IAEA Advisory Group Meeting on "Advanced Control Systems to Improve Nuclear Power Plant Reliability and Efficiency

March 13-17, 1995 Vienna

Status on the Finnish Activities regarding

QUALIFICATION OF PROGRAMMABLE AUTOMATION SYSTEMS

Kaj Juslin

VTT Automation, P.O.Box 1301, FIN-02044 VTT
E-mail: kaj.juslin&vtt.fi Fax: +3580.456.6475 Phone: +3580.456.6422

General

Safe and economic operation of a nuclear power plant sets high requirements for its supervision and control systems. A highly reliable automatic protection system is a necessity, but also many monotonous, time-critical or accuracy requiring control actions should be automated. Operation, maintenance and safety personnel should get sufficient, clear, timely and unambiguous information about the plant state and its coming development, procedures, design bases and technical specifications.

Modern computer and information technology opens up new means of constructing efficient supervision, control and protection systems. Their acceptance for safety-critical applications requires new verification and validation methods, rules and guidelines.

The objectives of our research work on Qualification of Programmable Automation Systems encompasses the development of methods, tools and practices for the evaluation and licensing of safety critical programmable automation systems. The results are applied by the safety authority and utility companies in connection with the development tasks in existing plants as well as during the construction of new plants.

Various aspects of the safety evaluation and licensing of programmable digital automation systems were studied. The studies clearly show that the safety assessment can not be based on conventional probabilistic methods because of the difficulties in quantification of the reliability of the software as well as the hardware. Additional means shall therefore be used to gain more confidence on the system dependability. None of the existing safety assessment methods alone is sufficient for the estimation of the safety and reliability. Each single method has more or less serious drawbacks either in applicability, coverage or in some other area. The problem concerns both methods for producing and methods for assessing the dependability of the system. For these reasons the principle to be applied should be the diversity in methods and techniques. Confidence in the dependability increases as the number of different assessing methods increases. The qualification process includes assessment of the development process, assessment of the product itself and testing of the system based on statistically selected test cases. In any case, several assessment methods give different points of view to the safety of the system, and therefore they produce a more credible safety estimate than one assessment method alone. The acceptability of the system is based on evidence of excellence of the design and implementation process and on independent analyses and testing of the end product.

Programmable digital systems are offered for both the operation and safety automation of the possible new nuclear power plant for Finland. This seems to be the largest technological leap in the otherwise quite customary plant concepts. In existing plants it probably also becomes necessary to replace analog systems with digital ones. Favorable experiences with these systems have been gathered from conventional power and other process plants. In the nuclear field the experience is limited, especially in safety-critical applications. In the Darlington plant in Canada an entirely digital reactor protection system has recently been licensed, and in the UK the Sizewell B plant will use digital operation automation and primary reactor protection systems. In the French N4-plants digital technology is used extensively. Partly due to problems with this technology, the commissioning of the first plant, Chooz B, has been delayed. Anyhow, the trend towards digital systems seems to be common in all leading nuclear countries.

There are many good reasons for using digital systems:

- they are more efficient, allowing the construction of many operator support and safety functions that would be impossible to realize with analog technology.

- they are more accurate and flexible and are easier to maintain.

- Digital systems require less space in buildings and can be commissioned in a shorter time span, and therefore lead to cheaper total construction costs of the plant.

- They can also, if properly designed and realized, be more reliable than analog systems.

One important reason is the common trend towards digital systems in other process industries; it would not be wise to deviate from this trend in the nuclear industry and thus miss the opportunity to benefit from the experiences of other branches.

The behaviour of programmable systems deviates to such an extent from the conventional analog systems that their licensing for safety-critical applications requires a new kind of approach. The objective of the work is to develop methods, tools and practices needed in an independent safety assessment of programmable automation systems in Finnish nuclear power plants. There is no single means of demonstrating that a programmable system is safe enough. Confidence in the "safety case" must be based on a multitude and diversity of means being employed to that end.

The research work has been realized in the projects UUTE/AJA and SAMA/AVV. The tasks have included:

- the analysis of the proposed automation concepts of the potential plant vendors and their technical, managerial and quality assurance aspects,
- the collection and analysis of applicable international standards and guidelines (Haapanen & Maskuniitty 1993a),
- familiarization and analysis of licensing practices in other countries (Haapanen et al. 1993),
- a study of the need and ways to use diversity (Haapanen & Maskuniitty 1993b),
- reliability analysis of software based safety functions (Pulkkinen 1993)
- fault tree and failure mode and effects analysis (Pulkkinen & Maskuniitty 1994)
- the development and trial use of a dynamic test harness for programmable systems (Haapanen & Korhonen 1994, Haapanen et al. 1995a, b & c).

Based on these studies common grounds will be synthesized for the requirements and guidelines to be applied by the utility companies and licensing authorities in the procurement and licensing of programmable systems.

Problems of safety assessment

"The safety assessment of a programmable system can not follow the conventional pattern because of the difficulties in quantification of the reliability of the software as well as the hardware". In the case of the software it is widely acknowledged that quantification is difficult. The hardware has equal problems due to the dependence of the effect of a hardware failure on the instruction being executed at the time of failure. There are two kinds of problems in producing safety-critical programmable systems; (a) difficulties in producing specifications for safe behaviour and (b) difficulties in assessing whether or not one has produced a system which is safe. One can never be certain that a software-controlled critical system is 'safe', but it is possible to systematically determine which techniques can, if properly applied, reduce uncertainties or doubts about system safety (McDermid 1991a).

'Failure modes' of the software are through design errors rather than random component failures caused by eg. wear. Redundant lines of the system will therefore fail simultaneously and some kind of diversity is needed to compensate for the residual errors contained in the system software. Software behaviour is

discontinuous: minor diversions may cause drastic changes in behaviour. Therefore it is not easy to extrapolate from known past to future performance, and it is difficult to predict the required high levels of software reliability by normal statistical techniques. Due to the uncertainty of the reliability assessment, additional means should be used to get more assurance about the systems ability to fulfill the basic safety requirements. Two basic principles for getting assurance are available: comprehension and diversity. "Assurance arises from comprehension of, and diversity in, the complete procurement process, and the methods and tools used in development and evaluation" (McDermid 1991a). One can either regard the assurance as a predictor of reliability or prefer the assurance before the reliability. The level of assurance achieved by using a particular set of means can be hypothesized to correspond to a certain level of reliability. Experimental validation of this hypothesis, however, seems not to be feasible in practice. The absence of a complete model of the operational environment for the system makes it impossible to compute with complete confidence the reliability of the system, and the discontinuities and non-determinism in software behaviour make it impractical to infer with complete confidence future reliability from past reliability. Therefore "the deployment decisions about critical systems are, and probably have to be, made on the basis of assurance, not reliability".

Achieving assurance

The main factor contributing to the assurance is the evidence produced during software development, and this in turn derives from the verification and validation activities. The goals of the development of a safety-critical system are (McDermid 1991a):

- to develop the software in such a way that it is impossible or extremely unlikely that its behaviour (execution) will lead to a catastrophic failure.

- to provide evidence that will convince both the developers and the assessment authority of the dependability of the software.

Due to the inconsistencies in the quantification of reliability, the software safety case has to be based on deterministic measures, with an accompanying confidence building element, although a limited reliability claim can be made based on the methods used in the production of the software. The main objective of the deterministic measures is to produce a high quality system which conforms to the specified requirements and expectations of the user in almost all respects. This implies a complete, correct and unambiguous requirements specification with the software being produced using the 'best' practices. This is then supported by confidence-building measures that allow greater precision to be set on the obtained but unknown reliability claim. The features of such a claim need to be the evidence of error avoidance, error detection and error tolerance during initial design and manufacture, through commissioning to final operation (Hunns & Wainwright 1992).

One widely promoted class of techniques is program verification. This involves expressing a specification in discrete mathematics, e.g. set theory and predicate calculus, and then using the inference rules of logic to show that a given program conforms to the specification. A suitably formulated program proof can show that the program conforms to its specification under all possible inputs, and all possible sequences of inputs. The analytical techniques, however, do not show that the software is safe; there may be flaws in the specification or inaccuracies in the model of the hardware implicit in the specification of the program, which can only be found by testing the programs in their operational environment - or at least on a representative hardware platform.

For complex artifacts it may be impossible to gain adequate comprehension directly, or it is simply more cost-effective to gain assurance in the process. In practice it is helpful to address assurance from both the process and the product points of view (McDermid 1991a).

Standards and guidelines

One key issue in the evaluation of the quality of the production process and thus the quality of the product is to check that in all phases of the process properly selected standards and guidelines are followed. Therefore the applicability of several international standards and guidelines (IAEA, ISO, IEC, IEEE) and their basic principles in different life cycle phases were analysed (Haapanen & Maskuniitty 1993a). In addition, practical studies of an automation system were carried through in order to identify the essential questions.

Most standards handling programmable automation approach the problem from the project control point of view. When systems are designed and realized starting from scratch the special needs of nuclear power application can be taken into account from the beginning. On the other hand these systems are prototypes having no earlier operating experiences.

In practice the situation may be different; the system is based on a mature automation system, which has a typical development history and operating experience from various branches of industry. In this case different activities of different organizations, such as system programming, application programming and maintenance can be found.

Standards for the project control can be used from the beginning of the project to make the programming cope with discipline, but applying them afterwards for qualification of earlier life cycle phases, eg. the system programs of existing automation systems, is more difficult. The basic standards determine three methods, which are the requirements for qualification of electrical items of safety systems and class E equipment. These are type testing, operational experience and analysis or combinations of those methods. Other items affecting the quality

of the system are the quality assurance system, project routines, safety classification and the safety architecture of the system.

Licensing practices

The Finnish licensing requirements should be a balanced synthesis of the best international practices tailored to the local circumstances. The methods and practices applied in seven leading nuclear countries (UK, Canada, USA, France, Japan, Germany and Sweden) for the licensing of programmable control and protection system were gathered and analyzed (Haapanen et al. 1993). The study was based on information presented in international conferences and seminars and published in periodicals and most importantly on information acquired during visits to the vendors, plants and licensing authorities by the representatives of the utility companies, Finnish authorities and VTT (in the framework of the joint AJA-project).

It turned out that the licensing practices in most countries are still under development and new systems have been taken into operation using special practices case by case. For the Finnish situation the most interesting case seems to be the Sizewell B licensing process now nearing completion in UK.

The second most interesting is the licensing of the reactor protection system of the Darlington plant in Canada. On the other hand, one can say that the methods and practices applied to the licensing can finally be fixed only during the actual licensing process itself. The common trend in the regulatory process also seems to lead towards non-prescriptive, performance based regulation, because performance objectives, if well conceived and stated, are less likely to be changed than are the particular means to achieve them.

Diversity

Software faults are systematic by nature as opposed to random component faults caused by ageing and wear. Therefore, all identical program versions fail simultaneously and redundancy can not protect against common mode failures caused by program faults. As it is impossible to produce and prove a program to be totally free from errors, a safety critical programmable application must have some error tolerance. The most critical functions must therefore be realized with diverse redundant systems.

Use of diversity and testing can be concluded to be central issues in producing safe programmable system and in proving them to be safe enough. The need of and ways to realize diversity have been studied (Haapanen & Maskuniitty 1993b).

A safety critical programmable system shall include diverse redundant parts so that no residual program fault can cause the failure of the intended function of the system. Diversity can be realized on functional level by introducing two or more different part functions for the same safety function. On the other hand, different part systems can be realized for the same function either by diverse programs or hardware. When diverse program versions are used one should assure that the versions really are different enough; two programs produced by two independent programming teams from the same specifications do not necessarily fulfil this requirement.

By combining functional and programming diversity in a suitable way one can produce a system that is safe enough without having a non-programmable back-up system, although in some cases this kind of back-up still are used (eg. Sizewell B; opposite examples are eg. Darlington and French N4-plants).

Applying diversity is also an important concept in confidence building measures for the implemented system. Independent analysis and testing of the system should use different methods and tools from those used in the production and validation process by the system vendor.

Reliability analysis

According to the Finnish regulatory guide, YVL Guide 2.8 (1987), the safety functions of nuclear power plant should fulfil some quantitative safety criteria. These criteria concern also the reactor protection systems independently of the techniques applied in their design. The conventional methods of Probabilistic Safety Assessment (PSA) are applicable in the analysis of usual mechanical systems controlled by hardwired control systems. Further, they may be applied in the analysis of operator errors and human reliability. However, these methods have not been applied in the analysis of software based or digital safety functions. To study the feasibility of various reliability engineering methods, a state of the art survey was made (Pulkkinen 1993).

The survey concentrated on the qualitative and quantitative software reliability analyses and models. The compatibility of software reliability models and PSA is also discussed, and some conclusions and recommendations are given.

The most important qualitative software reliability analysis methods are based on corresponding methods for conventional systems. The objectives of their application are the identification of the factors having impact on the safety of the system, allocation of safety criteria, support of the system design and the assurance of the software quality. Failure mode and effects analysis (FMEA) is a standardized method (see IEC 812 (1985)), which has been modified for the analysis of software by Reifer (1979) and developed further by several researches and practitioners. Software FMEA has been applied mainly in non-nuclear industry, e.g. in space applications.

Sneak Circuit Analysis, SCA (Taylor 1992), is a combination of FMEA-type approach and fault tree. In SCA, the outputs of the software leading to hazardous events are identified and their causes are studied. The findings of the identification are described by fault trees. Methods such as Preliminary Hazard Analysis, PHA, Software Requirements Hazard Analysis, SRHA, Subsystem Hazard Analysis, SSHA, System Hazard Analysis, SHA, and Operating & Support Hazard Analysis, O&SHA, are modifications of conventional ones (McKinlay 1991). The basis of the method developed by Toola (1992) for the analysis of accidents and disturbances of process automation is also conventional. These methods have been applied in non-nuclear industry, where extensive risk analyses are not usually made. In fact, it is likely that the analyses made in connection with a plant specific PSA provide the same information as the above mentioned methods.

Software fault tree describes the control structures of software by applying the gates of fault tree (Leveson et al. 1991). The software fault tree may be seen as a simplified and graphical form of axiomatic verification of software. Since it concentrates purely on the hazardous errors of the software, it may reduce the efforts needed in the verification process. Similar analyses may be performed by applying, for example, Petri nets (Leveson and Stolzy 1987) or other more formal methods. Formal methods are often applied in specification of safety critical software. The use of these methods makes it possible to verify the software more precisely. Applications of formal methods for nuclear power applications have been described by Parnas et al. (1991).

Software metrics describe the characteristics of software (size, complexity etc.) with simple numerical measures (Conte et al. 1986). The correlation between software metrics and the number of errors in software has been studied, for example, by Barnes et al (1987). It is possible to use the information from the metrics in prediction of the software reliability, but more empirical data is needed for that purpose.

The quantitative software reliability models have been discussed widely in the literature (Musa et al. 1987). Usually the software reliability models concentrate on the determination of the probability distribution of the number of errors in computer codes. Further, their predictions may be used in allocation of the testing efforts or other resources in the software development.

The reliability estimates needed in PSA are of the form: "the probability that the system does not operate at demand is p". The software reliability models give answers to questions like: "What is the probability that there are more that n errors in the software" or "What is the probability that the software operates without errors over a certain time period". Often software reliability models are based on various expert judgements, which have strong impacts on the final safety estimates. By combining traditional software reliability models, operational experiences, software metrics and experts judgements, it could be

possible to estimate the probabilities needed in PSA. This kind of safety arguments can be made explicit and they can be applied in the reliability evaluation (McDermid 1991b). However, there is a need for further research in this field. One possibility to evaluate the applicability and credibility of the combination of PSA and software reliability models is to organise Benchmark-studies.

The operational experience from the application of software based safety systems in industrial applications is still very limited. In order to increase the validity and credibility of the reliability analysis methods, one should be able to compare the model predictions with practical observations. For that purpose, there is a need to establish facilities for systematic data collection and analysis of operational experience.

PSA is the only method which gives an overview on the safety of the nuclear power plant by evaluating the relative importances of various systems and components. Thus it is reasonable to include also the digital safety systems in the PSA model. The quantitative risk estimates are based on various modelling assumptions and expert judgements, and the objectivity of the model is questionable. This does not restrict the usefulness of PSA models since sensitivity analyses can be made. A more problematic issue is the use of very strict safety goals.

The YVL Guide 2.8 (1987) requires that the quantitative safety criteria should be met with certain confidence. The requirement for rector protection signals is very strict, and by simple calculations it is possible to show that very extensive testing of the software is needed in order to prove that the safety goal is fulfilled. As a conclusion from the surveys of the software reliability analysis it is possible to recommend the revisions of the safety goals or principles in the YVL 2.8 Guide. It should be emphasized here that the criteria for conventional systems are not problematic. For the software based systems some clarifications are needed to make the criteria more operational.

Fault tree and failure mode and effects analysis

To get experience from the use and applicability of various reliability engineering methods in the analysis of digital safety systems, a case study on the application of failure mode and effects analysis (FMEA) and fault tree analysis was made. The case study is a continuation for the survey described in the previous section (Pulkkinen 1993). The main objectives of the case study were to evaluate the applicability of FMEA and fault tree analysis and to give recommendations and guidelines for full scale reliability analyses. Further, one objective was to modify the methods for the analysis of digital safety systems.

The fault tree approach was adopted in this study due to its compatibility with PSA. Furthermore, as an approach based on fault logic, fault trees provide a view parallel to the systems design. Since the aim of this study is not to study the dynamic failure behavior of the system, fault trees are sufficient.

The fault tree analysis of a digital safety system may be divided into two phases. First the fault trees for the selected protection signal passing through the system trains are constructed. As the second phase, the failure modes included in the fault tree are analyzed in more detail by applying a modified failure mode and effects analysis (FMEA). The minimal cutset analysis is essential in the identification of the most important events requiring detailed FMEA. The goal is to reveal the most important failure mechanisms including also the embedded software as well as to document the basic events and the gates of the fault tree. After finding minimum cut sets of the system the decision of the modules to be analyzed in more detail can be made. The most critical modules and basic events are analysed by applying FMEA.

The fault tree analysis of digital automation systems differs from that of typical hardware systems, and thus guidelines for performing such an analysis were given as a conclusion from the case study. In the following, an analysis procedure is shortly outlined .

The approach was found to be applicable in the reliability analysis of digital automation systems. However, the applicability is limited to the analysis of static aspects of the system, since the fault tree is always a static model.

The modelling of the software errors in the fault tree is somewhat problematic. First, the nature of software errors is not similar to that of mechanical and electrical faults. Secondly, the input signals to any modules or devices of digital automation systems are used as inputs of the software of the module. Thirdly, the software consists of the basic software and the user or application software, which have different functions in the system.

In the approach of the case study, the fault tree includes the software at a functional level. This may not be the correct approach, because the software functions may not actually be separated from the other parts of the software. It may not be possible to decompose the software into smaller units or functions as in the example software. The FMEA was applied here mainly to document the basic events of the fault tree. However, if sufficient background information is available, FMEA could also provide detailed analysis of system structure. The experiences from the FMEA in this case study are rather limited, and it is not possible to draw too strong conclusions.

The feasibility of quantitative reliability analysis of digital automation systems is questionable. However, it is possible to include the fault trees of these systems into the PSA models and make sensitivity studies on the relative importance of systems.

In order to be able to apply the methods in practical licensing work much research is still needed. Benchmark studies are good approaches for the comparison of various reliability engineering methods.

Dynamic testing

Traditionally, the development process incorporates testing in well defined phases. Unit testing, integration testing, factory acceptance testing and customer or site acceptance testing will form the core of the testing activities. For safety critical systems the ability to demonstrate a degree of independence in factory acceptance testing and site acceptance testing is important.

The dynamic testing at the end of the development process is aimed at demonstrating that the delivered system performs to its specification and meets customer requirements, that there are no functional errors in the software or the hardware and that the system interacts effectively. The operation of the system is addressed in realistic situations, with realistic operating conditions, with respect to the required reliability. Testing is intended to demonstrate that in a realistic situation, with real inputs, the system will behave as required over a prolonged period of time. Although the testing can not prove the system to be safe, each successful test case can increase the confidence about safety (Abbot 1992).

In general, the finding of errors at the final dynamic testing phase jeopardizes the whole acceptance of the system. The mere removing or correction of errors is not enough. The causes of their birth and going through the earlier testing and verification phases must be cleared and removed from the system production process.

An automatic test harness has been specified (Haapanen & Korhonen 1994). The test harness is applied in a trial testing of two pilot systems provided by ABB Atom and Siemens AG. The harness includes the generating and management principles for the test cases and the means for the verification of the test results. The most tedious task is the building of a logical model of the target system which can be used to verify the correctness of the test results.

Test harness

The test generator feeds the test sequences to the automation system to produce the target system responses. The same test sequences are used as inputs in the logical model of the target system ("test oracle") which produces expected test results. The results are then compared in the test result comparator and if the results are not equal, the system and the logical model have to be checked for the difference.

The same test sequences have to be driven through the target system and the model, but not necessarily in parallel or not even in the same test environment. The tests can be done at different times and if the results are recorded, they may afterwards be compared and checked for cases that gave dissimilar results. Recording of the results makes it also possible to run the cases in totally separate environments, physical connection between the environments is not needed. The recorded results can be moved from one environment to another using, for instance, floppy disks.

Test data generation

The ultimate goal is to define such a set of test cases that would reveal all faults and errors. If the knowledge about the system internal structure together with some continuity, majority etc. principle does not allow the extension of one single test to cover a wider range of test cases, a "complete" testing is required. This requires all possible input and internal state combinations to be covered. This is in practice not possible, since even in systems with a limited number of inputs and internal states the combination explosion would raise the required number of test cases far beyond any practical limits.

Another important goal is to define a statistically significant set of test cases for the estimation of the system reliability. When the requirements are very high, as is the case for the reactor protection system, even this significance usually is hard if not impossible to fulfil.

In a real project only a limited time period is available for the testing before plant start-up, and this time together with the performance of the testing system set the upper limit for the number of test cases. Thus the practical goal would be to define as many different test cases as can be run during the limited time period available for testing.

The generation of the test cases will start from a rather limited set of basic cases drawn from different sources available, such as:

- plant transient analyses,
- measurements from existing plants,
- models and simulators etc.,

and then multiplying these to a larger set by proper randomizing techniques in order the reach some statistical significance. Main methods for the randomizing are adding noise to the signals and pointing random sensor failures to various signals.

Test oracle

For creating a model of the target system a high abstraction level language is needed to describe its behaviour. The language shall be executable, which presupposes formal semantics. Usually formal languages have two different notations: one for describing the reactive part (i.e. the behaviour) of systems, and one for describing the data transformations. The reactive part is in general described using state machines and the lowest level activities using some high level language such as C.

The model of the target system should be based on the requirements specifications rather than on functional specifications which already may include some shortcomings or misinterpretations. There are several alternatives for creating the logical model of a typical target system. Perhaps the most generally used are the RTSA-method and Statecharts for embedded systems. Five methods were originally considered as candidates and ReaGeniX RT/SA-model and Statecharts-methods finally selected for further inspection. These two methods were applied for modelling of one target system and their weighted superiority determined using Analytic Hierarchy Process-method (AHP) by Saaty. The comparison showed the applicability of the methods to be rather close to one another. The drawbacks and advantages of the methods are partly compensative: while ReaGeniX has a stronger methodological background and is thus easier to apply, Statecharts has the better tool support. No serious weaknesses were found that would reject either of the methods. Without any doubt both methods can be used for modelling real-time safety critical automation systems. This has successfully been demonstrated in this project.

The on-line system

The on-line system is a common industry standard PC-computer equipped with proper I/O-cards for the connection to the test object. These include commercially available D/A-, A/D- and binary I/O-cards. Standard features also include a large hard disk and a DAT-recorder able to contain the large test data bases.

The input driver is a simple program loop timed by an external clock signal. At each time step it reads the precalculated integer and binary test signal values from the data tables and loads them to the input registers of the D/A-channels and binary output cards. Inside the same execution loop the output driver reads the momentary output signal values from the binary I/O-cars and stores them to output signal data tables for comparison with the expected response by the result comparator.

The result comparator compares at each time step the measured binary output signals from the test object with the corresponding expected response signal values in the test data base. As long as only binary signals must be compared the

task is quite straightforward. In case also analogue signals shall be compared to the expected response, the comparator algorithm becomes much more complicated, since also some amplitude deviations besides small differences on time behaviour must be allowed without an actual error is concluded to have taken place. This case, however, is not handled in the prototype test harness.

Pilot system testing

In order to get experience in the testing procedure and to identity further development needs of the test harness two representative pilot systems developed by ABB Atom and Siemens AG were tested using the prototype dynamic test harness specified and implemented in AVV project. The purpose of testing was not to validate the pilot systems.

The ABB pilot system consisted of one independent simplified branch (SUB) of an Average Power Range Monitoring System (APRM), whose prototype has been the APRM system recently implemented at the Swedish Barsebäck nuclear power plant. The Siemens pilot system was a protection system of a simple laboratory equipment simulating loss of coolant accident in a pressurizer system.

The pilot testing shows that the procedure is feasible and the ReaGenix is a valid tool for implementing the logical model of the tested system. The comparison of the responses of the tested system and its logical model turned out to be the critical task and more development is still needed.

Based on experience gathered the test harness can later be expanded and completed to a full-scope testing environment and used for testing real safety critical nuclear power plant applications when they arise either in existing or new nuclear power plants in Finland.

References

Abbot, A. 1992. The Role of Dynamic Testing in the Certification of Software Based Safety Critical Systems, IAEA Technical Committee Meeting on Safety Assessment of Computerized Control and Protection Systems, Wien, 12-16.10.-1992, 8 p.

Conte, S.D., Dunsmore, H.E & Shen, V.Y. 1986. Software Engineering Metrics and Models. The Benjamin/Cummings Publishing Company, Inc. Menlo Park, California. 396 s.

McDermid, J.A. 1991a. Issues in Developing Software for Safety Critical Systems, Reliability Engineering and System Safety 32 (1991), pp. 1 - 24.

McDermid, J.A. 1991b. Safety Arguments, Software and System Reliability. Proceedings 1991 International Symposium on Software Reliability Engineering, Austin, Texas, May 17-18, 1991. IEEE Computer Society Press, Los Alamitos, California. pp. 41-50.

Haapanen, P., Häll, L-E., Lucander, A. & Manninen, T. 1993. Licensing Practices for the Programmable Automation Systems (in Finnish). VTT Tiedotteita 1447, 44 p.

Haapanen, P. & Maskuniitty, M. 1993a. Standards and Guidelines Applicable for the Validation of Programmable Automation Systems (in Finnish). VTT Tiedotteita 1446, 24 p. + app. 48 s.

Haapanen, P. & Maskuniitty, M. 1993b. Diversity and Testing Requirements of Programmable Automation Systems (in Finnish). STUK-YTO-TR 52, 57 p.

Haapanen P. & Korhonen J. 1994. A Dynamic Test Harness for Programmable, Safety Critical Systems, Prototype Specifications. VTT Working Report. 37 p.+ app. 34 p.

Haapanen, P., Korhonen, J. & Maskuniitty, M. 1995a (to be published). Experimental testing of an ABB Master application. STUK-YTO-TR xx. xx p. + app. xx p.

Haapanen, P., Korhonen, J. & Maskuniitty, M. 1995b (to be published). Experimental testing of a Siemens AG application. STUK-YTO-TR xx. xx p. + app. xx p.

Haapanen, P., Korhonen, J., Maskuniitty, M. & Pulkkinen, U. 1995c (to be published). Experimental development of assessment methods for programmable automation systems; Final report of the AVV-project. STUK-YTO-TR xx. xx p. + app. xx p.

Hunns, D.M. & Wainwright, N. 1992. The uk regulator's approach to the acceptance of a software-based protection system for Sizewell B. OECD NEA & IAEA International Symposium on Nuclear Power Plant Instrumentation and Control, Tokyo, Japan, 18-22 May 1992, 11 p.

IEC 812. 1985. Analysis Techniques for System Reliability - Procedure for Failure Mode and Effects Analysis (FMEA). 41 p.

McKinlay, A. 1991. The State of Software Safety Engineering. In "Apostolakis, G (ed.) Probabilistic Safety Assessment and Management, Elsevier, New York". pp. 369-376.

Leveson, N.G. & Stolzy, J.L. 1987. Safety Analysis Using Petri Nets. IEEE Trans. Software Eng. Vol-SE-13, No. 3, pp. 386-397.

Leveson, N.G., Cha, S.S. & Shimeall, T.J. 1991. Safety Verification of ADA Programming Using Software Fault Trees. IEEE Software, July 1991, 49-59.

Musa, J.D., Iannino, A. & Okumoto, K. 1987. Software Reliability. Measurement, Prediction, Application. McGraw-Hill Book Company, New York. 621 p.

Parnas, D.L., Asmis, G.J.K. & Madey, J. 1991. Assessment of Safety-Critical Software in Nuclear Power Plants. Nucleart Safety, Vol. 32. No. 2, April-June 1991. pp. 189-198.

Parnas, D.L., Asmis, G.J.K. & Madey, J. 1991. Assessment of Safety-Critical Software in Nuclear Power Plants, Nuclear Safety, Vol 32, No 2, April-June 1991, pp. 189 - 198.

Pulkkinen, U. 1993. Reliability Analysis of Software Based Safety Functions. STUK-YTO-TR 53. Helsinki 1993. 64 p. (In Finnish).

Reifer, D.J. 1979. Software Failure Modes and Effects Analysis. IEEE Trans. Reliability, vol. 28 no. 3. pp. 247-249.

Sauter, G.D. 1991. Providing an Improved Regulatory Basis for New Nuclear Power Plants, ANSI Topical Meeting on "The Next Generation of Nuclear Power Plants", November 10-14, 1991, pp. 175-177.

Taylor, J.R. 1992. Human Reliability and Software Safety Analysis for Command and Control Systems. In "Petersen, K.E. & Rasmussen, B. (ed.) Safety and Reliability '92", Proceedings of the European Safety and Reliability Conference '92, The first Pan European conference on safety and reliability under auspices of ESRA (European Safety and Reliability Association, CEC), Copenhagen, Denmark 10-12 June 1992. pp. 762-771.

Toola, A. 1992. Safety Analysis in Conceptual Design of Process Control. Technical Research Centre of Finland, VTT Publications 117, Espoo. 100 p.

Ward, P.T., Mellor, S.J. 1985. Structured Development for Real-Time Systems. Volumes 1-3. Yourdon Press, New York.

YVL Guide 2.8. 1987. Probailistic Safety Analyses (PSA) in the Licensing and Regulation of Nuclear Power Plants. Finnish Centre for Radiation and Nuclear Safety, 18.11.1987. 19 p.