieved by injecting faults to corrupt the internal states of the software to test its fault tolerance. At the same time, this process yields realistic failure data for input to reliability models for improved accuracy of prediction.**

## I. INTRODUCTION

We report on a new area of SRE -- marrying software Fault Injection Technology (FIT) results with software reliability growth models (SRGMs). Both (FIT) [2] and SRGMs [1] have been applied independently to improve the reliability of software. However, to our knowledge, this is the first attempt to unite these two approaches to assure the reliability of software. At this stage of our research, we have developed the concepts of the approach. Our plan is to apply this approach to the Space Shuttle avionics software, where SRGMs have been applied [1] but not FIT. First, we describe FIT. Then, we show how it can be combined with SRGMs predictions. Next, we identify the types of software that are candidates for FIT. Last, we draw conclusions about the benefits of the combined approach.

## II. FIT TECHNOLOGY

Software FIT [2] provides a means for dynamically demonstrating that software quality (and in particular, *testability*, *robustness* and *fault tolerance*) is built and designed into code. FIT has long been used in the physical and hardware engineering communities for similar purposes (e.g., crash testing automobiles), and now, the philosophies that govern those approaches are available for software. Software FIT is a form of software testing that is distinct from traditional black-box, system-

level software testing; it allows software acquirers to determine software robustness when the software is fed anomalous input events. This is something that traditional software testing typically fails to address. Therefore, software FIT provides a means for assessing how problems propagate through and across software systems.

We now discuss how this applies to the advancement of SRGMs. Once FIT injects corrupt states into executing code, one of two events occurs: either the output of the software is modified from what it would have been had the corrupt state not been injected, or the software does not respond to the corrupt state and the output remains the same. Either way, information about the “anomaly revealing ability” of the test case used for that execution is produced (which is a function of what type of corruption was employed). What this demonstrates is the ability of that test case to force corrupt internal states to propagate in such a way that we can observe that the corrupt state existed at one time. By using FIT, we gain hints as to whether that test case can detect faults. If this process is replayed with enough test cases, and if we find that the test profile being employed does frequently propagate anomalies, we should be able to assess a higher reliability score for the software than the score that would result which is strictly based on the number of test cases in traditional SRGMs. Why? Because these test cases have demonstrated a greater than average ability to force propagation to occur (under the assumption that the injected corrupt states have similar qualities to the corrupt states that existing faults would create).

In addition, note one other benefit that FIT can offer for improving fidelity of the quantitative assessments of traditional SRGMs. If we employ the form of software FIT that injects anomalous input data coming into the software (instead of corrupting states inside of the code to simulate faults), and the software does not propagate into a failure, then we have demonstrated robustness (i.e., fault tolerance). This demonstrates that the software can produce

dependable service even in the face of a hostile external environment. Therefore SRGMs should take this information into account when making their quantitative assessments.

### III. USING FIT RESULTS IN SRGMS

The quality of SRGM predictions will be no better than the quality of failure data that is used to statistically estimate model parameters, typically failure rate parameters [1]. Tests that produce the failure data are typically driven by nominal operational profiles. In reality, the most significant threats to reliability occur under off-nominal conditions. Even if the profile includes anomalous inputs, the resultant tests are not guaranteed to reveal safety hazards in the code, as would be the case by using fault insertion. Model parameter estimates and reliability predictions would be more realistic, if test inputs are selected to test robustness under off-nominal conditions, combined with fault insertion to create corrupted states. If the software is not tolerant of anomalous inputs and corrupted states, this fact would be reflected in a higher failure occurrence than would be the case if a nominal profile were used. In this case, reliability predictions based on a nominal profile would be disastrous for the deployment of safety critical software. On the other hand, if the software is tolerant of anomalous inputs and corrupted states, this fact would be reflected in a lower failure occurrence than would be the case if a nominal profile were used. In this case, reliability predictions based on a nominal profile would be too pessimistic. In either case, FIT-based testing would produce more accurate reliability predictions of *future* reliability.

### IV. TYPES OF SOFTWARE THAT ARE CANDIDATES FOR THIS ANALYSIS

FIT analysis is applicable to: (1) commercial-off-the-shelf code and (2) source code that is accessible. In particular, it is useful for proprietary software of "unknown pedigree" that is slated to be embedded in a larger system. For such software, access to the interfaces between the binary formatted components is needed. (For example, a call to the operating system, a call to a database, and a call to a different component or a DLL are interfaces). At the interfaces, the approach used is termed Interface Propagation

Analysis (IPA), which injects anomalies into the data feeds that travel between the binary components. IPA then observes later downstream how far these anomalies have propagated and observes what new forms of anomalies might have been created. This information is then reported back to the user. This form of FIT is useful for impact analysis and robustness testing. When source code is available, the form of FIT is Propagation and Infection Analysis. Here, instrumentation is embedded directly into the source code that simulates the code containing programming and timing faults. Then, instrumentation is again used to determine what types of additional anomaly corruptions are present in the state of the executing software. Note that the majority of software FIT employed today is at the interface level and not at the source code level due to the large costs of performing it at the source level.

### V. SUMMARY

Thus, in summary, we contend that software FIT can augment current SRGMs with two additional pieces of information: (1) information concerning the quality of the test cases employed (and thus how much additional confidence we can place or not place in the existing reliability assessment), and (2) information concerning how robust the software is to anomalous inputs that are not part of the software's normal operational profile. We plan to validate the concept against experimental data, such as from the Space Shuttle.

### REFERENCES

- [1] Norman F. Schneidewind, "Reliability Modeling for Safety Critical Software", IEEE Transactions on Reliability, Vol. 46, No.1, March 1997, pp.88-98.
- [2] J. Voas and G. McGraw, Software Fault Injection: Inoculating Programs Against Errors. New York: John Wiley and Sons.