Supplementary material for article "ARCADE: a Framework for Integrated Management of Safety Assurance Information"

Camilo Almendra
Campus de Quixadá
Universidade Federal do Ceará
Quixadá, Brazil
camilo.almendra@ufc.br

Carla Silva
Centro de Informática
Universidade Federal de Pernambuco
Recife, Brazil
ctlls@cin.ufpe.br

1. Publication venue

This supplementary material refers to the article published in the 31st IEEE International Requirements Engineering Conference. Please cite as:

```
C. Almendra and C. Silva, "ARCADE: a Framework for Integrated Management
of Safety Assurance Information", 2023 IEEE 31st International Requirements
Engineering Conference (RE), Hannover, Germany, 2023
```

2. ARCADE Framework

Our framework is designed and implemented in the context of the target scenario illustrated in Figure 1.

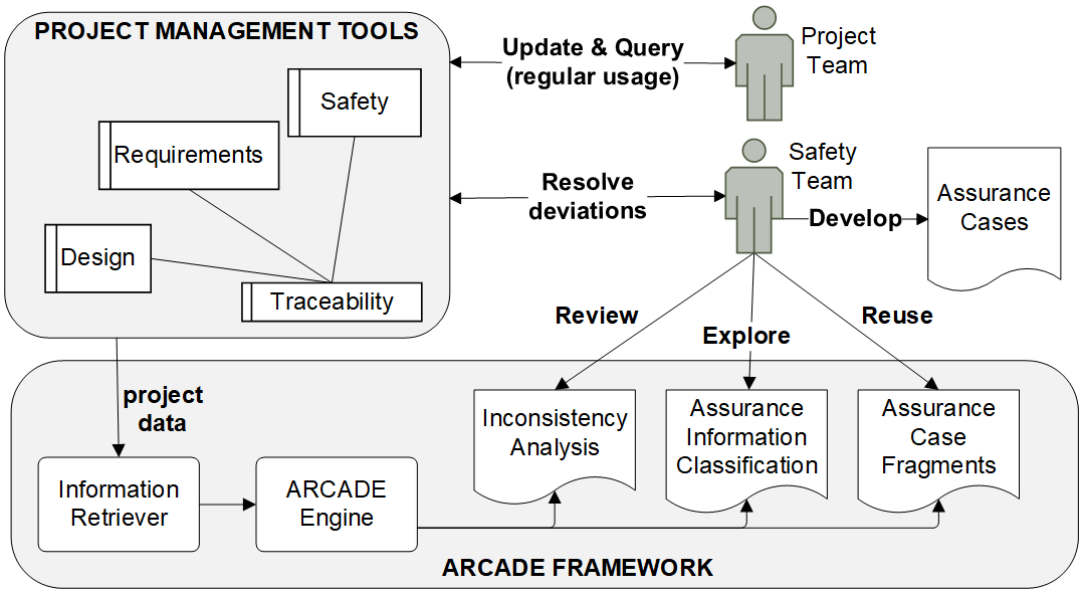

Figure 1. ARCADE framework usage scenario

2.1. Safety Assurance Information Model

2.2. Safety Assurance Ontology

2.2.1. Specification and knowledge gathering. The intended use of this ontology is to serve as a base system to integrate outputs of safety and software engineering activities performed by all kinds of personnel involved in SCS development (e.g. developers, safety engineers, assurance engineers). Intended users and their goals are:

- **Software engineers** working on requirements, design, tests and other kinds of artefacts who need to assess the consistency of software specifications and completeness of traceability with safety assurance artefacts.

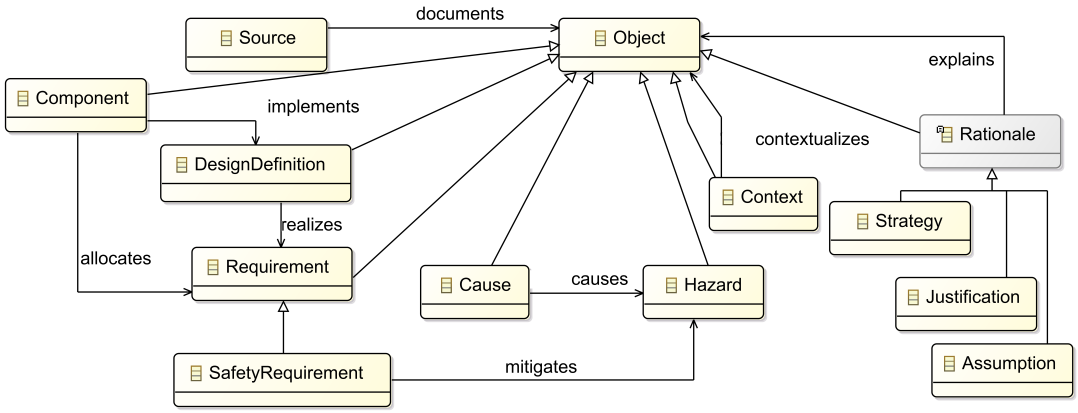

Figure 2. Safety assurance information metamodel

TABLE 1. INFORMATION MODEL CONCEPTS AND RELATIONS.

Concept	Description
Object	Input or output of the development process, in any level of granularity, that is managed in the ITS.
Source	An external artefact to the ITS.
Rationale	An explanation that provides additional information for an Object.
Strategy	A statement behind a project's decision. It can be a design decision, a requirements decision, or a mitigation strategy selected for a hazard.
Justification	A statement that provides an explanation for a project decision.
Assumption	A statement that reflects a common situation in the domain so ingrained into stakeholders' general beliefs, that it is unquestioned.
Hazard	A single hazard or category of hazards.
Requirement	A system requirement.
SafetyReq.	A requirement related to system safety. It represents actions and constraints that should or should not be performed to maintain the system in a safe state.
Context	An environmental or operational condition that may occur, and is relevant to safety.
Component	A unit of architecture, design or code of the system.
Relationship	**Description**
documents	It links a Source to Objects that it documents.
refines	It represents a refinement of Requirement or Hazard.
mitigates	It links a SafetyRequirement to the Hazards that it addresses.
causes	It links a Hazard to Objects identified as its cause.
explains	It links a Rationale to Objects it provides an explanation.
realizes	It links a Component to Requirements allocated to it.
contextualizes	It links a Context to Objects.

- **Safety engineers** working on safety analysis who need to assess the consistency of safety specifications and completeness of traceability with software specifications.
- **Assurance engineers** producing assurance cases or other certification documents who need to retrieve project information and assess its correctness and completeness with regard to safety assurance argumentation.

The knowledge sources used to conceptualize the ontology are described in Table 2.

TABLE 2. KNOWLEDGE SOURCES

Source description	Main contributions
Traceability reference model [1]	Core traceability concepts
Safety requirements metamodel [2]	Core safety concepts applied to software systems
GSN standard [3]	Assurance case concepts and rules of formation
RDF Schema [4]	Basic vocabulary to represent objects, object attributes and links in RDF
Open Patient Controlled Analgesia Pump [5]	Dataset with assurance cases and project information
Generic Patient Controlled Analgesia Pump [6]	Dataset with assurance cases and project information
Generic Insulin Infusion Pump [6]	Dataset with project information
Kelly's catalogue [7]	Assurance case argument patterns
Szczygielska's catalogue [8]	Assurance case argument patterns

The classes, object properties, data properties and rules that comprise the ontology specification are listed in Table 3. Classes and relationships shown in the conceptual model are not listed again in the table. All relationships have inverse. Rules from 1 to 20 are not restrictions, but instead classifications that will be inferred if the ontology reasoner finds items/links matching these conditions. Finally, rules 21 and 22 are restrictions, and if an object is classified as more than one of the disjointed classes, then inconsistency is triggered by the reasoner.

TABLE 3. FORMATION AND CLASSIFICATION RULES FOR SAFETY ASSURANCE ONTOLOGY

Rule	Description (classes in **bold**, relations in *italics*)
1.	**Object** that *allocates* other is a **Component**
2.	**Object** that *is allocated in* other is a **Requirement**
3.	If A *refines* B and B *is allocated in* C, then A *is allocated in* C
4.	**Object** that *causes* other is a **Cause**
5.	**Object** that *is caused by* other is a **Hazard**
6.	If A *contextualizes* B and B *is refined by* C, then A *contextualizes* C
7.	**Object** that *contextualizes* other is a **Context**
8.	**Object** that *documents* other is a **Source**
9.	**Object** that *explains* other is a **Rationale**
10.	**Object** that *implements* other is a **Component**
11.	**Object** that *is implemented by* other is a **Design Definition**
12.	**Object** that *documents* other is a **Source**
13.	**Object** that *mitigates* other is a **Safety Requirement**
14.	**Object** that *is mitigated by* other is a **Cause** or **Hazard**
15.	If A *mitigates* B and B *causes* C, then A *mitigates* C
16.	If A *refines* B and B *mitigates* C, then A *mitigates* C
17.	**Object** that *realizes* other is a **Design Definition**
18.	**Object** that *is realized by* other is a **Requirement**
19.	**Object** that *refines* or *is refined by* a **Hazard** is a **Hazard**
20.	**Object** that *refines* or *is refined by* a **Requirement** is a **Requirement**
21.	**Object** can only be classified as one of **Component**, **Design Definition**, **Hazard**, **Rationale** or **Requirement**
22.	**Object** can only be classified as one of **Assumption**, **Justification** or **Strategy**

2.2.2. Implementation. We illustrate how the model (Fig. 2) and rules (Table 3) are translated as OWL specification in Fig. 3. The object property named *mitigates* (lines 1 to 22), that is used to link **Safety Requirements** to **Causes** or **Hazards**, is specified as follow:

- Rule 13 is implemented in line 2 as a property domain.
- Rule 14 is implemented in lines 3 to 10 as a property range.
- Rule 15 is implemented as a property chain in lines 11 to 14.
- Rule 16 is implemented as a property chain in lines 15 to 18.

```
<owl:ObjectProperty rdf:about="arcade/sao#mitigates">
    <rdfs:domain rdf:resource="arcade/sao#SafetyRequirement"/>
    <rdfs:range>
        <owl:Class>
            <owl:unionOf rdf:parseType="Collection">
                <rdf:Description rdf:about="arcade/sao#Cause"/>
                <rdf:Description rdf:about="arcade/sao#Hazard"/>
            </owl:unionOf>
        </owl:Class>
    </rdfs:range>
    <owl:propertyChainAxiom rdf:parseType="Collection">
        <rdf:Description rdf:about="arcade/sao#mitigates"/>
        <rdf:Description rdf:about="arcade/sao#causes"/>
    </owl:propertyChainAxiom>
    <owl:propertyChainAxiom rdf:parseType="Collection">
        <rdf:Description rdf:about="arcade/sao#refines"/>
        <rdf:Description rdf:about="arcade/sao#mitigates"/>
    </owl:propertyChainAxiom>
    <rdfs:comment xml:lang="en">It links a SafetyRequirement to the
    Cause or Hazards that it addresses.</rdfs:comment>
    <rdfs:label xml:lang="en">mitigates</rdfs:label>
</owl:ObjectProperty>
```

Figure 3. OWL specification for Safety Requirement class and associated rules

2.3. Engine

We selected the OWL API for manipulating OWL specification and RDF data input, the HermiT reasoner for ontology reasoning, the OWL Explanation to compute explanations in case of inconsistencies, the ONT-API for manipulating OWL axioms as RDF, and Apache Jena for querying the ontology with SPARQL. Fig. 4 depicts the workflow of information from management tools to ARCADE. Fig. 5 depicts the internal workflow of the engine.

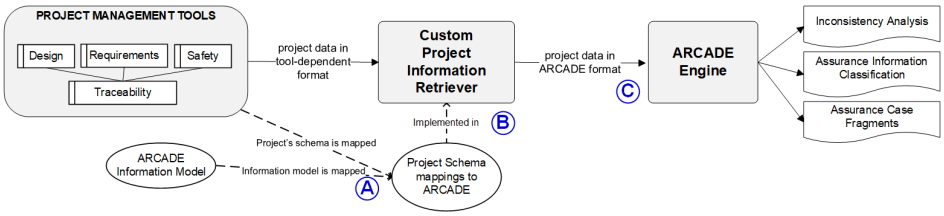

Figure 4. Information retrieval from Management Tools to ARCADE

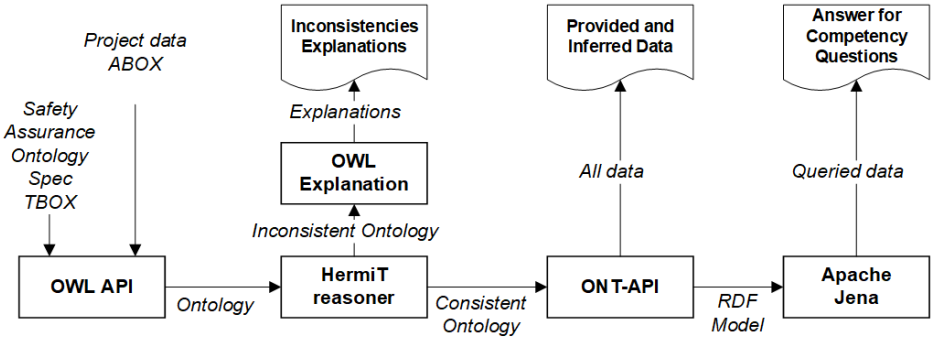

Figure 5. ARCADE Engine internal workflow

We defined a canonical format to allow the ARCADE Engine to handle a single input format, thus improving its reusability (see Fig. 6). This interchange format has the RDF Turtle syntax [9]. Turtle is a textual syntax for RDF that allows the specification of a data graph in a compact and natural text form. Besides, Turtle files are still machine-readable, yet providing human-readable representation, and can be used directly by third-party tools. This underlying syntax is practical to standardize the input format of the framework. The semantic schema of the interchange format is the ARCADE ontology.

```
:GPCA-10 rdf:type owl:NamedIndividual .
:GPCA-10 rdf:type sao:SafetyRequirement .
:GPCA-10 rdfs:label "Basal flow rate tolerance"@en .
:GPCA-10 rdfs:comment "The pump shall deliver basal infusion at the prescribed basal rate
infusion flow tolerance of tolerance = 0.5 ml/hour of the prescribed basal rate."@en .
:GPCA-10 sao:status "To Do"@en .
:GPCA-10 sao:creationdate "2020-09-28T14:52:08.896-0300"@en .
:GPCA-10 sao:lastupdate "2023-02-28T10:04:18.202-0300"@en .
:GPCA-10 sao:externaluri "https://arcade-dare.atlassian.net/browse/GPCA-10"@en .
:GPCA-10 sao:mitigates :GPCA-7 .
:GPCA-10 sao:mitigates :GPCA-19 .
:GPCA-10 sao:refines :GPCA-1 .
:GPCA-10 sao:isAllocatedIn :Component10005 .
:Component10005 sao:allocates :GPCA-10 .
:GPCA-10_justification rdf:type owl:NamedIndividual .
:GPCA-10_justification rdf:type sao:Justification .
:GPCA-10_justification sao:explains :GPCA-10 .
:GPCA-10 sao:isExplainedBy :GPCA-10_justification .
:GPCA-10_justification rdfs:label "the tolerance of rate is adequated to avoid under and
drug."@en .
```

Figure 6. ARCADE Interchange Format example

Figure 7 depicts the design patterns used to structure the assurance case generator.

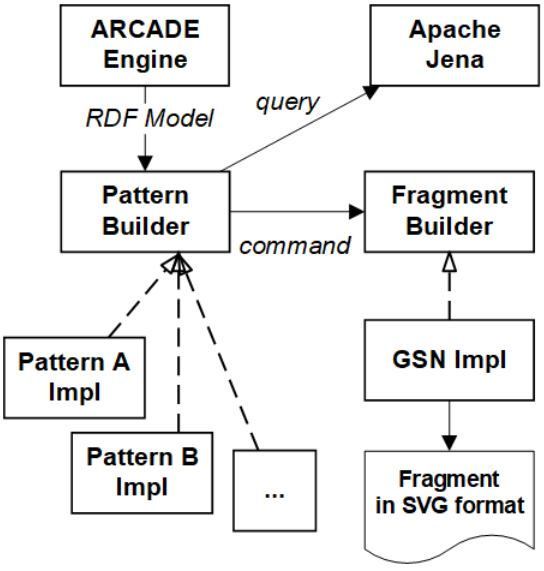

Figure 7. Assurance Case Generator workflow

3. ARCADE web tool usage

There are two ways to adopt ARCADE: configuring the automated retrieval or manually retrieving data for one-off analysis. In both cases, a preliminary step is to map the safety assurance information model to the representation schema used in the target project.

3.1. Mapping the information model

Figure 8 shows the customization of Atlassian Jira [10] – the most used agile project management tool nowadays [11].

Figure 8. Safety requirement represented as Story issue labelled as *Safety*, custom *Justification* field, and custom issue links *refines* and *mitigates*

3.2. Consistency analysis

ARCADE Engine presents the explanations showing the data entries and axioms involved separately. The axioms are termed "Rules" and data entries as "Project items" in the GUI to ease understanding by users. Fig. 9 shows a set of explanations generated upon inconsistencies found.

#	Items/Links involved	Rules impacted

Expl. #1	`GPCA-1 is a Requirement`
	`GPCA-10 refines GPCA-1`
	`GPCA-10 is a DesignDefinition`
		– Everything that refines or is refined by a Requirement, it's also a Requirement.
		– Project items can be classified to only one of these classes: Component, Context, DesignDefinition, Hazard, Rationale or Requirement (mutually exclusive).

Figure 9. Inconsistency explanation presented in ARCADE web tool

3.3. Knowledge exploration

The exploration of answers to the competency questions is provided in two perspectives: project items and links listing, and direct answers to questions.

Project Items	Item classification		Item relationships	
Item_ID	provided	inferred	provided	inferred
▼ GPCA-10 Name: Basal flow rate tolerance Description: The pump shall deliver basal infusion at the prescribed basal rate within a basal infusion flow tolerance of tolerance = 0.5 ml/hour of prescribed basal rate.	SafetyRequirement Requirement		mitigates GPCA-19 isAllocatedIn Component10005 mitigates GPCA-7 refines GPCA-1 isExplainedBy GPCA-10_justification	

Figure 10. Project data provided and inferred

Traceability Gap Questions	Status	Items founds
CQ13. Which are the Requirements that are not allocated in any Component	Gaps found!	GPCA-1 GPCA-2
CQ14. Which are the Safety Requirements that do not mitigate any Hazard	No gaps found.	

Figure 11. Answers for competency questions

3.4. Assurance case fragments

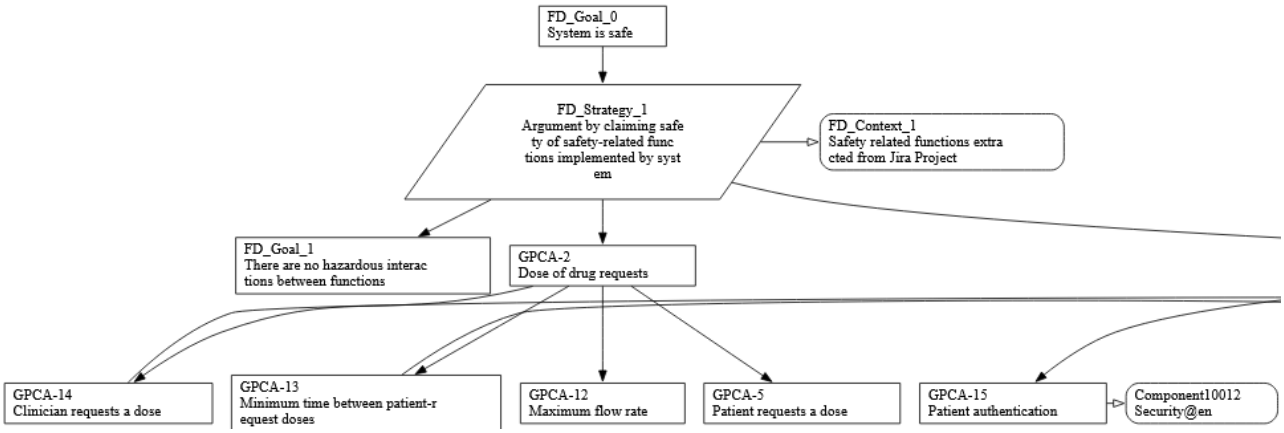

Figure 12. Assurance case fragment for Functional Decomposition pattern

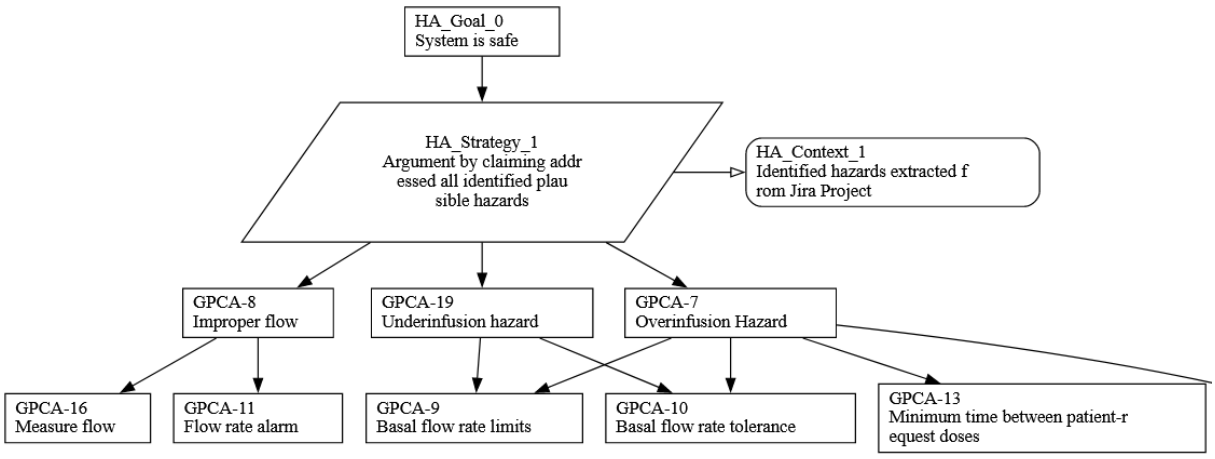

Figure 13. Assurance case fragment for Hazard Avoidance pattern

References

[1] B. Ramesh and M. Jarke, "Toward reference models for requirements traceability," *IEEE Transactions on Software Engineering*, vol. 27, no. 1, pp. 58–93, Jan 2001.

[2] J. Vilela, J. Castro, L. E. G. Martins, and T. Gorschek, "Safe-re: a safety requirements metamodel based on industry safety standards," in *Proceedings of the XXXII Brazilian Symposium on Software Engineering*. ACM, 2018, pp. 196–201.

[3] "Goal structuring notation community standard (version 3)," The Assurance Case Working Group, Standard, May 2021.

[4] W3C. Rdf schema 1.1. [Online]. Available: https://www.w3.org/TR/rdf-schema

[5] J. Hatcliff, B. Larson, T. Carpenter, P. Jones, Y. Zhang, and J. Jorgens, "The open pca pump project: an exemplar open source medical device as a community resource," in *Proceedings of the 2018 Medical Cyber-Physical Systems (MedCPS) Workshop*, 2018.

[6] Generic Infusion Pump Research Project. The generic infusion pump (gip) - a workbench for improving safety, security and usability of medical systems. Acessed: January 16, 2023. [Online]. Available: https://rtg.cis.upenn.edu/gip/

[7] T. Kelly and J. McDermid, "Safety case patterns-reusing successful arguments," in *IEE Colloquium on Understanding Patterns and Their Application to Systems Engineering*, no. 308, Apr 1998, pp. 3/1–3/9.

[8] M. Szczygielska and A. Jarzębowicz, "Assurance case patterns on-line catalogue," in *Advances in Dependability Engineering of Complex Systems*, W. Zamojski, J. Mazurkiewicz, J. Sugier, T. Walkowiak, and J. Kacprzyk, Eds. Cham: Springer International Publishing, 2018, pp. 407–417.

[9] D. Beckett, T. Berners-Lee, E. Prud'hommeaux, and G. Carothers, "Rdf 1.1 turtle - terse rdf triple language," World Wide Web Consortium, Tech. Rep., Feb 2014.

[10] Atlassian. Jira issue tracking system. [Online]. Available: https://www.atlassian.com/software/jira

[11] "13th annual state of agile report," CollabNet Version One, Tech. Rep., May 2019, accessed: 2019-08-07. [Online]. Available: https://explore.versionone.com/state-of-agile/13th-annual-state-of-agile-report