

# Quantitative Identification of Non-Dependable Modules and Signals in Software \*

Martin Hiller, Arshad Jhumka, Vilgot Claesson, Neeraj Suri  
Department of Computer Engineering  
Chalmers University of Technology, Göteborg, Sweden  
{hiller, arshad, vilgotc, suri}@ce.chalmers.se

## Abstract

We present a quantitative approach for identifying software modules and signals which will not be able to contain data errors that may be present in a software system, thus rendering the system non-dependable. Based on error propagation analysis in combination with error effect analysis we discuss how the results can be used to identify a) modules/signals which have a high “ability” to let propagating errors pass through them on their way through the system, and b) modules/signals which, when be subjected to errors, have a severe negative effect on the results produced by the system. This knowledge is very useful for directing and allocating resources for increased software reliability. Both the error propagation analysis and the error effect analysis are based on the Error Permeability measure. Using this measure we define a range of subsequent measures which allow us to quantify error propagation as well as error effect.

## 1. Motivation

As software-based functionality becomes pervasive, e.g., in embedded control systems, software usually comprises numerous discrete modules interacting with each other in order to provide a specific task or service. With an error present in a software module, there is a likelihood that this error can propagate to other modules with which it interacts. Knowing where errors propagate in a system is of particular importance for a number of development activities. Propagation analysis (see e.g. [2, 5]) may be used to find the module which are most exposed to errors in a system, and to ascertain how different modules affect each other in the presence of errors. In addition to knowing propagation characteristics it is also important to know where errors are likely to do the most damage. Note that those errors which are likely to propagate not always are those that are most likely to cause great damage. Thus it is important to do an

analysis of both these notions to identify the most vulnerable parts of a system. Furthermore, error propagation analysis and error effect analysis also gives an insight on locations in the system that would be best suited for placement of error detection mechanisms (EDM’s) and associated error recovery mechanisms (ERM’s), such as Executable Assertions (EAs, [1, 4]).

Error propagation and effect analysis may also be used as a means of obtaining information for use in decisions on where additional resources for dependability development are necessary and to determine where they would be most cost effective. Software is common not only in applications such as aircraft or other high-cost systems, but also in consumer-based cost-sensitive systems, such as cars. These systems often require both development costs and production costs to be kept low. Our approach complements other analysis activities, for instance FMECA (Failure Mode Effect and Criticality Analysis). Consequently, modules and signals found to be vulnerable and/or critical during propagation and effect analysis might be given more attention during design activities. Thus, error propagation and effect analysis, as a means of both system analysis and resource management, may be a very useful design-stage tool in such systems.

## 2. Approach

In our approach (initially introduced in [2]), we use the measure of *error permeability*, and based on it we define a set of related measures that cumulatively give an insight on the propagation characteristics and vulnerabilities of a system.

Consider the black-box software module in Fig. 1.



Figure 1. A basic black-box software module

\*Research supported in part by Volvo Research Foundation (FFP-DCN), by Saab Endowment, and by NSF Career CCR 9896321. © ISSRE and Chillarege Corp. 2001

For each pair of input and output signals, we can define *error permeability* as the conditional probability of an error occurring on the output given that there is an error on the input. Thus, for input  $i$  and output  $k$  of a module  $\mathbf{M}$  we define the *error permeability*,  $P_{i,k}^M$ , as follows:

$$0 \leq P_{i,k}^M = Pr\{\text{err in o/p } k | \text{err in i/p } i\} \leq 1 \quad (1)$$

This measure indicates how *permeable* an input/output pair is to errors occurring on the input.

Based on this fundamental measure we generate several subsequent measures for characterising error propagation (the actual mathematical definitions have been left out):

**Module error permeability** quantifies a module's "ability" to let propagating errors pass through it. The higher the permeability, the higher the chance of an error at the input to get through to the output.

**Module error exposure** quantifies the level of exposure to propagating errors. The higher the exposure, the more likely will the module be exposed to propagating errors (if there are errors in the system).

**Signal error exposure** is the equivalent of the *module error exposure* but at the signal level. Thus, with this measure we can identify which particular signals are more likely to be exposed to errors than others.

In addition to propagation analysis it is important to take into account the *what if*'s that may have a very low probability of occurring but still have a profound effect on system operation should they actually occur. Thus, it is necessary to do error effect analysis.

Extending the framework introduced in [2], we define a set of measures which, as were the propagation related measures, are based on *error permeability* (again, the mathematical definitions have been left out):

**Signal impact** quantifies the "effect" an error in a given signal would have on the output of the system. The higher the impact, the higher the chance of an error in the signal to affect the output of the system. This measure takes into account that an error in a given signal may find its way to the system output using several different paths.

**Signal criticality** is a way of biasing the impact values of signals based on which output signals they affect. For systems with multiple output signals, the system designer may wish to assign different levels of "importance" to these output signals, i.e., errors in one output signal may have a more severe effect than errors in another output.

Identifying candidate locations for ERM's and EDM's is a process where trade-offs have to be weighed against each other. Given the set of measures we have defined we can now set up the following guide-lines for interpretation:

- The higher the error exposure values of a module, the higher the probability that it will be subjected to errors propagating through the system if errors are indeed present. Thus, it may be more cost effective to place EDM's in those modules than in those with lower error exposure. An analogous way of reasoning is valid also for the signal error exposure.
- The higher the error permeability values of a module the higher the probability of subsequent modules being subjected to propagating errors if errors should pass through the module. Thus, it may be more cost effective to place ERM's in those modules than in those with lower error permeability.
- The higher the criticality of a signal (or impact if the system only has one output signal), the higher the probability of an error in that signal causing "expensive" damage from a system point-of-view. Thus, it may be more cost effective to equip those signals which have the highest criticality (impact) with EDM's and ERM's.

These rules may be conflicting with each other. Consider the case where a signal has a very low exposure but a high criticality (impact). A low exposure means that there is a low probability of errors propagating to that signal. However, a high criticality (impact) means that, should an error find its way into that signal, there is a high probability of that error causing damage which propagates beyond the system barrier into the environment. Thus, one may select signals with low exposure and high criticality (impact) as candidate locations for EDM's and ERM's.

Complementing this approach with approaches for streamlining EDM's and ERM's (such as [3]) will help in using resources available for dependability activities in a highly efficient way.

## References

- [1] Hiller M., "Executable Assertions for Detecting Data Errors in Embedded Control Systems", *Proc. DSN 2000*, pp. 24-33, 2000.
- [2] Hiller M., Jhumka A., Suri N., "An Approach for Analysing the Propagation of Data Errors in Software", *Proc. DSN 2001*, pp. 161-170, 2001.
- [3] Jhumka A., Hiller M., Claesson V., Suri N., "A Systematic Approach for Designing Consistent Executable Assertions for Embedded Software", *CE-TR 01-8*, Dept. of Comp. Engg., Chalmers, Sweden, 2001
- [4] Saib S.H., "Executable Assertions - An Aid To Reliable Software", *11th Asilomar Conference on Circuits, Systems and Computers*, pp. 277-281, 1978.
- [5] Voas J., "PIE: A Dynamic Failure-Based Technique", *IEEE Trans. on SE*, Vol. 18, No. 8, pp. 717-727, 1992.