Paper presented at the IAEA Technical Meeting on Common-Cause Failures in Digital Instrumentation and Control Systems of Nuclear Power Plants, 19-21 June 2007, Bethesda, Maryland, USA.

Proving absence of CCFs; a case for Open Source

Björn Wahlström, Olli Ventä, Janne Valkonen
Technical Research Centre of Finland
POB 1000
FI-02044 VTT (Espoo)
Finland

Abstract: Common cause failures have emerged as the major issue in licensing digital I&C systems. It is easy to use far-fetched scenarios to argue that there is a possibility that all redundancies will fail at the same time. The difficulty of countering such arguments has led to the introduction of various kinds of diversity, which increases both complexity and costs of the solutions. Risk-informed arguments can be used to show that selected scenarios are unlikely and not worth considering. This opportunity has not been used in actual projects due to obstacles for carrying out a detailed analysis. This situation would change if the target applications were built on Open Source solutions. The paper investigates some approaches to this problem when both the source code and its development history are available. More generally the paper argues that Open Source can offer the nuclear industry many benefits, especially when aiming at reusing earlier engineering solutions.

1 INTRODUCTION

Common cause failures (CCF) have emerged as one of the major issues for discussion in licensing digital I&C systems. It is easy to use some far-fetched scenario and argue that there always is a possibility that all redundancies will fail at the same time. The difficulty of countering such arguments has led to the introduction of various kinds of diversity, which increases both complexity and costs of the protective functions in consideration.

It is at least in principle possible to use risk-informed decision arguments, i.e. a combination of deterministic and probabilistic reasoning, to show that the likelihood of selected scenarios are below the rest risk level and, therefore, not worth considering. So far this opportunity has not surfaced to actual projects due to many obstacles in carrying out a detailed analysis of the software. The situation would at least partly change if the target applications were built on platforms implemented with Open Source[1] software modules.

More generally it can be argued that Open Source based solution could offer the nuclear industry many benefits, especially when aiming at a larger reuse of earlier designs in I&C projects [1]. The paper investigates some methodological solutions in approaching the problem of proving an absence of CCFs in software based systems. The proposed solutions rely on the availability of both the source code and its development history, which very seldom are available for proprietary software, but as a rule are available for Open Source software.

[1] The term Open Source is in this connection used in a generic meaning of software that is distributed under a user license, which gives free access to the source code and its development history.

2 OPEN SOURCE IN SOFTWARE DEVELOPMENT

The history of the Open Source movement can be found in many places on the Internet and will therefore not be repeated in this connection [2]. To summarise, one may say that the Open Source movement has grown from a hobbyist activity in the beginning of the 1990ies to a serious business, where today thousands of servers all over the world operate with Open Source software. To understand this success one has to go beyond the technical issues and consider also the driving motives of developers and users of Open Source software [3]. One indication of the seriousness of Open Source within the software industry is a recent special issue in the journal Management Science [4].

2.1 A new software development paradigm

A recent book gives a broad exposé of the Open Source movement in terms of motivations for developers, evaluations of features, used processes and tools, business models and societal issues [5]. Like any other new technology Open Source has its pros and cons, which have to be evaluated carefully before entering a path of large investments in time and resources. The openness however, provides an easy entrance, which can be used to evaluate the salient characteristics of Open Source software without too large investments.

The Open Source movement and its approach software development can be considered as a true paradigm shift [6]. As opposed to traditional software development in top-down organised teams, Open Source code is developed in a bottom-up fashion using interactions over the Internet involving people from the whole world. The success of many Open Source projects have also shown that the concept viable both technically and economically. Members of the traditional software community have tried to downplay the importance of this paradigm shift, by voicing doubts on the resulting quality of the Open Source products. The Open Source movement has also been referenced as a hacker community, which is destroying intellectual property. These negative characterisations seem however to be more based on concerns for loosing advantageous positions on the market than on real evidence.

2.2 Present views on Open Source

In assessing the usability of Open Source solutions for some specific application it is necessary to evaluate a mixture of technical, economical and social issues. One technical argument is that openness supports quality and this may be true to some extent, but quality relies more heavily on good programmers and development processes. It is therefore important to what extent a specific Open Source project can attract good programmers and build a well structured development process. Another argument in support of quality is that Open Source software gets tested more thoroughly as compared with proprietary software, but this is not necessarily true, because testing is a very mundane task in the developments process.

It can be argued that the Open Source movement may be the victim of its own success [7]. One danger is that intended users will feel the technology as a threat to their own career or business opportunities. Open Source may also become a part of the establishment that will diminish the interest of a new generation of creative minds in participating in the development activities. The workload on the key persons in the Open Source movement may become excessive, which may cause them to opt out of stressful positions. A proper balance between the ambitions of specific projects and their participants may be difficult to reach with the result that projects are not able to reach their goals.

2.3 Eclipse – a community for Open Source development

Eclipse [8] can serve as one example of development within the Open Source community, which may be of special interest for the nuclear community. The projects within Eclipse are focused on building an open development platform comprised of extensible frameworks, tools and runtime modules for building, deploying and managing software across the lifecycle. The Eclipse platform is supported, extended and complemented by a large group of major technology vendors, universities, research institutions, start-ups and individuals.

Eclipse can be considered as a kernel-type plug-in loader which is surrounded by hundreds (maybe thousands) of plug-ins, which are fundamental building blocks of the Eclipse platform. Each plug-in is providing services to other plug-ins and is also using services provided by several other plug-ins. The power of Eclipse comes from the fact that each plug-in is activated only when it is needed to contribute the functionality of the platform. This makes the design modular and easy to reuse. The unnecessary parts of software can be removed to simplify the implementation, which makes testing and verification more fluent.

The Eclipse community seems to be constantly growing, but mostly in areas outside the safety critical industries. However, some organisations involved in safety critical and embedded software development have shown their interest towards Open Source by joining the Eclipse Foundation. Universities and research organisations are utilising Eclipse and the number of available Eclipse applications is increasing.

The software development is facing increasingly stringent demands with regard to complexity, reliability and safety, especially in the areas of consumer electronics, automotive applications, military and aerospace technology as well as in telecommunications and data communications. To improve the applicability of the Eclipse platform in the above-mentioned applications, there is a description [9] of an integrated formal framework in the development of software for safety critical systems.

2.4 Doing business on Open Source solutions

Open Source does not automatically mean "available for free" and this fact is illustrated by many companies that make their living on Open Source solutions for their customers in different roles [10]. As one representative from these companies remarked, making money on Open Source is a different business, because only one in a thousands users becomes a paying customer [11].

The diversification of roles into service providers, integrators, module developers, testers, etc., often involves small and rapidly moving companies. These companies show an innovation potential and growth, which sometimes even could be characterised as astonishing. This development trend may be compared to large monolithic software houses, which seem to be interested only in areas of big business, such as banking, office applications and the entertainment industry.

If we try to look into the future of software development, there is a very clear trend towards an increasing complexity of systems, which also means increasing complexity of the V&V process of the software in consideration. One may also see large profit hungry software companies that in their marketing positions are forcing their customers to upgrade their systems at shortening product release intervals. The bearing principle also seems to be to make the releases before the software has matured enough to be stable. In this situation it is clear that there are an increasing number of openings for small and smart actors in the field.

3 PROS AND CONS OF USING OPEN SOURCE IN NUCLEAR I&C

In one earlier papers we argued that there would be many reasons for using Open Source software in the nuclear field [1]. These arguments are revisited as applied to nuclear I&C systems in the sections below. Common beliefs of the unsuitability for Open Source solutions for high reliability purposes are also countered to make the case that the licensing of safety critical software would be easier if the I&C systems relied on Open Source solutions.

3.1 Characteristics of the nuclear field

The nuclear field has many characteristics, which makes it different from other hazardous industries. The most important one is that it is not enough to have a safe nuclear installation is, but it has also to be proved to the licensing authorities that it is safe. In the nuclear field the challenge of achieving a high reliability with unreliable components has been solved by the so called *defence-in-depth* principle, which means that multiple barriers are built in against unwanted sequences of events. More concretely this principle relies on redundancy, separation and diversity, which aims to ensure that no single failure will pose a threat to safety.

The application of the defence-in-depth principle for digital I&C systems poses several difficulties, because it should be used to provide sufficient proofs that the I&C will provide all intended and no unintended functions. However, this is almost impossible due to the complexity of used hardware and software platforms, because it is always possible to argue that design errors in the hardware and software platforms may cause simultaneous failures of several critical functions. This difficulty is further aggravated if the source code and its development history cannot be analysed.

Another important characteristic is the long life-cycles of nuclear power plants. The nuclear power plants were typically designed with a life-time of forty years to which it is common to apply for a life-extension of twenty years. When this life-time is compared with the typical life-time of computer and software products, which may be as short as three years the discrepancy is evident. The I&C systems in use at the nuclear power plants were typically based on analogue technology, but an increasing obsolescence is now forcing plants to move to digital technology. I&C vendors today offer a system life-time of only twenty years, which means that a plant built today, would be forced to go through at least two large modernisations during their life-time.

Time will show what kind of systems will be offered by nuclear I&C vendors in the future, but it is immediately clear that costs of modernisations will be prohibitive if the design of the initial systems cannot be reused. One solution that can help is to build the system specifications to be independent of the used technology, but this solution will only take off a relatively small part of the costs connected to the verification and validation (V&V) processes. Another cost saving solution is to base the I&C systems on pre-developed or so called COTS (commercial-of-the-shelf) products. The problem however, is then to be able to demonstrate that these products fulfil the defence-in-depth principle. If the software specifications, the source code and its development history are available for the products it should be possible to arrive at some level of confidence in the software.

3.2 Arguments in favour of Open Source

There are many arguments in favour of the Open Source model in the nuclear field. The most important arguments are connected to software quality and dependability. Of these arguments the scalability is perhaps the most important. With Open Source it is easy to scale the software to the application in consideration by removing unused functionality. The scalability of the Open Source has thus the benefit of supporting flexibility and simplicity.

The second important feature of the Open Source solutions is the openness, which gives an independence of a specific vendor. Assuming that the development process has been supported with tools that have been created using the Open Source paradigm it is, at least in principle possible, that any new company takes over the development process from the point it was finished in the original development process. It is also possible to plug out old modules and tools, and plug in new and improved modules and tools at various points in the development and the V&V processes.

A final argument is that Open Source solutions would make it possible to ensure a larger modularity for the I&C architecture in large as well as for specific components used in specialised functions. The availability of the source code is a necessary, but not a sufficient condition for achieving a better analysability, but it will make it possible to introduce additional V&V efforts such as code analysis and testing. It would therefore at least in principle be possible to make comparisons and benchmarks between products that have been developed within a common Open Source framework.

3.3 Arguments against Open Source

One of the main arguments against Open Source is the same as one of the main arguments for Open Source. The business is handled by small companies on which you may not rely in the long run. On the other hand even large vendors may equally well move out from the nuclear field or move to other business areas as a part of acquisitions or mergers. If the nuclear could present itself as an own interesting business niche, it may generate more support from Open Source based companies as compared with companies marketing proprietary solutions. Considering the nuclear industry, the after-market in software maintenance and modifications may even be more profitable than initial deliveries.

Security concerns are the other large argument against Open Source, especially because the area has traditionally been associated with the hacker community. The perhaps most important counter argument is that security by obscurity cannot be a viable concept. The proprietary software may actually represent a larger threat, because large software companies are usually slow in revealing vulnerabilities in their solutions and in releasing necessary patches. One may argue that the source code should be open only to the nuclear community for security reasons, but this solution does not seem to be viable, because that would prevent other fields in using the code and still leave the need to assume that the code is known publicly.

3.4 Predicting the future of I&C in the nuclear field

In assessing the relative positions of Open Source and proprietary solutions it is necessary to make projections of how the nuclear I&C field will develop. One trend is a combined scaling down and scaling up. Safety critical functions will be scaled down to smaller platforms, which have the benefit of a larger simplicity and therefore an easier licensing process. The non-critical functions will be scaled up to rely on more complex software and hardware platforms, to make it possible to utilise the benefits of increased functionality.

In pursuing an increased re-usability of engineering design from earlier plants and systems, the concept of *design patterns* [12], may prove to be an interesting concept for sharing good practices of I&C and software design. Design patterns for safety critical I&C that on lower levels are supported by various Open Source solutions, may prove to be efficient solutions that are easy to verify and validate. If for example a convincing case can be made that some new I&C solution function for function is the same as the old, the V&V efforts of the old functions may be re-used for the new one.

In the future we will see more small I&C devices with embedded software. Such devices include smart sensors and transmitters, simple control devices, time relays, uninterruptible power supplies, etc. If they would rely on Open Source in their technical solutions, additional confidence in their functions could be obtained through testing and code inspections. More generally Open Source could open up the nuclear field for a use of standard industrial components both in non-safety and safety related applications.

For the high reliability protection systems, it would be beneficial to ensure that operating experience is obtained from as many applications as possible. If these systems would be attractive also for the conventional industry, many additional hours of operating experience would be obtained.

4 GENERAL PRINCIPLES FOR AVOIDING CCFS

A consideration of CCFs is very important within the nuclear field, because the application of the defence-in-depth principle calls for an elimination of mechanisms that simultaneously may initiate failure in several barriers. In software systems, a CCF will occur if the same software error in redundant channels will be triggered simultaneously. This means that CCFs can be avoided if either the software errors can be eliminated or if it can be assured that the triggering mechanisms will not influence the redundant channels.

4.1 Sound software engineering

In designing software based I&C systems, the most important principle is to avoid software errors by using sound software engineering methods. A commonly used method is to separate between the application and the platform, which also has the benefit that the assessment of a specific hardware and software platform can be reused for several applications. Present methods for building applications are well adapted to specific I&C functions, which implies that the application software typically is simple and efficient. It may even be able to argument that the application software is simple enough to be correct with a large likelihood.

The separation between the application and the platform does not solve the basic problem, but moves it to the V&V of the platforms. Most platforms of today rely on software that has been developed using standardised methods of software engineering. This implies that well defined software development models have been used. More generally international standards suggest a software life-cycle approach, a careful development of requirements specification, structured V&V, the use of methods such as FMEA and HAZOP as well as careful review and testing in all development phases.

Good software engineering practices include design for simplicity and modularity with clean interfaces between modules. Defence-in-depth can be supported by building in separation between unrelated functions and barriers for error propagation. In testing it is possible to insert temporary probes for diagnostic purposes and for the evaluation of the coverage of the testing efforts.

4.2 Special considerations for I&C systems

In the field of I&C systems, a number of good design practices have been suggested to avoid classes of design errors. For example, it is very usual that deterministic and synchronous behaviour is required for the real time part of the software. This solution gives a protection towards the time capacity problems for the processors, which may occur in certain plant transients.

A good practice in the design of I&C systems is also to separate between normal operation, expected disturbances and unexpected combinations of events. Normal operation would in this case be understood as the base operation of monitoring and control of signals without a consideration of possible plant transients. Expected disturbances would then be for example I&C component failures and switch-over to standby components. Some protection against unexpected combinations of events could be built in by a careful checking of signal validity in the interfaces between functions and modules. Another example is to restrict dimensionality of the state space of the I&C system by allowing only well defined state variables to be stored between time steps.

The efficiency and coverage of testing can be enhanced by statistical testing against a test oracle, which is built on another computer using a suitable high level language. A final test of the I&C system can be obtained by testing against a plant simulator that is taken through major plant transients.

4.3 Avoiding common triggering mechanisms

The second step in avoiding CCFs, is to design the I&C system in such a way that common triggering mechanisms can be avoided. One example is to disable temporal couplings between redundant functions within the I&C system. This could be ensured by using free running controllers all with their own clock and excluding the handling of dates on the lower level components.

Stochastic fluctuations in the measurement channels implies small differences in the input signals to redundant functions, which makes it very unlikely that redundant processors would operate with the same data streams. Such characteristics may also be straightforward to demonstrate with a plant simulator. Restricting operator inputs to one redundancy in a turn would similarly reduce the CCFs that may emerge from the human system interactions.

Uni-directionality of interactions between components and modules can ensure the integrity of higher safety classes and thereby the possibility of common triggering mechanisms. Uni-directionality however, implies that the higher safety class does not have any possibility to ensure that the data that is sent to a lower safety class has arrived correctly, because the use of any handshaking mechanism would destroy the uni-directionality. This restriction is however easy to live with, because the cyclic nature of I&C functions would correct missing or wrongly received signals in the next time step.

5 PROVIDING EVIDENCE THAT CCFS HAVE BEEN AVOIDED

For several reasons it may be a considerable challenge to provide evidence that developed software fulfils all intended and no unintended functions [13]. The approach for arguing that the likelihood of CCFs is very small, is to use a mine field analogy and argue that the well threaded paths of the software do not contain any unwanted surprises [14].

5.1 A method for analysing the possibility of CCFs

A method for analysing the possibility of CCFs has been proposed in a draft document [15], which is under development in a working group of the IAEA-TWG-NPPCI [16]. The method can briefly be described as follows:

– *Define the objective of analysis*. In this step, credible fault types and triggers are postulated to be analysed in subsequent steps.
– *Decompose the system to be analysed into blocks*. An arbitrary boundary within the system is drawn to define the part to be analysed. For the same system, there may be a need to developed alternate block representations depending on the nature of the CCF to be assessed.
– *Identify vulnerabilities*. This is done by finding prospective faults and trigger combinations that cannot be dismissed.
– *Determine if postulated CCFs can be prevented*. This may be achieved by additional protection against faults and trigger combinations.
– *Assess the need for additional defensive measures*. Can the protection against CCFs be considered good enough? If not, proceed with an assessment of additional fault types and triggers that should be postulated.

This method has to be supported by a risk informed approach in which both deterministic and probabilistic arguments are used. This reasoning is elaborated in more detail below.

5.2 Deterministic arguments

The deterministic arguments would be built on evidence collected during the software design process. The used software architecture and the module design may for example give evidence that certain error classes and triggering mechanisms have been successfully avoided. The simplicity reached by the removal of unnecessary functions within the software may support a careful inspection of the source code by available methods and tools. A restriction of the state space to the absolutely essential may provide enough evidence for the argument that data area and pointer overflows have been made impossible.

A record of statistical testing together with data from inserted test probes may provide arguments that the testing efforts have been reasonably complete. A combination of testing history and assurance that the input data streams are different enough could serve an argument that one specific triggering mechanism has been made impossible and so on.

5.3 Probabilistic arguments

For safety critical functions, it is necessary to be able to give some quantification of their reliability to be used in the plant specific probabilistic safety assessment (PSA). The statistical testing and the testing against a plant simulator can give some bounds on the reliability, but it is likely that these estimates have to be amended by expert judgment, which is based on deterministic reasoning about design solutions. Basically this reasoning would aim at giving evidence that specific failure mechanisms can be considered very unlikely or below the residual risk level as defined within the PSA.

The PSA can support a sensitivity analysis to calculate the levels at which the probability for a certain CCF will influence for example the core melt frequency. If deterministic arguments based on the system and software architecture, source code, testing arrangements and records, the quality of the development process, etc., can be used to build confidence in that the probabilities of certain CCFs are below other important sources of uncertainty, requirements on diversity may be relaxed. In some cases, it may even be argued that diversity will introduce higher risks due to the burden of increased complexity.

6 DESIGNING FOR VERIFICATION AND VALIDATION

At a more general level it may be argued that the need for V&V has to be built into the design of I&C systems already from the beginning. If this could be achieved it would make the reasoning about requirements, claims and evidence easier [17]. It would also introduce the need for bringing a larger formality in the requirements specifications, because specifications written in natural languages are always objects for interpretations. There is also a need for an integration of good software design practices into the requirements specification to make them easier to verify in a licensing process.

Basing nuclear I&C on Open Source solutions that are applied throughout the software development process, has at least in principle potential for ensuring a that a simplified V&V processes will give as good or better results than the processes that are in use today. Such solutions will also give a larger possibility to reuse earlier engineering efforts both in the design and in the V&V process for the modernisations that are necessary during the plant life-time.

If this argumentation can be applied, it removes unnecessary diversity, which now seems to be used as a solution to disagreements on possibility and likelihood of CCFs. However, it will not and should not remove diversity in places where it is needed. Diversity will for example be necessary in human work processes and in time to ensure that the same error is not introduced as a latent non-functionality in several redundancies. Functional diversity in shutdown systems is also well motivated, but it should not necessarily be implemented with diverse I&C platforms. The important issue is that the need for diversity can be assessed and analysed.

7 CONCLUSIONS

In the paper we have argued that Open Source can have a place in the I&C systems of nuclear power plants. We have also argued that a claim that certain CCFs are very unlikely cannot be given without a thorough analysis of the source code. We would in this connection like to go even further to recommend that the nuclear utilities should initiate a joint project to investigate the pros and cons of requiring Open Source solutions for their high reliability I&C solutions. If such a project would indicate a benefit, the easiest path to proceed would be to initiate the writing of requirements specification that would be based on Open Source solutions and a design that would support the V&V process. If these activities are gaining momentum, they will most likely attract vendor interests.

Regarding the licensing of digital I&C, it seems possible that the CCFs can be taken care of in an ordered manner, which is not to say that the reasoning would be easy. However, it seems to be the only way to generate the necessary reliability data for software based systems to be integrated in the PSAs of the nuclear power plants. This development would be important to enable a quantification among products to help architect engineers of the future nuclear power plants to select the best components for their I&C systems.

8 REFERENCES

[1] Olli Ventä, Björn Wahlström (2007). Investigating the case of Open Source applications within nuclear power, Enlarged Halden Programme Group meeting, Storefjell, Norway, 12-15 March.

[2] http://www.opensource.org.

[3] Steven Weber, *The Success of Open Source*, Harvard University Press, 2004.

[4] Georg von Krogh, Eric von Hippel, The promise of research on Open Source software, *Management Science*, Vol.52, No.7, pp.975-983, 2006.

[5] Joseph Feller, Brian Fitzgerald, Scott A. Hissam, Karim R. Lakhani (eds.) (2005). *Perspectives on Free and Open Source Software*, The MIT Press, Cambridge, Mass.

[6] Tim O'Reilly (2005). The Open Source paradigm shift, pp.461-481 in [5].

[7] Brian Fitzgerald (2005) Has Open Source a future? pp.93-106 in [5].

[8] http://www.eclipse.org/.

[9] Jiang Guo, Yuehong Liao, Raj Pamula (2005). Extending Eclipse to support object-oriented system verification. IRI -2005 IEEE International Conference on. Information Reuse and Integration, 15-17 Aug. 2005, pp.282-287.

[10] Sandeep Krishnamurthy (2005). An analysis of Open Source business models, pp.279-296 in [5].

[11] Personal communication with a representative from MySQL.

[12] Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides (1995). *Design patterns: Elements of reusable object-oriented software*, Addison Wesley.

[13] Björn Wahlström (2005). Risk assessment and safety engineering; applications for computer systems, SAFECOMP 2005, the 24th International Conference on Computer Safety, Reliability and Security, Fredrikstad, Norway.

[14] Thuy Nguyen, Ray Torok (2006). Assessment of Digital Equipment for Safety and High Integrity Applications, Defense-in-Depth and Diversity, presentation at the Joint IAEA – EPRI Workshop on Modernisation of Instrumentation and Control Systems in NPPs, 3-6 October, Vienna, Austria.

[15] IAEA (2006). TECDOC on Avoiding Common-Cause Failures in Digital I&C Systems of NPPs, draft 31.8.2006.

[16] http://www.iaea.org/NuclearPower/IandC/.

[17] P.-J. Courtois (2005). Towards a Deductive Approach for the Safety Justification of Computer Based Systems, IAEA Technical Meeting on Licensing Digital Instrumentation and Control Systems and Equipment in Nuclear Power Plants, Espoo, Helsinki, Finland, 22-25 November.