UNITED STATES
NUCLEAR REGULATORY COMMISSION
ADVISORY COMMITTEE ON REACTOR SAFEGUARDS
WASHINGTON, DC 20555 - 0001

June 9, 2004

Mr. Luis A. Reyes
Executive Director for Operations
U.S. Nuclear Regulatory Commission
Washington DC 20555-0001

SUBJECT: DIGITAL INSTRUMENTATION AND CONTROLS RESEARCH PROGRAM

Dear Mr. Reyes:

During the 513th meeting of the Advisory Committee on Reactor Safeguards on June 2-4, 2004, we reviewed the staff's research activities on risk assessment of digital instrumentation and controls (I&C) systems, which are part of the Digital I&C Research Program. Our Subcommittees on Reliability and Probabilistic Risk Assessment and on Plant Operations reviewed this matter during a meeting held on March 26, 2004. During these reviews, we had the benefit of discussions with the NRC staff and its contractors. We also had the benefit of the documents referenced.

CONCLUSIONS

We support the effort of the Digital I&C Research Program to develop more quantitative measures of digital system reliability.

DISCUSSION

As nuclear power plants move toward increased use of digital technology, new challenges are created due to analog technology obsolescence and the functional advantages of digital technology. The use of this technology may introduce new failure modes into plant systems. These must be understood and modeled for two major reasons that affect regulatory decisions:

1. Probabilistic risk assessment (PRA), the principal analytical tool supporting the Commission's risk-informed initiatives, must be modified to include models for the failure modes that digital software may introduce.

2. The current regulatory review of digital systems is based largely on controlling the process for software development without any assessment regarding failure modes or reliability.

The goals of the Digital I&C Research Program are to:

- Gain an understanding of how digital systems fail and how likely it is that they will fail in use.
- Develop methods and tools for including digital system models into PRA.
- Develop guidance for regulatory applications involving digital system reliability.

This program will provide additional information on digital I&C failures, digital failure assessment methods and system models, digital reliability assessment methods, and integration in PRAs. This information should be included in the staff's reviews. It is evident that no single type of information will be sufficient for regulatory decision-making. This program is pursuing multiple approaches. We strongly support the goals of this program.

Sincerely,

/RA/

Mario V. Bonaca
Chairman

Additional Comments by ACRS Member George E. Apostolakis

I agree with the Committee's conclusion. I offer two recommendations for the staff's consideration as it moves forward with this very important program:

A. The databases containing software-induced failures of technological systems should be reviewed and conclusions should be drawn regarding failure modes and their frequency of occurrence.

B. Available methods for the identification of failure modes and the assessment of the reliability of systems that are software driven should be reviewed critically. Their domains of validity should be determined by examining their assumptions and comparing them with the insights gained from the database review.

While I agree with the program's goals, I would like to see convincing evidence that funded projects will answer important questions that arise in the study of digital systems. These questions ultimately lead to fundamental issues related to the proper treatment of software "failures."

The literature on digital software (References 1-3) indicates that there are two main interpretations of the concept of software safety. The first interpretation views "failure" as a property of the software itself, just as the failure modes of hardware are considered properties of the components. This "software-centric" view is to be contrasted with the second interpretation, the "system-centric" view, which asserts that it is meaningless to talk about the failure of a piece of software in isolation. In this view, the concept of failure becomes meaningful only when the software is considered within a system, in which case one speaks of system failures. This approach is very similar to the modeling of human performance (Reference 3). An unsafe human act is considered meaningful only in the system context within which it occurs, an observation that has led the Office of Nuclear Regulatory Research to the development of the concept of "error-forcing context" (Reference 4).

A natural way to determine which interpretation of the concept of software failure is appropriate (and under what conditions) is to examine the available data that involve software failures. The staff's research program includes a task at Brookhaven National Laboratory that deals with databases. Although the Committee has not reviewed this effort in detail, I gather from the staff's presentation at the subcommittee meeting that this review was not intended to explore the concept of failure discussed above. The Committee was told that software failures have caused serious accidents in other industries and that an examination of licensee event reports has concluded that digital failures are approximately evenly divided among hardware, software, and human-system interface related failures. I recommend that the analysis be expanded to provide insights into which of the above interpretations would be the appropriate one to explain what happened.

There are many models for the evaluation of probabilities of software performance. These models fall naturally into two categories, depending on which interpretation of failure one adopts. The software-centric models borrow heavily from reliability models that have been developed for hardware components, e.g., exponential failure models. The system-centric models propose the expansion of standard system analysis tools, such as fault trees, to include software interactions with the hardware.

These models must be reviewed critically before the staff decides on which approach to adopt. This review should include an evaluation of the fundamental assumptions behind each model and a comparison with the insights gained from the review of the databases. For example, the staff told the Committee that "Markov-type modeling at the processor level appears to be capable of capturing digital design features." While this may, in fact, be a good conclusion, I would like to be convinced by arguments supporting the assumption of a constant rate of transition between "good" and "failed" states and by evidence from actual experience that supports this assertion. What kinds of events occurring in time at a constant rate does this model consider? Are errors in requirements and specifications included? These errors have been found to be the cause of a large number of software errors[1]. How are the interactions of the software with the rest of the system to be modeled? These interactions include potentially unexpected system conditions that exercise the software in a way for which it may not have been designed, thus leading to "wrong" responses. Past research results point to situations of this kind, where one could argue that the software did not fail, but nevertheless was induced to do the "wrong thing" by a design flaw left in the system or the software itself.

I believe that the implementation of the two recommendations that I have offered will provide a strong technical foundation for the achievement of the staff's goals in this program.

REFERENCES

1. National Research Council, *Digital Instrumentation and Control Systems in Nuclear Power Plants: Safety and Reliability Issues,* National Academy Press, Washington, D.C., 1997.

2. N.G. Leveson, *Safeware: System Safety and Computers*, Addison-Wesley, Reading, MA, 1995.

3. C. Garrett, and G. Apostolakis, "Context in the Risk Assessment of Digital Systems," *Risk Analysis*, 19:23-32, 1999.

[1] Some analyses of NASA failure experience indicate that nearly 75% of failures found in operational software are rooted in requirement errors (Reference 5).

4. S. E. Cooper, A. M. Ramey-Smith, J. Wreathall, G. W. Parry, D. C. Bley, W. J. Luckas, J. H. Taylor and M. T. Barriere, *A Technique for Human Error Analysis (ATHEANA).* NUREG/CR-6350, U.S. Nuclear Regulatory Commission, Washington, DC, 1996.

5. R. R. Lutz, *Targeting Safety-Related Errors during Software Requirements Analysis*, Proceedings of the First ACM SIGSOFT Symposium on the Foundations of Software Engineering, pp. 99-106, 1993.