Assurance Case Driven Design for Internet of Things

VLADIMIR SKLYAR, VYACHESLAV KHARCHENKO
Department of Computer Systems and Networks
National Aerospace University "KhAI"
17, Chkalova Street, Kharkiv, 61070
UKRAINE
v.sklyar@csn.khai.edu, v.kharchenko@csn.khai.edu

Abstract: - Assurance (Security and Safety) Case is a proven-in-use methodology to demonstrate a system compliance with security and safety critical requirements. An advanced approach to improve Assurance Case is proposed in a view of Assurance Case Driven Design (AC DD). A practical using of AC DD lays in cost-effectiveness improvement of certification and licensing processes. We analyze basic mathematical models and methods to improve a known formal notation at the top level. As a result we develop Claim-Argument-Evidence-Criteria (CAEC) notation as well as Development-Verification&Validation-Assurance Case (DVA) notation for AC DD implementation. This approach is implemented for the Internet of Things (IoT). Low-energy informed assessment has to be added to the IoT Assurance Case. Assurance Case concept for the IoT safety critical applications is developed and demonstrated.

Key-Words: - Internet of Things, Assurance Case, Safety and Security Life Cycle

1 Introduction

A goal of security and safety analysis is not only proving a conformance with requirements but mostly discovering gaps in such conformance assessment approach [1,2]. Assurance (Security and Safety) Case methodology contains a potential for improvement safety and security analysis techniques and tools [3]. We name a set of Assurance Case based techniques and tools as Assurance Case Driven Design (AC DD). A practical using of AC DD lays in improvement of certification and licensing processes [4].

From this prospective Assurance Case may be implemented for the earliest stages of life cycle activities to drive safety implementation from the scratch [5].

The main motivation of AC DD is the following:

– To develop a technique to assess safety and security features as soon as possible during development of a system concept (specification, design);

– To develop a technique to develop a system concept (specification, design) in a safe and secure manner.

AC DC also supports the following important topics:

– Research of integral security and safety features of modern critical control and communication systems and networks as an integral property; security importance increasing requests implementation of security requirements as a part of licensing issues; such approach is named as Security Informed Safety Case [6]; such approach is targeted to analyze safety and security in a structured way and creating Security Informed Safety Case that provide justification of safety taking into particular consideration the impact of security [7];

– Research of different type of embedded components, such as Field Programmable Gates Arrays (FPGAs) and microprocessor units (MCUs), which are applicable for Internet of Things (IoT) solutions;

– Research applications for specific market, for example, cloud computing, big data analytics and IoT with high level requirements to safety, security and quality of service (QoS).

At the present Assurance Case methodology progress lays in multidisciplinary dissemination of theory and experience [8]. Experts form different area may develop a general and cross-platform security and safety assurance approaches. At the same time there are some potential areas for Assurance Case improvement, such as:

– Assurance Case should faster find gaps in compliance with requirements than demonstrate such compliance;

– It is reasonable to implement Assurance Case from the earliest stage of life cycle; one more reason to do it is a prospective idea to combine of Assurance Case with argument based design

approach, what is a basis for elimination a board between design and modeling;

– Assurance Case should provide as many details as it is needed for comprehensive analysis;

– Assurance Case should support re-using of system safety and security files during system operation and maintenance;

– Assurance Case should support cost effectiveness of system life cycle;

– It is reasonable to improve formalism of Assurance Case against empirics in descriptions.

Assurance Case has two side of description and implementation:

1) A static part which describes an approach to combine arguments for assurance support;

2) A dynamic part to support a static part movement between stages of analyzed system life cycle.

There are the following notations for the Assurance Case static part:

Basic Toulmin notation [9], developed by author as an extension for a classic implication operation;

Claim-Argument-Evidence (CAE) notation [10,11] and Goal Structuring Notation (GSN) [12] based on Toulmin notation.

CAE and GSN formalisms are based on classical set theory, graph theory and relation algebra. Such relations tracing allows us to propose extensions for existing notations. In this article we discuss an approach to develop Claim-Argument-Evidence-Criteria (CAEC) notation as an extension of CAE notation [4,8].

The second side of Assurance Case implementation is dynamic application via life cycle stages. IDEF0 notation is considered as a fundamental for a formal description. Application of set theory and graph theory has been considered as a basis for IDEF0 notation. It allows proposing Development – Verification & Validation – Assurance Case (DVA) notation for description of dynamic Assurance Case application.

2 IoT Reference Architecture and Implementation of Safety and Security Requirements

Requirements for IoT components [13,14] have been identified by different vendors, system integrators, consortia etc. IoT Reference Architecture (IoT-RA) is a subject of standardization, what is developing now by International Electrotechnical Commission (IEC) and Institute of Electrical and Electronics Engineers (IEEE). The IoT-RA should describe the system or systems for the IoT, the major components involved, the relationships between them, and their externally visible properties. IoT-RA is presented at Fig.1. Existing layers and interfaces are points to implement and to assess safety and security and low-power (“green”) solutions [15,16].

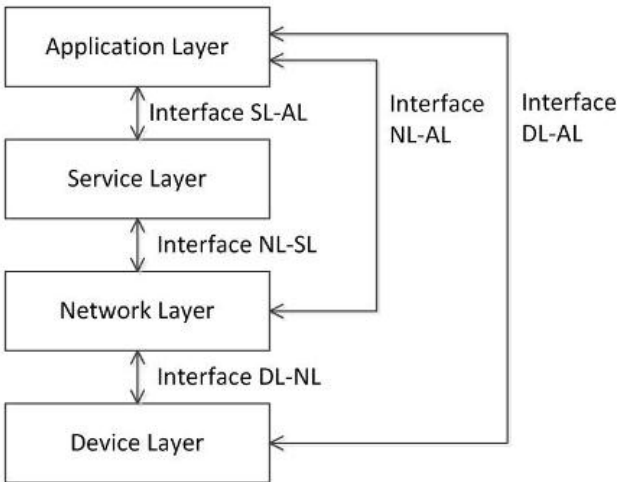

Fig.1. IoT Reference Architecture

Device layer is represented by sensor networks which are connected with minicomputers or controllers [17].

The modern researches discuss concerning high immaturity of the IoT market which is still be dynamical from the point of view of appearance and disappearance of the main market players. At the present, the Device Layer is the most predictable in IoT-RA. It is need to state, the Device Layer has a typical structure of the Computer Control System (CCS), like, for example, embedded systems, as it is represented on Fig.2. Control systems fundamentals lay in interaction with some processes of the real world via three the main parts which are sensors, controllers and actuators. For modern CCSs not mandatory but typically is a presence of Human-Machine Interface (HMI) with monitoring data transmission, processing and storage.

Popular hardware solutions, used today to implement the Device Layer of IoT, can be dived in two groups. The first group includes simple CPU-based boards fit for relatively small volume applications, for example, mbed NXP, Arduino family used Atmel CPUs.

The second group includes mini computers working on the base Linux operating systems, for example, Intel Edison, Intel Galileo, Raspberry Pi, Orange Pi, etc. Such devices can be used, from the one hand, as applications servers, from the other

hand, devices embedded features are able to process input and output digital and analog signals.

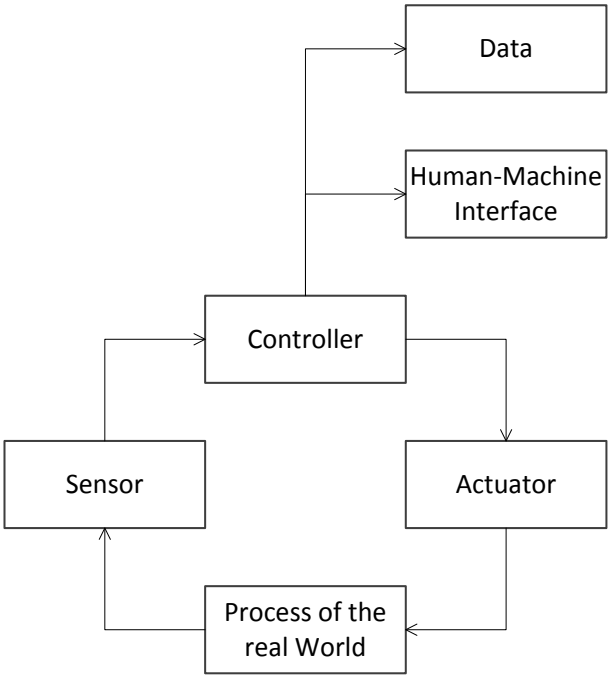

Fig.2. Typical Architecture of Computer Control Systems

3 General Approach to Implement AC DD

New methodology implementation requires not only technical measures but also organizational efforts to improve involved parts collaboration.

A chart on Fig.3 demonstrates such collaboration of the following three parts during AC DD implementation:

– Design team responsible for a product development;

– Quality Assurance (QA) and/or safety and security management team responsible for following all quality, safety and security procedures during development, verification and validation (V&V), configuration management, audits and other relevant activities;

– Assessment and certification team as a third part responsible for independent safety or security assessment of a product usually with issuing of a formal conformance document.

After establishment of organization and collaboration aspects let's analyze a general AC DD framework (see Fig.4).

Usually the first step in any system development is signing a contract. This contract is an input for system functional requirement as well as certification or licensing framework for safety and security critical applications. The Requirement Specification has to be developed on the base of contractual functional requirements.

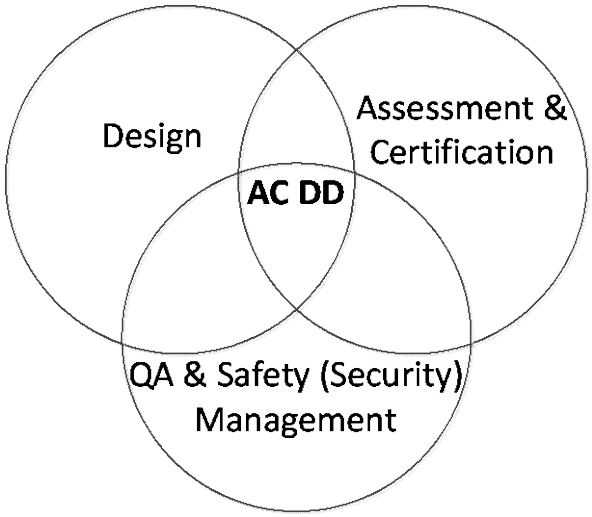

Fig.3. Assurance Case Driven Design Collaboration Chart

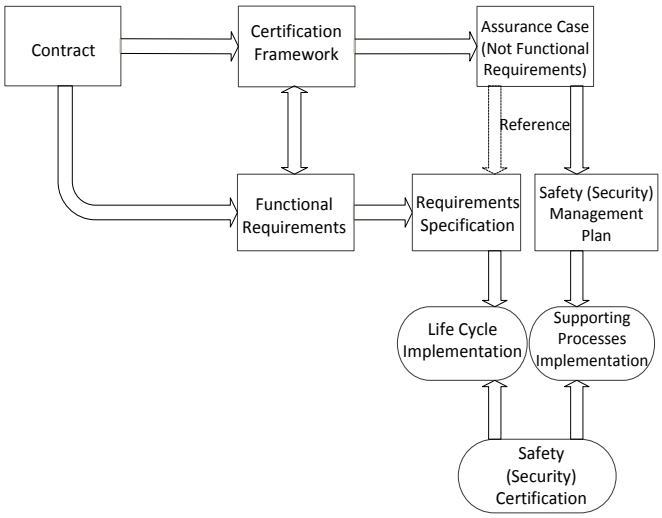

Fig.4. General Framework for Assurance Case Driven Design

Safety and security critical systems shall have an important addition to the Requirement Specification describing not functional requirements targeted to implement system integrity. AC DD approach proposes to present such requirements in a view of a preliminary Assurance Case. Such preliminary Assurance Case is not a result of assessment but a target which has to be achieved after the system implementation. Not functional requirements of Assurance Case are an input for Safety or Security Management Plan which has cover life cycle description with all development support processes.

Some parts of not functional requirements (for example, self-diagnostic requirements) may affect the Requirement Specification. After that staged life cycle with V&V and other supporting processes activities (Project Management, Configuration Management and other) has to be implemented in accordance with Safety (Security) Management Plan. After the contract and the Requirement Specification stages life cycle usually includes design, implementation, integration, validation, installation, and commissioning stages. Assurance Case activities have to be implemented after each of the stage. Safety or security certification has to finalize system life cycle before transfer it in operation at the customer site. Also during operation a periodical assessment or certification has to be done with associated update of Assurance Case.

Assurance Case structure depends from a type of the application. For example a typical structure of Assurance Case for industrial functional safety related application includes: security activities coordinated with safety, process implementation and assessment, and product implementation and assessment (Fig.4). Assessment can be done in a view of deterministic analysis, probabilistic analysis or demonstration.

4 Taxonomy of Functional Safety Standards

Typically, Assurance Case is built on the base of standards requirements. There are some standards which state requirements to safety of CCS. A vertical standard in this area is IEC 61508, Functional safety of electrical/ electronic/ programmable electronic safety-related systems, which include seven parts (see Fig.5).

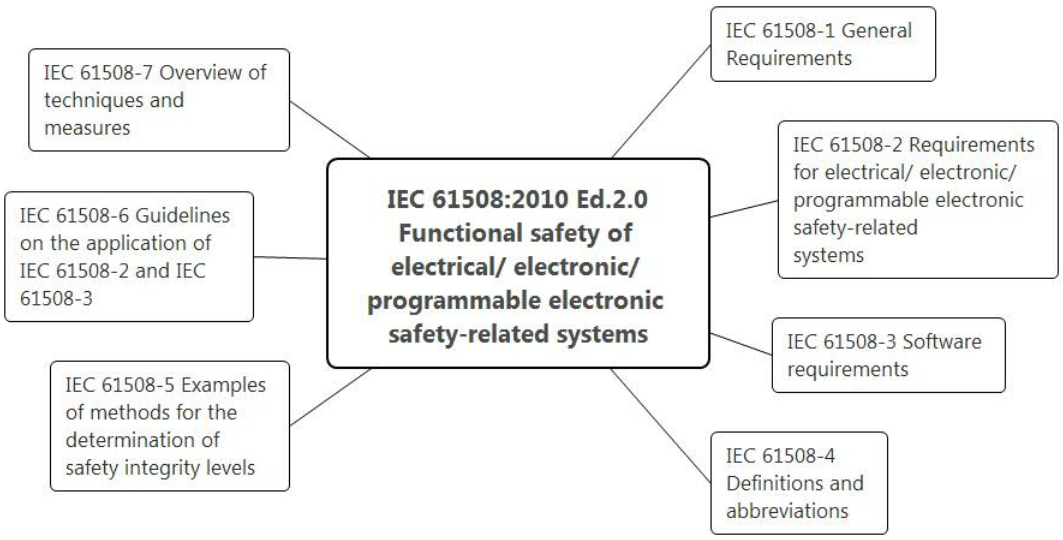

Fig.5. A Structure of the Standards Series IEC 61508

For separated domains there are the following standards:

– IEC 61511, Functional safety – Safety instrumented systems for the process industry sector;

– IEC 62061, Safety of machinery: Functional safety of electrical, electronic and programmable electronic control systems;

– IEC 61513, Nuclear power plants – Instrumentation and control for systems important to safety;

– ISO 26262, Road vehicles – Functional safety;

– EN 50129, Railway Industry Specific – System Safety in Electronic Systems;

– IEC 62304, Medical Device Software;

– RTCA DO-178C, Software Considerations in Airborne Systems and Equipment Certification;

– NASA STD 8719.13, Software Safety Standard.

Typical entities in area of functional safety assurance and assessment are hazards, harms, and risks, which together with equipment under control (EUC) protective measure are an ontology for safety critical plants (see Fig.6). Let's enter some definitions in accordance with IEC 61508 context.

Harm is physical injury or damage to the health of people or damage to property or the environment.

Hazard is potential source of harm.

Risk is combination of the probability $P(t)$ of occurrence of harm and the severity C of that harm.

EUC is equipment, machinery, apparatus or plant used for manufacturing, process, transportation, medical or other activities.

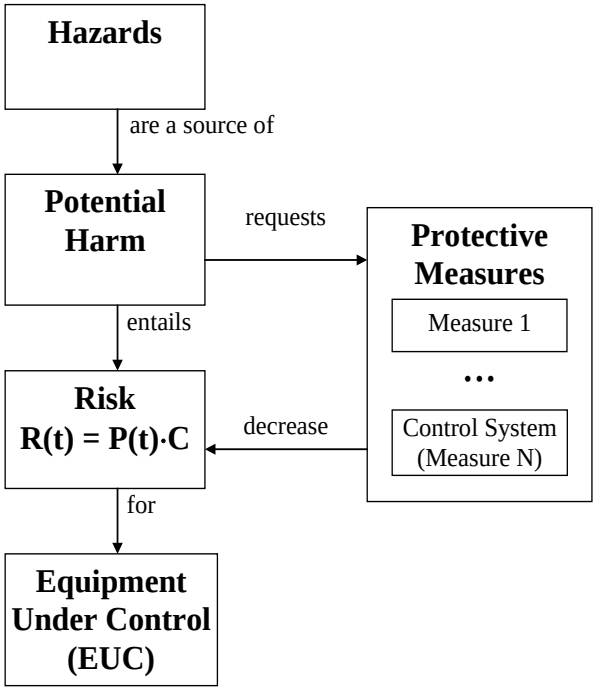

Fig.6. Ontology for Safety Critical Facilities

The next step in analysis of the IEC 61508 is taxonomy building for requirements to CCS functional safety. The main part of modern safety standards are based on a concept of a tolerable risk. CCS shall perform safety functions to ensure a tolerable risk level for the EUC and for the controlled facilities. To achieve the safety goals, CCS shall implement own critical failures risk not more than applicable tolerable level. To describe this issue, IEC 61508 endorses safety integrity as probability of a safety-related system satisfactorily performing the specified safety functions under all the stated conditions within a stated period of time. Safety integrity consists of random capability and systematic capability.

Random capability is an avoidance of random hardware failures which are occurring at a random time as results from one or more of the possible degradation mechanisms in the hardware. There are such measure and techniques as redundancy, separation, equipment qualification, self-diagnostics and other to avoid random failures.

Systematic capability requests an avoidance of systematic failures related in a deterministic way to a certain cause, which can only be eliminated by a modification of the design or of the manufacturing process, operational procedures, documentation or other relevant factors. There are the following approaches to resist systematic failures:

– Implementation of rigorous functional management as an umbrella process to launch project management, quality assurance, configuration management, documents control and other relevant activities with relations to safety lifecycle;

– Safety standards require V-shape lifecycle which consist staged development with verification and validation (V&V);

– Also different measures and techniques (i.e. formal and semiformal design methods, fault and error detection, software modularity, defensive programming, etc.) shall be implemented to avoid systematic failures at the level of the integrated system with software and hardware components.

Functional safety assessment shall be implemented as investigation, based on evidence, to judge the functional safety achieved by safety-related system.

The considered random and systematic capabilities together with functional safety assessment contains framework and taxonomy of CCS functional safety requirements implementation (see Fig.7).

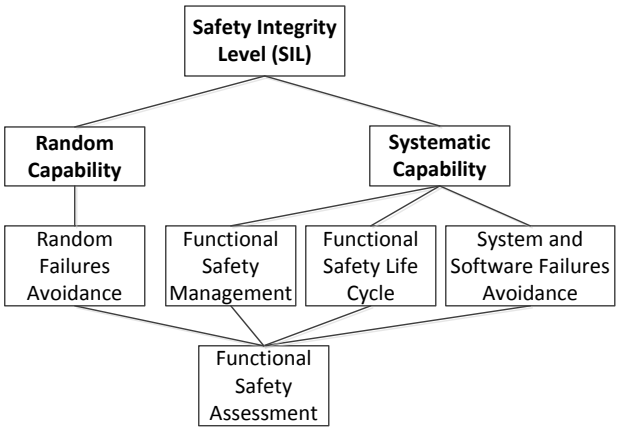

Fig.7. Taxonomy for Requirements to Functional Safety

Taking into account the above, it is possible to analyze and formally classify all the scope of requirement. For example, IEC 61508 contains seven parts (see Fig.5). Three the first parts consequentially describe requirements to safety critical facility (part 1), to system and hardware (part 2), and to software (part 3). Four other parts play mainly a secondary role. Each of the three first parts contains requirements to documentation, to

functional safety management, to safety life cycle and to functional safety assessment (see Fig.8).

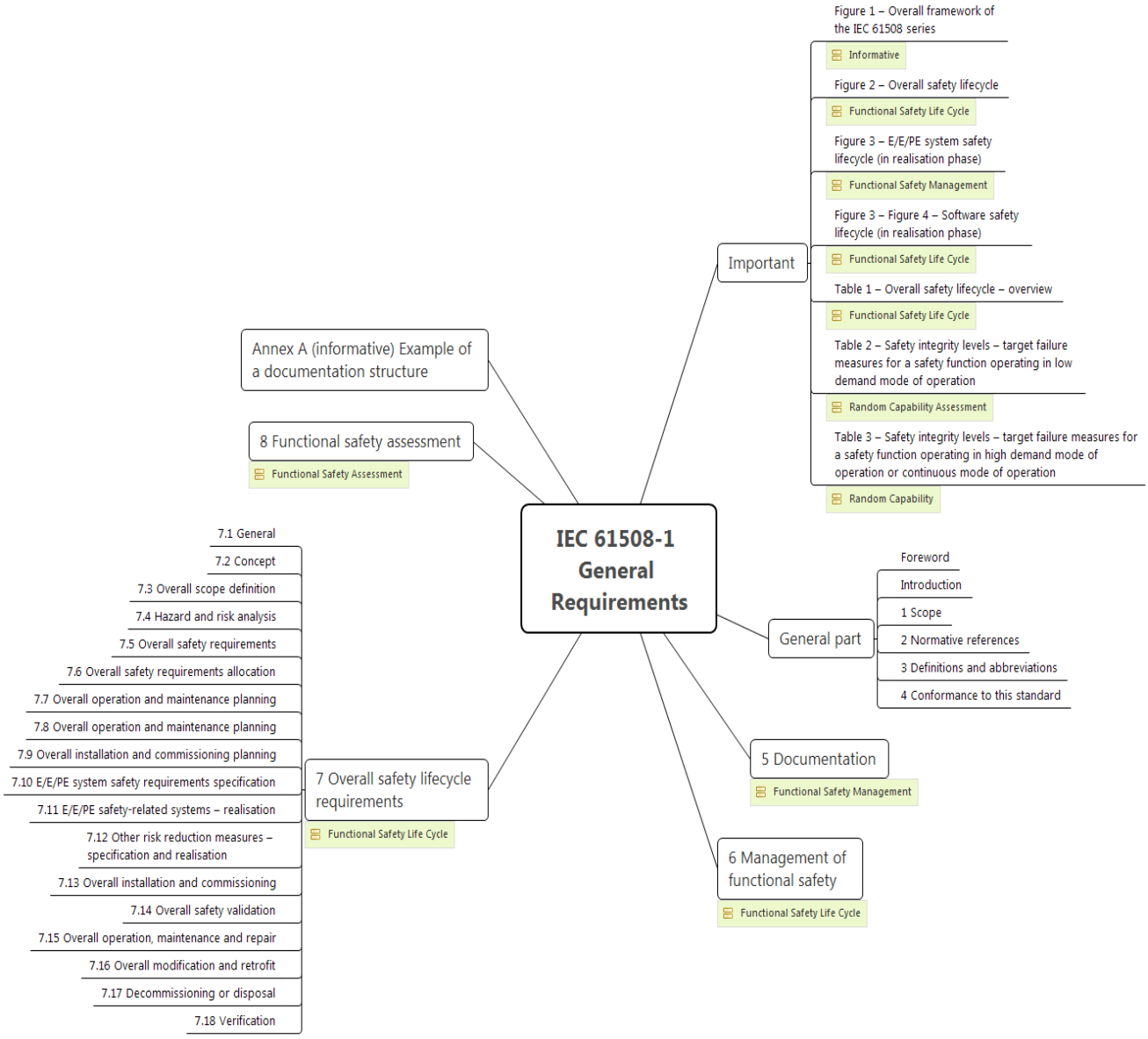

Fig.8. Structure of the Standard IEC 61508 – Part 1: General Requirements

5 Notations for a Static Part of Assurance Case

Firstly Assurance Case notation has been developed by Toulmin in [9] as an application of rhetoric and argumentation theory. It was a development of the classical predicative implication which was handled for some not trivial cases when the implication cannot be described just in "YES" and "NO" terms. Fig.9 presents authors view on not classical implication proposed by Toulmin.

Toulmin notation contains the following parts:

– Claim is a statement that something is so;

– Ground is the backing for the claim;

– Warrant is the link between the claim and the grounds;

– Backing is support for the warrant;

– Qualifier is the degree of certainty employed in offering the argument;

– Rebuttal is exceptions to the initial claim.

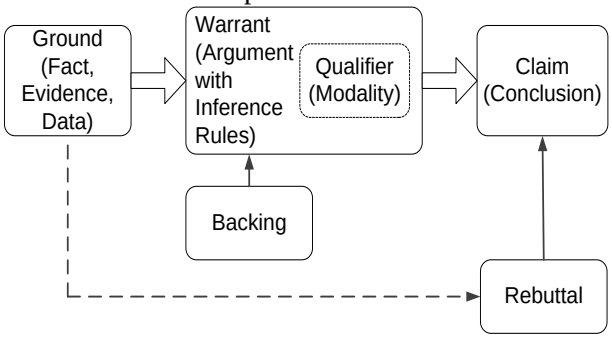

Fig.9. Basic Toulmin Notation for Claim-Argument-Evidence Model

It is clear, that the present notation is a kind of labeled graph. Graph nodes can be described as countable sets. Taking into account relations (edges) between graph nodes, the proposed Toulmin model can be described in terms of relational algebra with Entity-Relationship model (ER-model).

Posterior Assurance Case notations (CAE and GSN) [1] were developed also based on set theory, graph theory and relational algebra.

In the AC DD framework we propose some addition for Assurance Case CAE notations to be able assess specific features of critical systems. Acceptance criteria and coverage criteria are two additional entities which have to be taken into account for support arguments and evidences. Acceptance criteria are the conditions when stated requirements are met. From the point view of Assurance Case, acceptance criteria provide us ability to state the right arguments which are consistent with the claim and to provide the evidences which are consistent with the arguments. Coverage criteria describe how completely the claim is met.

From the point view of Assurance Case, coverage criteria provide us ability to state multiple arguments to completely cover all claim features and to provide multiple evidences which completely cover the arguments. Acceptance criteria for a claim can be extracted from both argument and/or evidence. In general case acceptance criteria provide a quantitative and qualitative description of a situation when the claim is met. A coverage criterion is a measure used to describe the degree to which evidence for specific arguments is provided. A modified CAE notation which we name Claim-Argument-Evidence-Criteria (CAEC) notation is given on Fig.10.

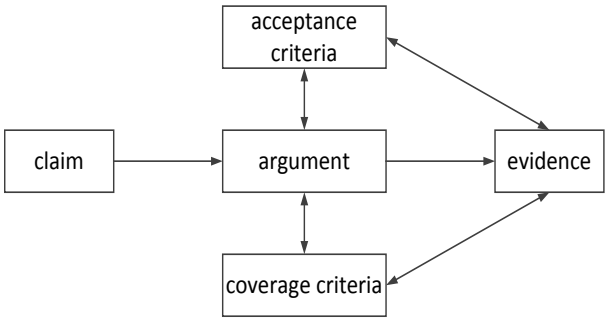

Fig.10. Claim-Argument-Evidence-Criteria (CAEC) Notation

6 Notations for a Dynamic Part of Assurance Case

The next step of CAE / CAEC notation development is to support activities of Safety & Security Life Cycle (SLC) stages with implementation of Assurance Case. Specification and design requirements are the inputs for each of the SLC stage. After any stage fulfillment, requirements implementation assessment has to be performed. This fundamental of the SLC has to be supplemented by the project specific products, processes, tools and techniques.

A sequence of SLC stages and relations between can be described with as named IDEF0 diagrams which specify an approach to functional modeling (see Fig.11). Each function entails four entities: inputs, outputs, control and mechanism of implementation.

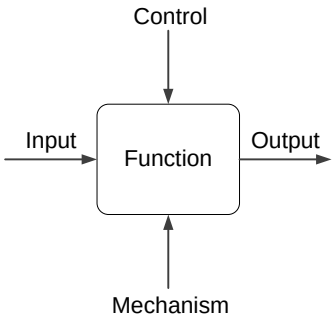

Fig.11. IDEF0 Notation

IDEF0 is also a kind of a labeled graph based on fundamentals of set theory, graph theory and relational algebra. The above demonstrate an adaptation of IDEF0 notation to dynamic Assurance Case application.

The following activities are mandatory for each of the SLC stage:

– Development targeted to move an implemented product representation stage by stage through SLC;

– V&V targeted to check conformance of the SLC stage development outputs to the SLC stage development inputs;

– Assurance Case update based on assessment of performed development and V&V activities.

The above can be represented as a diagram given on Fig.12.

The proposed DVA Development-V&V-Assurance Case (DVA) notation is based on following fundamentals:

– Safety & Security Life Cycle can be represented in a view of three components: Development (D), Verification and Validation (V&V) and Assurance Case (A);

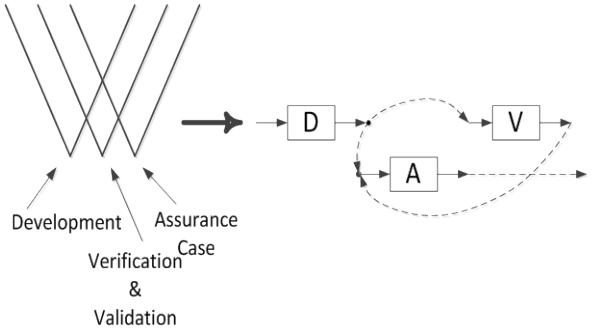

Fig.12. Transition from V-shape Life Cycle to Development-V&V-Assurance Case (DVA) Notation

– Development activities are staged implementation of requirements in design description of system, hardware and software, and after that implementation of requirements in a physical system, hardware and software;

– Development also covers processes implementation to support development of the product; processes also are described in a view of requirements which are collected in project plans;

– Typically requirements are represented and handled as database records; from this point of view the main operation with requirements are CREATE (to add), DELETE, MODIFY (if requirement needs some sense correction), EDIT (if requirement needs only editorial correction without changing of a sense);

– Forward and backward requirement tracing shall be implemented at each of Life Cycle stage to assure: 1) all previous stage requirements are implemented into the next stage documents; 2) no new requirement appears in the next stage documents; 3) all the requirements are verified or validated;

– Compliance of the product of next Life Cycle stage with the product of the previous Life Cycle stage is checked by implementation of V&V process;

– Compliance of processes implementation (including development and V&V processes) is checked by audits when processes implementation evidences are investigated against the project plans requirements; these audits can be a part of Assurance Case activities;

– All three D, V and A components of Safety & Security Life Cycle have specific inputs and outputs for each of the Life Cycle stage; so a diagram on Fig.12 represents DVA relations for some single stage.

To develop a graph and theoretical-set based model for DVA notation a diagram on Fig.11 should be elaborated to reflect feedback relations after V&V and Assurance Case performance (see Fig.13). Direct data transmission and feedback data are highlighted with different templates of lines. From the formalism prospective DVA notation can be described with using IDEF0 diagrams.

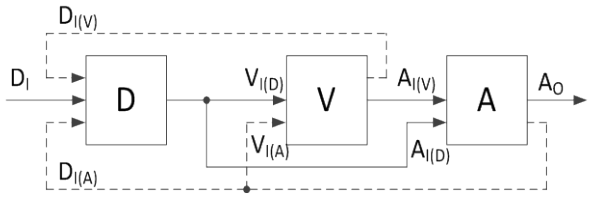

Fig.13. Graph and theoretical-set based description of DVA Notation

It is clear for Fig.12, input and output sets have some overlapping, so sets of DVA data flows ate described in terms of inputs. There are the following data sets transmitted between components of DVA:

– $D_I = \{d_{i1}, d_{i2}, \ldots, d_{iK}\}$ – a set of development process inputs transmitted from the out of the previous life cycle stage;

– $V_{I(D)} = \{v_{id1}, v_{id2}, \ldots, v_{idL}\}$ – a set of V&V process inputs transmitted from the out of development process;

– $A_{I(D)} = \{a_{id1}, a_{id2}, \ldots, a_{idM}\}$ – a set of Assurance Case process inputs transmitted from the out of development process;

– $A_{I(V)} = \{a_{iv1}, a_{iv2}, \ldots, a_{ivN}\}$ – a set of Assurance Case process inputs transmitted from the out of V&V process;

– $D_{I(V)} = \{d_{iv1}, d_{iv2}, \ldots, d_{ivP}\}$ – a set of development process inputs transmitted from the out of V&V process (a corrective feedback);

– $D_{I(A)} = \{d_{ia1}, d_{ia2}, \ldots, d_{iaQ}\}$ – a set of development process inputs transmitted from the out of Assurance Case process (a corrective feedback);

– $V_{I(A)} = \{v_{ia1}, v_{ia2}, \ldots, v_{iaR}\}$ – a set of V&V process inputs transmitted from the out of Assurance Case process (a corrective feedback);

– $A_O = \{a_{o1}, a_{o2}, \ldots, a_{oS}\}$ – a set of Assurance Case process inputs transmitted to the next life cycle stage after all the internal corrections.

7 Case Study: Application of Assurance Case Driven Design for Internet of Things

Let's consider the above concept for safety critical application of IoT Device Layer. For that we need to implement the following fundamentals:

– IoT applications have to comply with functional safety requirement, which, as it is shown above, include: Random Failures Avoidance, Functional Safety Management, Functional Safety Life Cycle, Systematic Failures Avoidance, and Functional Safety Assessment;

– To provide a full scope of IoT relevant requirement, it is needed to add two important areas those are security assurance and low power consumption; so these two issues have to be included to IoT Assurance Case;

– Each of the main issue of the IoT Assurance Case consists of a lot of atomic requirements, a static CAEC part of Assurance Case has to be considered and for each such requirement (see Fig.10);

– Assurance Case has to be sequentially implemented for *N* stages of IoT application Safety Life Cycle;

– A dynamic DVA part of Assurance Case has to be considered for each of the stage of Safety Life Cycle (see Fig.12, 13).

All the considered above statements are implemented in the IoT Assurance Case concept, as per Fig.14.

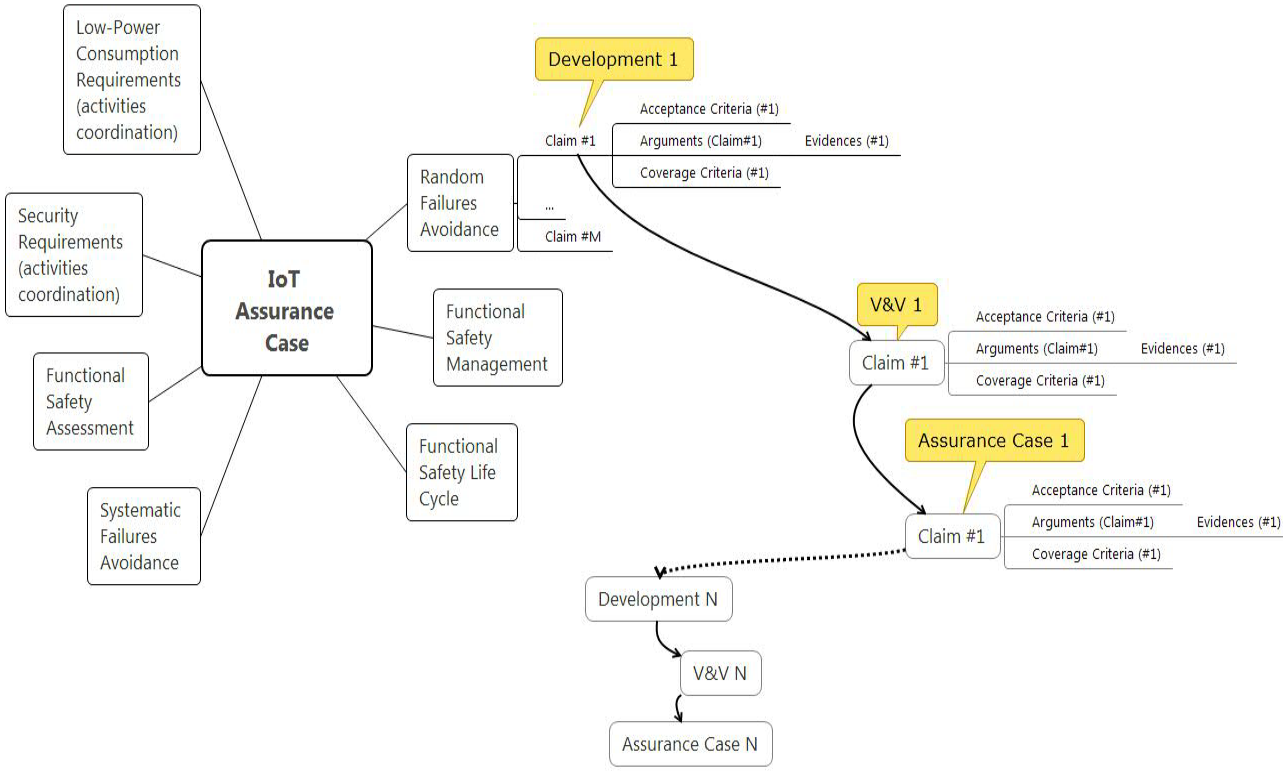

Fig.14. Assurance Case concept for the Internet of Things safety critical applications

8 Conclusion

The proposed AC DD approach may provide some benefits on the base of cost-effective "embedded certification" briefly described by CAEC and DVA notation. This cost-effective solution can work under conditions when the total cost of life cycle with application of "embedded certification" would be less than the cost of usual life cycle with usual after life cycle certification, i.e.: Cost (DVA Life Cycle) < Cost (DV Life Cycle) + Cost (Certification).

The next practical steps of AC DD development have to be directed to analyze existing Safety and Assurance Cases for cloud computing and big data analytics as well as to enforce Assurance Cases for IoT products with security and low-power informed approach.

For the last point is seems to be prospective to use typical safety and security building blocks [2]. A concept of safety and security building blocks describe typical component, architectural pattern, design solution, algorithm, protocol etc., such that it can be analyzed about at an abstract level (i.e., independent of a specific system). Proposed safety and security building blocks include, for example, encryption, signature generation and verification, node authentication, access control, traffic filtering, integrity protection, checksums, run-time monitoring and other.

References:

[1] T. Kelly, *Arguing Safety: A Systematic Approach to Managing Safety Cases: PhD thesis,* University of York, 1998.

[2] *D2.1 – Specification of Safety and Security Mechanisms*, Security and Safety Modelling (SESAMO): Artemis JU Grant Agreement no.: 295354, 2013.

[3] P. Bishop, R. Bloomfield, L. Cyra, Combining testing and proof to gain high assurance in software: A case study, *Proceedings of the IEEE International Symposium on Software Reliability Engineering (ISSRE 2013)*, 2013, pp. 248-257.

[4] V. Kharchenko, V. Sklyar, Assurance Case Driven Design for software and hardware description language based systems, *Radioelectronic and Computer Systems*, No.5(79), 2016, pp. 98-103.

[5] L. Duan, Sanjai R., M. Heimdahl, O. Sokolsky, I. Lee, Representation of Confidence in Assurance Cases Using the Beta Distribution, *Proceedings of IEEE 17th International Symposium on High Assurance Systems Engineering (HASE 2016)*, 2016, pp. 86-93.

[6] K. Netkachova, K. Müller, M. Paulitsch, R. Bloomfield, Security-Informed Safety Case Approach to Analysing MILS Systems, *Proceedings of International Workshop on MILS: Architecture and Assurance for Secure Systems*, 2015, pp. 12-18.

[7] V. Kharchenko, A. Kovalenko, O. Siora, V. Sklyar, Security assessment of FPGA-based safety-critical systems: US NRC requirements context, *Proceedings of the Information Digital Technologies International Conference*, 2015, pp. 132-138.

[8] V. Sklyar, Safety-critical Certification of FPGA-based Platform against Requirements of U.S. Nuclear Regulatory Commission (NRC): Industrial Case Study, *Proceedings of the 12th International Conference on ICT in Education, Research and Industrial Applications*, 2016, pp. 129-136.

[9] S. Toulmin, *The Uses of Argument*, Cambridge University Press, 1958.

[10] C. Holloway, Towards understanding the DO-178C / ED-12C assurance case, *Proceedings of 7th International Conference on System Safety, incorporating the Cyber Security Conference*, 2012.

[11] M. Napolano, F. Machida, R. Pietrantuono, D. Cotroneo, Preventing recurrence of industrial control system accident using assurance case, *Proceedings of IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW 2015)*, 2015.

[12] *GSN Community Standard*, Version 1, 2011.

[13] A. Sajid, H. Abbas, K. Saleem, Cloud-Assisted IoT-Based SCADA Systems Security: A Review of the State of the Art and Future Challenges, *IEEE Access*, Vol. 4, 2016, pp. 1375-1384.

[14] R. Islam, A. Kwak, H. Kabir, M. Hossain, K. Kwak, The Internet of Things for Health Care: A Comprehensive Survey, *IEEE Access*, Vol. 3, 2015, pp. 678-708.

[15] V. Sklyar, O. Odarushchenko, E. Bulba, R. Horbenko, A. Ivasyuk, D. Kotov, Assessment of Energy Consumption for Safety-Related PLC-Based Systems, In: V. Kharchenko et al. (eds.), *Green IT Engineering: Concepts, Models, Complex Systems Architectures*, Springler, 2017, pp. 269-281.

[16] V. Kharchenko, A. Gorbenko, V. Sklyar, C. Phillips, Green Computing and Communications in Critical Application Domains: Challenges and Solutions, *Proceedings of the Information Digital Technologies International Conference*, 2013, pp. 241-247.

[17] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, M. Ayyash, Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications, *IEEE Communications Surveys & Tutorials*, Vol. 17, Issue 4, 2015, pp. 2347-2376.