AMASS
Architecture-driven, Multi-concern and Seamless Assurance and Certification of Cyber-Physical Systems

AMASS Usage Scenario 2: Process & Product Configuration and Compliance Management

Second EAB Workshop
Västerås, Sept 17, 2018

Barbara Gallina, Ph.D.
WP6 Leader, T6.1-2 Leader, TM
Intro: Cross-&-Intra Domain Reuse - AREAS

- Process/Product/Assurance Case Line Specification
  - Variability Management for Cross and Intra Domain reuse
    - Process (P1) – families/line of processes
    - Product (P1, partly) – families/line of products
    - Argumentation (P2) – families/line of arguments
  - Measurement framework for Safety-oriented Process Line Engineering (SoPLE)
  - Compliance management: further developed vision
  - Argument fragments generation (Process and Product-based)
    - Towards fallacy-free process-based argumentation generation (P2)
  - Semantics-based equivalent standards mapping
  - Reuse assistant
    - Syntax-based Reuse Interface
    - Semantics-based Reuse Interface (P2)
Assumptions

- Who am I in this scenario?
  - A technical engineer having double expertise:
    - process engineering (process engineer)
    - product engineering (better a designer)
  - Domain of expertise: space-related development processes and systems design
  - Standards: ECSS
  - My company produces families of systems
    -> systematic reuse can be beneficial
  - How I decide?
    - First, I embrace a measurement framework
    - Then, if positive, I adopt the approach
AMASS Goal: G1 & G2

O3: consolidate a *cross-domain and intra-domain assurance* reuse approach to improve mutual recognition agreement of compliance approvals and to help assess the return of investment of reuse decisions.

**Measurement framework for SoPLE-SoPLE-targeted GQM Plus Strategies model**

As we had 6 common elements, **SoC computes to 6**. We have at least 4 single processes. The number of elements in the single processes for criticality levels A, B, C and D are 10 (f4 through f13), 10 (f4 through f13), 6 (f4, f5, f6, f9, f10 and f11) and 6 (f4, f5, f6, f9, f10 and f11) respectively. Thus, **PrR’s for single processes A, B, C and D are computed as 0.6, 0.6, 1 and 1 respectively**.

Let’s go

Our DSL = X \{ UMA, CHESSML, CACM-arg \}

Base model = X-compliant model

Resolved model = X-compliant model
Orthogonal variability management-P1

A software process modelled in EPF Composer

The achievement of error free models

Backward propagation of configured models

BVR Resolution editor

BVR VSpec editor

BVR Realization editor

2nd EAB Workshop, Västerås, Sept 17, 2018
Orthogonal variability management-P1
Exemplification at Product Level

- Small GEO product line has two main configurations:
  - FAST with a combination of chemical and electrical propulsion
  - FLEX based on only electrical propulsion for both orbit transfer and station-keeping

\[ (a) \quad \text{SoC} = \cap_{i=1}^{n} C_{pi} \quad (b) \quad P_{r} R_{i} = \frac{\text{SoC}}{|C_{pi}|} \]
Process & Product-based argument fragment generation – P1


F. U. L. Muram, B. Gallina and L. Gomez Rodriguez. Preventing Omission of Key Evidence Fallacy in Process-based Argumentations. 11th International Conference on the Quality of Information and Communications Technology (QUATIC), in press, Coimbra, Portugal, September 4-7, 2018

Second prototype (P1)
Other functionalities
Reuse assistant - P1

- Concept from Standard A ("from") to be Mapped
- Concept from Standard B ("to") to be Mapped
- Map Justification
- Map Postconditions (Compliance Gaps)
- Assurance Assets created as a Result of the Reuse
- Reusable Assurance Assets

2nd EAB Workshop, Västerås, Sept 17, 2018
Third prototype (P2)
Process & Product-based argument fragment generation

[Diagram showing the flow from Process Space to Normative Space through Mapping tables (Core), Argumentation about compliance (P1), Compliance checking (P2), and Ontology-based mapping (P2).]

Normative Space

Formalization

Norm(s)
F. UL Muram, B. Gallina and L. Gomez Rodriguez. Preventing Omission of Key Evidence Fallacy in Process-based Argumentations. 11th International Conference on the Quality of Information and Communications Technology (QUATIC), in press, Coimbra, Portugal, September 4-7, 2018
Fallacy detection and process-based argument generation

- Capturing standard requirements
- Modelling process lifecycle
- Mapping standard requirements
Fallacy detection and process-based argument generation

Modelling process lifecycle

Mapping standard requirements
Fallacy detection and process-based argument generation
Fallacy detection and process-based argument generation

Certification against following ROLES are INSUFFICIENT:
1. AOCSS ENGINEER
   DETECTED FALLACIES: Certifications against following competencies/requirements are omitted:
   - University degree in engineering
   - Several years of experience in the design, analysis and simulation of AOCSS systems in different project phases, Except
   - Working experience with Linux System, Matlab and SatSim.
   RECOMMENDATION: Add skill certifications against above omitted evidence for the AOCSS Engineer role
   to achieve sufficiency or provide rationale for its omission.

2. DEVELOPMENT TEAM LEADER
   DETECTED FALLACIES: Certifications against following competencies/requirements are omitted:
   - Management of Electra AOCSS SW development team
   - Working with Matlab/Simulink
   - Knowledge of design analysis and design test methodologies
   - Good analytical and problem-solving skills.
   RECOMMENDATION: Add skill certifications against above omitted evidence for the Development Team Leader role
   to achieve sufficiency or provide rationale for its omission.

Certification against following ROLES are SUFFICIENT:
3. AOCSS AIT ENGINEER
4. AOCSS SW ARCHITECT
5. AOCSS SW V&V MANAGER
Fallacy detection and process-based argument generation
Compliance checking + patterns

Automated Compliance Checking Vision


Automated Compliance Checking Vision

EPF Composer Modeling Capabilities

EPF Composer Customization

<table>
<thead>
<tr>
<th>EPF Composer</th>
<th>Compliance Information</th>
<th>Suggested Icons</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reusable Asset</td>
<td>Rule Set</td>
<td></td>
</tr>
<tr>
<td>Concept</td>
<td>Compliance Effect</td>
<td>![Arrow Icon]</td>
</tr>
<tr>
<td>Custom category</td>
<td>Standard requirement</td>
<td>![Target Icon]</td>
</tr>
</tbody>
</table>

AMASS Version of the Standards Mapping Method

Standards requirements
Lifecycle elements
Annotated Process
Automated Compliance Checking Vision

8.4.2 To ensure that the software unit design captures the information necessary to allow the subsequent development activities to be performed correctly and effectively, the software unit design shall be described using the notations listed in Table 7.

\[ r_3 : \text{performSpecifySwUnit} \Rightarrow [O] \text{selectMandatoryNotationsforSwDesign} \]
\[ r'_3 : \text{provideRationaleForNotSelectMandatoryNotationsforSwDesign} \Rightarrow [P] - \text{selectMandatoryNotationsforSwDesign} \]

Rules formalization [Castellanos2017]

Plugin modeling

Process Elements
- Method Content
  - Content Packages
    - Process Elements
      - Roles
    - Tasks
      - Design Software Unit
      - Specify software unit design
      - Start Software Unit Design Process
  - Work Products
    - Software Architectural Design
    - Software Safety Requirements
    - Software Unit Design

Standards requirements

Lifecycle elements

Annotated Process

Task: Start Software Unit Design Process
- Relationships
  - Mandatory: Software Architectural Design, Software Safety Requirements
  - Software Unit Design Process > Start Software Unit Design Process

More Informe
- addressSoftwareUnitDesignAndImplementationPhase
- performProvideAssociatedSoftwareSafetyRequirements
- performProvideSoftwareArchitecturalDesign

Effects
- addressSoftwareUnitDesignAndImplementationPhase
- performProvideAssociatedSoftwareSafetyRequirements
- performProvideSoftwareArchitecturalDesign
Automated Compliance Checking Vision

### Regorous report

#### Compliance Check Results:
- **Process is non-compliant.**
- **Description:** Unfulfilled obligation to `selectMandatoryNotationsForSwDesign` (Achievement, non-pre-emptive, non-persistent).
- **Element name:** Specify Software Unit.

<table>
<thead>
<tr>
<th>Custom Categories</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Standard Requirements ISO 26262 Software Unit Design</strong></td>
</tr>
<tr>
<td>- R1. Address software unit design process</td>
</tr>
<tr>
<td>- r1.1 Address software unit design process</td>
</tr>
<tr>
<td>- addressSwUnitDesignProcess</td>
</tr>
<tr>
<td>- R2. Specify software units</td>
</tr>
<tr>
<td>- r2.1. Complete requirements for specifying software units</td>
</tr>
<tr>
<td>- performProvideSwArchitecturalDesign</td>
</tr>
<tr>
<td>- performProvideSwSafetyRequirements</td>
</tr>
<tr>
<td>- performSpecifySoftwareUnit</td>
</tr>
<tr>
<td>- R2.2. Incomplete requirements for specify software units</td>
</tr>
<tr>
<td>- -performSpecifySoftwareUnit</td>
</tr>
<tr>
<td>- R3. Describe software unit specification</td>
</tr>
<tr>
<td>- r3.1 Strict software unit specification description</td>
</tr>
<tr>
<td>- selectMandatoryNotationsForSwDesign</td>
</tr>
<tr>
<td>- r3.2. Taylored software unit specification description</td>
</tr>
<tr>
<td>- provideRationaleForNotSelectMandatoryNotationsForSwDesign</td>
</tr>
<tr>
<td>- selectMandatoryNotationsForSwDesign</td>
</tr>
</tbody>
</table>

- **Rule Set - ISO 25862-Software Unit Design**
The process, product and argumentation models can be linked to enable
- impact analysis
- process engineers, designers and assurance managers to work separately
Reuse discovery

IoT: Internet Of Tools

Knowledge-centric Systems Engineering

OSLC KM:
Represent any type of system artefact
&
Access any (delegated) operation

Knowledge Management processes

...to support...
via
...interoperability...

Reuse discovery and selection

A shape: System Representation language

A set of reuse operations on top of the industrial knowledge graph...

<table>
<thead>
<tr>
<th>Operation</th>
<th>URI Template</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base URI/prefix</td>
<td><a href="http://www.reusecompany.com/oslc/km/operations">http://www.reusecompany.com/oslc/km/operations</a></td>
</tr>
</tbody>
</table>
| Search artifact   | <base_uri>/sas/search
Query params: query={text}
Body params: srl={srl content}                                      |
| Filter            | <base_uri>/sas/filter
- Similar to query capabilities
- Similar to Linkedin: { (key=value,)+ }                              |
Interplay of the functionalities - Reuse assistant perspective
Thank you for your attention!

Any questions