NKS-361
ISBN 978-87-7893-445-1

Modelling of DIgital I&C, MODIG — Interim report 2015

Stefan Authén[1]
Ola Bäckström[2]
Jan-Erik Holmberg[3]
Markus Porthin[4]
Tero Tyrväinen[4]

[1]Risk Pilot AB
[2]Lloyds Register Consulting
[3]Risk Pilot AB Suomen sivuliike
[4]VTT Technical Research Centre of Finland Ltd

March 2016

Abstract

The NKS-project MODIG (MODelling of DIGital I&C) aims to get a consensus approach for a reliability analysis of a plant design with digital I&C, improved integration of probabilistic and deterministic approaches in the licensing of digital I&C, improved failure data collection including software failure probability quantification, and a practical application of probabilistic safety assessment (PSA).

A survey of the defence-in-depth (DiD) framework and PSA's role in it has been made. The assessment of DiD and diversity is in principle straightforward for PSA, e.g., risk metrics can be used to evaluate DiD levels 3, 4 and 5. A PSA model always includes uncertainties, which needs to be accounted for especially when comparing with deterministic assessment. Regarding digital I&C, the focus of the assessment is on the DiD levels 1, 2 and 3. In addition the logic model of PSA can be used in the assessment of deterministic failure criteria.

Spurious actuation is a functional failure mode when a component performs a function without a real demand. Spurious actuations are of special interest for I&C due to complex effects via system dependences and due to a huge number of possible failure locations. There is a need to develop a reasonable but comprehensive approach both for deterministic and probabilistic analyses. Analysis requirements have been compiled, and a generic failure modes taxonomy and an analysis approach have been outlined.

The software reliability task has been working on the confidence building in the method to estimate application software failure probability. The impact of pooling data from high and low demand systems is discussed. The principle of the probability estimation has been adjusted from the approach developed in the DIGREL project. A solution for the software complexity assessment has been prepared.

I&C failure data is one of the information sources needed for the assessment of I&C reliability. Vendors have data sources as they typically have access to experience data from many plants, have needed insight on the software development processes and are capable to analyse the causes of the detected failures. International collaboration and discussions are still needed in order to forward the use of I&C failure data in PSA.

Key words

Digital I&C system, probabilistic safety assessment, reliability, nuclear power plant safety

NKS-361
ISBN 978-87-7893-445-1
Electronic report, March 2016
NKS Secretariat
P.O. Box 49
DK - 4000 Roskilde, Denmark
Phone +45 4677 4041
www.nks.org
e-mail nks@nks.org

Modelling of DIgital I&C, MODIG — Interim report 2015

Report from the NKS-R MODIG activity (Contract: AFT/NKS-R(15)116)

Stefan Authén[1]
Ola Bäckström[2]
Jan-Erik Holmberg[3]
Markus Porthin[4]
Tero Tyrväinen[4]

[1]Risk Pilot AB, Parmmätargatan 7, SE-11224 Stockholm, Sweden
[2]Lloyds Register AB, P.O. Box 1288, SE-172 25 Sundbyberg, Sweden
[3]Risk Pilot AB Suomen sivuliike, Metallimiehenkuja 10, FI-02150, Espoo, Finland
[4]VTT Technical Research Centre of Finland Ltd, P.O. Box 1000, FI-02044 VTT, Finland

Table of contents

Abbreviations

A/D	Analog/digital
ACP	AC power system
AIM	Analog input module
ALOCA	Large loss-of-coolant accident
AOM	Analog output module
APU	Acquisition and processing unit
APU-AS	APU application-specific software module
APU-FRS	APU functional requirements specification module
AS	Application software (module)
BBN	Bayesian belief network
BWR	Boiling water reactor
CCF	Common cause failure
CCI	Common cause initiator
CCW	Component cooling water system
CD	Core damage
CDF	Core damage frequency
COM	Communication link module
COTS	Commercial off-the-shelf
CPU	Central processing unit
CSNI	Committee on the Safety of Nuclear Installations (OECD/NEA)
DBC	Design basis condition
DCS	Data communication software
DEC	Design extension condition
DFLT	Default value
DF	Detected fault
DiD	Defence-in-depth
DIM	Digital input module
DOM	Digital output module
DCU	Data communication unit
DLC	Data link configuration
ECC	Emergency core cooling system
EDF	Électricité de France
EF	Elementary function
EFW	Emergency feedwater system
ESFAS	Engineered safety features actuation system
ET	Event tree
FMEA	Failure mode and effects analysis
FC	Fractional contribution
FRS	Functional requirements specification
FT	Fault tree
FTD	Fault tolerant design
GRS	Gesellschaft für Anlagen- und Reaktorsicherheit, Germany
HSI	Human-system interface
I&C	Instrumentation and control
I/O	Input/output
IAEA	International Atomic Energy Agency
ICDE	OECD/NEA International Common-cause Failure Data Exchange Project
IE	Initiating event

IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronics Engineers
ISO	International Organization for Standardization
ISTec	Institut für Sicherheitstechnologie
LMFW	Loss of main feedwater
LOCA	Loss-of-coolant accident
LOOP	Loss-of-offsite power
LERF	Large early release frequency
LRF	Large release frequency
MFW	Main feedwater system
MOV	Motor operated valve
MU	Manual control unit (I&C unit for main control room operations)
NEA	OECD Nuclear Energy Agency
NKS	Nordic nuclear safety research
NPP	Nuclear power plant
NPSAG	Nordic PSA Group
NRC	U.S. Nuclear Regulatory Commission
NSA	Not-self-announcing (fault of software)
OECD	Organisation for Economic Co-operation and Development
pfd	Probability of failure per demand
PSA	Probabilistic safety assessment
QA	Quality assurance
RCS	Reactor control system
RDF	Risk decrease factor
RIF	Risk increase factor
RHR	Residual heat removal system
RLS	Reactor limitation system
RPS	Reactor protection system
RT	Reactor trip
SA	Self-announcing (fault of software)
SAFIR	Finnish Research Programme on Nuclear Power Plant Safety
SCM	Signal conditioning module
SICA	Simple complexity analysis
SSM	Strålsäkerhetsmyndigheten, Swedish Radiation Safety Authority
SSMFS	SSM regulation series
SW	Software
SWS	Service water system
SyS	System software
TXS	TELEPERM® XS, product of AREVA
VU	Voting unit
VU-AS	VU application-specific software module
VU-FRS	VU functional requirements specification module
V&V	Verification and validation
VTT	Technical Research Centre of Finland
WENRA	Western European Nuclear Regulators Association
WGRISK	OECD/NEA CSNI Working Group on Risk Assessment
YVL	Ydinvoimalaitos (nuclear power plant), STUK's regulatory guide series for nuclear facilities

Summary

The NKS-project MODIG (MODelling of DIGital I&C) aims to get a consensus approach for a reliability analysis of a plant design with digital I&C, improved integration of probabilistic and deterministic approaches in the licensing of digital I&C, improved failure data collection including software failure probability quantification, and a practical application of probabilistic safety assessment (PSA) to compare design alternatives. In 2015, MODIG explored the assessment of defence-in-depth by PSA with an emphasis on I&C, outlined an approach to analyse spurious actuations, developed further the confidence on the software reliability method proposed in the previous NKS project DIGREL and prepared a proposal for an international collaboration on the development of a systematic approach for the diversity assessment of digital I&C systems for PSA (OECD/NEA Working Group RISK task proposal). A joint workshop together with the NKS project PLANS was organised with more than 40 participants from seven European countries.

A survey of the defence-in-depth (DiD) framework and PSA's role in it has been made. The assessment of DiD and diversity is in principle straightforward for PSA, e.g., risk metrics can be used to evaluate DiD levels 3, 4 and 5. A PSA model always includes uncertainties, which needs to be accounted for and argumented, especially when comparing with deterministic assessment. Regarding digital I&C, the focus of the assessment is on the DiD levels 1, 2 and 3. DiD level 4 (severe accident management) is also assessed, but it is quite simple from I&C point of view. In addition the logic model of PSA can be used in the assessment of deterministic failure criteria.

Spurious actuation is a functional failure mode when a component performs a function without a real demand. Spurious actuations are of special interest for I&C due to complex effects via system dependences and due to a huge number of possible failure locations. There is a need to develop a reasonable but comprehensive approach both for deterministic and probabilistic analyses. Analysis requirements have been compiled, a generic failure modes taxonomy has been outlined based on von Wright's theory on concept of action, and an analysis approach has been outlined.

The software reliability task has been working on the confidence building in the method to estimate application software failure probability. The impact of pooling data from high and low demand systems is discussed. Also the principle of the probability estimation has been adjusted from the approach developed in the DIGREL project. A solution for the software complexity assessment has been prepared.

I&C failure data is one of the information sources proposed to be used for the assessment of I&C reliability. Vendors of digital I&C have shown to be potential data sources as they typically have access to experience data from many plants, have needed insight on the software development processes and are capable to analyse the causes of the detected failures. The OECD/NEA ICDE project has also started collection of digital I&C related failure data in 2015. ICDE's primary focus is on understanding of failure causes and ways of prevention, and reliability quantification is not pursued. Thus, the digital I&C related failure data collected by ICDE does not likely support PSA in the best possible way. International collaboration and discussions are still needed in order to forward the use of I&C failure data in PSA. The MODIG project partners strive to foster such discussions e.g. through active participation in the WGRISK group and international seminars and conferences.

Acknowledgements
The work has been financed by NKS (Nordic nuclear safety research) and SAFIR2018 (The Finnish Research Programme on Nuclear Power Plant Safety 2015–2018). NKS conveys its gratitude to all organizations and persons who by means of financial support or contributions in kind have made the work presented in this report possible.

1. Introduction

The NKS project DIGREL (2010–14) developed guidelines for analysis and modelling of digital systems in probabilistic safety assessment (PSA) for nuclear power plants. The project consisted of three interrelated activities. First, a taxonomy for failure modes of digital I&C systems was developed by a task group of OECD/NEA Working Group RISK (OECD 2015). Second, a fictive digital I&C PSA-model was developed for the demonstration and testing of modelling approaches (NKS-230, NKS-261, NKS-277, NKS-302, NKS-330). Third, a method was developed for the quantification of software reliability in the context of PSA (NKS-304, NKS-341).

As a result of DIGREL, there is a good understanding of sufficient level of details for PSA modelling and an approximate idea of treatment of software failures. The DIGREL scope was, however, limited to a simple reactor protection system architecture, and, also human reliability analysis was out of the scope. Based on conclusions from the DIGREL-project, issues left out of the scope and discussions with stakeholders, a number of relevant issues have been identified to be studied further in the MODIG (Modelling of Digital I&C) project. A further important aspect of the MODIG project is to foster international collaboration, which is considered essential in a safety assessment area, where no consensus has not yet been reached.

This report provides interim results from the MODIG project. Chapter 2 defines scope and objectives of the project. Chapter 3 provides main definitions used in the report. Chapter 4 describes the survey of defence-in-depth framework and its relationship with PSA. Chapter 5 discusses analysis of so called spurious actuations, which is a specific category of failure modes, which can be harmful for I&C. Chapter 6 presents results from the further development of the software reliability analysis method. Chapter 7 discusses the challenges related to I&C failure data and recent efforts of its collections. Chapter 8 concludes the work.

2. Scope and objectives

The MODIG project is part of a SAFIR2018 (the Finnish Research Programme on Nuclear Power Plant Safety 2015−2018) research project "Integrated safety assessment and justification of nuclear power plant automation" (SAUNA). The overall objective of the SAUNA project is to develop principles and methods to design and assess nuclear power plant process and automation together with respect to the fulfilment of the defence-in-depth principle.

MODIG focuses on PSA, which is the main method to be further developed. One of the objectives is to get a consensus approach for a reliability analysis of a plant design with digital I&C, improved integration of probabilistic and deterministic approaches in licensing of digital I&C, improved failure data collection including software failure probability quantification, and a practical application of PSA to compare design alternatives.

In 2015, MODIG started with the topics on the assessment of defence-in-depth, diversity and complexity, analysis of spurious actuations, software reliability analysis and digital I&C failure data. The example model of DIGREL has been further developed to test and demonstrate the methods.

3. Definitions

Active failure: An active failure leads to a spurious actuation of a function.

Application function: function of an I&C system that performs a task related to the process being controlled rather than to the functioning of the system itself. Also referred to as I&C function.

Application software (module): piece of software that is represented by a specific group of lines of source code (or equivalent graphical representation, e.g. function diagrams) and has a specific functionality. The application software is the representation of the application functions in form of code. The application software is executed and controlled by the system software (run time environment) during an operating cycle.

Common cause failure: Failure of two or more structures, systems and components due to a single specific event or cause. (IAEA 2007)

Complex function block: In the SICA method, a complex function block is a function block that uses internal memory or for which the sum of the number of inputs, outputs and parameter is over ten.

Complexity (of software): A single metric that expresses how many parts software contains (e.g. code lines), how much different parts are connected to each other and how diverse different parts are. There are several alternatives to a complexity metric.

Connected function blocks: In the SICA method, function blocks are defined connected if they affect the same output of the software module.

Data communication software (DCS): This software module implements the data communication protocol. It is part of the platform software.

Data link configuration (DLC): This software module is provided in the form of a data table. It specifies the nodes that can be part of a given network, and the data messages that can be exchanged between the nodes of the network.

Demand: A plant state or an event that requires an action from I&C. Note: A state of the I&C system requiring an action of an active fault tolerant design feature is not considered a demand. In this report, "demand" is used in the same meaning as in the reliability metric "probability per demand", and is a specific uncovering situation of a failure, distinct from dedicated failure detection mechanisms such as online and offline monitoring.

Diversity: The presence of two or more redundant systems or components to perform an identified function, where the different systems or components have different attributes so as to reduce the possibility of common cause failure, including common mode failure. (IAEA 2007)

Fail safe: Pertaining to a functional unit that automatically places itself in a safe operating mode in the event of a failure (ISO/IEC/IEEE 2010); "system or component" has been replaced with "functional unit") Example: a traffic light that reverts to blinking red in all

directions when normal operation fails. Note: In general fail safe functional units do not show fail safe behaviour under all possible conditions.

Failure: Termination of the ability of a product to perform a required function or its inability to perform within previously specified limits (ISO/IEC 2005). "Failure" is an event, as distinguished from "fault" which is a state.

Failure effect: Consequence of a failure mode in terms of the operation, function or status (IEC 2006, “of the system” removed).

Failure mode: The physical or functional manifestation of a failure (ISO/IEC/IEEE 2010).

Failure mechanism: Relation of a failure to its causes.

Fatal failure: The I&C unit or the hardware module stalls. It ceases functioning and does not provide any exterior sign of activity. Fatal failures may be subdivided into:

> **Ordered fatal failure:** The outputs of the I&C unit or the hardware module are set to specified, supposedly safe values. The means to force these values are usually exclusively hardware. Equivalent to the definition “Halt/abnormal termination of function with clear message” (Chu et al. 2006).
>
> **Haphazard fatal failure:** The outputs of the I&C unit or the hardware module are in unpredictable states. Equivalent to the definition “Halt/abnormal termination of function without clear message” (Chu et al. 2006).

Fault: Defect or abnormal condition that may cause a reduction in, or loss of, the capability of a functional unit to perform a required function (IEC 2010b; “defect” added). Note: "Failure" is an event, as distinguished from "fault" which is a state.

Fault tolerance: The ability of a functional unit to continue normal operation despite the presence of failures of one or more of its subunits. Note: Despite the name this definition refers to failures, not faults of subunits. It is therefore distinct from the definition in (ISO/IEC/IEEE 2010). Possible means to achieve fault tolerance include redundancy, diversity, separation and fault detection, isolation and recovery.

Feedback loop (in software logic diagram): A path in a logic diagram that starts and ends with the same point (any point in the feedback loop can be considered as a start/end point).

Function block: reusable, closed, and classifiable piece of software, capable of processing signals, from which I&C functions can be assembled using function diagrams. Function blocks operate in a closed and well-defined manner. Also called elementary or library function.

Function diagram: diagram that specifies the application software to be run within an I&C system by connecting function blocks with each other and with external signals.

Functional requirements specification (FRS): documentation that describes the requested behaviour of an engineering system and includes the operation and activities that a system must be able to perform.

Initiating event: An initiating event is an event that could lead directly to core damage (e.g. reactor vessel rupture) or that challenges normal operation and which requires successful mitigation using safety or non-safety systems to prevent core damage (IAEA 2010).

Non-fatal failure: The I&C unit or the hardware module fails but it continues to generate outputs. Non-fatal failures may be subdivided into:

> **Failures with plausible behaviour:** I&C runs with wrong results that are not evident (Chu et al. 2006). An external observer cannot determine whether the I&C unit or the hardware module has failed or not. The unit is still in a state that is compliant to its specifications, or compliant to the context perceived by the observer.
> **Failures with implausible behaviour:** I&C runs with evidently wrong results (Chu et al. 2006). An external observer can decide that the I&C unit or the hardware module has failed. The unit is clearly in a state that is not compliant to its specifications, or not compliant to the context perceived by the observer.

Not-self-announcing (NSA) fault: A not-self-announcing application software module fault is a fault that cannot be detected by the I&C system itself. NSA faults with a passive failure can only be revealed by an observer in case of a demand. NSA faults with active failure could be observed both during plant operation and at a demand, since these would lead to an unexpected signal.

Passive failure: A passive failure leads to an unavailability of the output signal, i.e. failure to actuate.

Proprietary software: code that is embedded in specific hardware modules, different from the microprocessor module of acquisition and processing unit (APU), voting unit (VU) and data communication unit (DCU), and that performs a function of its own. It can be also designated as "software in COTS" (Commercial off-the-shelf). This software is proprietary and its source code is generally not available for the end user.

Redundancy: Provision of alternative (identical or diverse) structures, systems and components, so that any one can perform the required function regardless of the state of operation or failure of any other. (IAEA 2007)

Self-announcing (SA) fault: A self-announcing application software module fault is a fault which is detected by the I&C system via self-monitoring. The fault is displayed in an interface such that the operator can exactly find the location of the fault in the I&C system.

Spurious actuation: A failure where an actuation of an I&C function occurred without a demand. Spurious actuation can be caused by any failure between the process measurement sensors and the actuator, including erroneous operator command or failure of watchdogs.

System software: The operating system and runtime environment (interaction between application and operating system).

4. Defence-in-depth

4.1 Introduction

Defence-in-depth (DiD) is a widely applied safety and security principle in all safety-critical technological areas even if it may be called differently in some context. In safety management context, defence-in-depth means having more than one protective measure for a given safety objective, such that the objective is achieved even if one of the protective measures fails. The protective measures can be anything from inherent safety features, use of multiple barriers, engineered safety features, principles and procedures followed in design, construction, operation, maintenance and decommission of the system.

Objective of this chapter is to describe defence-in-depth concept with emphasis on the role of I&C and probabilistic assessment of it. DiD relies on the application of multiple barriers, physical separation, redundancy and diversity. From the I&C architecture point of view this multitude of principles and necessity to implement several barrier functions means a complex design task. It is also challenging to make the safety demonstration both deterministically and probabilistically.

PSA can be used in this context both as a probabilistic tool and a logic model tool. The probabilistic usage is the ordinary way of using PSA as done in various risk-informed applications (e.g. to show compliance with numerical risk criteria, to compare design alternatives, etc.). This can be needed if and when deterministic criteria are not fully met. PSA can also be used "deterministically" by utilizing the logic model, which captures the system and scenario dependences. Both usages are discussed in this chapter.

4.2 Regulatory requirements

4.2.1 IAEA definition

The IAEA safety guide INSAG-10 (IAEA 1996) defines defence-in-depth as follows "A hierarchical deployment of different levels of diverse equipment and procedures to prevent the escalation of anticipated operational occurrences and to maintain the effectiveness of physical barriers placed between a radiation source or radioactive material and workers, members of the public or the environment, in operational states and, for some barriers, in accident conditions."

The objectives of defence in depth are:

- To compensate for potential human and component failures;
- To maintain the effectiveness of the barriers by averting damage to the facility and to the barriers themselves;
- To protect workers, members of the public and the environment from harm in accident conditions in the event that these barriers are not fully effective.

Defence-in-depth is usually described in two ways for nuclear safety (IAEA 1999):

- a system of successive physical barriers isolating the radioactive fuel from the environment (fuel matrix, fuel rod cladding, primary coolant boundary, reactor confinement)
- a system of successive levels of protection following the logic of the accident model
    - Level 1: Prevention of abnormal operation and failures.

- Level 2: Control of abnormal operation and detection of failures.
- Level 3: Control of accidents within the design basis.
- Level 4: Control of severe plant conditions, including prevention of accident progression and mitigation of the consequences of severe accidents
- Level 5: Mitigation of radiological consequences of significant releases of radioactive material.

In the safety analyses, it shall be demonstrated that for an identified set of initiating events:

- deterministically, unwanted consequence is avoided even if one or more failures are postulated in the barriers. Failure criteria depend on the frequency of the (postulated) initiating event
- probabilistically, the frequency of the unwanted consequence is less than the numerical criterion.

Defence-in-depth is in practice incorporated everywhere in the safety management of nuclear power plant, like regulatory requirements and oversight procedures, QA activities at various system life cycle phases, safety classification of systems, structures and components, organizational structure and responsibilities and safety culture considerations. Fleming & Silady (2002) point out this all makes defence-in-depth as a multi-faceted framework. They distinguish three usages of the term defence-in-depth:

1. Design defence-in-depth: design feature to have multiple and physical lines of defence between the hazard and the public
2. Process defence-in-depth: Incorporation of the defence-in-depth thinking into the licensing requirements. Although there is a relationship between these requirements and the detailed design features that are reflected in design defence-in-depth, they are not one in the same as they are controlled by different stakeholders in the process.
3. Scenario defence-in-depth: strategies to prevent initiating events from occurring and from progressing to accidents, and strategies to mitigate the consequences of events and accidents.

It should be finally noted that while the IAEA definitions can be considered generally accepted and followed in all countries, nuclear regulatory requirements differ between countries. Above all, the requirements for new reactors are much harder than they were for older reactors. One of the recent extensions, is the requirement for the so called design extension cases, which also needs to be managed. It splits the level 3 into 3a and 3b, adding one more level in the DiD hierarchy (see e.g. WENRA 2013; STUK 2013a)

4.2.2 Finnish and Swedish regulatory requirements

Finnish nuclear safety requirements related to defence-in-depth principle are given in the Government decree 717/2013 which are further specified in the regulatory guide YVL B.1 (STUK 2013a). Swedish requirements are given in the codes SSMFS 2008:1 (SSM 2009a) and SSMFS 2008:17 (SSM 2009b).

Generally, defence-in-depth principle is an overall design principle to be followed and it is defined similarly to e.g. IAEA and WENRA. In addition, many detailed requirements are given how the principle shall be applied in the design of specific levels of DiD as well as in the demonstration of the safety. In this respect, Finnish and Swedish requirements are written

quite differently, even though the underlying intention may be same. As an example, Finnish requirements specify in detail failure criteria for various initiating event categories. Swedish requirements specify in detail criteria for manual actions.

4.3 Basic principles of defence-in-depth

4.3.1 Multiple barriers

Defence-in-depth is based on an accident model consisting initiating events and following event sequence where depending on the success of failure of barriers the end state is an accident or a safe state. Initiating events can be called threats or hazards. Barriers can be called protective layers or safety functions or safety systems, and they should not only be understood as physical hinders. Accident is a negative outcome from the point of view human beings, environment or economy of an enterprise. Figure 1 illustrates a simple defence-in-depth solution consisting of three barriers.

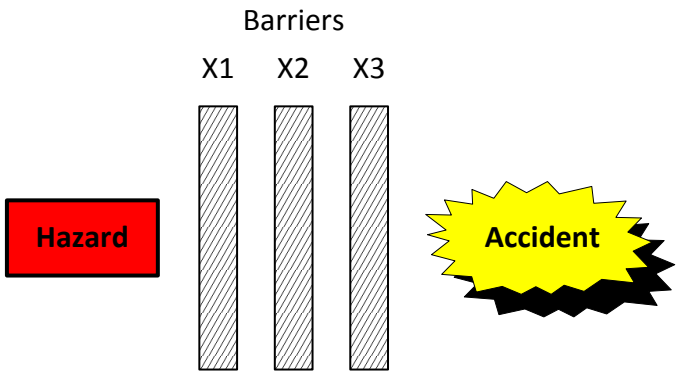


Figure 1. Defence-in-depth as a design of multiple barriers.

4.3.2 Redundancy, diversity and physical separation

Redundancy is duplication or generally multiplication of components of a system to increase the reliability of the system. Redundancy is especially effective if redundant components do not have dependencies, such as common support system, common maintenance or common environment. Dependencies can be avoided by introducing technological diversity between the components, by physical separation and by functional isolation (no common support systems).

The redundancy principle is applied in defence-in-depth at two dimensions. To have several barriers is a kind of overall redundancy principle, and at that dimension the diversity principle is followed as much as reasonably possible. On the other hand, redundancy principle can be applied for each barrier to increase its reliability. At the barrier level, diversity is not necessarily required.

Probabilistically, the effect of redundancy can be represented as follows. Let a protection system consists of n barriers $X_i, i = 1, \dots, n$, each of which has a failure probability p_i. Given that they form a serial system, the failure probability of the system is

$$P(S) = P(X_1 \cap \dots \cap X_n), \quad (1)$$

where $P(S)$ is the system failure probability and $P(X_1 \cap \dots \cap X_n)$ is the probability that all the barriers $X_1, \dots, X_n$ fail.

If the barriers are independent, the system failure probability is a product of the failure probabilities of the barriers

$$P(S) = P(X_1) \cdot \ldots \cdot P(X_n). \quad (2)$$

Evidently under the condition of independent barriers, the system failure probability decreases when barriers are added in the protection system. In reality it can be hard to design fully independent barriers, and then the following relationship will hold

$$P(S) = P(X_1 \cap \ldots \cap X_n) > P(X_1) \cdot \ldots \cdot P(X_n). \quad (3)$$

To have independent barriers is thus a desired property of defence-in-depth.

Diversity or diverse redundancy is a central principle to reduce the influence of common cause failures. It uses different technology, design, manufacture, software, etc. Depending on the type of safety function there are different possibilities to achieve diversity. Confinement type of purely passive physical structures can be diversified by building several different types of consecutive layers, which is a strategy for e.g. nuclear spent fuel final repository. Electromechanical safety systems depend on power supply and I&C, which considerably limits the degree of diversity that can be achieved. One strategy could be to combine electromechanical safety system barrier with some type of passive system. Otherwise we may not be able to claim full diversity.

Physical separation principle comes partly from the avoidance of hazards that can destroy several redundancies at the same time. Fire, flooding due leaking fluid systems and missiles from breaking structures are examples such hazards. Physical separation is especially meaningful principle for barriers which are electromechanical systems.

There are several strategies how to implement physical separation. Figure 2 shows three basic alternatives. In the first case barrier systems at different levels of defence-in-depth are separated from each other. For instance, it is a common requirement that non-safety (operational systems) are physically separated from safety systems (to the extent reasonably possible). In the second case, redundancies within each barrier level are separated but different barriers are not separated. The logic is that even if a hazard can eliminate one redundancy at all levels of defence-in-depth, some defence-in-depth and diversity still remains. In the third case both redundancies and barrier levels are separated from each other. This is most effective way of separation but it is also most expensive and can be practically difficult to build.

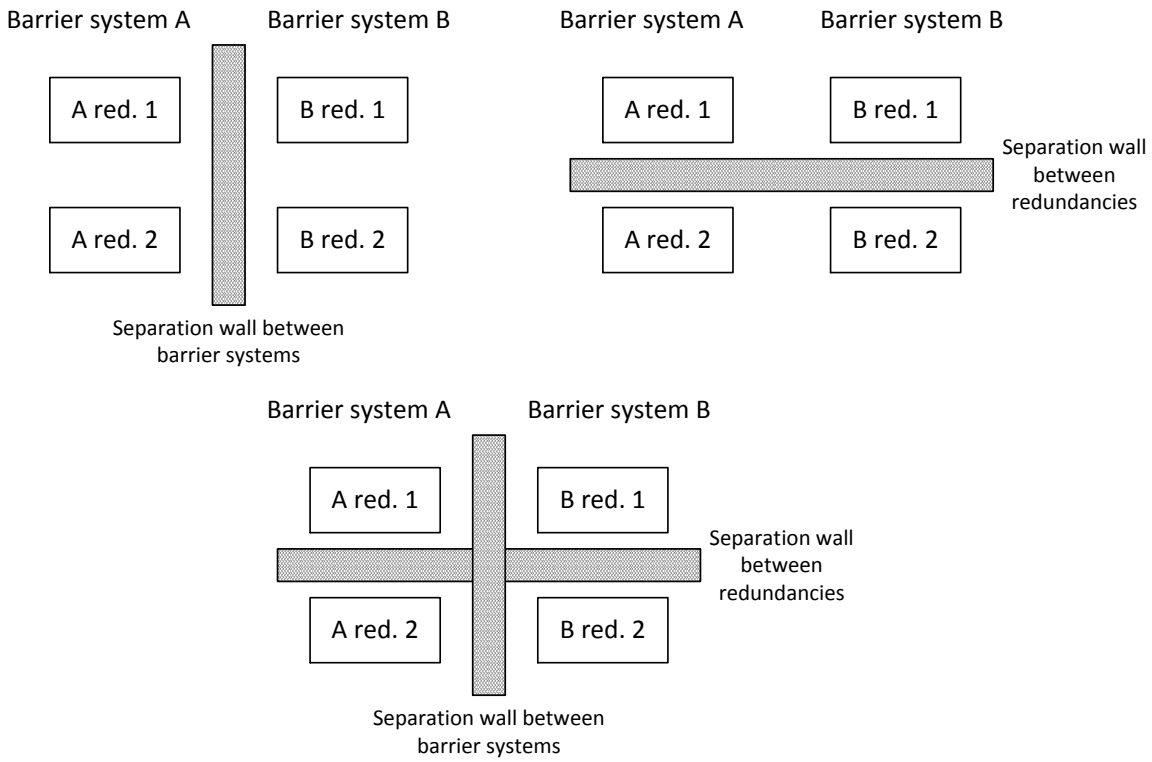


Figure 2. Implementation strategies for physical separation between barrier systems and their redundant subsystems.

The second alternative is common at modern nuclear power plants so that redundancies are located in physically separated compartments around the plant, but one compartment can include components from several DiD-levels. Typically it means a four-redundant design that safety important system located outside of the containment are distributed into four compartments, and the two-redundant systems are distributed between pairs of divisions AB and CD (Figure 3). Physical separation between four compartments can be well applied for major components and power supply, but cannot be fully achieved for all details. Inside the containment, physical separation is also followed, but it cannot be as strict as outside of the containment since containment is one atmospheric volume.

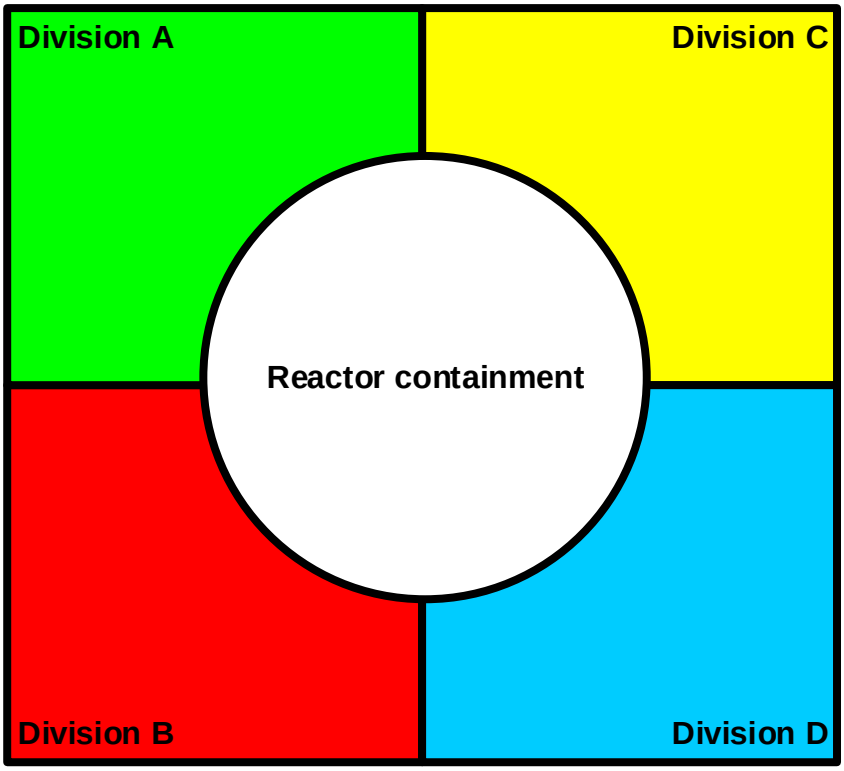


Figure 3. Physical separation of a four-redundant design into four compartments around the reactor containment.

4.3.3 Principle of successive barriers and reducing consequences

In defence-in-depth each barrier is effective in a certain order, i.e., they are successively challenged by the threat. Principle of reducing consequences is associated with the usual property of defence-in-depth that each barrier has different functional meaning. This principle is related to the requirement to have a barrier with respect to each consequence category.

Figure 3 presents an event tree, which illustrates the relationships between levels of PSA and levels of defence-in-depth. It also shows the principle of successive barriers and reducing consequences at NPP. Functional meaning of each DiD level are given in the headings of the event tree branches.

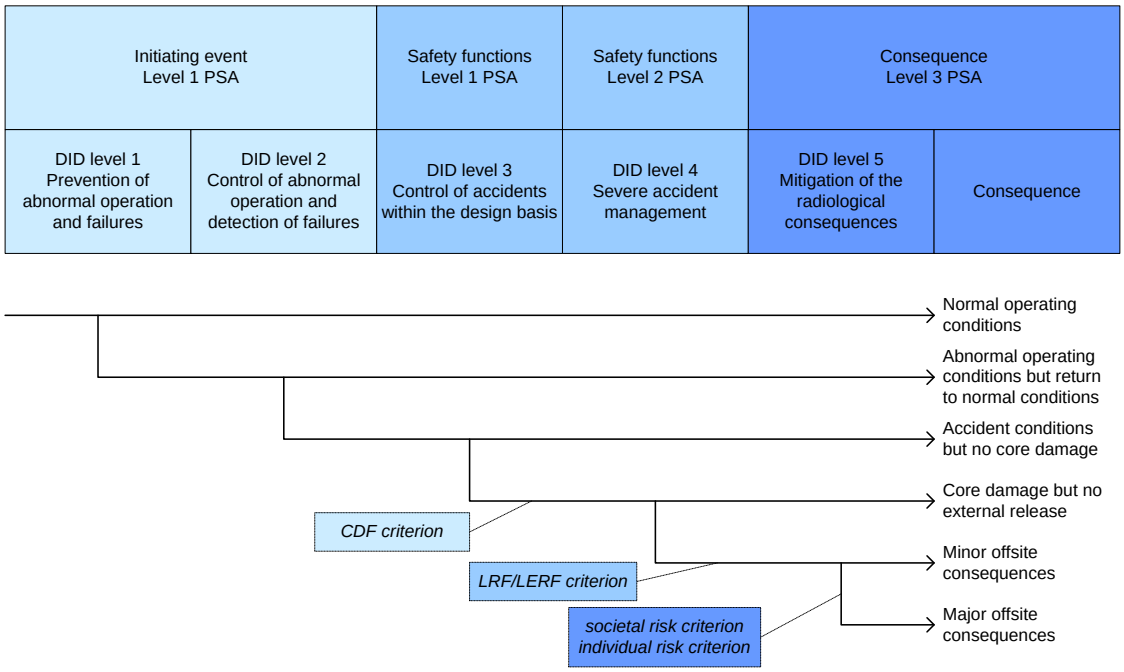


Figure 4. Levels of defence-in-depth, PSA and risk criteria.

Probabilistically, numerical criteria can be defined for various consequence categories, e.g. core damage frequency criterion and large release frequency criterion. Defence-in-depth can thus be seen as an implementation to achieve safety goals.

Safety goals and related risk acceptance criteria are overall qualitative and numerical targets given by the society or regulation for safety-critical installations. Qualitative target can be given like "Individual members of the public should be provided a level of protection from the consequences of nuclear power plant operation such that individuals bear no significant additional risk to life and health" (U.S.NRC 1986). Numerical target can be, e.g., individual fatality risk shall be less than 10^{-5} per year (Trbojevic 2005).

4.3.4 Accident prevention and mitigation

Principle of accident prevention and mitigation is a developed form of the principle of reducing consequences. It is based on an accident model, where we distinguish between events and barriers before an accident and events and barriers after an accident. If the accident prevention succeeds, the event is only an incident (near-miss). Otherwise an accident happens, and the effectiveness of accident mitigation barrier determines the level of consequences. Figure 4 illustrates an event tree for such a defence-in-depth strategy.

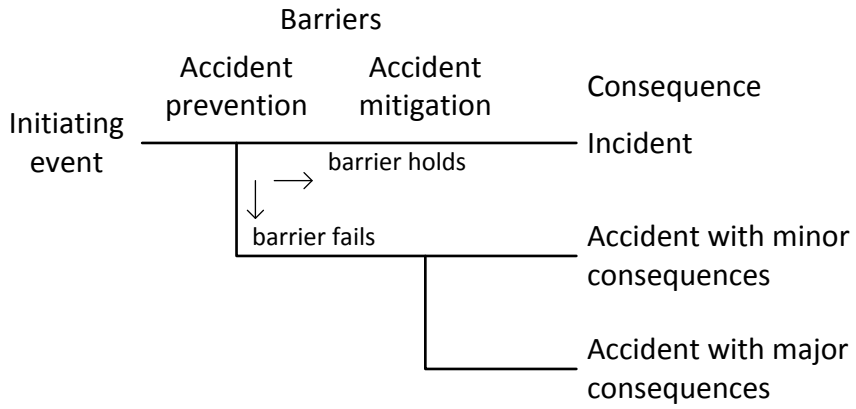


Figure 5. Event tree for an accident prevention and mitigation system.

Meaning of accident resp. incident is application specific and there can be ambiguity in the terminology. In the nuclear power plant safety nomenclature, an event is an accident if it involves a radioactive release (IAEA 2014).

The defence-in-depth principle requires that there are both accident prevention and mitigation systems, and that there is sufficient independence between them. Accident prevention and mitigation systems may have different technological and reliability requirements.

4.3.5 Classification of barriers

Defence-in-depth is more than having multiple barriers even if a redundancy is one of the central principles. In defence-in-depth the barriers have usually different functional role, i.e., we distinguish between accident prevention and accident mitigation barriers. In nuclear power plants five levels of barriers are considered (IAEA 1996).

Besides distinguishing between accident prevention and mitigation barriers, the barriers can be classified according to the implementation or activation principle such as degree or levels of passivity or degree of automation. In many industrial applications, requirements are set for the implementation principle of different barriers. For some places, e.g., a clearly physical, passive barrier is required. For other places, automated protection system is required. There are also design rules when and when not manual intervention can be or must be counted on.

Degree of passivity depends on the need of a system or component to have any external input to operate (IAEA 1991). If a component is not passive, it is necessarily an active one. The concept of passivity can be considered in terms of several categories. A system can have passive and active characteristics at different times. For example the active opening of a valve initiates subsequent passive operation by natural convection.

4.3.6 Safety classification

Safety classification of system, structures and components is means to manage the quality requirements of the items. In nuclear field, there is a national variation in the safety classification. For example, the International Electrotechnical Commission categorization (IEC 2009) defines three safety categories A, B and C, where the American standards of Institute of Electrical and Electronics Engineers uses classification process that only distinguishes between safety and non-safety systems (IEEE 2003). The IAEA has generally adopted a distinction between safety systems, safety related systems and non-safety systems (IAEA 2002).

The relationship between defence-in-depth and safety classification is immediate in the sense that each barrier is assigned with a safety class, and all items belonging to one barrier have the same safety class (at least in the nuclear field applications). It follows that all items within one barrier have the same requirements for design, qualification, regulatory review and QA procedures during all life cycle phases. Barriers can belong to different safety class, and in fact this is considered beneficial both from the diversity point of view and from the optimal resource allocation point of view (graded QA).

4.3.7 Weaknesses and limitations

Defence-in-depth is a powerful design against hazards as long as barriers are intact. In reality, the barriers have defects, which makes the barrier system like a Swiss cheese having many holes. According to the Swiss cheese model, an accident happens when the holes of barriers holes in many layers momentarily line up to permit a trajectory of accident opportunity — bringing hazards into damaging contact with victims (Reason 2000).

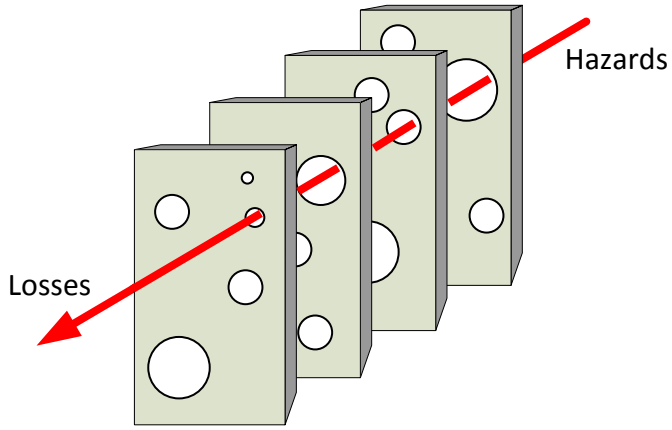


Figure 6. Swiss cheese model.

Following the Swiss cheese model we can see that there are two ways of minimizing the accident probability: 1) minimizing the likelihood of holes and hazards and 2) minimizing the dependencies between hazards and holes being in the same trajectory, i.e., minimizing the dependencies between them.

The first target includes usual design and operational challenge for socio-technological systems, i.e., how to avoid active failures and latent conditions making the system unreliable (Reason 2000). For both types of failures any measures to minimize the failure probability or rate is relevant and there are number of activities which need to be accounted during the life cycle of the system (specification, design, manufacturing, commissioning, operation, maintenance).

To avoid latent conditions, the question is how to detect them. Most problematic cases are design errors and erroneous or unintended system modifications. In those cases, the normal, planned methods to detect the availability of the barrier may be by-passed.

Regarding active failures, the system should be designed to tolerate single failures. This can be often achieved by introducing a transition to safe-state, building redundancy in the system (switch-over is made in case of failure), having an alarm system for operators including

procedures to cope with the situation in the required time frame, and so on. Usually active failures are not considered as dangerous as latent conditions, but they may also include deceptive features, if the organisation does not learn from repeating failures.

The independence target with a multi-barrier systems implies a complexity challenge for the design since it is hard to avoid different types of dependences between the barriers. If we are required to have more than two barriers like in a nuclear power plant, we begin to be limited by space, physical separation means, material choices, technology providers, support system choices, maintenance and testing couplings, etc. The more barriers are required, it does not only lead to complex and expensive design but there are practical limitations how independent barriers can be.

Complexity of defence-in-depth strategy also leads to the question of balancing between safety/security and availability. Defence-in-depth implies thus a cost factor. Without a proper risk assessment cost-ineffective barriers may be implemented.

Defence-in-depth may be also criticized for the fact that increasing protection does not always increase safety. Besnard and Hollnagel (2012) explain that increasing protection affects the perceived risk exposure, which affects the behaviour so that increase performance efficiency while keeping the perceived level of risk constant.

A related feature of a multi-layer protection system is that it may be difficult to notice a local violation of a barrier since it does not necessarily have any immediate, visible effect. In systems designed according to the defence-in-depth strategy, the defences are likely to degenerate systematically through time, when pressure toward cost-effectiveness is dominating (Rasmussen 1997).

Effective preventive barriers may also lead to less experience feedback from management of incidents and accidents. From learning and alertness point of view it might be better that people get occasionally experience how to handle dangerous situations.

Defence-in-depth principle can be characterized as a deterministic and qualitative principle, which explicitly does not take into account probabilistic reasoning. It is a strict requirement to have each barrier. Weakness of one barrier cannot be compensated by making another barrier stronger. A nuclear power plant must have a reactor containment building regardless of the probability for a core damage accident. The reason for this is the underlying uncertainty that even a strongest barrier may fail in unexceptional conditions. It is hard to prove that one barrier is extremely reliable. However, from risk decision making point of view it may be non-optimal require a multi-barrier system than a single strong barrier.

4.4 Defence-in-depth and I&C

IAEA (1999) defines three roles for I&C systems of an NPP:

1. They are the ‘eyes and ears’ of the operator. If properly planned, designed, constructed and maintained, they provide accurate and appropriate information and permit judicious action during both normal and abnormal operation. They are therefore, with the human operator, vital for the safe and efficient operation of the plant.
2. Under normal operating conditions they provide automatic control, both of the main plant and of many ancillary systems. This allows the operator time to observe plant

behaviour and monitor what is happening so that the right corrective action can be taken quickly, if required.

3. The I&C safety systems protect the plant from the consequences of any mistakes which the operator or the automatic control system may make. Under abnormal conditions they provide rapid automatic action to protect both the plant and the environment.

All of those three roles can be associated with defence-in-depth principles. For the design of safety I&C, the defence-in-depth principle involves an obvious challenge since it requires consideration of multiple constraints. Implementation of basic principles of defence-in-depth in nuclear I&C is discussed below.

Design of I&C is based on the general concept of having five defence-in-depth levels, multiple barriers responsible for accident prevention or mitigation. There are different I&C systems performing functions at different levels of DiD, although in practice there is some overlap.

Redundancy is implemented in most I&C systems, e.g., by having multiple sensors, processors and buses. Redundancy requirements depend on the level of DiD, and follows the analysis of initiating events (plant design basis and associated postulated initiating events).

Diversity is required between different levels of DiD to avoid CCF:s. Normal and safety I&C should be implemented differently. In addition safety functions I&C needs to be implemented with diverse technologies (preventive, protection, diverse protection functions). Diversity requirements lead to need of having several platforms, possible to implement hardwired back-ups for software based systems, use of diverse process measurements to actuate safety functions and considerations regarding manual back-up for automation functions. Full independence between levels of DiD cannot be achieved, which leads to a judgement on the sufficiency of the degree of diversity.

Physical separation is implemented between divisions so that hazards should lead only to loss of one redundancy. Physical separation affects, e.g., cable routing, cabinet placement and placement of main control room and reserve control room. Physical separation may be required also between different levels of DiD.

Depending on the function, I&C can be performing the function or it can be just monitoring it. Manual interventions (operator controls) must be considered both w.r.t possibility to recover I&C errors and elimination of adverse effect of human errors. Conditions for manual interventions depend on the plant operating state (POS), i.e., role of manual intervention can be different during power operation from low power and shutdown conditions.

Safety classification follows the defence-in-depth levels and associated functions, e.g., DiD level 3 functions are performed by cat. A systems (IEC 2009). Each function and system belongs to certain safety class, which affects e.g. the qualification requirements and V&V principles. Due to practical overlap between DiD levels, there may be challenges in the classification.

Risk criteria imply reliability requirements for I&C. System specific reliability requirements can also come from the safety class, e.g. SIL classes of IEC 61508 (IEC 2010a).

It is thus evident that the design task is complex and there is a need to make compromises between different design objectives. Earlier generation nuclear power plants it was sufficient to basically distinguish between normal operation functions and safety functions, and to have two categories of systems. Today, this is far too simple approach, and the functional thinking must be widened to all levels of DiD. One of the problems here is that in reality safety functions and different levels of DiD have some overlap. As discussed in (EPRI 2014), examples of interconnections can include:

- Human-system-interface (HSI) for the control systems (level 1) and the limitation systems (level 2) may be combined
- There may be one-way connection from the protection system to the normal control system HSI, allowing the HSIs to integrate information across all three levels of DiD.
- Sensors and actuators may be shared by level 3a and 3b, but signal branching circuitry priority logic is provided to eliminate failure propagation between I&C systems.

Table 1 discusses specific I&C issues for each defence-in-depth level.

Table 1. Issues related to design of I&C for different levels of DiD.

DiD level	Notes
1	• Normal operation I&C should be designed to achieve good availability of the plant (low frequency of initiating events), and to eliminate the propagation of failures to safety I&C. • Operational I&C functions can have safety-related functions. Systems/functions are not necessarily only non-safety classified • Potential common cause initiator (CCI). It should be demonstrated that normal I&C cannot interfere safety I&C • Could operational I&C be credited in safety analysis (usually not in deterministic safety assessment but maybe in PSA)?
2	• Includes a number of preventive functions to avoid operational transients. Also important to the availability of the plant and has functional dependences with level 1. • Functions are safety classified but not in the highest safety category. It has joint objectives with DiD level 3 and can have functional dependencies • Plays usually minor role in PSA (part of the initiating event frequency) • Redundancy (1-o-o-2 success criterion) may be required
3	• Includes Reactor trip and ESFAS (Engineered safety features actuation system) and belongs to the highest safety class. • Nowadays four-redundant systems • In addition diversity may be required (DiD level 3b, design extension cases)
4	• From I&C point of view many functions are rather simple both including passive features (no or little I&C), manual functions (relaying on monitoring of status of the plant) • Safety classification varies • DiD level 3 functions/systems play significant role at this level, too
5	• Includes alarming, monitoring and communication functions • May be considered quite separate from the other systems

4.5 Defence-in-depth and PSA

4.5.1 Probabilistic use of PSA

PSA provides many type risk metrics for the assessment of individual levels of DiD as well as end state. Holmberg & Nirmark (2008) and Hellström (2015) have studied principles and concepts to assess DiD in several manners. A probabilistic verification of defence-in-depth is done by assessing frequencies of different end states, as illustrated in Figure 3:

- level 1 PSA and core damage frequency (CDF) verifies DiD levels 1–3
- level 2 PSA and large release frequency (LRF) verifies DiD levels 1–4
- level 3 PSA and societal risk metric verifies DiD levels 1–5.

There is a variation how numerical risk targets are set (Bengtsson et al 2011). For instance, in UK and the Netherlands it is set for the whole defence-in-depth system. In most other countries, societal or individual risk criteria are not applied, but the core damage frequency

(CDF) and large release frequency/ large early release frequency (LRF/LERF). These may be interpreted as subsidiary criteria for risk of offsite consequences in countries where level 3 PSA is not required. Subsidiary criteria are preferred due to the uncertainties in the risk assessment of offsite consequences (e.g. societal and individual risk) and that they explicitly put focus on defence-in-depth, in particular attention is paid to the accident prevention and mitigation.

There is rather good consensus on the meaning of CDF. For level 2 PSA, apparently more than one release category is needed to cover the spectrum of various possible release types. This is an open issue. For level 3 PSA, lessons learnt from Fukushima is that the fatality risk (individual or group) is far too limited risk metric. Societal impact including environmental impact should be accounted for, too. Definition of appropriate risk metric for level 3 PSA is also an open issue. Besides, level 3 PSA is not yet required in many countries, and level 2 PSA risk metrics are used as surrogates (Caldwell et al. 2014).

Risk importance measures can be used to study the importance of various barriers and safety functions. While there are appealing features to do so, it is difficult to define "numerical risk criteria" for risk importances. For instance, suppose we have a requirement on balanced design, e.g., meaning that no initiating event should dominate the result. However, the assessment of balanced design depends on the way initiating events are grouped. The role of risk importances is therefore more to provide qualitative risk insights than to be used to show compliance with strict design criteria. There is however need to provide further guidance on the use of probabilistic criteria for individual DiD levels.

Appendix A demonstrates how the DIGREL example PSA (Authén et al. 2015) has been used to assess risk importances with a level 1 PSA. PSA provides a variety of metrics such as fractional contribution of basic events or group of basic events, risk increase factors of basic events, and conditional core damage probability for initiating events. Ahonen (2011) applied Birnbaum importance measure to assess the degree of diversity between systems responsible for common safety function.

4.5.2 Deterministic use of PSA

Defence-in-depth is basically linked with the deterministic safety assessment, including the categorisation of initiating events and associated failure criteria. The logic model of PSA can be used to analyse the failure criteria. This "deterministic" application of PSA is common to safe shutdown analyses, e.g., related to fire hazards. In this application, each fire scenario together with postulated failure criterion is examined and based on minimal cut sets it is judged whether safe shutdown can be reached (Cederhorn & Frisk 2014). The method could be analogously applied to failure tolerance analyses as required by STUK (2013a) in the Guide YVL B.1. Even though today's PSA:s are quite complete, the application of PSA for failure tolerance analyses requires development of features in the model (more details, considerations on how to handle operator recoveries, etc.).

In appendix A, a test with the DIGREL example model was done with respect to the so called N+2 failure criterion in case of anticipated operational occurrences and design basis accidents (DBC2–4). (N+2) failure criterion means that "*it must be possible to perform a safety function even if any single component designed for the function fails and any other component or part of a redundant system – or a component of an auxiliary system necessary for its operation – is simultaneously out of operation due to repair or maintenance.*" It must be, e.g., shown that "*it shall be possible to accomplish decay heat removal from the reactor and containment by one*

or several systems that jointly meet the (N+2) failure criterion…" This assessment can be made excluding common cause failures from the model, and in this example model, this requirement is fulfilled. There are no such minimal cut sets.

It was also analysed whether the failure criterion associated with the Design extension category A (DEC-A) is fulfilled. DEC-A refers to an accident where an anticipated operational occurrence or class 1 postulated accident involves a common cause failure in a system required to execute a safety function. A diverse N+1 system is needed to reach a safe state. In the example case, this means that the DBC2 events (anticipated operational occurrence) combined with a CCF must be examined. A large number of minimal cut sets can be found not fulfilling the criteria, but then it depends on how the criterion is actually interpreted for cases like:

- which software CCF should be counted as a common cause failure, e.g., system software CCF or CCF between nearly identical application software modules?
- can a system requiring an operator action be (N+1) failure tolerant?
- shall the (N+1) failure criterion be applied to structures like the demineralized water tank?
- can recovery be accounted for the loss-of-offsite power (LOOP), i.e., could LOOP be classified into different categories depending on the duration of the LOOP (only short time LOOPs would be DBC2)?

The above examples of the deterministic use of PSA is very limited but give an idea of capabilities. The topic will be further discussed in the next chapter regarding analysis of spurious actuations.

4.6 Conclusions

It can be expected that deterministic DiD principles cannot fully be satisfied or that the deterministic demonstration is limited and may be biased by postulation criteria. PSA is therefore needed in a probabilistic manner to support safety demonstration. PSA mostly focuses on modelling DiD levels 3 and 4, while levels 1 and 2 are often implicitly covered by the initiating event analysis. A more detailed analysis of systems and functions involved in the DiD levels 1 and 2 could provide useful information both for plant safety and availability perspectives.

PSA can also be used in a "deterministic" manner, by utilizing the logic model to analyse the failure criteria. This application of PSA is common to safe shutdown analyses, e.g., related to fire hazards. In this application, the properties of the minimal cut sets are analysed with respect to each initiating event category. The method could analogously be applied to failure tolerance analyses as required by STUK. Even though today's PSA:s are quite complete, the application of PSA for failure tolerance analyses requires development of features in the model. More detailed analysis of I&C at DiD levels 1–3 may be needed.

A crucial issue for the results is the assessment of the independence between barriers (diversity), which can be associated with the postulation and quantification of common cause failures. Today's PSA may ignore inter-system CCFs. For hardware, a thorough analysis of common subcomponents between systems should be required. This can theoretically mean very large CCF groups which are impractical to model as such. It would be anyway important to identify such CCF groups and then make case-by-case judgement what should be included

in the model. Software CCF:s are discussed in chapter 6.8 of this report. Diversity assessment is also addressed.

DiD requirements can lead to complex design solutions. It is an open questions how the impact of complexity should be accounted when comparing designs. Complexity is an indirect factor affecting the system reliability, similar to organizational factors. It might however be assumed that higher complexity correlated with higher unreliability due to more complicating operation and maintenance. There is no unique metric for complexity and there is little statistical data to estimate the correlation. Impact of complexity is also discussed in chapter 0 from the software reliability point of view.

5. Analysis of spurious actuation

5.1 Definition for spurious actuation

Spurious actuation is a functional failure mode when a component performs a function without a real demand. As an example, a transmitter erroneously sends an actuation signal opposite to the state of the process. Spurious actuation is also called inadvertent operation.

Meaning of "spurious", "inadvertent" and "demand" depend on the analysis perspective. Transmitter may send the actuation signal due to failed sensor, and in this sense it works correctly. However, from the process control point of view, the function of the transmitter is spurious (inadvertent).

In PSA, spurious actuation is usually a complementary failure mode to "failure to actuate when demanded". In the fault tree analysis and FMEA, both failure modes must be tentatively considered, and causes for spurious actuation can be analysed top-down from the actuators down to support systems, I&C systems and power systems. Spurious actuation can be omitted if the effect of the actuation has no negative impact on system safety, e.g., reactor scram may be considered a fail-safe case.

Spurious actuations are of special interest for I&C systems and for fire initiating events (hot shorts). In the area of human reliability, errors of commission are also kind of spurious actuations.

Spurious actuations are by nature more complex to analyse than "failure to actuate". If a system/component has several functions spurious actuation may mean several things. It is not self-evident when spurious actuation should be considered, and if considered, how failure rate can be estimated. In addition, it is questionable whether common cause failure assumption should be applied to spurious actuation (simultaneous spurious actuation of several components).

A special category of spurious actuation are failures of safety related systems causing an initiating event and at the same time deteriorating the performance of safety functions, i.e., so called common cause initiators. There is a need to develop a method to analyse common cause initiators related to safety I&C.

5.2 Regulatory requirements and guidelines

The analysis of spurious actuations is required in various ways in the international guidelines and regulatory requirements. In the IAEA guide (IAEA 2012), it is required to take into account spurious operation and unsafe failure modes when considering the reliability of items important to safety (Requirement 23). Requirement 25 defines the single failure criterion, and it includes also "Spurious action … to be one mode of failure when applying the concept to a safety group or safety system."

WENRA (2013) guidelines includes a design requirement that systems, structures and components (SSC) important to safety, allocated to different levels of DiD shall be functionally isolated. This includes prevention from the propagation of failure or spurious signals from one system to another.

In the STUK (2013a) regulatory Guide YVL B.1, there are the following requirements:

- 353. A common cause failure analysis shall be drawn up for initiating events in design basis categories DBC 2 and DBC 3. For the common cause failure analysis, the implementation of the safety functions shall be presented for each initiating event in a manner that indicates the use of the systems implementing the principles of diversity and redundancy. The common cause failure analysis shall address one safety function, or part of it, at a time with due regard to the systems implementing the function and the related auxiliary systems. The analysis shall address the common cause failures of all components whose common cause failures or *spurious actuation* may affect the performance of the safety function. The common cause failure analysis shall consider the initiating event, interdependencies between initiating events as well as common cause failures between components sharing a similar property, i.e. components that are similar or contain a significant number of similar parts.
- 432. No single anticipated failure or *spurious action* of an active component taking place during normal plant operation shall lead to a situation requiring intervention by systems designed to manage postulated accidents.

The Guide YVL B.3 (STUK 2013b) on deterministic safety analyses for a nuclear power plant has a requirement that "304. The inadvertent actuation of every system accomplishing a safety function shall be addressed as an initiating event."

Swedish SSM regulatory guides do not address spurious actuations.

5.3 Generic failure modes taxonomy

In this section, a general definition is given for the concept "spurious actuation". The definition is based on von Wright's theory on concept of action (von Wright 1968). This logic theory enables us to claim that the space of considered failure modes is exhaustive and exclusive, and that we have a logically sound meaning for "spurious actuation".

von Wright's theory is based on the concept of change being defined as a temporal succession of two states. von Wright's elementary action is an action effecting a *change* or a *not-change* in the physical world. An *elementary change* is a succession in time of two contradictorily opposed states in the physical world. In an *elementary not-change* a given state remains unchanged.

There are four possible changes $\neg pT\neg p, \neg pTp, pT\neg p, pTp$, where $\neg p$ and p denote the two possible states of the world and T is a temporal operator. An action has also a counterfactual aspect because the change would not occur unless the action was done. From this perspective actions can be categorised into interventions and omissions. In an intervention, an agent affects the state of the world so that opposite state does not happen. In an omission, an agent lets a change or not-change happen. This leads to eight possible elementary action types (Table 2).

Table 2. Elementary action types.

Interventions		Omissions	
Schema	**Action type**	**Schema**	**Action type**
$\neg pTpI\neg p$	Produce p	$\neg pTpIp$	Let p happen
$pTpI\neg p$	Maintain p	$pTpIp$	Let p remain
$pT\neg pIp$	Destroy p	$pT\neg pI\neg p$	Let p disappear
$\neg pT\neg pIp$	Suppress p	$pT\neg pI\neg p$	Let p remain absent

p, $\neg p$ = complementary states of the world
T = temporal operator
I = agent operator

This taxonomy of elementary action types can be translated into generic failure modes by interpreting $\neg p$ and p as two possible states of a component, e.g., pump is stand-by or running, valve is closed or open, etc. A "failure" is then an agent making an *intervention*. An *omission* means an "absence of failure". Subsequently we can define that elementary actions "Produce" and "Destroy" are "spurious failure modes" while "Maintain" and "Suppress" are "failures of actuation when demanded".

Table 3 gives examples of relationships between the elementary actions and failure modes of typical components. This demonstrates the completeness of the taxonomy for components whose state space is bimodal. For multi-state components, the space of states must be divided into two exclusive and exhaustive sets.

Table 3. Von Wright's elementary actions and generic failure modes.

Elementary action	**Component type**	**State before**	**State after**	**Failure mode**
Productive	Pump, fan, diesel generator	Stand-by (off)	Running	Spurious start to function
	Valve	Close/open	Changed position	Spurious change of state
	Processor, sensor, I&C module	State 0 (no actuation signal)	Actuation signal	Spurious actuation signal
Destructive	Pump, fan, diesel generator	Running	Off	Spurious stop
	Valve	Close/open	Changed position	Spurious change of state
	Processor, sensor, I&C module	State 1 (actuation ON)	State 0	Spurious loss of actuation signal
Suppressive	Pump, fan, diesel generator	Stand-by (off)	Off	Failure to start when demanded
	Valve	Close/open	Position remained	Failure to change position
	Processor, sensor, I&C module	State 0 (actuation OFF)	State 0	Failure to provide actuation signal
Sustaining	Pump, fan, diesel generator	Running	Running	Does not stop to function when demanded
	Valve	Close/open	Position remained	Failure to change position
	Processor, sensor, I&C module	State 1 (actuation ON)	State 1	Failure to reset the actuation

Appendix A contains an analysis of risk contribution from spurious actuation in the DIGREL example model. It should be noted that "spurious actuations" are not specifically addressed in this analysis, but they are part of the normal analysis of various possible failure modes.

It is important to ensure the completeness of the failure modes considered and that fault tolerance principles of the I&C are properly accounted for. Another important judgement is how the failure detection is determined for various failure modes, in particular related to the hardware modules which are the smallest entities considered in the analysis. For software modules, the taxonomy of dividing failures into fatal and resp. non-fatal failures resolves the issue of failure detection. Spurious actuations are generally related to the failures detected by on-line monitoring followed by a "fail-safe" actuation or they are related to wrong input signals, which can be classified as self-revealing failures. Latent hardware module failures will be detected by a failure per demand or test.

The example evaluation indicated a large contribution from spurious reactor protection system (RPS) signals. The spurious actuations are mainly caused by the fail-safe design that may occur (default value 1), while spurious behaviour of hardware and software causing spurious actuations is insignificant.

5.4 Suggested analysis approach for spurious actuations

Certain spurious actuations are already well covered by the ordinary systems reliability analyses (failure modes and effects analysis and fault tree analysis) and should be in the PSA model, when done at appropriate level of details and when considering fault tolerant features properly. The analysis needs to be done in several steps as demonstrated with the DIGREL example (actuators, signals, I&C units, I&C modules). Both impacts of single failures and CCF must be considered. The usual practice followed in PSA is thus suggested.

There are a number of challenging issues, which require further consideration, e.g.:

- How to identify possible common cause initiators comprehensively?
- To what extent CCF causing a spurious actuation should be considered? It can make difference whether multiple failures occur simultaneously or if there is a time difference. Simultaneous CCF is much more unlikely than CCF within a longer time window.
- Likelihood of a fire caused hot short, i.e., can a power or I&C cable fire cause a spurious actuation, how to analyse multiple hot shorts? This has been discussed a lot in the context of fire PSA methods, see e.g. (U.S.NRC 2005)
- Spurious actuations caused by human errors of commission. This has been discussed a lot in the context of human reliability analysis, see e.g. (U.S.NRC 2012)

In order to keep an analysis manageable both top-down and bottom-up approaches are needed. By top-down approach a screening of irrelevant system failures or I&C function failures can be performed

- what is the effect of a spurious I&C function?
- what could cause a prevention of an I&C function?

Bottom-up approach can be applied to the critical functions. As an example the reactor scram can be screened out since a spurious scram may be considered a safe failure. For core cooling and residual heat removal, the analysis needs to be broken down into smaller functional entities to make the judgements.

6. Software reliability

6.1 Software reliability in nuclear PSA

In the context of PSA for NPPs, there is an on-going discussion on how to treat software reliability in the quantification of reliability of systems important to safety. It is mostly agreed that software could and should be treated probabilistically (Dahll et al. 2007, Chu et al. 2009) but the question is to agree on a feasible approach.

Software reliability estimation methods described in academic literature are not applied in real industrial PSAs for NPPs. Software failures are either omitted in PSA or modelled in a very simple way as common cause failures related to the application software (AS) of operating system (platform). It is difficult to find any basis for the numbers used except the reference to a standard statement that 1E-4 per demand is a lower limit to reliability claims, which limit is then categorically used as a screening value for software CCF.

The software reliability estimation approach described in this report is a continuation of the approach previously developed and described in (NKS-341).

6.2 Software quantification method

The software quantification method is based on the defined cases from (NKS-341) and (OECD 2015). The following software modules are considered:

- System software (SyS).
- Elementary functions (EFs).
- APU functional requirements specification modules (APU-FRS).
- APU application software modules (APU-AS).
- Proprietary software in I&C.
- VU functional requirements specification modules (VU-FRS).
- VU application software modules (VU-AS).
- Data communication software (DCS).
- Data link configuration (DLC).

Depending on the location of the software fault, failure effect and system architecture, one or more units in one or more subsystems can be impacted. The report *Failure modes taxonomy for reliability assessment of digital I&C systems for PRA* (OECD 2015) presents a list of maximum failure extents of a postulated event. Because it would be impractical to take all of them into consideration in the PSA model, the most relevant can be identified. The software faults and effects presented in Table 4 are considered further in this report.

Table 4. Generic software failure modes and effects.

Effects	Definition of effects	Software fault location					
		SyS	APU-FRS	APU-AS[1]	VU-FRS	VU-AS[1]	DCS
SYSTEM	Loss of complete system	case 1					case 1
1SS	Loss of one subsystem	case 2a	case 2a		case 2a	case 2a	case 2b
1APU-1SS	Loss of one group of redundant APU in one subsystem		case 3a	case 3a			
1VU-1SS	Loss of one group of redundant voters in one subsystem				case 3b	case 3b	
1AF-1SS	Loss of one function in all divisions of one subsystem		case 4a	case 4a	case 4b	case 4b	
1AF-1D-1SS	Loss of one function in one division of one subsystem		case 4c	case 4c			

Cases 1, 2a and 2b are system failure modes, representing fatal failures. Case 2a means loss of one subsystem (represented by a fatal failure of APUs and VUs) and case 2b means loss of communication within a subsystem. The difference between 2a and 2b is that in case of fatal failure in DCS or DLC (b), VUs run and can take safe fail states. In case (a), the whole subsystem stops running and also takes a safe state.

Cases 3 and 4 are application module failure modes and can be fatal or non-fatal failures. A fatal failure is a failure where the process stalls, which means that it will be possible for an external observer to know that the system has stopped (Case 3). A non-fatal failure does not stop the process but can yield an incorrect set of signals (Case 4).

The quantification method depends on the type of software module. System software (types 1 and 2 in Table 4) and application software modules (types 3 and 4 in Table 4) are considered relevant to model and quantify in PSA. The other SW modules could be ignored since their faults are covered by other cases.

The software failure modes are also intended to be defined in such a way that they allow the PSA model to be used for analysis of defence-in-depth analysis (see section 4). Thereby the PSA model and the defined software failure modes would be useable also in the safety verification of the software system.

6.3 System software (SyS) quantification

The failures of SyS should preferably be estimated for the system in question from operational history, since it is practically impossible and not meaningful to analyse system software more in detail (it is a “black box”).

Fatal failure of SyS is assumed to cause at least the failure of one subsystem (1SS). With sufficient data (even though it may be hard to find such data), this failure mode should be possible to estimate. The value calculated from operating experience represents thus the unavailability of one subsystem.

---

1 Note that the APU-AS and VU-AS software modules consider the elementary functions (function blocks) involved in the application functions implemented in the APU/VU.

For SyS type of failures, it can be assumed that the failures could happen regardless if the plant is in normal operation or during a transient. The run time environment and messages sent between units will still be performed with the same frequency. Failures of the SyS will be discovered, since they should lead to a fatal failure.

The SyS faults that shall be estimated for the PSA are following:

- SYSTEM-SyS fatal CCF (case 1)
- 1 SubSystem – 1SS Sys fatal CCF (case 2a)
- 1 SubSystem – 1SS-DCS fatal CCF (case 2b)

From the TXS[1] experience, following estimates of failure probabilities (per demand) have been done for TXS (NKS-341):

- P[System-SyS, case 1] ≈ 2E-9
- P[1 SubSystem, case 2a] ≈ 2E-6
- P[1 SubSystem, case 2b] ≈ 1E-5.

The failure probability is calculated based on an estimate of the failure rate and an exposure time. Fatal failures of the system software, as well as communication software, should be possible to estimate based on the complete operational experience, as discussed above. The failure rate estimates for the TXS software are based on the operational experience, and assuming at least one failure (should none have been observed). This is quantified using a one-stage Bayes model:

$$\lambda = \frac{2n+1}{2T}, \quad (4)$$

where n = number of failures, T = observation time

This yields an estimate on a failure rate, which is the multiplied with the assumed transient time for the PSA (assumed to be 24 hours in (NKS-341)) to get the failure probability during transient of the software.

6.4 Application software quantification method

6.4.1 General

Whilst the system failures can be estimated using operation experience, this is not the case for the application software. The approach for application software quantification therefore needs to be treated separately.

A fundamental question with regard to application software quantification is to what extent information from other software can be used to predict the behaviour of your specific software, as the different software may be doing vastly different tasks.

---

[1] TELEPERM® XS, product of AREVA

To take into account interdependencies in between different software the application software has to be split in to what is defined as application software modules.

6.4.2 Application software failure modes

The failure modes for application software can be divided into fatal and non-fatal failures (case 3 and case 4 in Table 4). The fatal failures in application software will affect all on-going processes as these will stop the ongoing processes and thereby also affect the other application software running on the processor.

The non-fatal failures will only affect the output of the current application software. The effect of a fatal and a non-fatal failure may have the same impact on the signal generated, this is simply a matter of configuration and definition of end-states.

As the fatal failures will affect all ongoing processes, these types of failures should be handled by the process and the error treatment. As the error treatment needs to be comprehensive in a reactor protection system, the fraction of failures leading to exceptions is expected to be small. Because of this the fatal failures' fraction of application software failures was estimated to be below 5% in (NKS-341).

6.4.2.1 Definition of application software module

In the quantification method for non-fatal failure modes the application software is suggested to be split into application software modules. An application software module (AS module) is a piece of software that is representing a specific functionality. The application software is executed and controlled by the system software (run time environment) during an operating cycle.

Each AS module usually corresponds to one individual function diagram group dedicated to a specific task. Depending on the specific case the application software can be represented by one or more AS modules.

An AS modules shall be defined so that one AS module is only used as a complete entity in the analysis. In Figure 7 an AS is presented. The AS module could be defined according to the red line, or as three separate modules according to the red dotted line. As the output from this software is through the same interface, there is no reason to split the AS module into more than one AS module, so a proper definition would be according to the red line in this example.

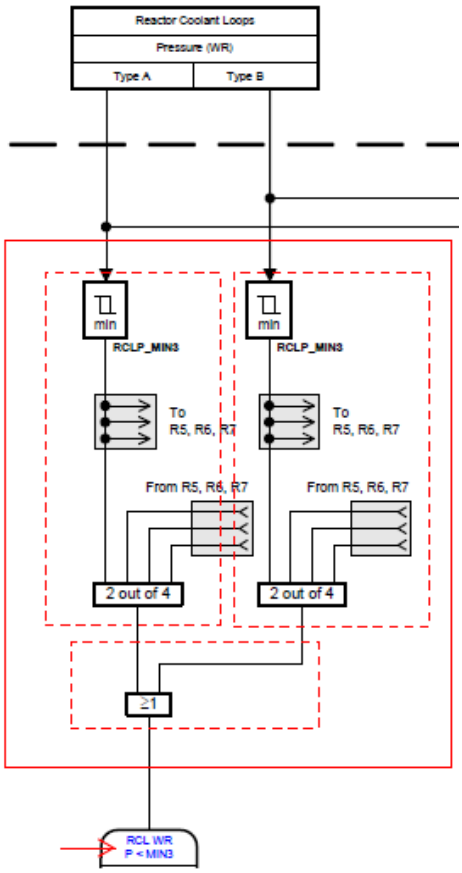


Figure 7. Illustration of the definition of application software modules.

An estimate of the number of application software modules that form an average application software function is in the range of 5−20.

6.4.3 Quantification method

There are different types of methods possible for approaching the quantification of application software. Previously in (NKS-341) a method based on failure detection mechanisms was outlined as the main alternative. This has been further evaluated and the suggested approach is in this report based on direct failure mode estimation.

The failure mode estimation approach directly estimates fatal failures and non-fatal failures, without a need for expert judgement on split fractions between these categories as in other studied approaches. The approach may also help in the classification of failures. The suggested approach is described in the following sections.

6.4.4 Estimation of fatal failure probability

Fatal failures in any application software should be treated by the error handling system to avoid a fatal failure that would affect other ongoing processes. The error handling has to handle some type of faults that could occur in the application software, but these potential faults should be recurring in all processes. Hence, it should be reasonable to claim that these failures can be considered to be of same type and therefore analysed together.

The situations where fatal failures could be observed are:

- Failures during operation
- Failures during demands

Optimally, the estimate should be based on failures during demands in addition to information from failures during operation. There is more operational experience during operation, but it could be claimed that this operational experience does not cover potential failures at demands. The amount of demands is however not easily estimated. Assuming that the error handling will be challenged during operation in the same way as during a demand, the addition of failure probability from demands will be negligible compared to the probability of failure during operation. This assumption, which is used in this analysis, should be further justified.

The fatal failures are estimated using the operational experience for processors. According to (Jockenhoevel-Barttfeld 2014) the TXS experience is 44 million hours for processors running application software (considering 4 redundancies since we are assuming CCF, which means that only experience from one train can be claimed). No failures were observed.

Using a one stage Bayesian approach (see section 6.3) this yields an estimate of 1.1E-8 failures/hour and processor. Considering 24 hours mission time, the fatal failure probability per processor (CCF is assumed between redundant units) is estimated to:

$$P_{\text{fatal, per processor}} = 1.1\text{E-}8 \cdot 24 \approx 3\text{E-}7 \text{ per demand for redundant units in TXS system.}$$

6.4.5 Estimation of non-fatal failure probability

The estimation of non-fatal failure probability is challenging, since this type of failure is not automatically detected by the system and may be very hard for an observer to identify. The probability estimation for non-fatal failures has to therefore be based on engineering judgements. There is simply insufficient operational experience to make good claims for non-fatal failures.

The potential impacts of non-fatal failures are also more challenging to estimate, as these can potentially generate either no signal- or spurious signal scenarios. To be able to estimate the impact of the application software faults, the application software has to first be split in modules, see next section.

6.4.6 Quantification method for non-fatal failures

The approach suggested is a Bayesian Belief Network (BBN) approach (Figure 8).

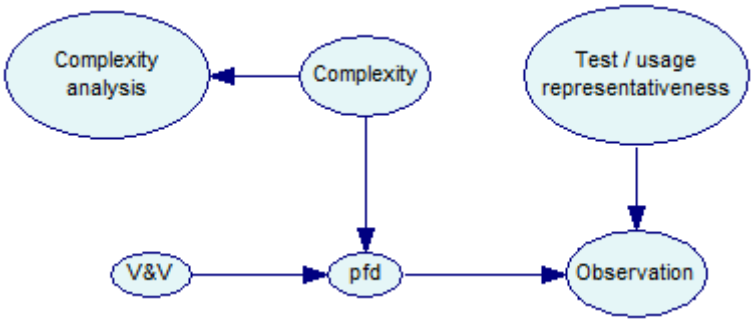


Figure 8. A BBN for assessing software reliability using V&V class, software complexity and usage and test observations as evidence.

The complexity is measured on the scale low, medium and high (see section 6.7). The V&V is measured from a scale from 0-4 according to Table 5.

Table 5. V&V level.

V&V	Safety class in nuclear (IEC 2009)
0	*Non-nuclear safety*
1	C
2	B
3	A
4	-

The estimate of the failure probability for an application software module is then suggested to:

$$E[P_{NSA} \mid F] = 1E\text{-}6 * F, \quad (5)$$

where F is a shaping factor.

The shaping factor F is defined according to Table 6.

Table 6. Shaping factor F.

		Complexity		
		High	**Medium**	**Low**
V&V	**0**	10000	1000	100
	1	1000	100	10
	2	100	10	1
	3	10	**1**	0.1
	4	1	0.1	0.01

The distribution for the estimated failure probability is assumed to be a beta distribution Beta(α,β) with the above mean, and with an α of 0.5 and $\beta = \alpha(1 - \text{mean value}) / \text{mean value}$

to represent a wide distribution. The prior suggested above ranges (if F=1) from approximately 4E-9 to 4E-6 (5^{th} and 95^{th} percentile respective).

The derived non-fatal failure probability can then be updated using a Bayesian approach, should there be sufficient data available. In case a Bayesian update is performed, pooling of operational experience is a very important aspect, see section 6.6.

The non-fatal failure probability estimated using the above method is then split in spurious signal and no-signal failure probability respectively using the fraction 0.2 for spurious (active failure) and 0.8 for no signal (passive failure), illustrated by Figure 9 (NKS-341).

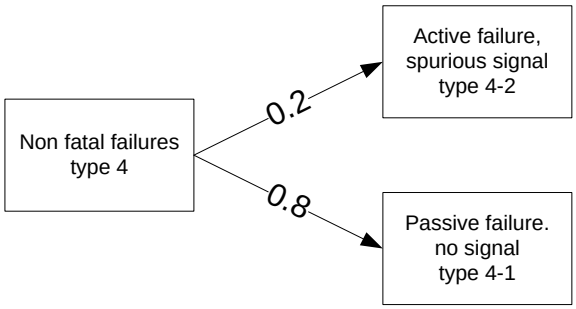


Figure 9. Split of non-fatal failures in active and passive failures.

6.5 Comparison of the application software quantification method with existing data

As the fatal failure probability is based upon operational experience, this estimate is consistent with operational experience. The below justification is hence focused on the non-fatal failure probability.

An estimate of the failure probability per demand is collected by Areva and presented in (NKS-341). No non self-announcing faults have been observed for the TXS system from a number of nuclear power plants. The demands that have been analysed are two low demand systems (reactor protection system (RPS) and reactor limitation system (RLS)) and one high demand system (reactor control system (RCS)). Demands have been estimated to:

- D_{RPS} = 3.4E+3
- D_{RLS} = 2.4E+3
- D_{RCS} = 7.0E+6.

Using a one stage Bayesian model, the failure probability is estimated

$$P(AF\ NSA) = \frac{2n_{NSA}+1}{2D}. \quad (6)$$

The failure probability is, as can be seen, dependent on the pooling of the data. In this case the most relevant question is whether or not the high demand rate system can be included in the estimation of the failure probability of the application software in RPS. In this discussion two estimates are calculated, one including RCS and one excluding RCS. The failure probability

at demand of the application software function is estimated to 9E-5 and 7E-8 respectively (where the lower figure is including RCS information).

The above operational experience is compared with the data derived from the BBN method suggested in this report. In the comparison we use V&V level 3 and the complexity is assumed as "medium". The reason for selecting medium for complexity is that the complexity estimation method shall be correlated such that the AS modules in RPS are typically medium or low. Assuming medium complexity, the BBN would yield a failure probability for an application software module of 1E-6 per demand. Assuming an average amount of application software modules of 10 in an application software function, the estimate would for the application software function be 1E-5 per demand. When the method's estimate is compared with the information operational experience above, it can be concluded that it is in the same range. The estimate calculated using the method of the non-fatal failure probability is clearly above the operational experience estimate when high demand system (RCS) is included (7E-8), but slightly below the estimate for RPS+RLS (9E-5). The estimate for RPS and RLS is though based on too few demands to be able to make a reasonable judgement for very low probabilities. The data collected for TXS is therefore considered at least not to be contradicting the estimate of the current baseline estimate.

In (Jänkälä, 2010) the failure probabilities used for the software at digital safety I&C were discussed. The probability of software failures affecting more than one division is 5E-5 (within same automation system). This figure is not easily comparable to the estimate above, as the estimate covers both application software failures and system failures and all failure modes. Also, it shall be observed that more than one application software function may be run one processor. Assuming only one AS function, the failure probability (fatal and non-fatal) would be 3E-7 (fatal failures, see section 6.4.4) plus 1E-5 (non-fatal failures estimated above). The SyS failures (2a and 2b), see section 6.3, would sum to 1.2E-5 and therefore the failure probability of one processor covering both application software failures and SyS failures is roughly 2.2E-5. The method is therefore considered to yield an estimate that is in line with the judgement in (Jänkälä, 2010).

The above claim is that, at least for an RPS type of system, the method proposed will generate failure probabilities that are reasonable compared to operational experience (TXS data) and that the method proposed also generates failure probabilities that are reasonably comparable to current PSA (compared to Jänkälä, 2010).

6.6 Bayesian update, pooling of data

6.6.1 Software and pooling

When failure data are collected, it is typically collected for groups of statistically homogeneous components. For pumps this can mean type of pump and its capabilities, e.g. horizontal centrifugal pump, in standby, with delivery head 8 bar and 120 kg/s.

When the data is collected and evaluated it can then either be pooled, i.e. several components are considered identical, or the data can be treated individually for each component. If the data is pooled, then the operational history is added up (sum of failures and hours) to one "super component". A condition for pooling is that the data is homogeneous, i.e. that the components are failure prune to the same extent. Homogeneity tests are then preferably performed to verify that the observed failures can be pooled (which typically means that it

cannot be proved that the data is inhomogeneous). If homogeneity cannot be demonstrated then the data needs to be treated as separate components (or divided in smaller groups where homogeneity can be demonstrated). A two-staged approach could also be considered (especially the approach used in the T-book (2010), where every object is considered as its own entity). This approach could be especially interesting if data from several vendors would be available for the same type of application software.

The collection and data analysis for software application modules have some challenges, of which following is of especial interest:

- Each software module is its own entity, and the software modules are therefore not (generally) exactly the same
- There are very few failures observed. Further, if a failure was observed, the software would be repaired (fault would be removed)

Generally, each separate software module could, based on the first bullet, be argued being treated as its own individual. We should also try to define groups that are assumed to be reasonably homogeneous.

The factors that previously (NKS-341) have been identified for defining the priori function are:

1. Failure mode
2. Complexity
3. Level of V&V.

What might need to be considered are also the some other factors that could be argued to be affecting the probability. One factor not considered in (NKS-341) was:

4. Operational profile of application software module (demand frequency).

The testing frequency could be seen as important to have some coverage of the scenarios that could be relevant. It must be remembered that the situations that are sought for with regard to RPS function are generally situations where the system is not normally triggered (that is, the normal functions are working).

The type of software system (vendor specific) could also affect the prior, due to different design of systems, error treatment, tool support for design of system etc, but this has not been considered in the analysis.

In (NKS-341) data was collected for three different groups of NSA failures, failure per demand. They were based on tests of RPS, RLS and RCS systems.

6.6.2 Pooling of data for software components

Pooling of software modules cannot be justified by statistical data with the scarce data available (see discussion in Appendix C). If homogeneity is claimed it should be based on

engineering judgements of the actual software modules and evidence for the statement should be discussed/presented.

6.6.3 Bayesian update of data

In section 0 the method for non-fatal failure estimate is presented. The method includes an Bayesian update of the prior data. A one staged Bayesian approach would not improve the estimate for low demand rate modules. This is simply because to modify the prior, which is in the range of 1E-6 would require more than 100 000 demands to significantly having an impact on the quantification. Hence, low demand rate modules are not relevant for a one staged Bayesian update. If it cannot be claimed that the group of software modules that may be pooled has more than 100 000 demands, then it will not be relevant.

A two staged Bayesian approach could be applied but it would still require that the data could be claimed to represent reasonably similar software. A two staged Bayesian could potentially be used if data from different systems (vendors), operating under similar conditions (separated between low demand systems and high demand systems) were available.

6.7 Complexity

To be able to use the quantification method presented in section 6.4.6, the complexity of software needs to be estimated. Complexity analysis of application software modules was studied in the DIGREL project (NKS-341), and different complexity indicators and analysis methods were identified. However, consensus on complexity classifications has not been achieved. While there is good agreement on distinguishing *high* from *low* complexity, the line between *low* and *medium* complexity and *medium* and *high* complexity is not very clear. Rough categorisation (low-medium-high) is considered sufficient because the uncertainties are high and the significance of an individual software module with regard to PRA results is supposedly quite small. It is also practical because the analysis can be performed faster than if a numeric metric was calculated, and because it is difficult to give an accurate definition of complexity as a numeric metric.

In the software reliability report of DIGREL (NKS-341), two complexity analysis methods were considered, the ISTec's method (Märtz et al. 2010) and SICA (SImple Complexity Analysis) method (NKS-341). In these two methods, the analysis is based on software logic diagrams. In the ISTec's method, a software complexity metric is calculated based on nine indicators using a Bayesian belief network model. All the details of the ISTec's method are not publicly known and it is complicated to calculate by hand. In SICA method, complexity analysis is performed by a visual assessment using simple decision rules. In addition, TOPAAS method for software reliability analysis accounts software complexity (TOPAAS 2011).

Five example software modules were analysed using SICA and a modified version of the ISTec's method in the DIGREL report (NKS-341). The ISTec's method produced higher complexities in some cases. It seemed that modules of medium and high complexity are rare according to SICA, and it was noticed that SICA method does not capture some aspects of complexity in the best possible way. Hence, in this section, SICA method is developed further to match the results of the "modified ISTec's method" better.

SICA aims for simple complexity analysis that an expert can perform by a short visual assessment of a logic diagram. SICA accounts the complexity of function blocks, the

interconnections between function blocks and inputs and outputs in the determination of the complexity category of a logic diagram.

Based on experience gathered from model checking (Lahtinen et al. 2010, Lahtinen et al. 2012, Björkman et al. 2009), feedback loops and some function blocks (e.g. time-related blocks, flip-flops and modified function blocks that implement non-standard functionality) increase the software complexity. In addition, function blocks that perform complex computation are likely to affect the complexity.

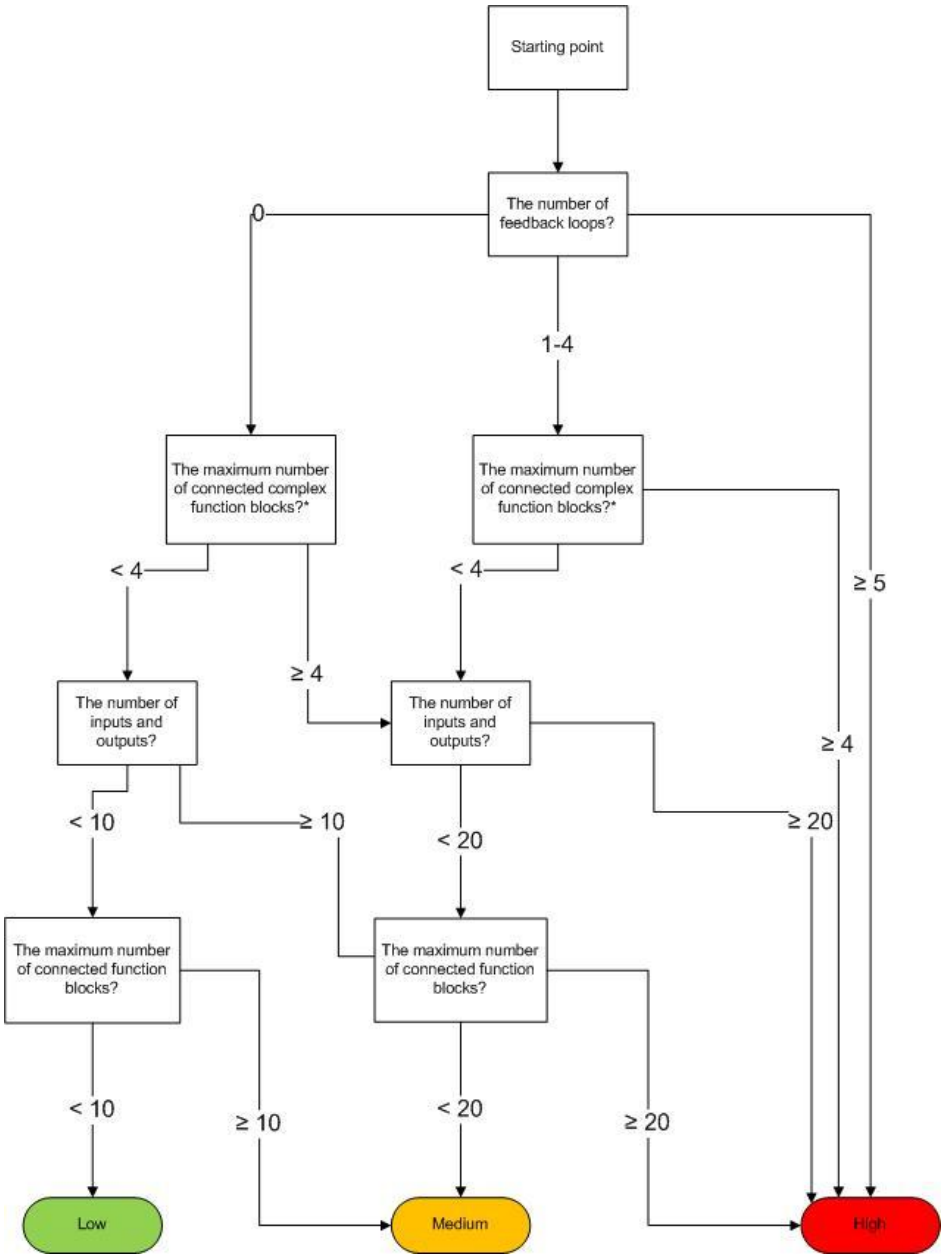
In the SICA method, all the function blocks that use internal memory are categorised as *complex function blocks*. Also those function blocks for which $IN + OUT + PAR > 10$ are categorised as *complex function blocks*.

In the SICA method, the complexity of an application software module is determined based on the number of inputs and outputs in total, the number of feedback loops (not including feedback loops inside function blocks) and the maximum number of "*connected"* complex function blocks. Function blocks are defined to be "connected" if they affect the same output signal, i.e. function blocks located in the same signal path that are involved in the processing of the signal. Note that it is required that the complex function blocks are on the same signal path because if they are in different paths, the software consist of many not so complex parts and is not complex (except maybe due to complex input/output relations). The number of connected function blocks can be calculated for each output signal involved in the software module and the maximum value is used in complexity analysis.

SICA has previously accounted only *complex function blocks*. To take other function blocks into account too, a rule on the maximum number of connected function blocks is added to the new version of SICA. This will not complicate the analysis much in most cases, because checking if the number is larger or smaller than 10 or 20 is very straightforward. The largest complication is that there is one more rule (with two variations) for the analyst to learn.

The decision parameters of the number of inputs and outputs, and the maximum number of connected complex function blocks are lowered from the DIGREL report (NKS-341) so that the results will be closer to the results of the ISTec's method. A user of the SICA method should notice that the parameter values and decision rules are only suggestions of the authors. These rules and parameters have been seen suitable based on the analysis of numerous software diagrams. Still, they are just expert judgements and there are no unambiguously correct rules to define complexity categories. A user of SICA is allowed to choose different parameters and modify the decision rules if they seem to be more fitting for his/her purposes.

The decision rules to categorise software modules into complexity classes, *low*, *medium* and *high*, are presented in Figure 15.



* all function blocks that use internal memory or those for which $IN + OUT + PAR > 10$, where IN is the number of inputs, OUT is the number of outputs and PAR is the number of parameters.

Figure 10. Rules to identify application software complexity in the SICA method.

In the following, few concepts related to the SICA analysis are explained more in detail:

- **Feedback loop** is a path in a logic diagram that starts and ends with the same point (any point in the feedback loop can be considered as a start/end point). The path can continue through internal outputs and inputs of the diagram to form a feedback loop. A feedback loop is formed by a single path. If a part of a feedback loop has a "redundant" part (another path with the same start point and end point), there are two feedback loops. In other words, multiple feedback loops can have a common part.
- **Input** of a logic diagram is a single signal coming to the diagram from outside of the diagram (e.g. from another diagram or measurement sensor). A logic diagram can also

include internal inputs that come from the same diagram, but they are not counted as inputs of the diagram in the SICA method.

- **Output** of a logic diagram is a single signal going out of the diagram (e.g. to another diagram or device). A logic diagram can also include internal outputs that lead to internal inputs in the same diagram, but they are not counted as outputs of the diagram in the SICA method.
- A logic diagram can include **hidden redundancies**, i.e. only one of identical parts in the diagram is explicitly shown in documentation. Inputs, outputs, function blocks and feedback loops must be calculated from all redundant parts of a diagram in the SICA method even if they are not explicitly shown in documentation.

The new version of SICA is more comprehensive and more conservative in that it gives higher complexity classification for some modules. The SICA method takes most of the factors affecting complexity into account. The complexity of interconnections between function blocks is measured by counting connected function blocks. It is quite simplified metric for that but a more detailed solution would make the analysis complicated. Inputs and outputs are treated together for simplicity and because there have not been claims that they should have different weights. The ISTec's method counts upstream and downstream diagrams, but SICA neglects them to keep the analysis simple enough. It is not considered a big defect because the number of upstream and downstream diagrams correlates with the number of input and outputs.

SICA analysis can be considered simple. Learning of the method could take some time because there are several decision rules and parameters to remember. Complex function blocks also need to be identified, and it might be good practise to do that before the analysis of diagrams. But after learning the method and knowing which function blocks are complex, the categorisation of most of the software modules is just a matter of visual assessment of few seconds.

SICA analysis of nine software modules from a fictive reactor protection system is presented in Appendix B.

6.8 Software CCF

CCF between identical software modules in redundant trains is a general assumption. Failures in the SyS software is assumed to apply to all trains, in which the SyS software is used (dependent on the case studied, see Table 4). With regard to AS faults, the definition of fatal failures is assuming that all redundant processors are affected (conditional CCF probability = 1). A redundant processor is defined as running the same type of application software as the other trains (there may be small deviations in the running AS, but the majority should be the same). Non-fatal failures are considered to be completely dependent for each similar AS module.

The above assumptions are considered conservative, since the few failures observed in TXS operational experience are not CCF related. However, there are very scarce data and since the modules are operating under the same conditions, with same signal trajectories, then it would require justification to claim that they are not affected by CCF. Because the different trains are running with some time offset, it could be claimed that the CCF probability is not 1.

There could also potentially be CCF between AS modules (using same elementary functions, partly being the same, using almost same inputs etc.). Generally, when data are pooled (see section 6.6) this should indicate that there is likelihood of a CCF. The components are assumed to fail under similar conditions.

To account for these types of dependencies it is suggested that CCF between AS modules are considered. A simple Beta-factor method is suggested. CCF between AS modules is only relevant to consider when the functions are used as input to redundant systems (from a plant level perspective). It shall be noticed that the suggested CCF method is actually CCF between CCFs.

The estimation of the Beta-factor is based upon the assumption that a CCF between software would be triggered by a number of inputs that are in states that are not properly treated. Therefore, the CCF would be triggered by identical inputs. Given this assumption, if there are no identical inputs, the Beta-factor would be 0 and if all inputs are the identical the Beta-factor could in worst situation be 1.

The estimate of the Beta-factor could be based on some indicators (like in HRA methods). In the rough method below, only two indicators are used for screening of the similarity:

- Amount of identical Inputs
- Level of similarity of the functions

If identical functions and inputs are used, the Beta-factor should be assumed to be 1.0. If the AS´s are similar, but not exactly the same, a rough estimate is suggested below. "Similar" is in current method assumed as a screening rule based on that the logic is reasonably same. Formal rules for "similar" is yet to be developed.

Table 7. Rough estimation of Beta-factor when AS1 and AS2 is similar and not exactly the same.

		AS2 complexity		
		High	**Medium**	**Low**
AS1 complexity	**High**	InputsIdentical / InputsTotal	(InputsIdentical/ InputsTotal) /2	Beta = 0
	Medium		InputsIdentical/ InputsTotal	(InputsIdentical/ InputsTotal) /2
	Low			InputsIdentical/ InputsTotal

If the two AS are similar and they are of the same complexity – the Beta factor is estimated based on the amount of same inputs in relation to total amount of inputs. If the AS:s are of different complexity, the treatment of the signals will automatically have to be different and the Beta-factor should be reduced. In the suggested approach the Beta-factor is divided by a factor of 2.

As an example, assume two similar AS modules denoted by AS1 and AS2. AS1 has 5 inputs and AS2 has 8 inputs. 3 of the inputs are identical. Both of the application software modules are judged to be of medium complexity. The two AS modules are judged to be similar (due to that the logic using the identical inputs are very similar). Therefore it is decided that there is a risk of CCF. The beta factor is calculated as:

- InputsIdentical = 3
- InputsTotal = 5 + 8 – 3 = 10 (reduced with 3 to not double count same inputs).

The CCF factor (beta factor) can hence be calculated as 3/10 = 0.3. This factor would then be applied in the modelling of the application software modules in the fault tree.

6.9 Justification of the software reliability model

A remaining task that is planned for the project in future is to further define the claims and evidence for the software approach used. The list of claims below is a draft list, which will need to be refined. The claims are defined on three levels a) software system claims b) software failure mode claims and c) failure data claims.

Software system claims are:

- Hardware and software failure modes can be treated separately.
- From a reliability analysis perspective, it is meaningful to define failure modes for a software system as it is done for hardware systems to be accounted for in a PSA.
- The software system can, from a system failure perspective, be divided in software subsystems (module) which are failing independently.
- The software subsystem can be divided into SyS module, DCS module and AS modules.
- Failures in Elementary Functions can be properly covered by failures in SyS, DCS and AS.

Software failure mode claims are:

- The failure modes of the system can be described by Fatal and Non-Fatal failures.
- Fatal failures ending up in a non-defined state can be neglected.
- Non-fatal failures can be described by either no-signal or spurious signal scenarios.
- SyS software failures are fatal failures and non-fatal failures can be neglected.
- DCS failures are fatal failures and non-fatal failures can be neglected.
- AS failures can be both fatal and non-fatal failures.
- Software failures can conservatively be considered to be CCFs for same software.
- Non-similar software CCF can be neglected.

Failure data claims are:

- The failure probability for SyS can be estimated using operational experience gathered during operation.
- The failure probability for DCS can be estimated using operational experience gathered during operation.
- AS Fatal failures probability can be estimated using operational experience gathered during operation.

- AS Non-fatal failures probability are dependent on complexity, level of verification and operational profile of software.
- AS modules can be pooled (even though the software is not exactly the same) when the reliability is estimated.

These claims (assumptions) have been discussed during the development of the method. However, they have not been put as claims and evidence. The use of "claims and evidence" is considered to be a good way to describe the assumptions made during the development of the method, to have a transparent approach.

7. Failure data collection

7.1 The role of failure data in assessment of I&C reliability

I&C failure data is one of the information sources for the assessment of I&C reliability. For the data to be useful in the quantification of I&C reliability for the use in PSA, there needs to be enough relevant data which is collected, categorized and processed in a useful manner. The OECD/NEA Working Group on Risk Assessment (WGRISK) DIGREL task group developed a failure mode taxonomy for reliability assessment of digital I&C systems for use in PSA (OECD 2015). It was developed to support modelling and quantification efforts and to help define a structure for data collection. The OECD DIGREL task group recommended the applicability of the taxonomy in data collection to be tested.

Hardware failure data is usually provided by the vendor of the equipment as a standard requirement in the contract between the utility and the vendor. The data provided by the supplier sets in practice the limit for the detail of the PSA. For software failure data, it is less evident how the data should be collected and processed, by whom as well as to what extent the data should be the accessible to other parties. In the DIGREL project, operating data on software failures concerning the TELEPERM® XS (TXS) platform was analysed by AREVA (Bäckström et al., 2015). The collaboration showed that the vendors of digital I&C are potential data sources as they typically have access to experience data from many plants, have needed insight on the software development processes and are capable to analyse the causes of the detected failures. However, the data is typically not freely available to other parties.

For software, the number of observed failures in I&C systems in operation is very low. Thus the main issue in failure data collection is to collect enough exposure data relevant for the examined systems. Operating experience has been judged to suit well for assessment of reliability of system software, whereas its significance for the assessment of application software is more limited. In practice, there is not enough data available to justify low failure rates for low-demand-rate application software based on data only, unless the data is pooled with the one for high-demand-rate systems. As discussed in Chapter 6, there are several open questions regarding the justification of data pooling. Is it justifiable to pool operating data between different software modules, between different systems, plants with same I&C provider or plants with different I&C provider?

7.2 The International Common-cause Failure Data Exchange (ICDE) project

The OECD/NEA International Common-cause Failure Data Exchange (ICDE) project was initiated in 1994 and is since 1998 formally operated by the NEA. The participating countries in ICDE are Canada, Czech Republic, Finland, France, Germany, Japan, Republic of Korea, Spain, Sweden, Switzerland, United Kingdom and United States.

The objectives of the ICDE project are to:

- collect and analyse CCF events over the long term so as to better understand such events, their causes, and their prevention;
- generate qualitative insights into the root causes of CCF events which can then be used to derive approaches or mechanisms for their prevention or for mitigating their consequences;

- establish a mechanism for the efficient feedback of experience gained in connection with CCF phenomena, including the development of defences against their occurrence, such as indicators for risk based inspections;
- generate quantitative insights and record event attributes to facilitate quantification of CCF frequencies in member countries; and
- use the ICDE data to estimate CCF parameters.

The ICDE data base currently consists of 12 individual component specific databases (e.g. pumps, emergency diesels, MOVs etc.). The data collection is organized in a way that each country participating in the ICDE project can decide for which "component" and for which time period it is willing to share data. The data from one component database and time period are then available for all countries which participate in this particular "component" data exchange as soon as the country has provided respective own data.

In 2012, ICDE decided to start collecting I&C system related data. The project has prepared Coding Guidelines for Digital Instrumentation and Control Equipment (Kreuser and Stiller, 2013) and started collection of digital I&C related failure data in 2015. After this first data exchange for Digital I&C is completed the ICDE is planning to have a workshop to discuss insights from the collected events, preliminary to be held in 2017.

While ICDE takes advantage of the OECD DIGREL taxonomy to some extent, the primary focus of ICDE is to describe the observed failures in detail in order to understand their root causes. The ICDE does not endeavour to collect exposure data associated with the observed failures nor does it process the data for the purpose of failure frequency quantification. Thus, the ICDE I&C data collection seems insufficient for the need of PSA. Therefore, international collaboration and discussions are still needed in order to forward the use of I&C failure data for the needs of PSA. The MODIG project partners strive to foster such discussions e.g. through active participation in the WGRISK group and international seminars and conferences.

8. Conclusions

A digital I&C reactor protection system will form a core in a modern nuclear power plant. A failure in the hardware or software has a potential to propagate and affect the overall safety of the station. The design of the system needs to meet the deterministic criteria and due to the complexity of the system the use of PSA technique could be very relevant to demonstrate the DiD concept. The digital I&C system should also be included in the PSA evaluation, as it has a potential to significantly affect the overall plant reliability.

MODIG is an international collaboration project focussing on risk analysis methods and application for modern nuclear power plants with digital automation systems. The objective is to get a consensus approach for a reliability analysis of a plant design with digital I&C, improved integration of probabilistic and deterministic approaches in the licensing of digital I&C, improved failure data collection including software failure probability quantification, and practical application of PSA to compare design alternatives.

In 2015, MODIG explored the assessment of defence-in-depth by probabilistic safety assessment (PSA) with emphasis on I&C, outlined an approach to analyse spurious actuations, developed further the confidence on the software reliability method proposed in the previous NKS project DIGREL and prepared a proposal for an international collaboration on the development of a systematic approach for the diversity assessment of digital I&C systems for PSA (OECD/NEA Working Group RISK task proposal). A joint workshop together with the NKS project PLANS was organised with more than 40 participants from seven European countries.[1]

Defence-in-depth is a fundamental safety principle for nuclear power plants, and similar principle are applied in all safety-critical technological areas even if it may be called differently in some context. Defence-in-depth is basically linked with the deterministic safety assessment, including the categorisation of initiating events and associated failure criteria.

DiD requirements can lead to complex design solutions. It's an open question how the impact of complexity should be accounted when comparing designs. Complexity is an indirect factor affecting the system reliability, similar to organizational factors. It might however be assumed that higher complexity correlated with higher unreliability due to more complicating operation and maintenance. There is no unique metric for complexity and there is no statistical data to estimate the correlation.

It can be expected that deterministic principles cannot fully be satisfied or that the deterministic demonstration is limited and may be biased by the postulation criteria. PSA is therefore needed to complement the safety demonstration. PSA mostly focuses on modelling DiD levels 3 and 4, while levels 1 and 2 are often implicitly covered by the initiating event analysis. A more detailed analysis of systems and functions involved in the DiD levels 1 and 2 could provide useful information both for plant safety and availability perspectives.

PSA can also be used in a "deterministic" manner, by utilizing the logic model to analyse the failure criteria. This application of PSA is common to safe shutdown analyses applied, e.g., in assessment of fire hazards. In this application, the properties of the minimal cut sets are

[1] http://www.nks.org/download/modigplans_workshop_2015092930_notes_u001.pdf

analysed with respect to each initiating event category. The method could analogously be applied to failure tolerance analyses as required by STUK in the Guide YVL B.1. Even though today's PSAs are quite complete, the application of PSA for failure tolerance analyses requires development of features in the model. More detailed analysis of I&C at DiD levels 1–3 may be needed. One crucial issue for the results will be the assessment of the independence between barriers (diversity), which can be associated with the postulation and quantification of common cause failures.

Spurious actuation is a functional failure mode when a component performs a function without a real demand. Spurious actuations are quite well covered in today's PSAs, when done at appropriate level of details and when considering fault tolerant features properly. Open issues include how to identify possible common cause initiators comprehensively and to what extent CCF causing a spurious actuation should be considered. Both questions are relevant to digital I&C due to possibly complex effects via system dependences and due to huge number of possible failure locations.

There is a need to develop reasonable but comprehensive approach to analyse spurious actuations both for deterministic and probabilistic analyses. Analysis requirements have been compiled, generic failure modes taxonomy has been outlined based on von Wright's theory on concept of action, and an analysis approach has been outlined. A combination of top-down and bottom-up approaches is suggested to keep the analysis manageable. By a top-down approach a screening of irrelevant system failures or I&C function failures can be performed. Bottom-up approach will be applied to the critical (non-screened) functions.

The use of the PSA model for both DiD and PSA requires a proper definition and handling of the software system failure modes. To be able to define relevant software failure modes the system needs to be split into a number of entities. The entities used within this report are basically system software and application software. The system software can be further split into the run time environment and communication software. The failure modes applicable for each type of software differ. The analysis of software reliability is a continuation of the study performed in (NKS-341).

For system software failures the reliability is suggested to be estimated based on operational experience. The analysis of application software reliability is also suggested to be performed based on operational experience with regard to fatal failures. Estimation of the probabilities for application software non-fatal failures can hardly be performed based on operational experience, as it is hard to collect this data on an appropriate level and even if you could — you would not have sufficient amount of operational experience. Therefore the method suggested is an analytical approach using a metrics of complexity and verification and validation. If operational experience data is available, those could be applied in a Bayesian manner. Pooling of data is an open issue and the way it is done can significantly affect the results.

The SICA method is suggested to be used to estimate the level of complexity for application software. It has been improved so that it is more comprehensive and more conservative in that it gives higher complexity classification for some software modules. The SICA method is very easy to apply, and the analysis can be performed without in depth knowledge about the code.

Estimation of CCF probabilities between software failures is discussed. Generally the assumption is between identical software modules a complete CCF is postulated. A method to assess CCF between non-identical but similar application software modules is also outlined.

Software reliability is a highly controversial area, and therefore it is important to provide the claims and evidence behind the assessments. As part of the work performed within this report, a list of high level claims used in the reliability assessment has been compiled.

Hardware I&C failure data is usually provided by the vendor of the equipment as a standard requirement in the contract between the utility and the vendor. The data provided by the supplier sets in practice the limit for the detail of the PSA. Vendors of digital I&C have shown to be potential data sources also for software failures as they typically have access to experience data from many plants, have needed insight on the software development processes and are capable to analyse the causes of the detected failures.

The OECD/NEA ICDE project has also started collection of digital I&C related failure data in 2015. Their primary focus is on understanding of failure causes and ways of prevention, and reliability quantification is not pursued. Thus, the digital I&C related failure data collected by ICDE is likely to be of limited use from the PSA point of view. International collaboration and discussions are still needed in order to forward the use of I&C failure data in PSA. The MODIG project partners strive to foster such discussions e.g. through active participation in the WGRISK group and international seminars and conferences.

For software, the number of observed failures in I&C systems in operation is very low, emphasizing the importance of exposure data collection. Operating experience has been judged to suit well for assessment of fatal failure reliability of system and application software, whereas its significance for the assessment of non-fatal failures for application software is more limited. Pooling of data is one way to deal with the scarcity of observed failures. However, there remain several open questions regarding the justification of data pooling, concerning which data may be pooled and on which conditions.

9. References

Ahonen, E. 2011. Studying of the Failure Tolerance with the Probabilistic Risk Assessment. Master's Thesis, Lappeenranta University of Technology (In Finnish).

Authén, S., Björkman, K., Holmberg, J.-E. & Larsson, J. 2010. Guidelines for reliability analysis of digital systems in PSA context — Phase 1 Status Report. NKS-230 Nordic nuclear safety research (NKS), Roskilde.

Authén, S., Gustafsson, J. & Holmberg, J.-E. 2012. Guidelines for reliability analysis of digital systems in PSA context — Phase 2 Status Report. NKS-261 Nordic nuclear safety research (NKS), Roskilde.

Authén, S. & Holmberg, J.-E. 2013. Guidelines for reliability analysis of digital systems in PSA context — Phase 3 Status Report. NKS-277, Nordic nuclear safety research (NKS), Roskilde.

Authén, S., Holmberg, J.-E., Lanner, L. & Tyrväinen, T. 2014. Guidelines for reliability analysis of digital systems in PSA context — Phase 4 Status Report. NKS-302, Nordic nuclear safety research (NKS), Roskilde.

Authén, S., Holmberg, J.-E., Tyrväinen, T., Zamani, L. 2015. Guidelines for reliability analysis of digital systems in PSA context – Final Report, NKS-330, Nordic nuclear safety research (NKS), Roskilde.

Besnard, D. & Hollnagel, E. 2012. Some myths about industrial safety. CRC technical Report.

Bengtsson, L., Holmberg, J.-E., Knochenhauer, M. & Rossi, J. 2011. Probabilistic Safety Goals for Nuclear Power Plants; Phases 2-4 / Final Report. NKS-226, Nordic nuclear safety research (NKS), Roskilde.

Björkman, K., Frits, J., Valkonen, J., Heljanko, K., Niemelä, I. 2009. Model-based analysis of a stepwise shutdown logic. MODSAFE 2008 Work Report, VTT working papers 115, VTT, Espoo. 42 p.

Bäckström, O., Holmberg, J.-E., Jockenhövel-Barttfeld, M., Porthin, M. & Taurines, A. 2014. Software reliability analysis for PSA. NKS-304, Nordic nuclear safety research (NKS), Roskilde.

Bäckström, O., Holmberg, J.-E., Jockenhövel-Barttfeld, M., Porthin, M., Taurines, A. & Tyrväinen, T. 2015. Software reliability analysis for PSA — Final report. NKS-341, Nordic nuclear safety research (NKS), Roskilde.

Caldwell, A., Olsson, A., Nordqvist, M., Johanson, G. Holmberg, J.-E., Sunde, C., Karanta, I. 2014. Addressing off-site consequence criteria using Level 3 PSA — Phase 1 Status Report, NKS-303, Nordic nuclear safety research (NKS), Roskilde.

Cederhorn, E. & Frisk, M. 2014. Experiences from Developing and Implementing Shutdown Fire PRA at Forsmark NPP. Proc. of 12th International Probabilistic Safety Assessment and Management Conference, 22-27.6.2014, Honolulu.

Chu, T.L., Martinez-Guridi, G., Yue, M. & Lehner, J. 2006. A review of software induced failure experience. 5th NPIC HMIT meeting, November 2006, BNL-NUREG-77124-2006-CP.

Chu, T.L., Martinez-Guridi, G., Yue, M., Samanta, P., Vinod, G., and Lehner, J. 2009. Workshop on Philosophical Basis for Incorporating Software Failures into a Probabilistic Risk Assessment, Brookhaven National Laboratory, Technical Report, BNL-90571-2009-IR, November.

Dahll, G., Liwång, B. & Pulkkinen, U. 2007. Software-Based System Reliability, Technical Note, NEA/SEN/SIN/WGRISK(2007)1, Working Group on Risk Assessment (WGRISK) of the Nuclear Energy Agency, Paris.

EPRI. 2014. Principles and Approaches for Developing Overall Instrumentation and Control Architectures that Support Acceptance in Multiple International Regulatory Environments. 3002002953, Electric Power Research Institute, Palo Alto.

Fleming, K.N., Silady, F.A. 2002. A risk informed defence-in-depth framework for existing and advanced reactors, Reliability Engineering and System Safety 78 205–225.

Hellström, P. 2015. DiD-PSA: Development of a Framework for Evaluation of the Defence-in-Depth with PSA. SSM Report 2015:04, Strålsäkerhetsmyndigheten, Stockholm.

Holmberg, J.-E., Nirmark, J. 2008. Risk-informed assessment of defence in depth, LOCA example. Phase 1: Mapping of conditions and definition of quantitative measures for the defence in depth levels. SKI Report 2008:33.

IAEA. 1991. Safety related terms for advanced nuclear plants, IAEA-TECDOC-626, International Atomic Energy Agency, Vienna.

IAEA. 1996. Defence-in-depth in nuclear safety. INSAG-10. International Atomic Energy Agency, Vienna.

IAEA. 1999. Modern Instrumentation and Control for Nuclear Power Plants: A Guidebook, Technical reports Series No. 387, International Atomic Energy Agency, Vienna.

IAEA. 2002. Instrumentation and Control Systems Important to Safety in Nuclear Power Plants, IAEA Safety Standards Series No. NS-G-1.3, International Atomic Energy Agency, Vienna.

IAEA. 2007. IAEA safety glossary. Terminology used in nuclear safety and radiation protection, 2007 edition, International Atomic Energy Agency, Vienna.

IAEA. 2010. Development and application of level 1 probabilistic safety assessment for nuclear power plants for protecting people and the environment, IAEA Specific Safety Guide No. SSG-3, International Atomic Energy Agency. Vienna.

IAEA. 2012. Safety of Nuclear Power Plants: Design-Specific Safety Requirements, No. SSR-2/1, International Atomic Energy Agency, Vienna.

IAEA. 2014. The Use of the International Nuclear and Radiological Event Scale (INES) for Event Communication Guidelines and Good Practices for Setting up a National Framework on the Effective Use of INES for Event Communication, IAEA, Vienna.

IEC. 2009. Nuclear power plants — Instrumentation and control important to safety — Classification of instrumentation and control functions, IEC 61226, ed. 3.0, International electrotechnical commission, Geneva.

IEC. 2010a. Functional safety of electrical/electronic/programmable electronic safety-related systems, Part 2: Requirements for electrical/electronic/programmable electronic safety related systems. IEC 61508-2, ed. 2.0. International Electrotechnical Commission, Geneva.

IEC. 2010b. Functional safety of electrical/electronic/programmable electronic safety-related systems, Part 4: Definitions and abbreviations. IEC 61508-4, ed. 2.0. International Electrotechnical Commission, Geneva.

IEEE. 2003. IEEE Standard for Qualifying Class 1E Equipment for Nuclear Power Generating Stations," IEEE Std. 323-2003, Institute of Electrical and Electronics Engineers.

ISO/IEC. 2005. Software engineering — Software product quality requirements and evaluation (SQuaRE) — Guide to SQuaRE, ISO/IEC 25000:2005. International Electrotechnical Commission, Geneva.

ISO/IEC/IEEE. 2010. Systems and software engineering – Vocabulary, ISO/IEC/IEEE 24765:2010. International Electrotechnical Commission, Geneva.

Jockenhoevel-Barttfeld, M. 2014. Estimations of baseline probabilities of application software failures AREVA, D02-IN-PEPS-G-14-0007 rev B.

Jänkälä, K. 2010. Reliability of New Plant Automation of Loviisa NPP, Presentation at NKS seminar, Espoo, September 2010.

Kreuser, A, Stiller, J. 2013. Coding Guidelines for Digital Instrumentation and Control Equipment. International Common-Cause Failure Data Exchange, ICDECG14.

Lahtinen, J., Valkonen, J., Björkman, K., Frits, J., Niemelä, I., Heljanko, K. 2012. Model checking of safety critical software in the nuclear engineering domain. Reliability Engineering and System Safety, 105, Special Issue ESREL 2010, 104 – 113.

Lahtinen, J., Björkman, K., Valkonen, J., Frits, J., Niemelä, I. 2010. Analysis of an emergency diesel generator control system by compositional model checking. MODSAFE 2010 work report, VTT Working Papers 156, VTT, Espoo. ISBN 978-951-38-7497-1, 35 p.

Märtz, J., Miedl, H., Lindner, A., Gerst, Ch. 2010. Komplexitätsmessung der Software Digitaler Leittechniksysteme, ISTec-A-1569.

OECD. 2015. Failure modes taxonomy for reliability assessment of digital instrumentation and control systems for probabilistic risk analysis, NEA/CSNI/R(2014)16, OECD/NEA/CSNI, Paris.

Persson Sunde, E. 2012. Funktionsorienterad klassificering av komponenter – tillämpning på T-bokens pumpar, Scandpower AB Rapport 210528/R1.

Rasmussen, J. 1997. Risk management in a dynamic society: a modelling problem, Safety Science 27(2/3) 183–213.

Reason, J. 2000. Human error: models and management, BMJ Mar 18; 320(7237): 768–770.

SSM. 2009a. Swedish Radiation Safety Authority Regulatory Code. The Swedish Radiation Safety Authority's Regulations and General Advice concerning Safety in Nuclear Facilities. SSMFS 2008:1. Consolidated version with amendments made up to and including SSMFS 2010:3, Swedish Radiation Safety Authority, Stockholm.

SSM. 2009b. Swedish Radiation Safety Authority Regulatory Code. The Swedish Radiation Safety Authority's Regulations concerning the Design and Construction of Nuclear Power Reactors. SSMFS 2008:17, Swedish Radiation Safety Authority, Stockholm.

STUK. 2013a. Safety design of a nuclear power plant, Guide YVL B.1, Radiation and Nuclear Safety Authority in Finland, Helsinki.

STUK. 2013b. Deterministic safety analyses for a nuclear power plant, Guide YVL B.3, Radiation and Nuclear Safety Authority in Finland, Helsinki.

T-book. 2010. Reliability Data of Components in Nordic Nuclear Power Plants, 7th edition, The TUD Office, Vattenfall Power Consultant, Stockholm.

TOPAAS. 2011. TOPAAS: Een structurele aanpak voor faalkansanalyse van software intensieve systemen. Rijkswaterstaat Ministerie van Verkeer en Waterstaat.

Trbojevic, V.M. 2005. Risk Criteria in EU, ESREL'05, Poland, 27-30 June 2005.

U.S.NRC. 1986. Safety Goals for the Operation of Nuclear Power Plants. 10 CFR Part 50, U.S. Nuclear Regulatory Commission, Washington D.C.

U.S.NRC. 2005. EPRI/NRC-RES Fire PRA Methodology for Nuclear Power Facilities, Final Report, NUREG/CR-6850, EPRI 1011989, U.S. Nuclear Regulatory Commission, Washington D.C.

U.S.NRC. 2012. EPRI/NRC-RES Fire Human Reliability Analysis Guidelines, NUREG-1921, U.S. Nuclear Regulatory Commission, Washington, D.C.

Von Wright, G.H. 1968. An essay in deontic logic and the general theory of action. Acta Philosophica Fennica, 21.

WENRA 2013. Safety of new NPP designs. Study by Reactor Harmonization Working Group RHWG, Western European Nuclear Regulators' Association.

Appendix A. DIGREL model example

A.1 Probabilistic assessment of defence-in-depth

In the previous NKS DIGREL project an example PSA model was developed representing a fictive BWR NPP with four redundant safety systems and a diversified four redundant reactor protection system. The assumed design of the plant and the PSA model is described in the report NKS-330, see especially appendix A of the report (Authén et al. 2015).

The PSA model covers level 1 PSA, i.e., the end state "core damage", and includes four initiating events:

- Large Loca (ALOCA)
- Loss of main feedwater (LMFW)
- Loss of offsite power (LOOP)
- General transient (TRAN).

Event trees are shown in the figures below. Consequence "CD" refer to core damage, "CD1" to core damage due to failed reactivity control, "CD2" to core damage due to failed core cooling", and "CD3" to core damage due to failed residual heat removal.

Large Loss Of Coolant Accident	Reactor Scram	Emergency Core Cooling	Residual Heat Removal				
ALOCA	C	V	W1	No.	Freq.	Conseq.	Code
				1	1,00E-04	OK	
				2	8,48E-09	CD,CD3	W1
				3	1,46E-08	CD,CD2	V
				4		CD,CD1	C

Figure A-1. DIGREL example model event tree for large LOCA (ALOCA).

Loss of Main Feed Water	Reactor Scram	Emergency Feed Water	Depressurization	Emergency Core Cooling	Residual Heat Removal	Filtered Containment Venting				
LMFW	C	U	X	V	W1	W3	No.	Freq.	Conseq.	Code
							1	5,00E-01	OK	
							2	5,03E-05	OK	W1
							3		CD,CD3	W1-W3
							4	8,67E-05	OK	U
							5	7,49E-09	CD,CD3	U-W1
							6	3,17E-08	CD,CD2	U-V
							7	9,47E-07	CD,CD2	U-X
							8	3,16E-11	CD,CD1	C

Figure A-2. DIGREL example model event tree for loss of main feedwater (LMFW).

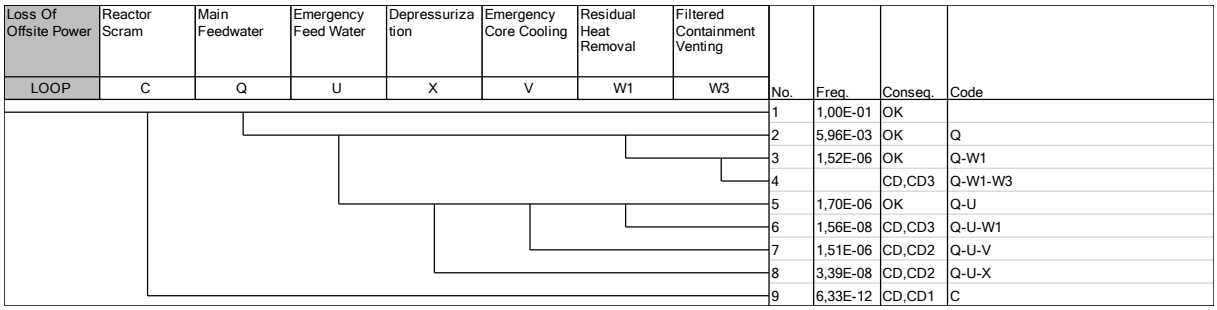


Figure A-3. DIGREL example model event tree for loss of offsite power (LOOP).

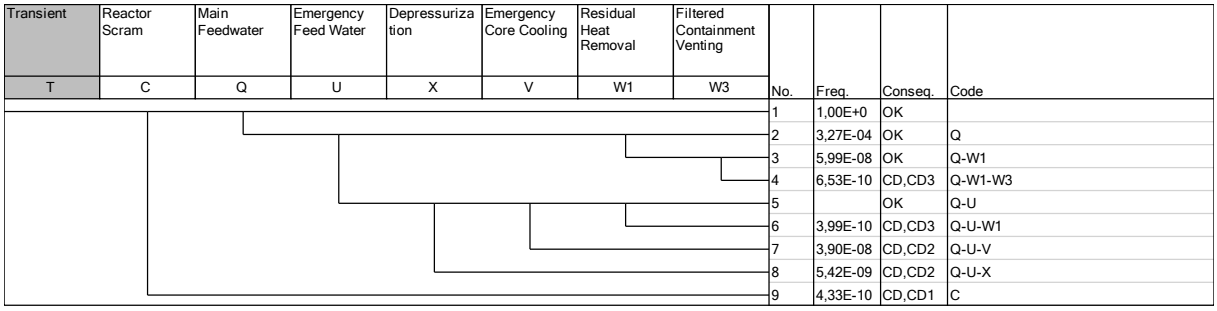


Figure A-4. DIGREL example model event tree for general transient (TRAN).

The model includes three safety functions corresponding with the CD categories CD1, CD2, CD3: reactivity control, core cooling and residual heat removal. Reactivity control has been modelled simply to cover only the automatic actuation of the reactor scram by the redundant subsystems of the reactor protection system: RPS-A and RPS-B (Figure A-6). Reactor scram is successful if RPS-A or RPS-B generates the scram signal. Conditions for actuation of scram are specific to initiating event categories, and 2-o-o-4 logic is applied.

Table A-1 presents the success criteria for the safety functions core cooling and residual heat removal per each initiating event. Three systems are available for the core cooling: main feedwater system, emergency feedwater system and emergency core cooling system. Two systems are available for the residual heat removal: the residual heat removal cooling via the condensation pool and filtered venting, which can be used when the core cooling happens from an external water source (feed-and-bleed operation).

Table A-1. Success criteria for core cooling and residual heat removal

<table>
<tr><td colspan="2">Safety function</td><td colspan="3">Core cooling</td><td colspan="2">Residual heat removal</td></tr>
<tr><td colspan="2">System</td><td>Main feedwater</td><td>Emergency feedwater</td><td>Emergency core cooling</td><td>Residual heat removal</td><td>Filtered cont. venting</td></tr>
<tr><td rowspan="4">IE</td><td>ALOCA</td><td rowspan="2">No credit</td><td>No credit</td><td>1-o-o-4</td><td rowspan="3">1-o-o-4*</td><td>No credit</td></tr>
<tr><td>LMFW</td><td rowspan="3">1-o-o-4*</td><td rowspan="3">1-o-o-4*, depressurisation 4-o-o-8 valves</td><td rowspan="3">1 line</td></tr>
<tr><td>LOOP*</td><td>2-o-o-3*</td></tr>
<tr><td>TRAN</td><td colspan="2"></td></tr>
<tr><td colspan="2">Actuation</td><td>RPS-A</td><td>RPS-B</td><td>RPS-A</td><td>RPS-A</td><td rowspan="3">passive system</td></tr>
<tr><td colspan="2">*LOOP power supply</td><td>Gas turbine</td><td colspan="3">Gas turbine or emergency diesel generator</td></tr>
<tr><td colspan="2">Component cooling</td><td>Not modelled</td><td>Air cooler</td><td colspan="2">Component cooling water system</td></tr>
</table>

For time being, the model can only be used to analyse defence-in-depth levels 1–3, and in fact only systems belonging to DiD level 3 are modelled in detail. The model includes some simple reliability models for systems belonging to DiD levels 1 and 2 (main feedwater, gas turbine), which also has some role as functions of "control of accidents within design basis" although the system do not belong same safety class DiD level systems.

Since this is only a level 1 PSA, the probabilistic evaluation is simple: CDF of the fictive model with fictive data is 3,2E-6 per year, which satisfies the usual CDF criterion 1E-5 per year (Bengtsson et al. 2011). Here we assume that the fictive model is complete (it covers all initiating event categories and all plant operating modes).

There are number of ways of further analysing risk importances of the design. In this example, the following elements are assessed:

- contribution of various core damage categories
- importance of initiating events
- importance of systems

Importance of components of I&C systems have been analysed in (Authen et al. 2015).

It should be noted that there are no strict numerical criteria when analysing risk importances. Risk importances give a relative ranking with respect to different reliability features of the items. The result of the assessment is that some items can be considered more important than others, but it cannot be said e.g. what is acceptable or not acceptable.

Importances of core damage categories and initiating events are shown in Figure A-5. Core damage sequences are grouped into three categories based on failed safety function: reactivity control (CD1), core cooling (CD2), and residual heat removal (CD3). The failure of core cooling contributes most to CDF, the failure of residual heat removal is the second, and the failure of reactivity control has insignificant contribution. This risk information can be used to conclude that core cooling is maybe the relatively speaking weakest safety function and reactivity control is strongest, with respect to the defence-in-depth level 3.

In the left hand side diagram, results from each initiating event category are plotted in an initiating event frequency – conditional core damage probability scatter plot. Initiating events LOOP and LMFW contribute most to CDF, and ALOCA and TRAN have a little contribution. Y-axis values show the margin to core damage given an initiating event. ALOCA has clearly smallest margin (least barrier), and LOOP has second smallest margin.

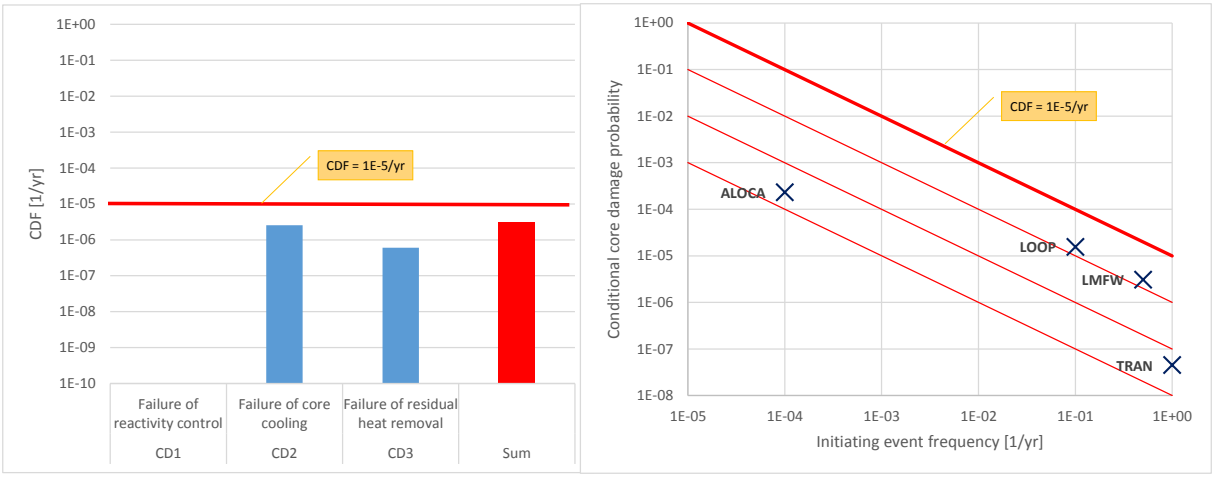


Figure A-5. Importances of the core damage and initiating event categories in the fictive DIGREL PSA model.

A typical way to assess system importances is to look at risk importances of basic events associated with a system. Sum of fractional contributions is a popular risk metric. A fractional contribution of a basic event represents how much the minimal cut sets including the basic event contribute to the top event probability (CDF). Equivalently, the fractional contribution expresses how much the top event probability decreases when the basic event probability is set 0. Fractional contribution gets values between 0 % and 100 %. For a system, the fractional contribution is usually calculated as a sum of basic events' contributions, which is not exactly correct, but it is sufficient to get a qualitative ranking (e.g. high importance, medium importance, low importance).

Table A-2 presents the ranking of the systems based on the sum of fractional contributions (FC) of the system related basic events. AC power system is most important. This includes for instance diesel generators and gas turbine, which are critical components in case of loss of offsite power (initiating event LOOP or consequential LOOP in case of other initiating events).

Table A-2 also presents a qualitative ranking using ranges 10−100% for high importance, 1−10% for medium importance, 0.1−1% for low importance and <0.1% for very low importance. The scale is quite arbitrary but it demonstrates a simple way classification of systems according to their risk importance.

Table A-2. Ranking of the systems based on the sum of fractional contributions (FC) of the system related basic events (excluding possible contribution via initiating events).

ID	Description	FC	Qualitative ranking
ACP	AC power system	5,1E-1	High importance
ADS	Automatic depressurisation system	2,8E-1	
RPS-B	Reactor protection system B	1,9E-1	
FCV	Filtered containment venting system	1,8E-1	
RHR	Residual heat removal system	1,6E-1	
EFW	Emergency feedwater system	1,1E-1	
RPS-A	Reactor protection system A	2,7E-2	Medium importance
SWS	Service water system	2,6E-2	
ECC	Emergency core cooling system	1,3E-2	
RPV	Reactor pressure vessel instrumentation	8,4E-3	Low importance
DWS	Demineralized water system	6,1E-3	
HVA	Room cooling system	4,3E-3	
CCW	Component cooling water system	6,5E-4	Very low importance
CPO	Condensation pool	1,8E-4	
DCP	DC power system	1,3E-4	
MFW	Main feedwater system	9,7E-5	
RCO	Condensation pool instrumentation	2,2E-7	

It should be noted that this kind of analysis of system importances may require further processing of the PSA-model, since e.g. system failures related to initiating events are usually not indicated and the human errors can or cannot be associated with some particular system. For instance in this example, the initiating event LMFW is related to the main feedwater system, but is not necessarily fully caused by failures of the system (MFW system importance in Table A-3 does not include this contribution). If there is an interest to analyse systems involved in defence-in-depth levels 1 and 2, the PSA-model should be made more detailed in this respect.

A.2 Probabilistic assessment of defence-in-depth

In the DIGREL example, there are four initiating events of which LMFW, LOOP and TRAN can be classified as operational transients and ALOCA is a design basis accident. In the Finnish regulation (STUK 2013a), LMFW, LOOP and TRAN belong the design basis category DBC2, and ALOCA is DBC4.[1]

For DBC2–4 events it must be, e.g., shown that "*it shall be possible to accomplish decay heat removal from the reactor and containment by one or several systems that jointly meet the (N+2) failure criterion…*" (N+2) failure criterion means that "*it must be possible to perform a safety function even if any single component designed for the function fails and any other component or part of a redundant system – or a component of an auxiliary system necessary for its operation – is simultaneously out of operation due to repair or maintenance.*" This assessment can be made excluding common cause failures from the model, and in this

[1] DBC2 = anticipated operational occurrence; DBC3 = Class 1 postulated accidents, which can be assumed to occur less frequently than once over a span of one hundred operating years, but at least once over a span of one thousand operating years; DBC4 = Class 2 postulated accidents, which can be assumed to occur less frequently than once during any one thousand operating years; DEC-A = Design extension condition, where an anticipated operational occurrence or class 1 postulated accident involves a common cause failure in a system required to execute a safety function. (STUK 2013a)

example model, this requirement is fulfilled. There are no such minimal cut sets, since the front-line safety systems and their support systems are four-redundant.

For DEC-A events (DBC2 or DBC3 and a common cause failure in a system required to execute a safety function), it must be shown that there is a diverse N+1 system to reach a safe state. In the example case, this means that the initiating events LMFW, LOOP and TRAN combined with a CCF must be examined. A large number of minimal cut sets can be found not fulfilling the criteria, but then it depends on how the criterion is actually interpreted for cases like:

- which software CCF should be counted as a common cause failure, e.g., system software CCF or CCF between nearly identical application software modules?
- can a system requiring an operator action be (N+1) failure tolerant?
- shall the (N+1) failure criterion be applied to structures like the demineralized water tank?
- can recovery be accounted for the LOOP, i.e., could LOOP be classified into different categories depending on the duration of the LOOP (only short time LOOPs would be DBC2)?

A.3 Analysis of spurious actuations

In the DIGREL model, spurious actuations are modelled both for failures of running components to stop and failures of I&C to cause spurious actuation signals. These failure modes are considered as so called mission time failures which may cause failure of a safety function after an initiating event. Common cause initiators caused by spurious failures are not considered in the example model, which will be an issue for the further development of the model.

The architecture of the safety I&C is presented in Figure A-6. The protection system is divided into two subsystems, called RPS-A and RPS-B. In addition to the APU:s and VU:s, the I&C architecture includes an I&C unit for operator actions, abbreviated by MU. MU is relevant for the manual actuation of the primary circuit depressurization and manual actuation signal of main feedwater pumps. See (Authén et al. 2015) for the complete description of the example.

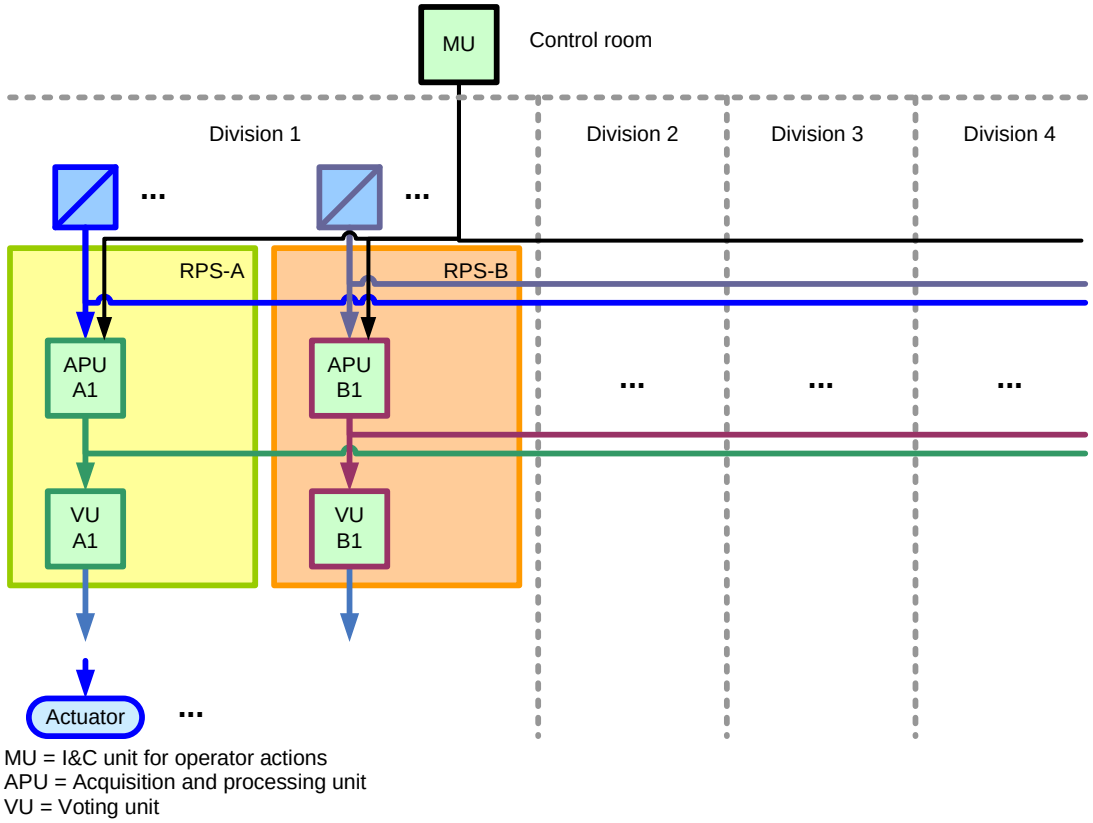


Figure A-6. DIGREL example I&C architecture.

Impacts of I&C system failures should be analysed both from the actuator perspective (which signals are required for the function and which may hinder the function) and from the I&C unit perspective (what are the effects of various module level failures modes). In the assessment, the fault tolerance features of the design must be taken into account, since they will determine the output in case of a detected failure. Fault processing is implemented in the design of the hardware circuits and the software logic, and it can be defined on a case-by-case basis how the logic shall react if invalid input signals are present, and how output signals shall be set in case of faulty logic signals. In general, the following applies for detected failures of the example I&C protection system:

- Detected failure in input signals, in intra I&C unit signal processing or in inter I&C unit signal exchange will cause corresponding signals to be replaced by a default value of 0 or 1.
- Complete, or fatal, failure of an I&C unit, e.g. processor failure or power supply failure, will cause all output channels of the I&C unit to 0 and controlled actuators will go to the predefined fail-safe state.

There are different solutions for voting applied in the safety I&C system for actuation signals to the actuators:

- Hardwired 2/4 voting by relays or pilot valves (e.g. scram)
- Software 2/4 voting of I&C units possible treatment of degraded voting logic

The fail-safe actions are separately defined for each I&C function and for each actuation signal. I&C functions using the same inputs, may apply different default values and different types of voting logic.

As an example, the emergency feedwater system (EFW) pump has the failure modes failure to start and spurious stop with respect to the core cooling safety function. With respect to the overpressure protection (primary circuit integrity) spurious start or failure to stop are critical failure modes, but this safety function has not been considered in the DIGREL example (so far). Failure modes of the EFW system are analysed in the Table A-5 to A-9 of the report NKS-330 in the following order:

- Table A-5: failure modes, causes and functional effect of the actuators (pumps and valves)
- Table A-6: failure modes and causes of voting units (VU) and acquisition and processing units (APU) controlling the actuators of the EFW system (single failures)
- Table A-7: functional failure effects of CCF between redundant VU:s, APU:s and communication links (both detected and undetected failures are considered to account for the fail-safe principles)
- Table A-8: generic failure modes of hardware modules included in VU:s and APU:s
- Table A-9: failure modes and effects on software module failures (system software, data communication software and various application software modules related to I&C functions).

In the end of appendix A of the report NKS-330, an example extract from the fault tree model structure is given. The example demonstrates an example path of the modelling of dependencies starting from the actuator failure down to sensor failures. The logic of the fault tree model follows the failure modes and effects analysed in Tables A-5 to A-9.

It should be noted that "spurious actuations" are not specifically addressed in this analysis, but they are part of the normal analysis of various possible failure modes. Therefore it is important to ensure the completeness of the failure modes considered and that fault tolerance principles of the I&C are properly accounted for.

Another important judgement is how the failure detection is determined for various failure modes, in particular related to the hardware modules which are the smallest entities considered in the analysis. For software modules, the taxonomy of dividing failures into fatal and resp. non-fatal failures resolves the issue of failure detection. Failure detection is widely discussed in the report NKS-330, and e.g., Table A-10 lists how the detection coverage is considered for various hardware modules. Generally, it can be said that spurious actuations are related to the failures detected by on-line monitoring followed by a "fail-safe" actuation or they are related to wrong input signals, which can be classified as self-revealing failures. Latent hardware module failures will be detected by a failure per demand or test.

The impact of spurious signals from the RPS on the CDF have been studied by use of the DIGREL model. For the purpose of the evaluation a RPS configuration with default value 1, i.e. actuation, at occurrence of detected failures was chosen. See NKS-330 for more details on the DIGREL model.

The evaluation indicates a large contribution from spurious RPS signals, in fact, the contribution of 11% from the digital RPS to the CDF is to 95% due to failure modes causing spurious signals. The contribution from software failures causing spurious outputs is insignificant, though due to the fail-safe actions taken at detected software failures e.g. in input signals, in intra I&C unit signal processing or in inter I&C unit signal exchange, software failures causing loss of output have a significant contribution to spurious signalling and also contributes with 5% to the CDF. The remaining 6% contribution to the CDF is due to detected hardware failures causing loss of I&C functions which due to the incorporated fail-safe design are replaced by default values of 1, with spurious actuations as a result.

In summary, the fail-safe design is the cause of more than 99,9% of the spurious actuations that may occur, while spurious behaviour of hardware and software causing spurious actuations is insignificant. The DIGREL model concurrently shows a lower CDF when an RPS configuration with default value 0 is applied.

Appendix B. SICA analysis of fictive software modules

This appendix presents software logic diagrams of a fictive reactor protection system. The modules are analysed using the SICA method. The overall module structure of the example is presented in Figure B-1. All the modules of the example system are not presented in figures and included in the analysis because some of them are very similar. Descriptions of the function blocks of the examples are presented in Table B-1. The example system including diagrams and descriptions was created by Jussi Lahtinen, VTT.

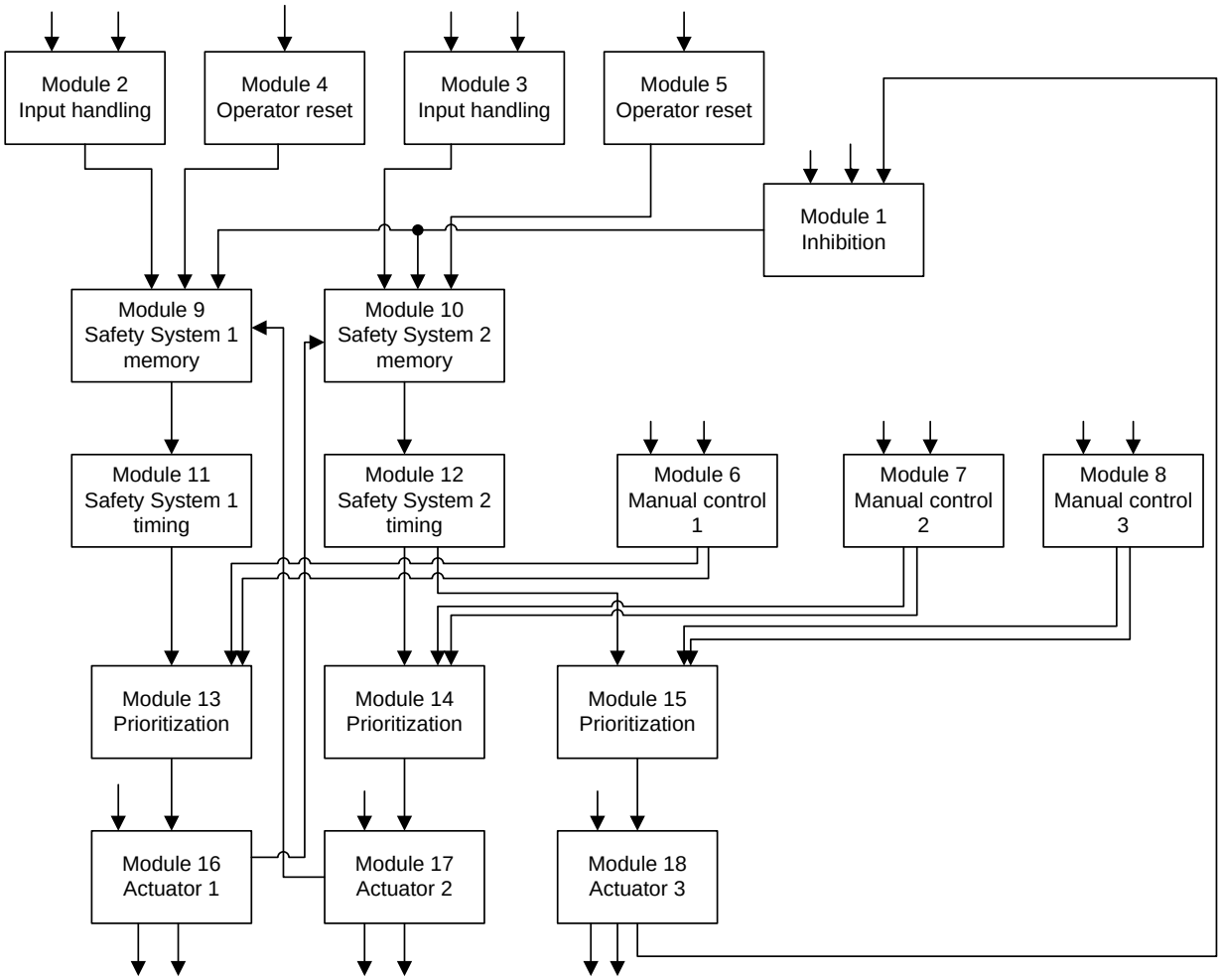


Figure B-1. Module structure of the example system.

Table B-1. Descriptions of the function blocks.

Function block	Description
COMPLIMIT	Analog signal is transformed into a binary value based on a limit value.
TON	The output is set to 1 after the input has been 1 for the specified time.
AND	The output is 1 only if all inputs are 1.
OR	The output is 1 if any of the inputs is 1.
VOTE	The output is 1 if sufficient number of inputs are 1.
TOF	The output turns to 1 if the input turns to 1. The output stays in value 1 for the specified time after the output has turned to 0.
PULSE	The output is set to 1 for the specified time when the input turns to 1. After the specified time, the output turns to 0 regardless of the input value.
COMP	The output is set to 1 if the input is the specified value.
Set Reset	The output turns to 1 if the set input turns to 1, and the output turns to 0 if the reset input turns to 1.
MEMORY	The output is 0 if the input has always been 0, otherwise the output is 1.
DELAY	The output equals to the input value with the specified delay.
NOT	The output is negated input value.

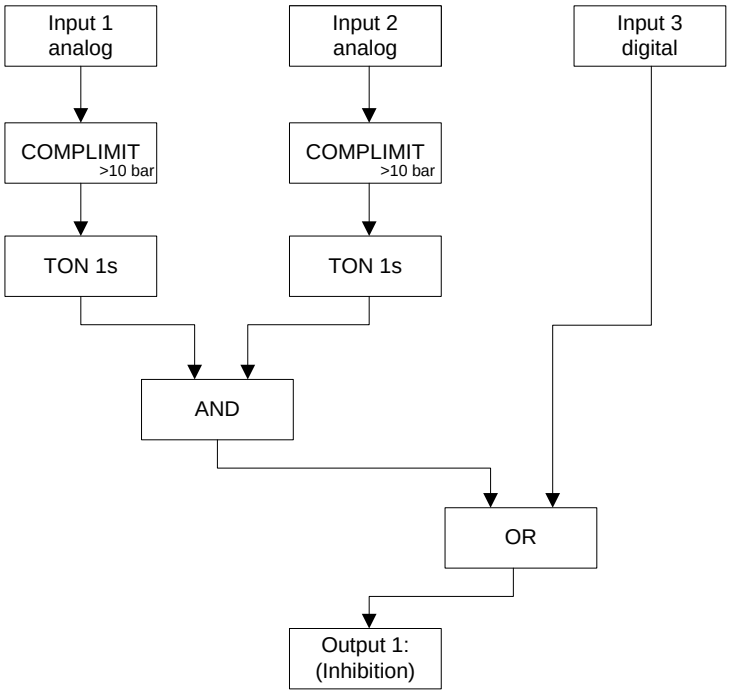


Figure B-2. Module 1. Inhibition. The idea is to inhibit the safety functions when the measurements (inputs) indicate that there is a failure in the safety actuator equipment.

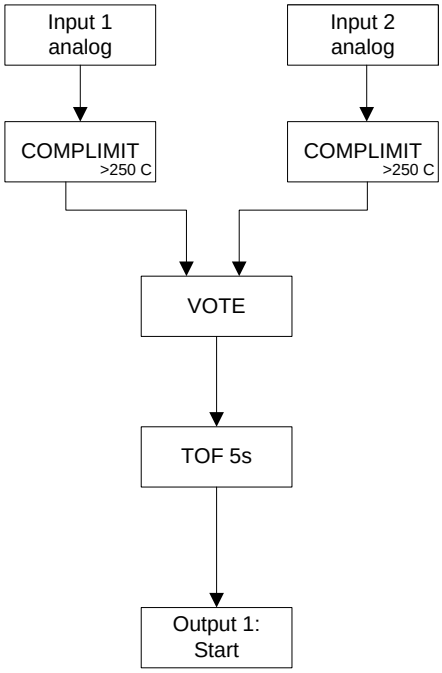


Figure B-3. Module 2. Input processing. Inputs are two redundant temperature measurements. If they are above 250 ºC, the safety function is started.

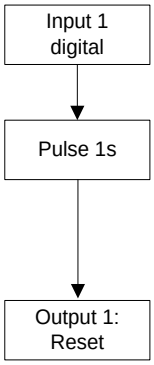


Figure B-4. Module 4. Operator reset. After the safety function has been started, the command is memorized (flipflops in modules 9 and 10). When the demand for the safety function disappears, the operator can reset the flipflops manually. This results in a short pulse to the reset inputs of the flipflops.

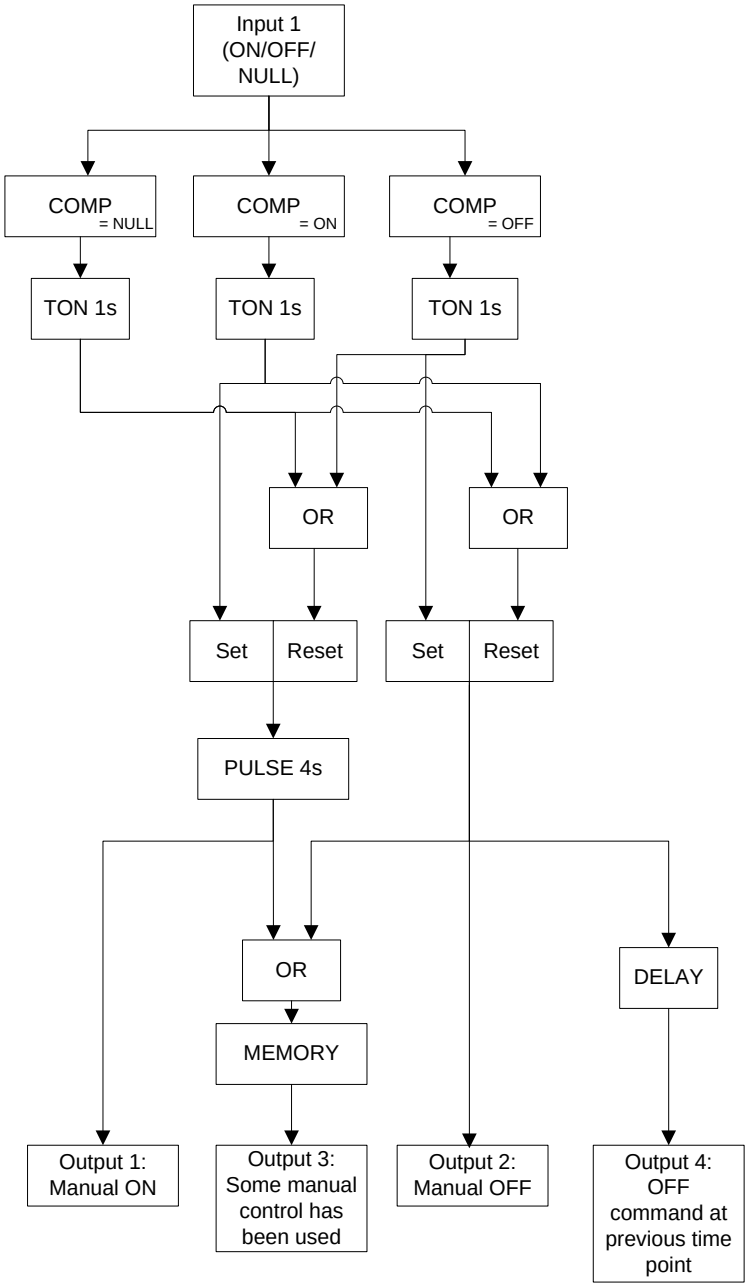


Figure B-5. Module 6. Manual commands. Operator manual command (ON/OFF/no command) is given using a device with three possible positions. The module controls the actuator in module 16. This actuator shall not receive too long starting signals. The starting signal length is limited by using the 4 s PULSE function block.

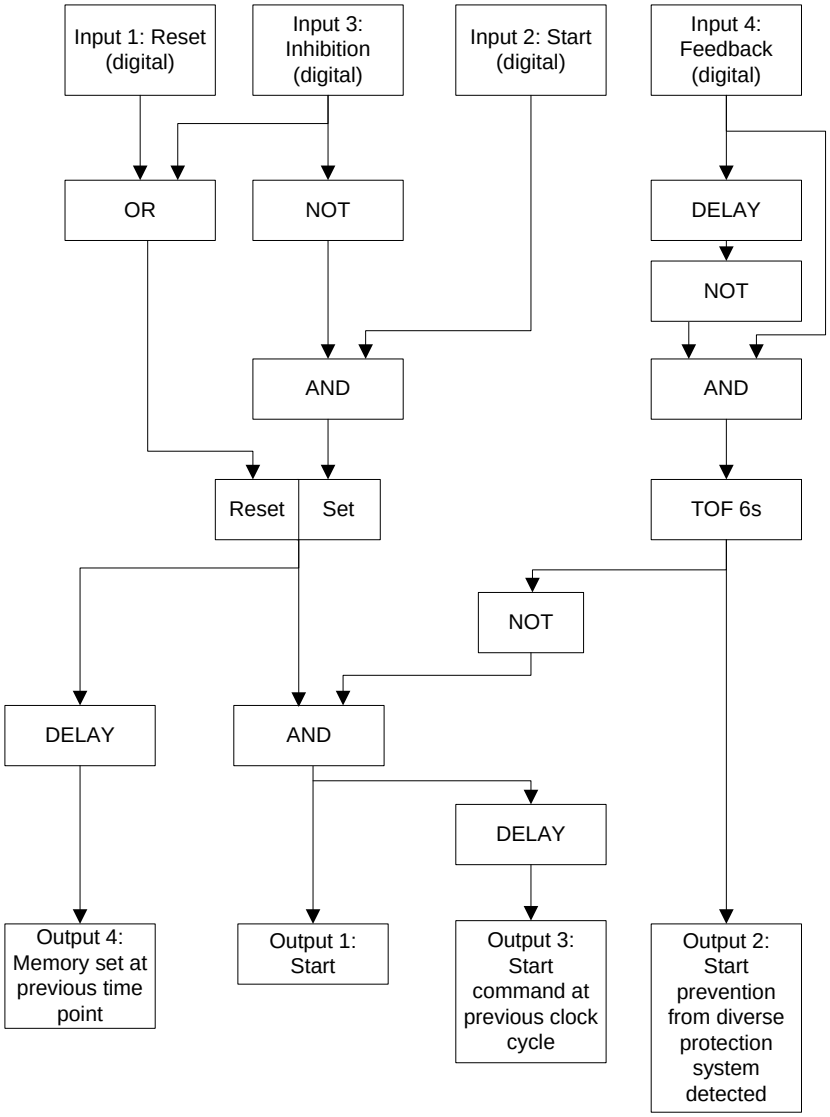


Figure B-6. Module 9. Safety system 1 memorization. The inhibition signal prevents a new start even though it is demanded. The idea with the feedback is that if the diverse safety system is starting up, this system is prevented from starting. However, if the start command is already being given, the starting sequence is not interrupted.

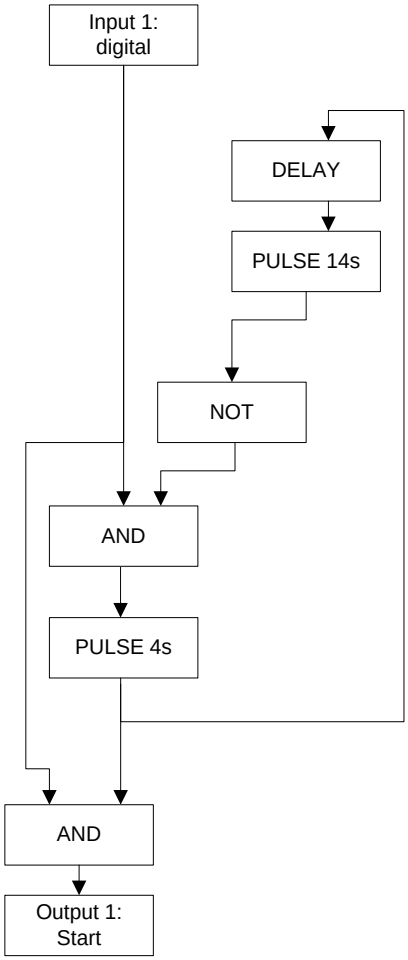


Figure B-7. Module 11. Timing of the safety system 1. When the safety function is started, this timing module outputs 4 s pulses with 10 s resting time between the pulses.

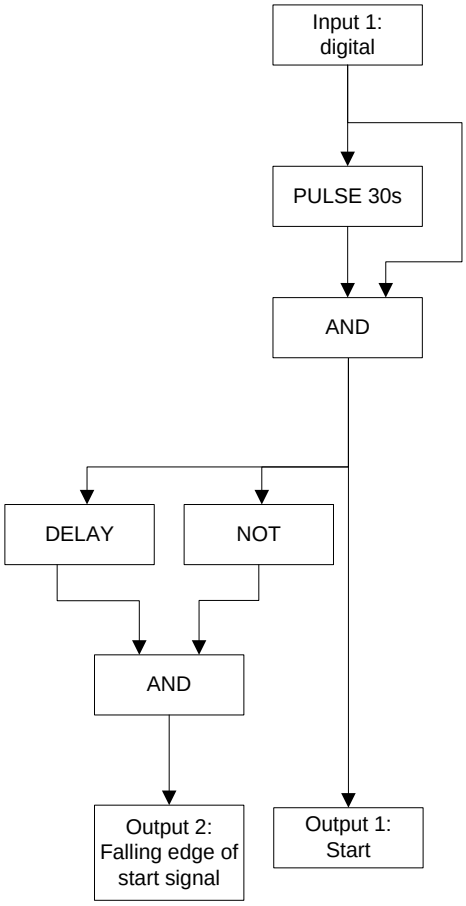


Figure B-8. Module 12. Timing related to safety system 2 start. Whenever a rising edge is detected, a 30 s long pulse is given as output to start the device.

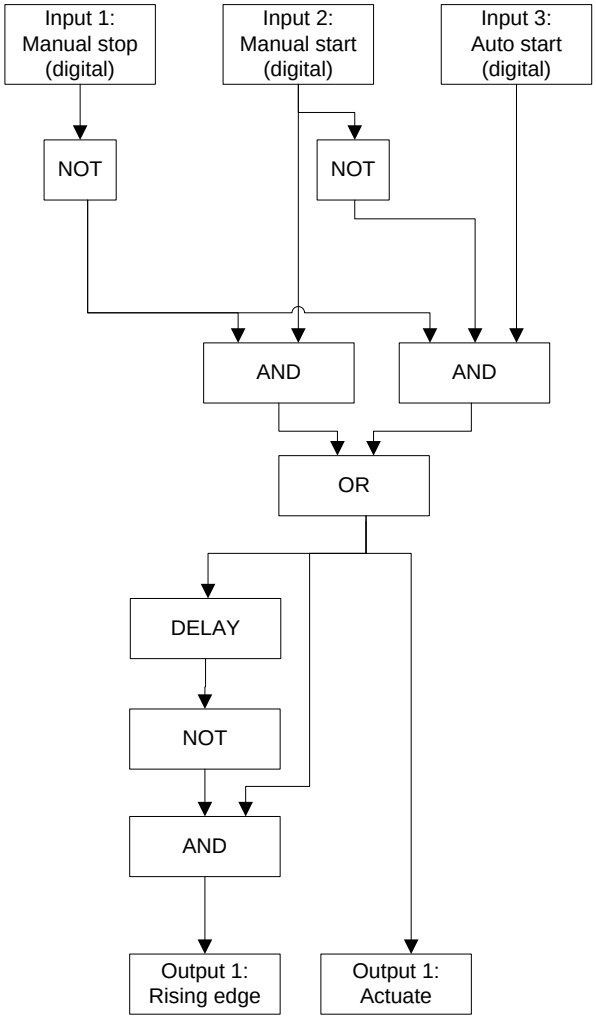


Figure B-9. Module 13. Signal prioritization. Prioritization function has three inputs: stop command from operator (manual STOP), start command from operator (manual START), and start signal from automation. The priority order of these signals is: 1. Manual STOP (highest), 2. Manual START, 3. AUTO START.

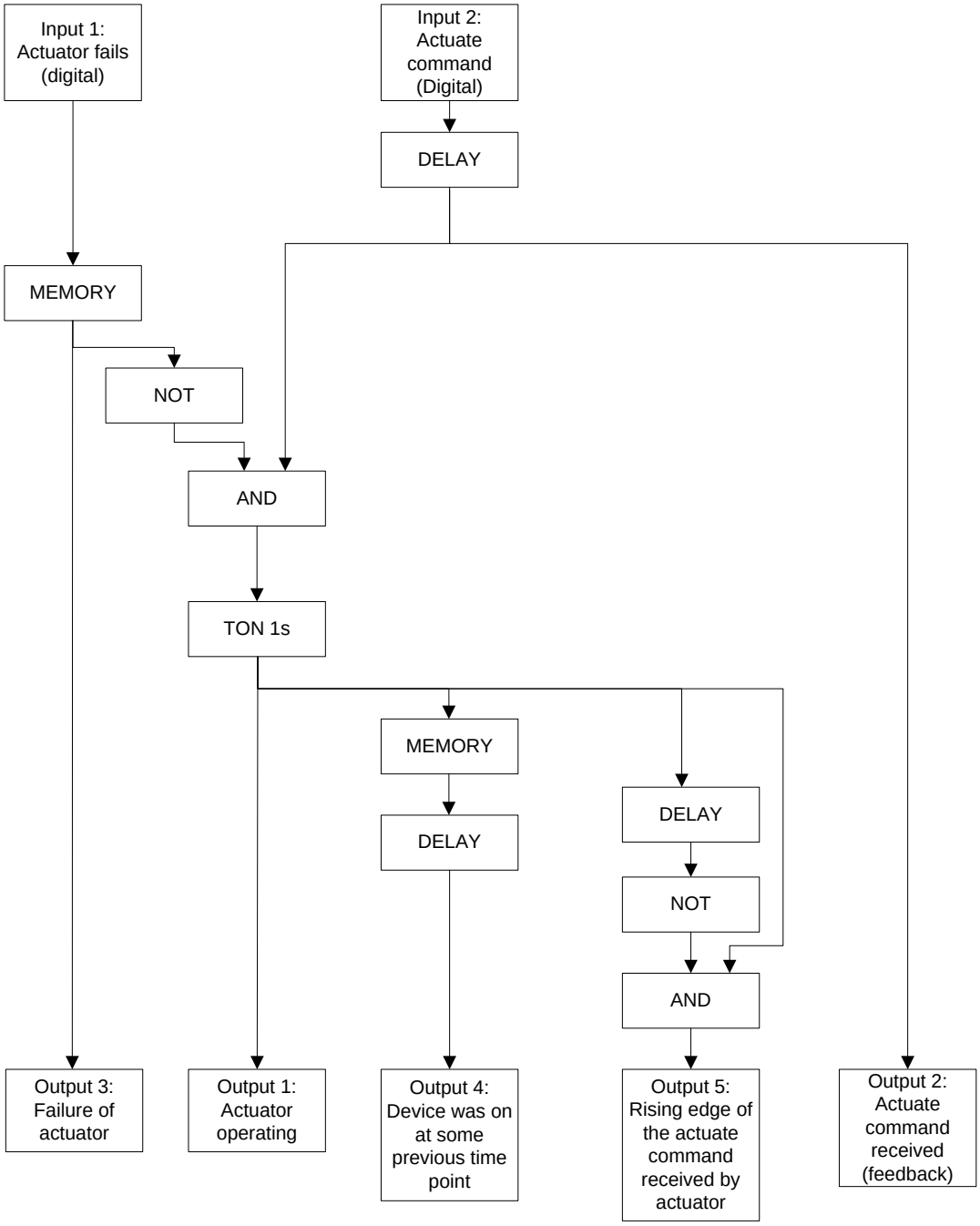


Figure B-10. Module 16. Actuator. The received actuation command is output as a feedback to the diverse safety system. The actuator takes 1 s to start operating. If the actuation command is received for 1 s, the output 1 is set. If the actuator fails, output 1 is set to 0.

Table B-2. SICA analysis results. Red colour indicates that a limit value is exceeded.

Module	The number of feedback loops	The maximum number of connected complex function blocks	The number of inputs and outputs	The maximum number of connected function blocks	Complexity category
1	0	2	4	6	Low
2	0	1	3	4	Low
4	0	1	2	1	Low
6	0	7	5	13	Medium
9	0	4	8	11	Medium
11	1	3	2	6	Medium
12	0	2	3	5	Low
13	0	1	5	8	Low
16	0	5	7	8	Medium

Complex function blocks of the examples are:

- TON
- TOF
- PULSE
- Set/Reset
- MEMORY
- DELAY

These function blocks are complex because they use internal memory.

Appendix C. Test of homogeneity based on statistical data

The objective with this appendix is to demonstrate that pooling of operational experience data for per demand failures is not justifiable from a statistical point of view when the experience is from systems with different operational profile. Specifically, the problem is that no failures have been observed.

In (Persson Sunde, 2012) a method is described that is used to test if a group of components are homogenous based on the number of failures and demands and/or stand-by time. When the objective is to test if the components in a group are homogenous with regard to their respective number of failures and demands it would yield a homogenous test where the following null hypothesis is tested; "all the components have the same failure probability".

Following the method in (Persson Sunde, 2012) it is then assumed that the expected number of failures for component $\boldsymbol{i}$ is $\boldsymbol{E_i = \frac{N}{T} t_i}$, where $\boldsymbol{T}$ is the total number of demands, $\boldsymbol{N}$ is the total number of failures and $\boldsymbol{t_i}$ is the number of demands for component $\boldsymbol{i}$. This corresponds to the assumption that all components are assumed to have the same failure probability, $\boldsymbol{\frac{N}{T}}$. The total number of observed failures, $\boldsymbol{N}$, is assumed to be multinomial distributed over the components in the group.

To test the null hypothesis a test quantity $\boldsymbol{Q}$ is used;

$$Q = \sum_i \frac{(O_i - E_i)^2}{E_i}. \qquad \text{(C-1)}$$

Where O_i is the observed number of failures for component i. It is common to approximate Q with a Chi-Square distribution when there is a large number of observations. When that is not the case, the work around used in (Persson Sunde, 2012) is to perform Monte Carlo simulations to calculate the expected distribution of Q given N failures. From the cumulative distribution of Q, $CDF(Q)$, it is identified if the observed Q-value, Q_{obs}, is significant based on the chosen significance level, usually set to 0.05. If $1 - CDF(Q_{obs})$ is smaller than 0.05 the homogenous test indicates that the components in the group are inhomogeneous on a 95% significance level. Hence, the null hypothesis of all components in the group having the same failure probability is not likely given the observed failures and demands.

Example

From the TXS experience, with regard to NSA failures following operational history was collected:

1. D_{RPS} = 3.4E+3
2. D_{RLS} = 2.4E+3
3. D_{RCS} = 7.0E+6.

Until end of 2013 no NSA application software failures had been observed. Since no failures are identified, it will not be possible to state whether or not these groups of functions are homogeneous. So instead we are assuming that we would have observed 1 failure in each group following then following results are obtained:

Table C-1. Test of homogeneity based on statistical data.

Group	Demands	Comment
D_{RPS}, D_{RLS}, D_{RCS}	3 400; 2 400; 7 000 000	Not homogeneous
D_{RPS}, D_{RLS}	3 400; 2 400	Homogeneous
D_{RPS}, D_{RCS}	3 400; 7 000 000	Not homogeneous
D_{RLS}, D_{RCS}	2 400; 7 000 000	Not homogeneous

Since the data available are very scarce, the above does not claim to show homogeneity. The purpose is merely to show that we should not assume homogeneity between these groups, without further justification.

Title	Modelling of DIgital I&C, MODIG — Interim report 2015
Author(s)	Stefan Authén[1], Ola Bäckström[2], Jan-Erik Holmberg[3], Markus Porthin[4], Tero Tyrväinen[4]
Affiliation(s)	[1]Risk Pilot AB, [2]Lloyds Register Consulting, [3]Risk Pilot AB Suomen sivuliike, [4]VTT Technical Research Centre of Finland Ltd
ISBN	978-87-7893-445-1
Date	March 2016
Project	Modelling of DIgital I&C (MODIG)
No. of pages	81
No. of tables	13
No. of illustrations	31
No. of references	55

Abstract
max. 2000 characters

The NKS-project MODIG (MODelling of DIGital I&C) aims to get a consensus approach for a reliability analysis of a plant design with digital I&C, improved integration of probabilistic and deterministic approaches in the licensing of digital I&C, improved failure data collection including software failure probability quantification, and a practical application of probabilistic safety assessment (PSA).

A survey of the defence-in-depth (DiD) framework and PSA's role in it has been made. The assessment of DiD and diversity is in principle straightforward for PSA, e.g., risk metrics can be used to evaluate DiD levels 3, 4 and 5. A PSA model always includes uncertainties, which needs to be accounted for especially when comparing with deterministic assessment. Regarding digital I&C, the focus of the assessment is on the DiD levels 1, 2 and 3. In addition the logic model of PSA can be used in the assessment of deterministic failure criteria.

Spurious actuation is a functional failure mode when a component performs a function without a real demand. Spurious actuations are of special interest for I&C due to complex effects via system dependences and due to a huge number of possible failure locations. There is a need to develop a reasonable but comprehensive approach both for deterministic and probabilistic analyses. Analysis requirements have been compiled, and a generic failure modes taxonomy and an analysis approach have been outlined.

The software reliability task has been working on the confidence building in the method to estimate application software failure probability. The impact of pooling data from high and low demand systems is discussed. The principle of the probability estimation has

been adjusted from the approach developed in the DIGREL project. A solution for the software complexity assessment has been prepared.

I&C failure data is one of the information sources needed for the assessment of I&C reliability. Vendors have data sources as they typically have access to experience data from many plants, have needed insight on the software development processes and are capable to analyse the causes of the detected failures. International collaboration and discussions are still needed in order to forward the use of I&C failure data in PSA.

Key words Digital I&C system, probabilistic safety assessment, reliability, nuclear power plant safety