Use of Evidence Based Arguments in Standard Compliance

Managing Safety Case Relations to System Models

Andrzej Wardziński
Gdańsk University of Technology, Poland
ARGEVIDE sp. z o.o., Gdańsk, Poland

SCSSS 2017
23 May 2016, Stockholm
Part 1: Use of Evidence Based Arguments in Standard Compliance
1. Evidence based arguments
2. Standard structure and requirements
3. Demonstrating compliance and making assessment
4. Managing standards

Part 2: Managing Safety Case Relations to System Models
1. References to the system context
2. System model
3. Establishing and maintaining relations
Evidence based arguments

- Argument structure based on Toulmin’s argument model
  - comply with ISO 15026 and OMG SACM
- Argument premises may be supported by evidence

Assurance case argument

Evidence
Available argument notations:

- **graphical** notations (GSN, CAE)
- **tabular** notation
- **hierarchical** textual notation
  - **TCL** – Trust Case Language
    - developed at Gdańsk University of Technology in 2007
Prescriptive vs. goal based standards

Use of evidence based arguments (assurance cases) is already required by some goal based standards

<table>
<thead>
<tr>
<th>Requirements of a standard</th>
<th>Prescriptive standards</th>
<th>specify precisely what should be demonstrated</th>
<th>Goal based standards</th>
<th>specify goals and allow different ways how it is achieved</th>
</tr>
</thead>
<tbody>
<tr>
<td>How to demonstrate compliance</td>
<td>Provide evidence the requirement is satisfied</td>
<td>1. Define strategy how the goal is achieved 2. Justify the strategy is effective 3. Provide evidence the strategy is followed</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Argument hierarchy can represent **structure** of a standard
- directly or with mapping

Leaves of the argument represent **requirements** of the standard

Users can provide **evidence** to demonstrate compliance

Argument can be extended with **additional information** like:
- guidance for standard users
- assessment procedures and criteria
Argument structure example

ACQ.3.BP1: Negotiate the contract/agreement

Negotiate all relevant aspects of the contract/agreement with the supplier.

[OUTCOME 1]

NOTE 1: Relevant aspects of the procurement may include: system requirements, acceptance criteria and evaluation criteria, linkage between payment and successful completion of acceptance testing, process requirements, process interfaces and joint processes.
Compliance for each requirement of the standard can be evaluated separately.

Different assessment methods can be used, for example:

- Dempster-Shafer method permits to represent uncertainty (e.g. missing information)
- SPICE is using 0..100 scale with four levels of compliance (N-P-L-F)
- Rating scale is using number for evaluation
- 3-value scale (noncompliant, partially compliant, compliant)
Assessment results can be
- represented with a color scale
- reported to MS Excel, XML, PDF
1. Define structure of a standards (conformance case template)

2. Plan your compliance project (start with an empty compliance case)

3. Provide evidence and compliance argument

4. Make assessment (self assessment, certification assessment)

5. Report progress and level of the compliance

6. Maintain compliance
Applications

The approach has been applied by commercial users for standards:

- Hospital Accreditation Standards (NCQA, Poland)
- ISO 9001 Quality management systems
- ISO 14001 Environmental Management Systems
- OHSAS 18001 Occupational Health and Safety Management
- ISO 27001 Information Security Management
- IEC 62443 Security for industrial automation and control systems
- EN/IEC 61511 Functional safety – Safety instrumented systems for the process industry sector
- ISO 26262 Road vehicles – Functional safety
- ISO/IEC 17065 Conformity assessment — Requirements for bodies certifying products, processes and services
Experiences

- Argumentation structure is easier to comprehend than traditional documentation of standards
  - Users better understand the standard requirements
- You can create an integrated compliance environment consisting of:
  - Requirements of the standard
  - Guidance, best practices, evidence samples
  - Compliance evidence and descriptions
  - Assessments and comments
- The approach helps to maintain consistency in conformance projects
- Online cooperation improves communication between organizations
Standards going electronic

- Traditional document structure of standards is
  - optimal for technical publication (and will not disappear)
  - not optimal for using it and for managing

- Standards logical structure and dependencies become more and more complex
  - maturity levels, SILs, EALs, process areas, practices, etc.

- Argumentation structure is a step in the right direction to represent logical structure of a standard
  - More advanced data structures may also be useful

- It helps to manage complex standards

- XML representation makes possible exchange of compliance information between systems and organizations
Part 2

Managing Safety Case Relations to System Models
Safety argument in the context

- Argument context includes…
  - System structure, elements and their properties
  - Behaviour (events, processes)
  - Risk model (hazards, causes, safety requirements)
  - Environment structure and properties
  - System life cycle activities and artefacts

- A valid safety argument needs the context to be correct and consistent
How can the context be managed?

- Informal references
  - Use context names in argument elements
    - Example claim: Speed sensor S17 failure rate is below 10^-6

- Distinct context elements
  - GSN Standard specifies a Context element
    - A context, presents a contextual artefact. This can be a reference to contextual information, or a statement.

- Model generated argument
  - Automatic safety argument generators ensure argument consistency with system models used.

- Direct references to system model elements
For the presented fragment of an argument:

- **Claim1**: Hazardous situation \( \{H\} \) is mitigated
  - **Context1**: Severity: \( \{\text{Sev}\} \)
  - **Context2**: Hazard \( \{H\} \) description
  - **Argument1**: Argument strategy over hazard causes
    - **Justification1**: Hazard is mitigated by providing control measures for all its causes
    - \([1..*]\) **Claim1.1**: Cause \( \{C\} \) is addressed by control measures

The goal is:

- to establish references to valid elements of the risk model
- to ensure referenced elements relations hold (e.g. we refer to causes of the hazard specified in the parent claim)
- to maintain correctness of the references and to be informed when it is challenged (e.g. elements of the risk model are modified)
System metamodel defines an abstract schema for system models

- It defines entities, attributes and relations
- UML class diagram can be used to present a metamodel

Example:
System metamodel enables establishing references to:
- elements of a given type
- elements in a specified relation with context elements

We extend the safety argument parameters with:
- a model type
- a selector which specifies an element type or relation

Claim 1: Hazardous situation \{H: HModel: Hazard\} is mitigated

Context 1: Severity: \{Sev: HModel: SeverityOfHazard(H)\}

Context 2: Hazard \{H\} description

Argument 1: Argument strategy over hazard causes

Justification 1: Hazard is mitigated by providing control measures for all its causes

[1..*] Claim 1.1: Cause \{C: HModel: CausesOfHazard(H)\} is addressed by control measures
An intermediary named *Model interface* can:

- provide information about system metamodel classes and relations
- give lists of elements which satisfy the reference requirement
- verify if a given element or relation is up to date
The minimal model of a model interface which permits to establish and maintain references to system models.
Integration process

Pre-development phase steps
1. System metamodel specification
2. Model interface development
3. Argument pattern development

Development phase steps
4. System modeling
5. Assurance case development (instantiation)
6. System models and assurance case maintenance (iteration of steps 4 and 5)
The relations data are maintained in:

- abstract reference table

<table>
<thead>
<tr>
<th>Pattern element id</th>
<th>Reference name</th>
<th>Model type</th>
<th>Element selector</th>
</tr>
</thead>
<tbody>
<tr>
<td>Claim 1</td>
<td>H</td>
<td>HModel (the risk model)</td>
<td>Hazard</td>
</tr>
<tr>
<td>Context 2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Context 1</td>
<td>Sev</td>
<td>HModel (the risk model)</td>
<td>SeverityOfHazard( H )</td>
</tr>
<tr>
<td>Claim 1.1</td>
<td>C</td>
<td>HModel (the risk model)</td>
<td>CausesOfHazard( H )</td>
</tr>
</tbody>
</table>

- concrete(instantiation) reference table

<table>
<thead>
<tr>
<th>Argument element id</th>
<th>Reference name</th>
<th>Model name</th>
<th>Model element id</th>
<th>Element name</th>
</tr>
</thead>
<tbody>
<tr>
<td>C1</td>
<td>H</td>
<td>PCAHazardTable.xml</td>
<td>H1</td>
<td>Air in line</td>
</tr>
<tr>
<td>Ctxt 2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ctxt 1</td>
<td>Sev</td>
<td>PCAHazardTable.xml</td>
<td>S1</td>
<td>Critical</td>
</tr>
<tr>
<td>C2</td>
<td>C</td>
<td>PCAHazardTable.xml</td>
<td>C1</td>
<td>Sensor failure to detect air bubble</td>
</tr>
<tr>
<td>C3</td>
<td>C</td>
<td>PCAHazardTable.xml</td>
<td>C2</td>
<td>Safety subsystem failure to stop the pump</td>
</tr>
<tr>
<td>C4</td>
<td>C</td>
<td>PCAHazardTable.xml</td>
<td>C4</td>
<td>Pump does not stop on request</td>
</tr>
</tbody>
</table>
Prototype solution

- Manual specification of argument pattern parameters
- Prototype instantiation tool reads / writes SACM 1.1 arguments
- The model interface implemented for XML risk model and OSATE AADL models (partially)
Experiences

- **Conclusions**
  - Uniform model interface is sufficient for establishing and maintenance of assurance case relations to system model
  - Use of GUIDs in system models is essential for references maintenance

- **Further work**
  - Case studies for other types of models
  - Verification function to detect model changes
  - Maintenance of the instantiation reference tables
  - Integration with SACM 2.0 (Terminology package)
Summary

- Uniform model interface will facilitate establishing and maintaining assurance case relation to system models
  - We expect this to be easier for safety engineers

- The established relations are:
  - correct as they rely on directly on existing models
  - up to date (this can be verified at any moment of time)

- System model changes can be propagated to the safety argument
Thank you for your attention