AMASS
ECSEL Joint Undertaking

Safe Cooperating Cyber-Physical Systems
using Wireless Communication | SafeCOP

MÄLARDALEN UNIVERSITY SWEDEN

Tool-Supported Safety-Relevant Component Reuse: From Specification to Argumentation

Irfan Sljivo, Barbara Gallina, Jan Carlson, Hans Hansson, Stefano Puri

{Irfan.sljivo, Barbara.Gallina, Jan.Carlson, Hans.Hansson}@mdh.se,

stefano.puri@intecs.it

23rd International Conference on Reliable Software Technologies – Ada-Europe 2018

Lisbon 2018

Outline

- Safety-critical systems and safety cases
- Safety-critical systems and reuse
  - Safety Element out-of-Context
- Contract-based design
- The AMASS Platform overview
- Support for product-based SEooC reuse in the AMASS platform
- Loading Arm Controller Unit Case Study
- Conclusions and future work

Safety-Critical Systems and Certification

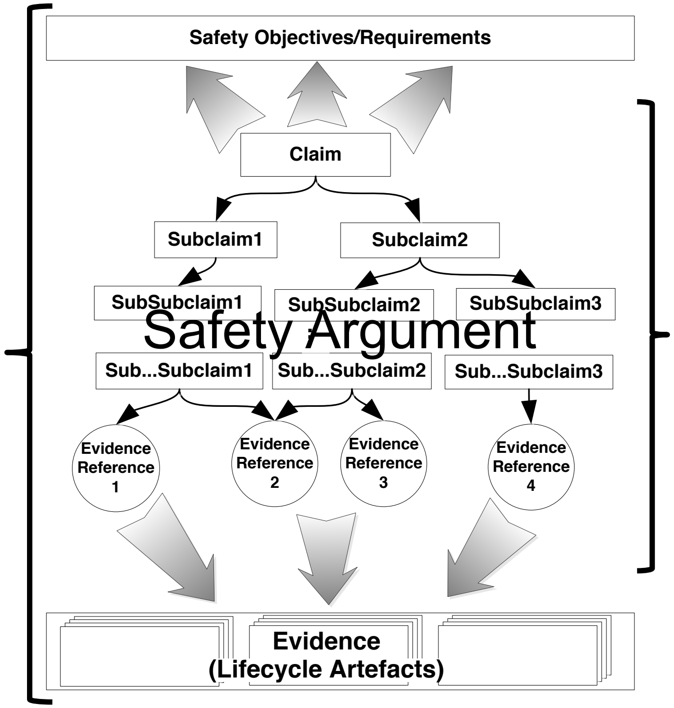
- *Safety-critical systems*
  - Malfunctioning can result in harm or loss of human life, or damage to property or the environment
    - *Sometimes the harm can be done even in absence of failures!* (Safety of the intended function)
    - **Functional safety**: absence of unreasonable risk caused by *hazards* due to malfunctioning behaviour
  - Usually need to comply with domain-specific safety standards

Avionics standard – DO-178C
Automotive – ISO 26262
Railways – EN 50128…

- Some safety standards require a safety case to show that the system is acceptably safe

Safety Case

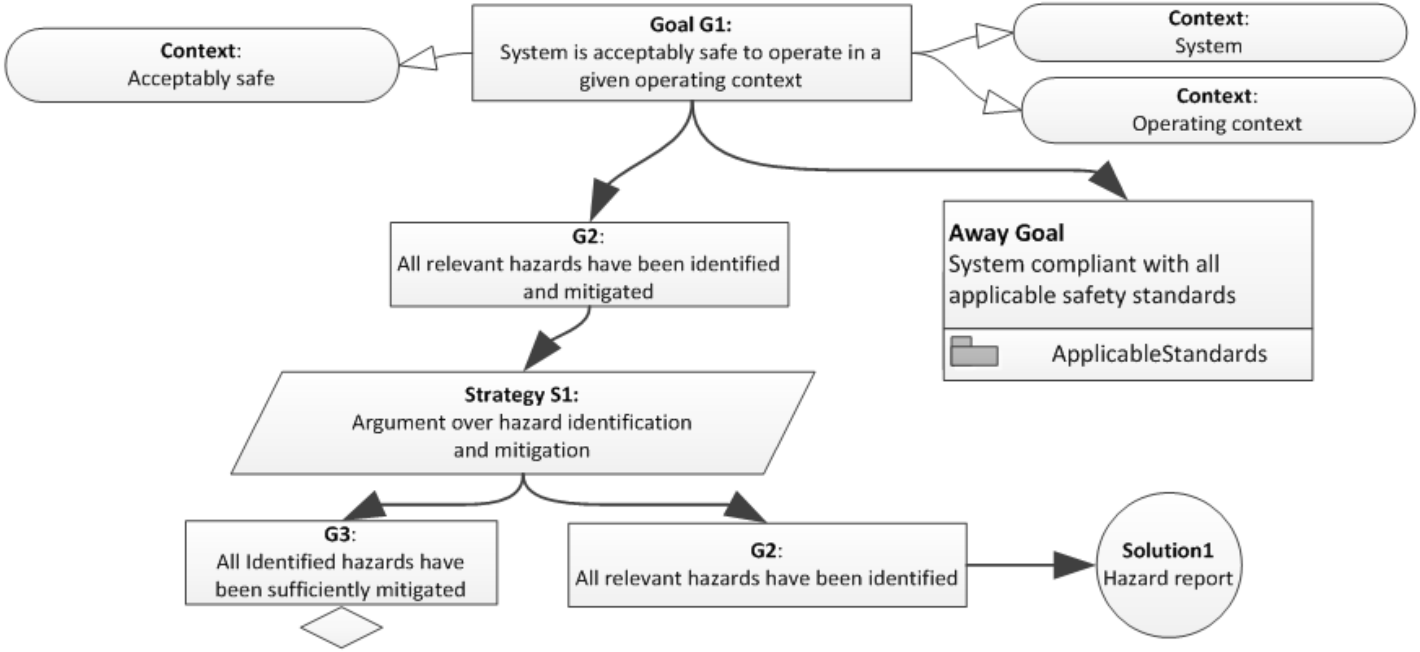
- A *safety case* is documented in form of a structured argument to clearly communicate that the system is acceptably safe to operate in a given context [Kelly, 1998]



- *Safety argument* is the "spine" of the safety case showing how safety objectives/ requirements are connected with evidence

- *Assurance case – safety case generalisation*
- *Goal Structuring Notation* (GSN) - a graphical argumentation notation that can be used to specify elements of any argument [GSN, 2011]

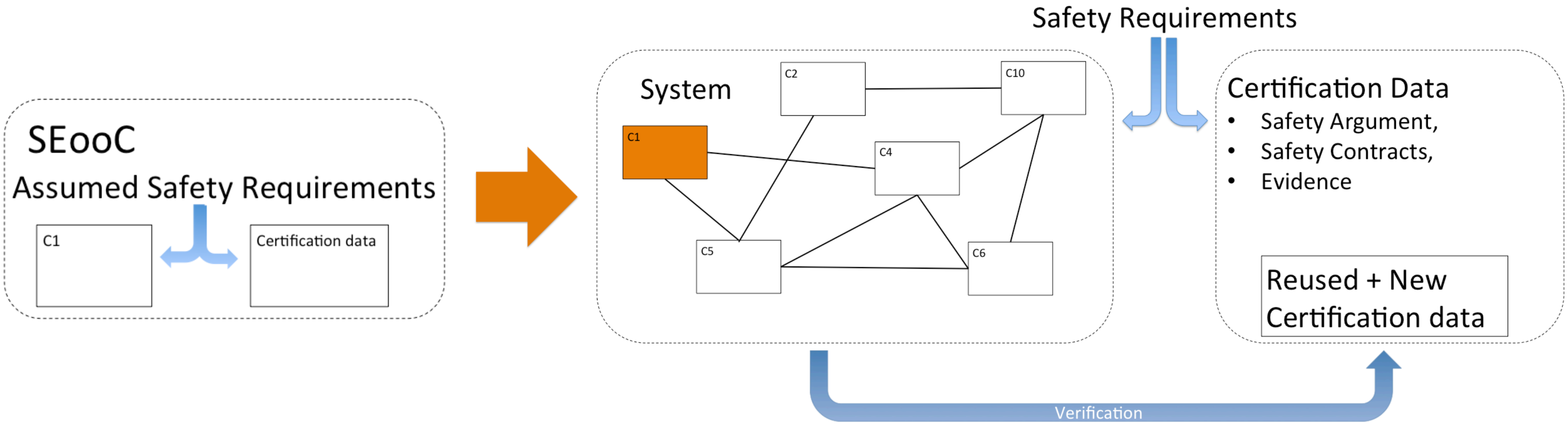
GSN – An Argument Example


Goal G1:
System is acceptably safe to operate in a given operating context
Context:
Acceptably safe
Context:
System
Context:
Operating context
G2:
All relevant hazards have been identified and mitigated
Away Goal
System compliant with all applicable safety standards
ApplicableStandards
Strategy S1:
Argument over hazard identification and mitigation
G3:
All Identified hazards have been sufficiently mitigated
G2:
All relevant hazards have been identified
Solution1
Hazard report

- We can read the initial goal structure as follows:
  - *The system is acceptably safe to operate in a given operating context when all relevant hazards have been identified and the system is compliant with all applicable safety standards.*
    - The context statements define what acceptably safe, system and operating context mean.

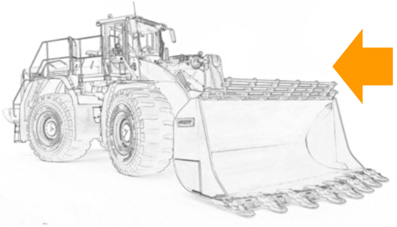
Safety-Critical Systems and Reuse

- To fully benefit from reuse of safety elements, safety artefacts describing safety-relevant reasoning should be reused as well, e.g.,:
  - Safety case arguments
  - The supporting certification data (e.g., system models, specifications, test cases, simulation results etc)
- *Safety Element out of Context (SEooC)* is a notion of reusable components proposed in Automotive ISO 26262 functional safety standard



Reusable Safety Element Example

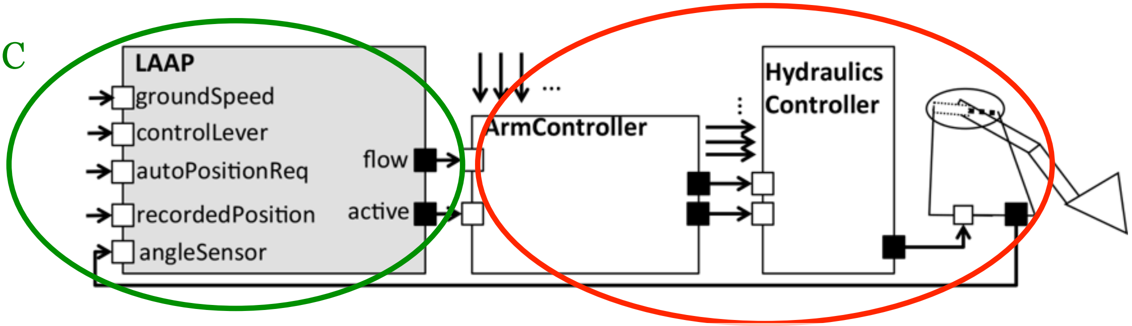
- Lifting Arm Automatic Positioning (LAAP)
- Different target systems (e.g., some wheel-loaders support “settable” automatic positioning and some have fixed position)



GWL – used in controlled environments for heavy lifting

CWL – used for tasks that require high precision

We develop LAAP as SEooC



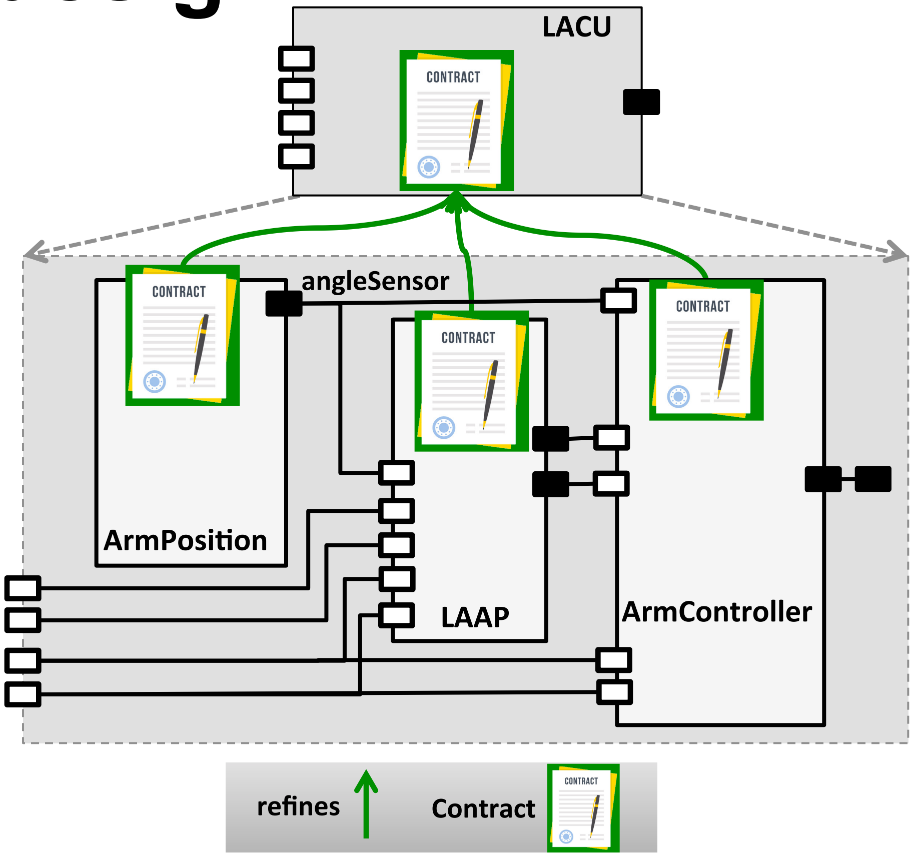
Information related to the system design and functional safety concept shall be:

- Assumed
- Considered
- Validated

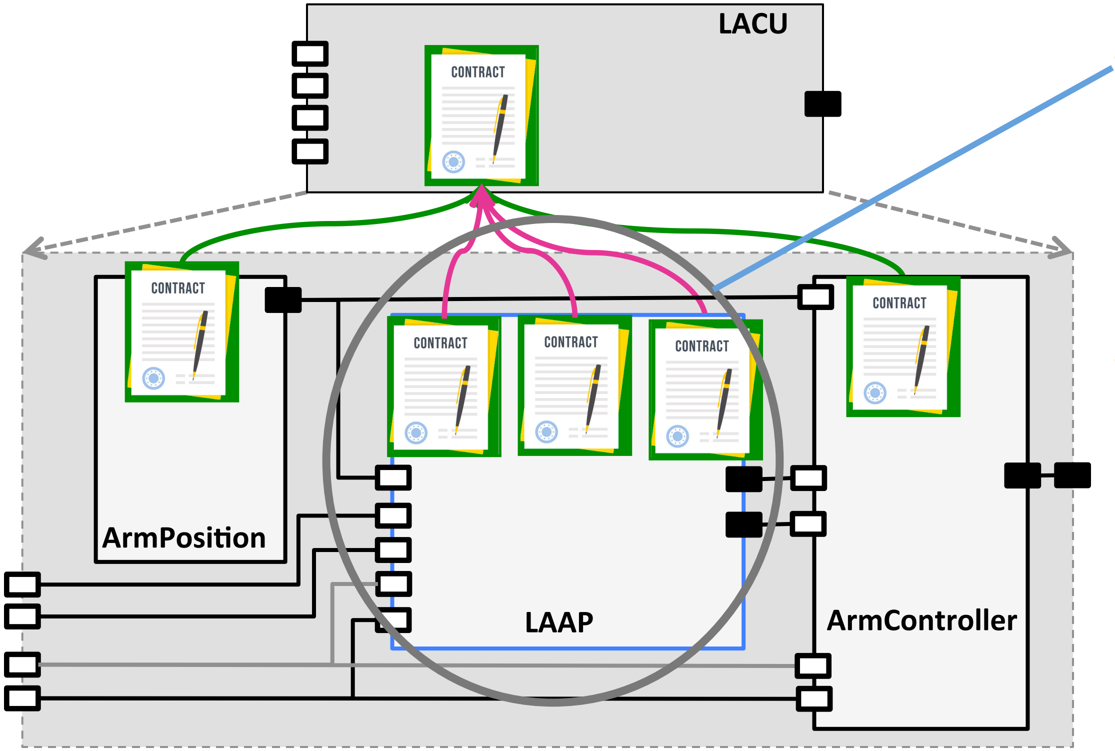
Contract-based design through assumption-guarantee contracts offer a way to achieve this!

Contract-based design

- *A contract of a component is a pair of assumptions and guarantees*
  - Specified on each component in hierarchy
  - Contracts on different levels explicitly connected with refinement relationship
    - E.g., following requirement decomposition



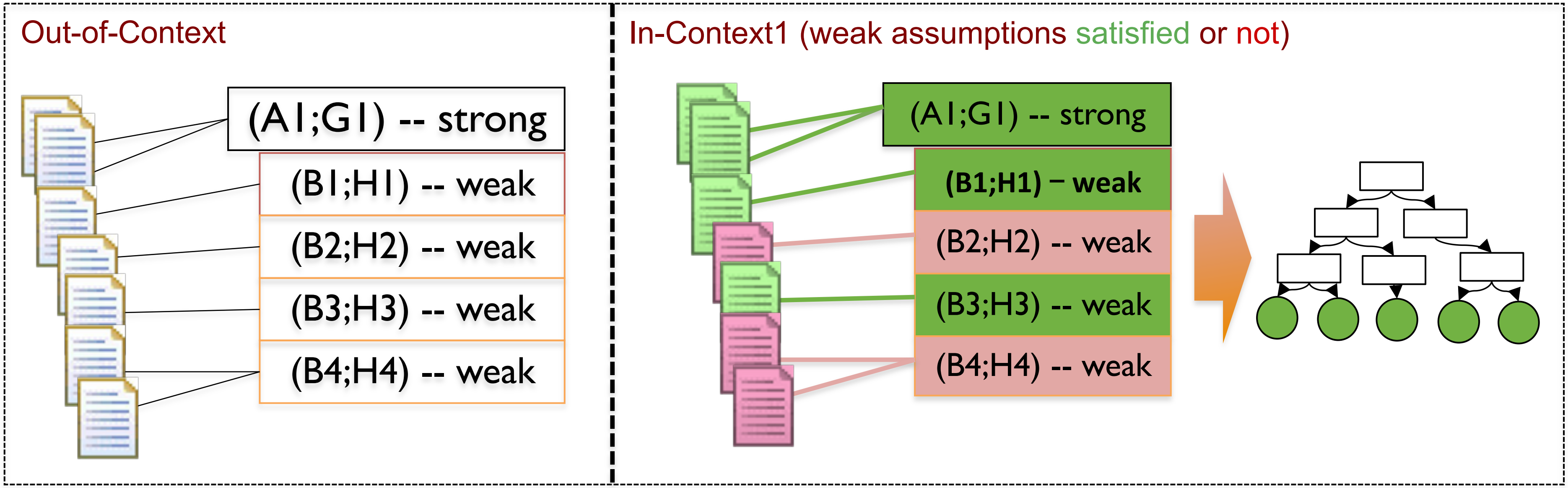
Reusable components in Contract-based design



- Reusable component
  - It has a set of weak contracts and not all are relevant/applicable in this environment
- To successfully check refinement, the "right" weak contracts should be identified and selected for this environment

- Strong and weak contracts were introduced as a way to provide this variability management
  - Strong contracts should be satisfied in every context in which the component is used
  - Weak guarantees are context-specific. They are offered only when besides the strong assumptions, weak assumptions are satisfied as well

Product-based reuse in AMASS platform: The idea



Out-of-Context

(A1;G1) -- strong

(B1;H1) -- weak

(B2;H2) -- weak

(B3;H3) -- weak

(B4;H4) -- weak

In-Context1 (weak assumptions satisfied or not)

(A1;G1) -- strong

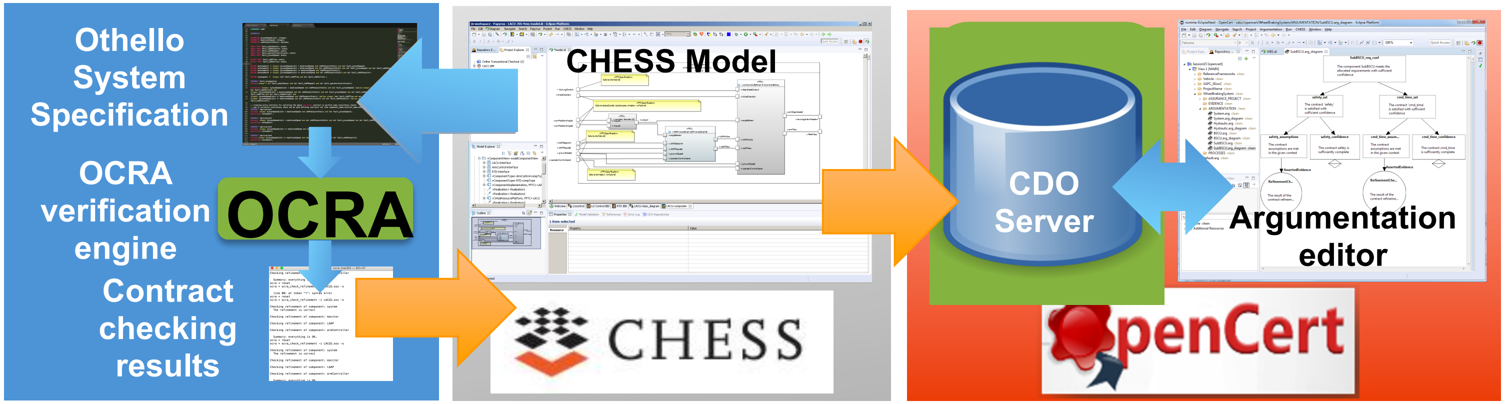
(B1;H1) – weak

(B2;H2) -- weak

(B3;H3) -- weak

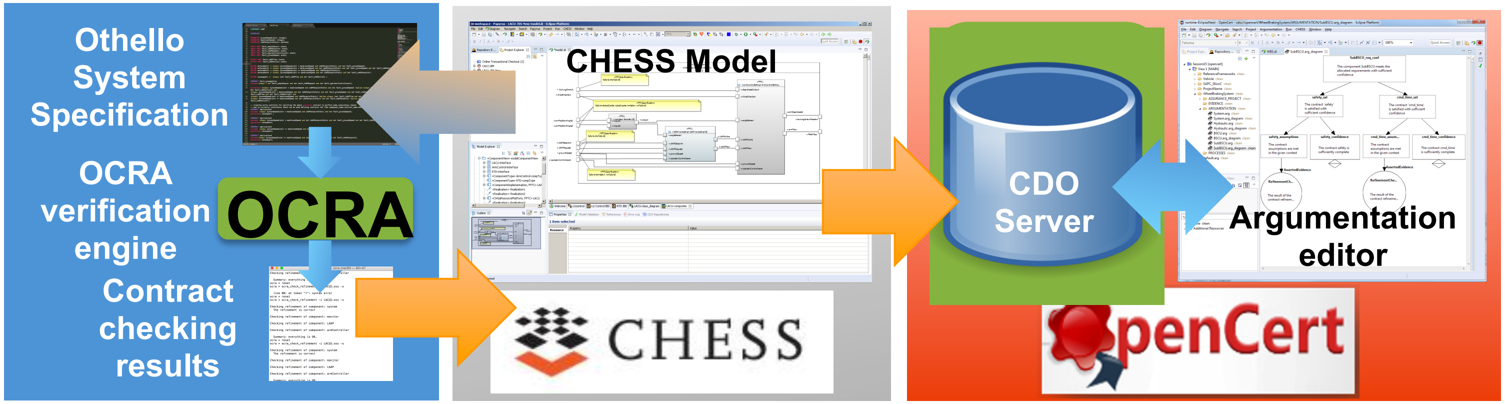
(B4;H4) -- weak

AMASS Platform: Supporting SEooC Reuse



- To support SEooC reuse and assurance we needed to:
  - Enrich CHESS meta-model to capture all the different certification data
  - Connect with OpenCert to automate transformation of the data from CHESS model to the corresponding Safety Case in OpenCert
  - Utilise OCRA contract checking to identify the relevant specification, hence the certification data
- AMASS Platform: https://www.polarsys.org/opencert/

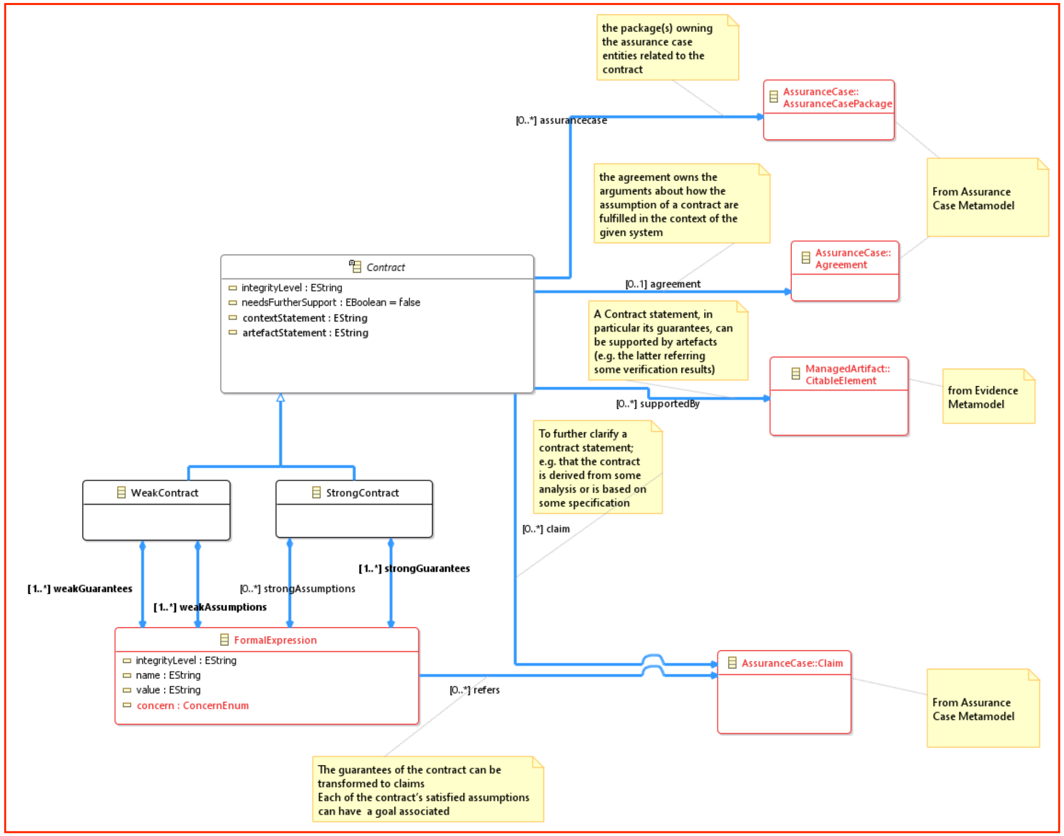
AMASS Platform: Supporting SEooC Reuse



- To support SEooC reuse and assurance we needed to:
  - Enrich CHESS meta-model to capture all the different certification data
  - Connect with OpenCert to automate transformation of the data from CHESS model to the corresponding Safety Case in OpenCert
  - Utilise OCRA contract checking to identify the relevant specification, hence the certification data

Extending the CHESS MM

- We associate assurance assets with the contracts to support automated assurance from such contracts



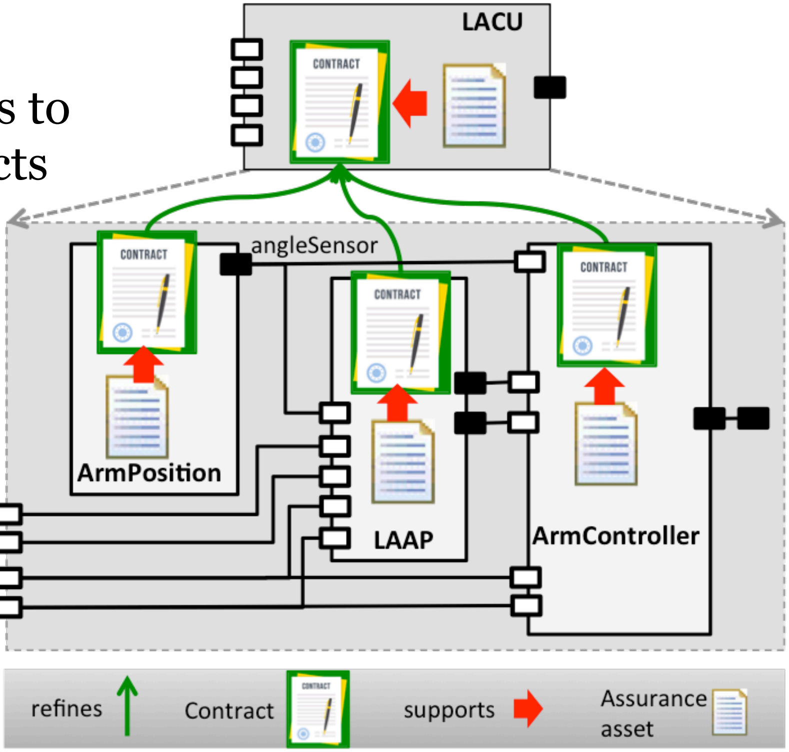
the package(s) owning the assurance case entities related to the contract

the agreement owns the arguments about how the assumption of a contract are fulfilled in the context of the given system

A Contract statement, in particular its guarantees, can be supported by artefacts (e.g. the latter referring some verification results)

To further clarify a contract statement; e.g. that the contract is derived from some analysis or is based on some specification

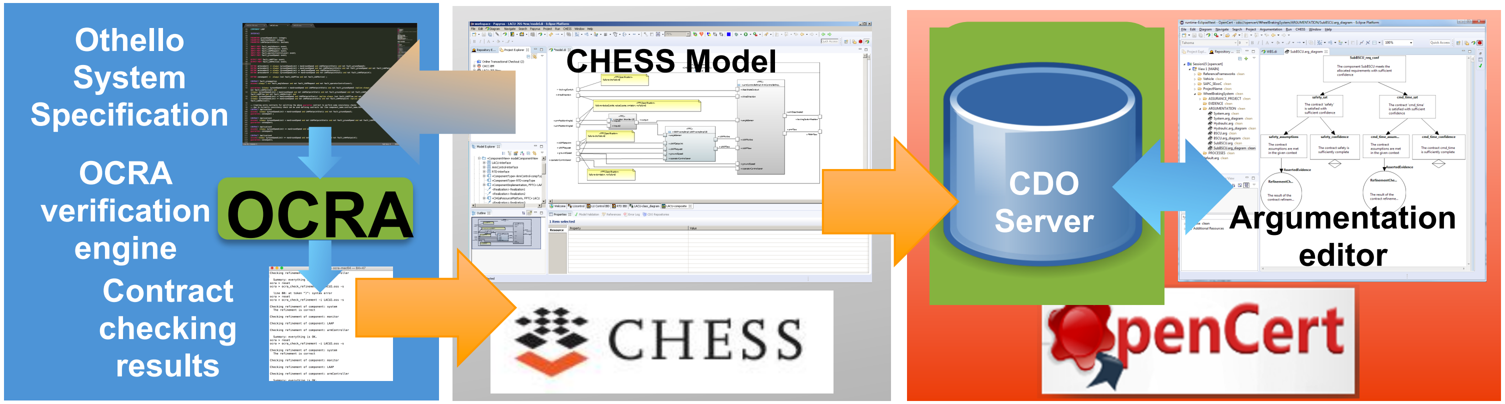
The guarantees of the contract can be transformed to claims
Each of the contract's satisfied assumptions can have a goal associated



Key MM relations for assurance

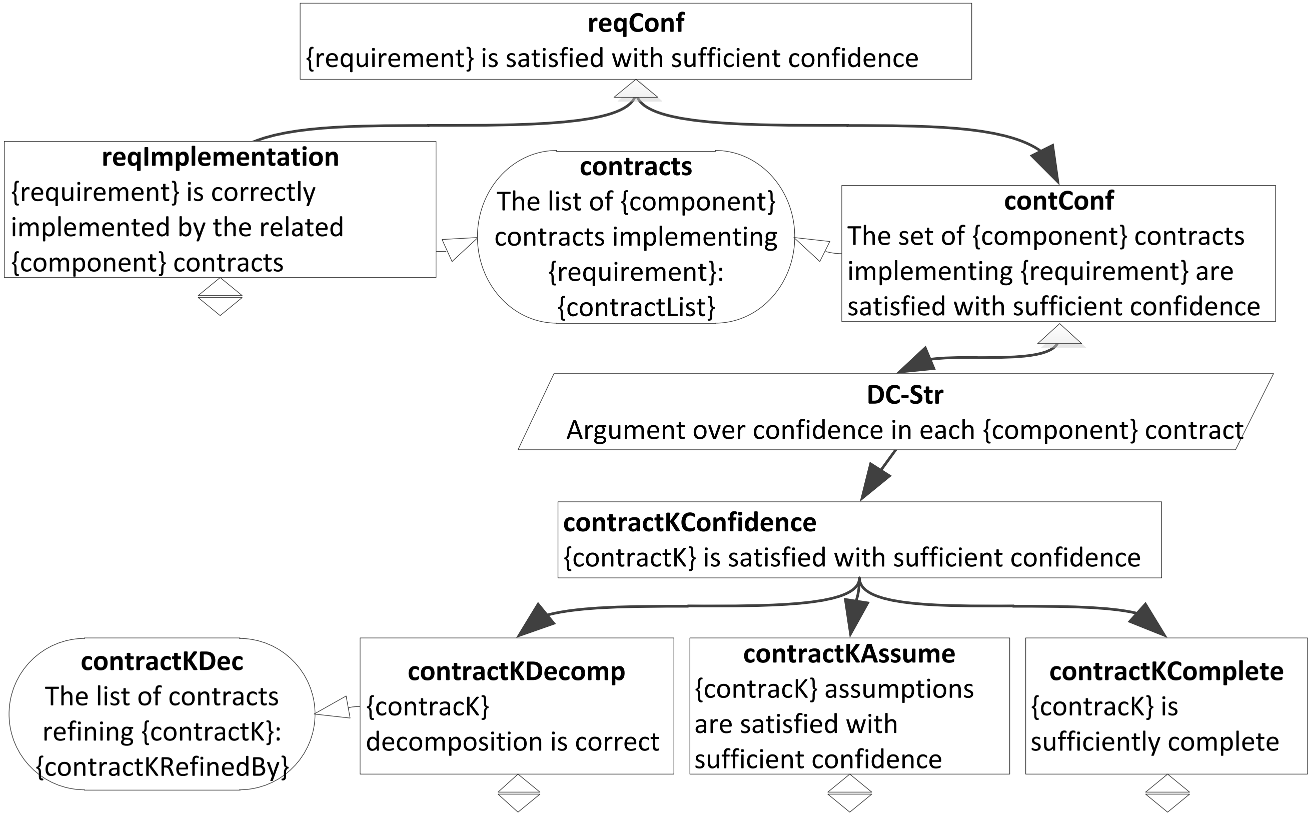
- Requirement to contract
- Context statement to contract
- Assurance elements (the certification data) to contracts

AMASS Platform: Supporting SEooC Reuse



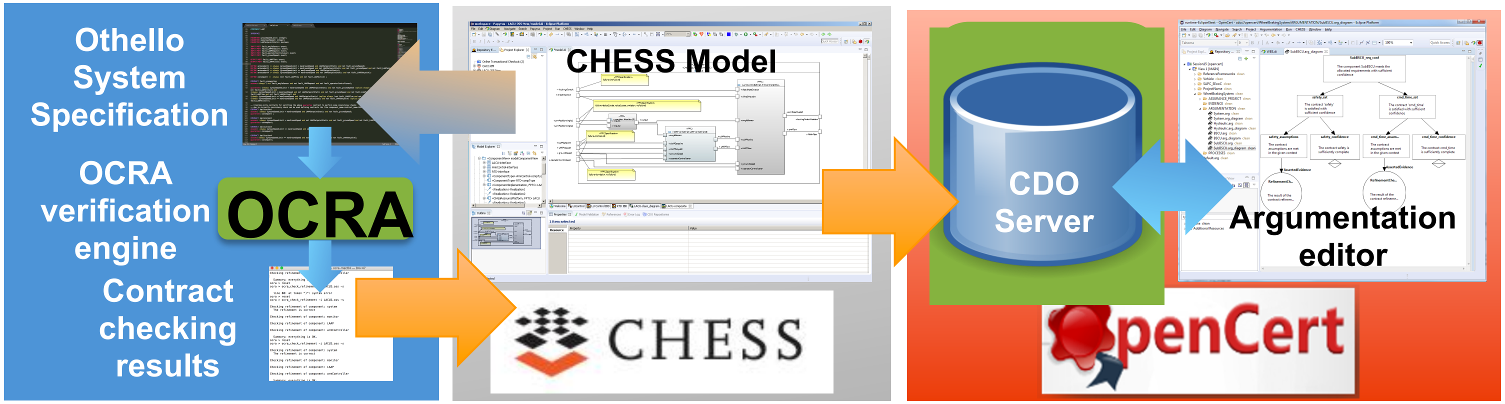
- To support SEooC reuse and assurance we needed to:
  - Enrich CHESS meta-model to capture all the different certification data
  - Connect with OpenCert to automate transformation of the data from CHESS model to the corresponding Safety Case in OpenCert
  - Utilise OCRA contract checking to identify the relevant specification, hence the certification data

The target argumentation pattern


reqConf
{requirement} is satisfied with sufficient confidence
reqImplementation
{requirement} is correctly implemented by the related {component} contracts
contracts
The list of {component} contracts implementing {requirement}: {contractList}
contConf
The set of {component} contracts implementing {requirement} are satisfied with sufficient confidence
DC-Str
Argument over confidence in each {component} contract
contractKConfidence
{contractK} is satisfied with sufficient confidence
contractKDec
The list of contracts refining {contractK}: {contractKRefinedBy}
contractKDecomp
{contracK} decomposition is correct
contractKAssume
{contracK} assumptions are satisfied with sufficient confidence
contractKComplete
{contracK} is sufficiently complete

The automated generation of argument-fragments is done by automatically instantiating the given pattern for each component in the CHESS model

AMASS Platform: Supporting SEooC Reuse



- To support SEooC reuse and assurance we needed to:
  - Enrich CHESS meta-model to capture all the different certification data
  - Connect with OpenCert to automate transformation of the data from CHESS model to the corresponding Safety Case in OpenCert
  - Utilise OCRA contract checking to identify the relevant specification, hence the certification data

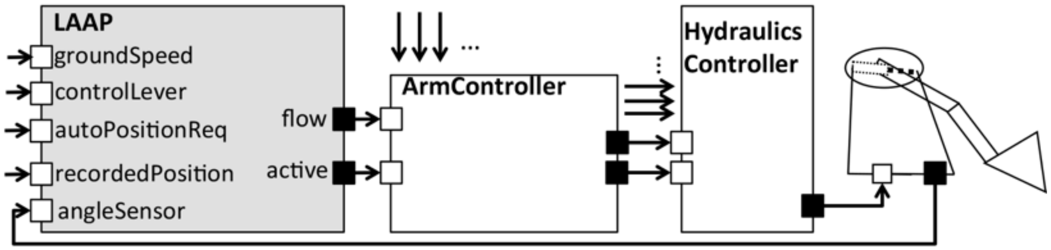
Contract variability in CHESS/ OCRA

- OCRA does not explicitly provide support for strong and weak contracts
- We have added two ways of checking refinement with strong/weak contracts in CHESS
  - Transform all weak contracts into strong
    - Refinement can be performed without prior selection for all strong/weak contracts
    - Identification of the relevant weak is still needed for contract-based assurance
    - Supports refinement check with no manual effort, but not fine-grained reuse
  - Preselect only the relevant weak contracts and treat them as strong without transformation
    - We've added support for automatic selection of the relevant weak contracts
    - The current automated selection is done based on the weak contract validity, but does not include relation to requirements. Some manual adaptations of the selection might be needed.
    - Supports both refinement check and fine-grained reuse

Product-based reuse in AMASS platform: Summary

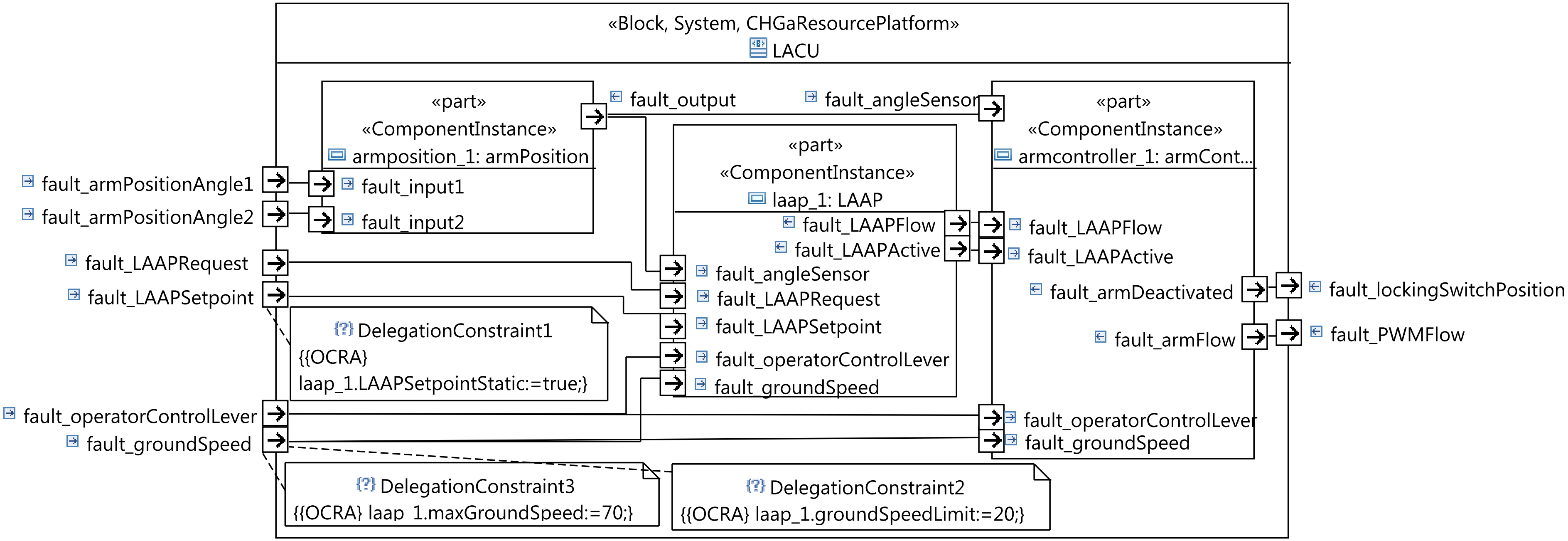
- What we had before in AMASS platform:
  - Specify strong and weak contracts
  - Translate to OCRA for refinement checks with manual selection of weak contracts
- What we contributed to AMASS Platform:
  - Assurance elements to contracts traceability (the metamodel extension)
  - Automatic selection/filtering of the weak contracts applicable in the given environment
  - Strong/weak contract transformation for OCRA verification engine
  - Automated argument-fragment generation from the filtered/selected contracts/evidence (pattern instantiation)

Loading Arm Case Study



- Hazards:
  - Unintended arm movement
  - Arm movement during high speed
- Some of the safety requirements:
  - SR1: The stop position of the loading arm shall not deviate more than +-0.04 rad
  - SR2: The loading arm shall be disabled during high speed

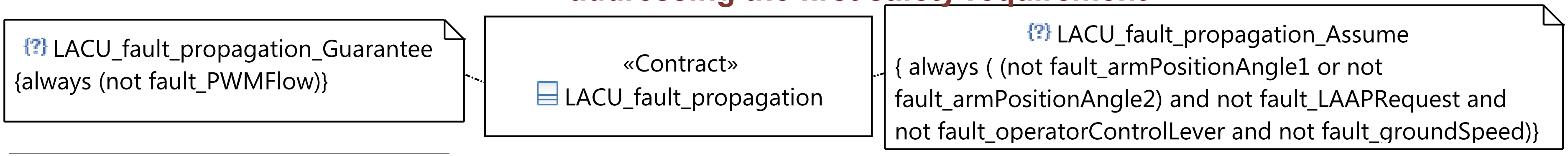
CHESS model



At this point we set the values of various configurable parameters used by the reusable component, e.g., LAAPSetpointStatic, maxGroundSpeed, groundSpeedLimit

The contracts based on failure propagation analysis

LACU – the software controller of the loading arm system and its contract addressing the first safety requirement

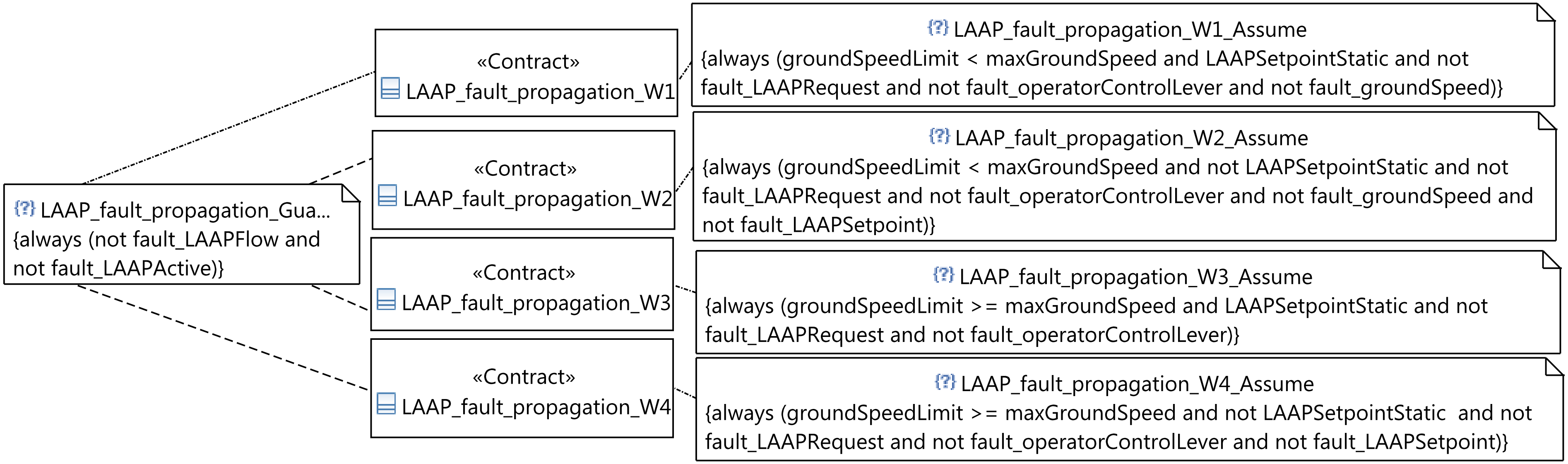


LACU_fault_propagation_Guarantee
{always (not fault_PWMFlow)}

«Contract»
LACU_fault_propagation

LACU_fault_propagation_Assume
{ always ((not fault_armPositionAngle1 or not fault_armPositionAngle2) and not fault_LAAPRequest and not fault_operatorControlLever and not fault_groundSpeed)}

LAAP – the subcomponent of LACU and its weak contracts refining the LACU contract



LAAP_fault_propagation_Gua...
{always (not fault_LAAPFlow and not fault_LAAPActive)}

«Contract»
LAAP_fault_propagation_W1

LAAP_fault_propagation_W1_Assume
{always (groundSpeedLimit < maxGroundSpeed and LAAPSetpointStatic and not fault_LAAPRequest and not fault_operatorControlLever and not fault_groundSpeed)}

«Contract»
LAAP_fault_propagation_W2

LAAP_fault_propagation_W2_Assume
{always (groundSpeedLimit < maxGroundSpeed and not LAAPSetpointStatic and not fault_LAAPRequest and not fault_operatorControlLever and not fault_groundSpeed and not fault_LAAPSetpoint)}

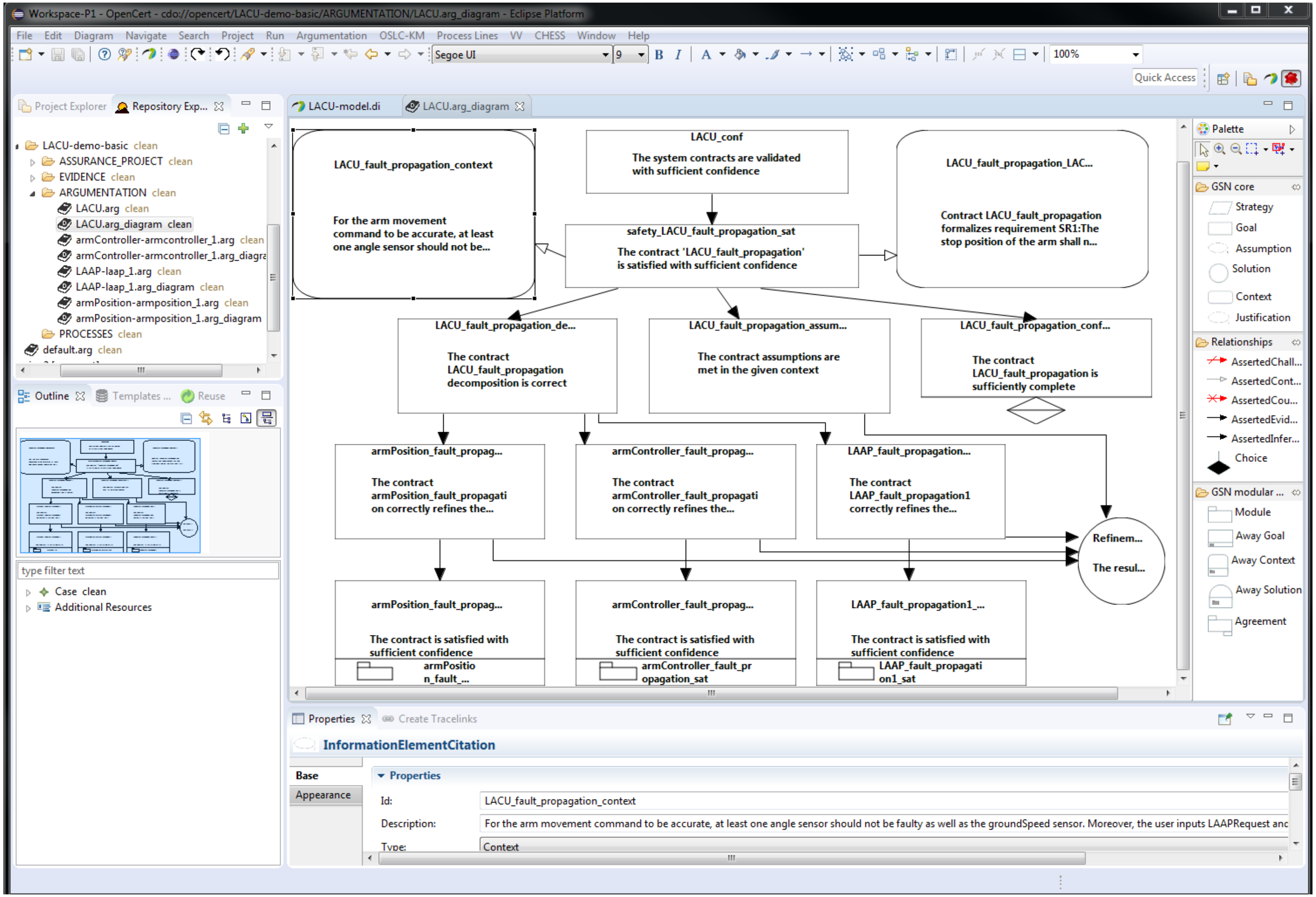
«Contract»
LAAP_fault_propagation_W3

LAAP_fault_propagation_W3_Assume
{always (groundSpeedLimit >= maxGroundSpeed and LAAPSetpointStatic and not fault_LAAPRequest and not fault_operatorControlLever)}

«Contract»
LAAP_fault_propagation_W4

LAAP_fault_propagation_W4_Assume
{always (groundSpeedLimit >= maxGroundSpeed and not LAAPSetpointStatic and not fault_LAAPRequest and not fault_operatorControlLever and not fault_LAAPSetpoint)}

OpenCert argument



LACU_fault_propagation_context

For the arm movement command to be accurate, at least one angle sensor should not be...

LACU_conf

The system contracts are validated with sufficient confidence

LACU_fault_propagation_LAC...

Contract LACU_fault_propagation formalizes requirement SR1:The stop position of the arm shall n...

safety_LACU_fault_propagation_sat

The contract 'LACU_fault_propagation' is satisfied with sufficient confidence

LACU_fault_propagation_de...

The contract LACU_fault_propagation decomposition is correct

LACU_fault_propagation_assum...

The contract assumptions are met in the given context

LACU_fault_propagation_conf...

The contract LACU_fault_propagation is sufficiently complete

armPosition_fault_propag...

The contract armPosition_fault_propagation correctly refines the...

armController_fault_propag...

The contract armController_fault_propagation correctly refines the...

LAAP_fault_propagation...

The contract LAAP_fault_propagation1 correctly refines the...

Refinem...

The resul...

armPosition_fault_propag...

The contract is satisfied with sufficient confidence

armPositio n_fault_...

armController_fault_propag...

The contract is satisfied with sufficient confidence

armController_fault_pr opagation_sat

LAAP_fault_propagation1_...

The contract is satisfied with sufficient confidence

LAAP_fault_propagati on1_sat

Id: LACU_fault_propagation_context

Description: For the arm movement command to be accurate, at least one angle sensor should not be faulty as well as the groundSpeed sensor. Moreover, the user inputs LAAPRequest anc

Type: Context

Conclusions and Future Work

- Contract-based design inherently supports reuse of component implementations
  - We've shown that it can be used to support fine-grained reuse of assurance assets (the certification data)
- The AMASS Platform enables the basic support for contract-driven reuse and assurance by tightly coupling system and assurance modelling via contract-based design

- Future directions
  - Enriching the underlying metamodels to allow for capturing additional information needed for reuse and assurance
  - The contract specification "layer" as a place where different requirements are formalised could be used for analysis of the interplay of multiple concerns such as safety, security and performance

Thank you!

Questions and comments?

SAFECOMP 2018

2018
September 18

Aros Congress Center
Västerås, Sweden

6
Workshops

3
Keynote speakers

MÄLARDALEN UNIVERSITY
SWEDEN

Keynotes:

- **Robyn Lutz**, "*Software Engineering for Safety in Molecular Programmed Systems*"
- **Uma Ferrell**, "*Reviews?! We do that! Cross-domain reuse of engineering knowledge and evidence*"
- **Richard Hendeberg**, "*Experiences from the industry, design and application of a control system platform for safety of machinery*"

Fast abstracts call still open until July 02
http://www.es.mdh.se/safecomp2018/fast-abstracts-call