MDEP Generic Common Position

Multinational Design Evaluation Programme
Generic Common Position
DICWG No6 – PUBLIC USE

Date: 13 March 2013
Validity: **until next update or archiving**
Version C

MDEP Generic Common Position No DICWG-06

Related to : Digital Instrumentation and Controls Working Group activities

COMMON POSITION ON PRINCIPLE ON SIMPLICITY IN DESIGN

Participation

Countries involved in the MDEP working group discussions:	Canada, Finland, France, India, Japan, People's Republic of China, Republic of Korea, Russian Federation, South Africa, the U.A.E., the U.K. and the U.S.
Countries which support the present common position	Canada, Finland, France, India, Japan, People's Republic of China, Republic of Korea, Russian Federation, South Africa, the U.A.E., the U.K. and the U.S.
Countries with no objection:	
Countries which disagree	
Compatible with existing IAEA related documents	Yes

Multinational Design Evaluation Programme

Digital Instrumentation and Controls Working Group

GENERIC COMMON POSITION DICWG NO6: COMMON POSITION ON PRINCIPLE ON SIMPLICITY IN DESIGN

Summary

The Digital Instrumentation and Controls Working Group (DICWG) has agreed that a common position on this topic is warranted given the increase of use of Digital I&C in new reactor designs, its safety implications, and the need to develop a common understanding from the perspectives of regulatory authorities. This action follows the DICWG examination of the regulatory requirements of the participating members and of relevant industry standards and IAEA documents. The DICWG proposes a common position based on its recent experience with the new reactor application reviews and operating plant issues[1].

Context

The use of digital technology typically allows the achievement of more complex functionality. This increase in functionality can accommodate both essential and non-essential functions associated with safety. Although the increased functionality can result in benefits, the increased complexity can also have negative effects. Requirements that are unnecessary or that specify unnecessarily stringent performance criteria cause extra work and add complexity. Complexity can generate additional faults in design, difficulty in detecting and correcting faults, introduction of failure modes and effects that are not present in simpler design, and challenge in demonstrating conformance to safety system design criteria such as independence, testability and reliability. It can also increase licensing uncertainty during the review by the regulatory authorities. The actual licensing experience by some of the regulatory authorities has shown that simplicity provides greater licensing certainty. This common position provides the agreed-upon principle of the MDEP DICWG member states on simplicity for the design of the digital systems of the highest classification. Other design principles (e.g., independence and redundancy) for essential safety functions should continue to be met as this common position is applied.

Definition of terms

Complexity:

1. The degree to which a system or system component has a design or implementation that is difficult to understand and verify

[1] The goal of MDEP is not to independently develop new regulatory standards. Common Positions are not legally binding and do not constitute additional obligations for the regulators or the licensees but are guidelines, recommendations, or assessments that the MDEP participants agree are good to highlight during their safety reviews of new reactors. Any MDEP member may decide to implement the common positions through its national regulatory process.

2. Pertaining to any set of structure-based metrics that measure the attribute in definition 1.

[IEEE Std 7-4.3.2 (2010), IEEE Std 610 (1990) and IEC 61513 (2011)]

Simplicity:

The degree to which a system or component has a design or implementation that is straightforward and easy to understand. Contrast with complexity. [IEEE Std 610 (1990)]

Generic Common Position on Treatment of Simplicity in Design:

1. Design of digital systems for the highest classification shall be as simple as practical.

2. All unnecessary complexity shall be avoided both in the functionality of the system and in its implementation.

3. All features should be demonstrated to be beneficial to safety in consideration of the impact of their added complexity to the design. This complexity cannot lead to violation of other design principles (for example, independence, redundancy, diversity).

References

NS-G-1.1, “Software for Computer Based Systems Important to Safety in Nuclear Power Plants,” 2000

NS-G-1.3, “Instrumentation and Control Systems Important to Safety in Nuclear Power Plants,” 2002

IEEE 610.12, “IEEE Standard Glossary of Software Engineering Terminology,” 1990

IEEE 7-4.3.2, “IEEE Standard Criteria for Digital Computers in Safety Systems of

Nuclear Power Generating Stations,” 2010

IEC 61513, “Nuclear power plants - Instrument and control for systems important to safety - General requirements for systems,” 2011

IEC 60880, “Nuclear power plants - Instrumentation and control systems important to safety - Software aspects for computer-based systems performing category A functions,” 2006

Four Party Report, “Four Party Regulatory Consensus Report on the Safety Case for Computer-Based Systems in Nuclear Power Plants”

Seven Party Report, “Licensing of safety critical software for nuclear reactors. Common position of seven European nuclear regulators and authorised technical support organizations”

Annex 1

Simplicity in design (IEC 60880)

The disadvantages introduced by diversity may include a greater overall complexity (extracted from 880)

Simplicity in design (IEC 61513)

The choice of system architecture may be restricted in order to limit the complexity to facilitate implementation of functions of high safety category (extracted from IEC 61513)

Simplicity in design (NS-G-1.1)

3.2. It should be demonstrated that all unnecessary complexity has been avoided both in the functionality of the system and in its implementation. This demonstration is important to safety and is not straightforward, as the use of digital programmable technology permits the achievement of more complex functionality. Evidence of obedience to a structured design, to a programming discipline and to coding rules should be part of this demonstration.

3.3. For safety systems, the functional requirements that are to be fulfilled by a computer system should all be essential to the achievement of safety functions; functions not essential to safety should be separated from and shown not to impact the safety functions.

3.4. For computer based system applications, top-down decomposition, levels of abstraction and modular structure are important concepts for coping with the problems of unavoidable complexity. They not only allow the system developer to tackle several smaller, more manageable problems, but also allow a more effective review by the verifier. The logic behind the system modularization and the definition of interfaces should be made as simple as possible (for example by applying ‘information hiding’ (see Section 3.3.4 of Ref. [4])).

3.5. In the design of system modules, simpler algorithms should be chosen over complex ones. Simplicity should not be sacrificed to achieve performance that is not required. The computer hardware used in safety systems should be specified with sufficient capacity and performance to prevent software from becoming too complex.

Simplicity in design (7 party report)

2.12.3.6 The *systems and software architecture* design shall have the minimum complexity commensurate with the design requirements.

2.2.3.8 It shall be ensured that the use of these fault tolerant, exception handling and hazard mitigating mechanisms is appropriate and that they do not introduce unnecessary complexity.

2.3.2.4 Despite all best endeavours to produce fault free software through good design practices and thorough testing, there is always the potential for unforeseen error conditions to arise. Therefore the technique of incorporating error checking (which may be based on formal assertions) into software is regarded as a sound policy. This technique is known as defensive programming. It should cover both internally and externally arising exceptions, without adding unnecessary complexity to the software.

Simplicity in design (4 party report)

5.1.3 Minimising faults in the design

(a) complexity avoidance;

5.2.4 System design principles

(b) avoidance of complexity, so far as is practicable, should be the guiding aim;