SOFTWARE RELIABILITY MEASUREMENT SESSION

John D. Musa
Session Chairman
Bell Labroatories
Whippany, New Jersey 07981

Many people think of reliability as a devoutly wished for but seldom present attribute of a program. This leads to the idea that one should make a program as reliable as one possibly can. Unfortunately, in the real world software reliability is usually achieved at the expense of some other characteristic of the product such as program size, run or response time, maintainability, etc. or the process of producing the product such as cost, resource requirements, scheduling, etc. One wishes to make explicit trade-offs among the software product and process rather than let them happen by chance. Such trade-offs imply the need for measurement. Because of mounting development and operational costs, pressures for obtaining better ways of measuring reliability, have been mounting. This session deals with this crucial area.

The first paper, by Bev Littlewood, presents a model for software reliability in which early failure corrections cause greater reductions in the failure rate throughout test and debugging. Whether or not there exists a differential effect significant enough to warrant a more complex model is not known at the present time, but is definitely an important area for study.

The second paper by Musa and Iannino describes a methodology for handling variations in program size in software reliability modeling. Traditional software reliability models assume that systems are stable. However, most systems are integrated sequentially and many systems undergo design changes. The methodolgy described in this paper thus provides a means for substantially improving the estimation of software reliability quantities in large classes of practical systems.

The last paper by Goel and Okumoto discusses the very practical problem of how much testing should be done before the release of a system. The paper utilizes a cost model based on the authors' software reliability model. The optimal software release time is thus based on cost considerations.

ABSTRACT

OPTIMAL TESTING POLICIES FOR SOFTWARE SYSTEMS

By Amrit L. Goel, Syracuse University
Kazu Okumoto, Rutgers University

An important problem of practical concern is to determine how much testing should be done before a system is considered ready for release. This decision, of course, depends on the model for the software failure phenomenon and the criterion used for evaluating system readiness.

In this paper, we first develop a cost model based on the time dependent failure rate function of Goel and Okumoto. Next, we derive policies that yeild the optimal values of the level of test effort (b*) and software release time (T*). The sensitivity of the optimal solution is also numerically evaluated.

SOFTWARE RELIABILITY MODELING ACCOUNTING FOR PROGRAM SIZE VARIATION DUE TO INTEGRATION OR DESIGN CHANGES

By J.D. Musa and A. Iannino
Bell Laboratories
Whippany, New Jersey 07981

ABSTRACT

Estimation of software reliability quantities has traditionally been on stable systems; i.e., systems that are completely integrated and are not undergoing design changes. Also, it is assumed that test results are completely inspected for failures. This paper describes a method for relaxing the foregoing conditions by adjusting the lengths of the intervals between failures experienced in tests as compensation. The resulting set of failure intervals represents the set that would have occured for a stable system in its final configuration with complete inspection. The failure intervals are then processed as they would be for a complete system. The approach is developed for the execution

time theory of software reliability, but the concepts could be applied to many other models the estimation of quantities of interest to the software manager are illustrated.

A BAYESIAN DIFFERENTIAL DEBUGGING MODEL FOR SOFTWARE RELIABILITY

By B. Littlewood
The City University,
London, England

ABSTRACT

An assumption commonly made in early models of software reliability is that the failure rate of a program is a constant multiple of the number of bugs remaining. This implies that all bugs have the same effect upon the overall failure rate. The assumption is challenged and an alternative proposed. The suggested model results in earlier bug-fixes having a greater effect than later ones (the worst bug show themselves earlier and so are fixed earlier), and the DFR properly between bug-fixes (confidence in programs increases during periods of failure-free operation, as well as at bug-fixes). The model shows a high degree of mathematical tractability, and allows a range of reliability, and allows a range of reliability measures to be calculated exactly. Predictions of total execution time to achieve a target reliability, are obtained.