AMASS

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

Architecture-Driven Assurance

First EAB Workshop
Trento, September 11, 2017

Stefano Puri
WP3 Leader

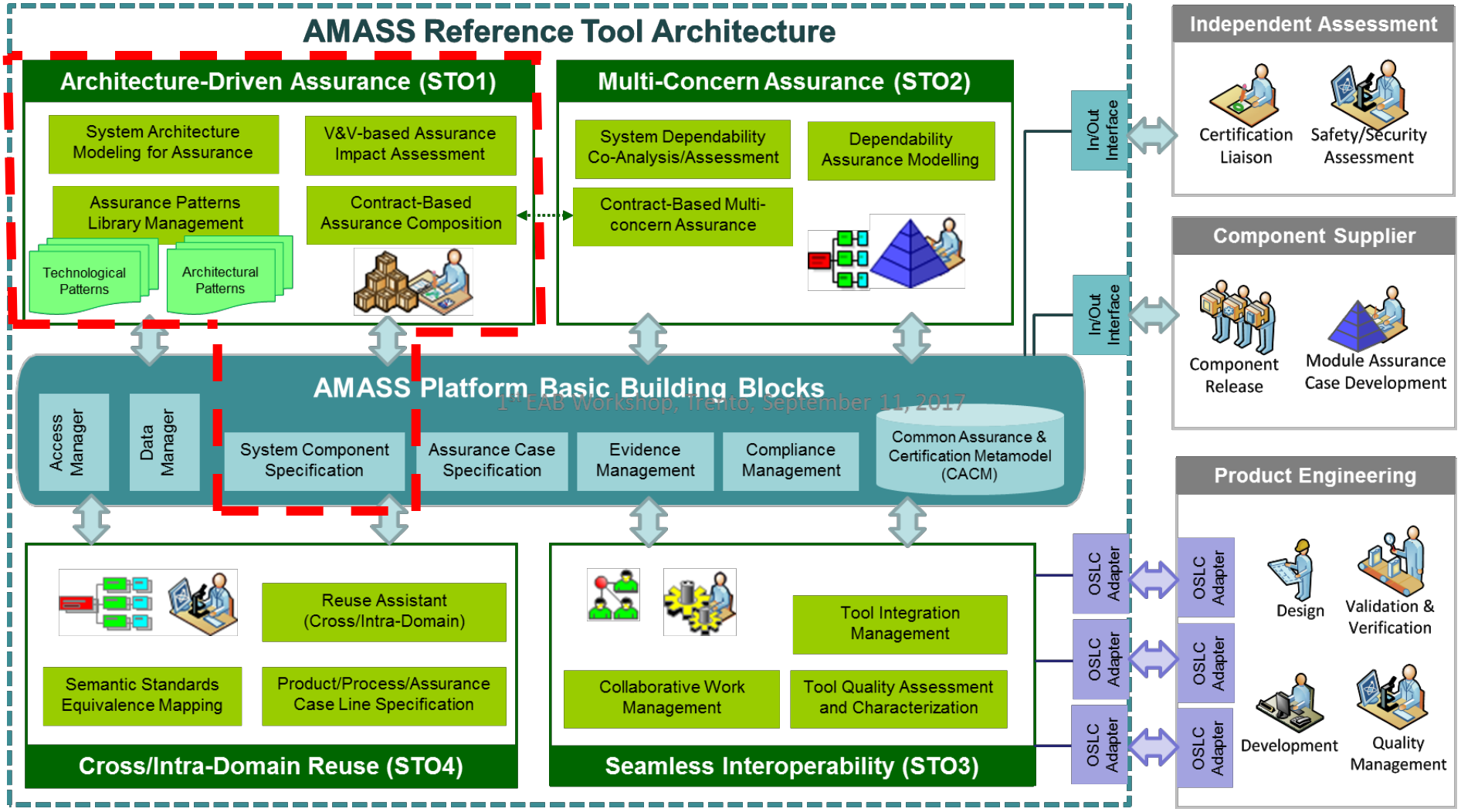
Architecture-Driven Assurance Objectives

Architecture-Driven Assurance (AiDA): an assurance that exploits and is linked to the system architecture in order to provide more structured evidences and arguments to show that the system is free of vulnerabilities.

Main specific objectives:

1. Determine the needs and constraints based on which the approach will be developed.
2. Provide a conceptual framework that meets the information needs for architecture-driven assurance.
3. Specify the design of AMASS building blocks for applying the architecture-driven approach.
4. To implement the AMASS tools above.
5. To provide methodological guidance for the application of the approach and the maintenance of its results.

AMASS Reference Tool Architecture: AiDA Scope

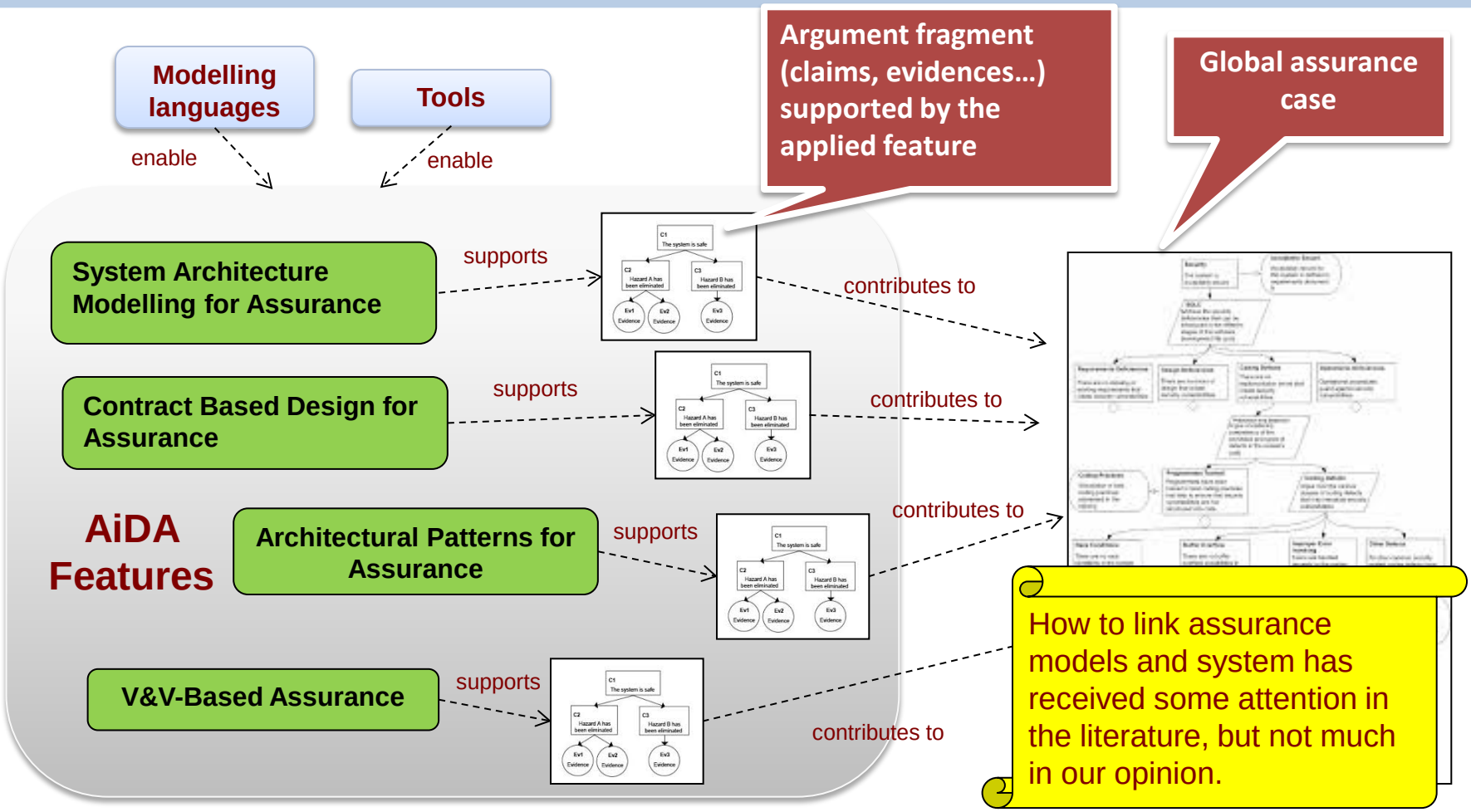


AiDA work areas

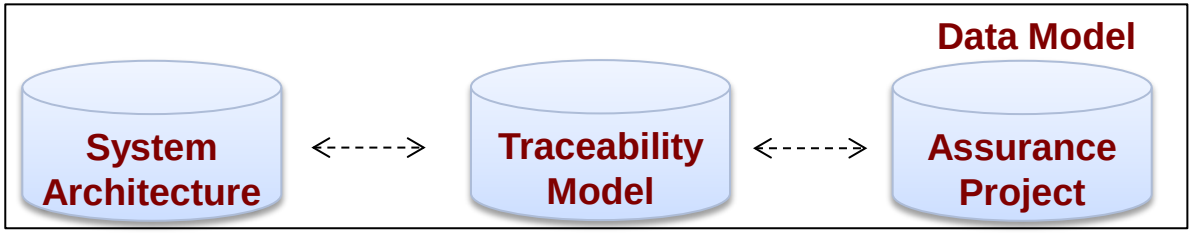
- System Architecture Modelling for Assurance
  - Exploit the system architecture in the assurance case
  - System architecture languages
  - Architecture trade-off and comparison
- Architectural Patterns for Assurance
  - Interaction between assurance and architectural patterns
  - Architectural patterns from standards
- Contract-based assurance
  - Assurance patterns for contract-based design
  - Enrich evidence produced by contract-based design
- V&V-based assurance
  - Enrich V&V techniques

<<Contract>>
Formalize properties of system and components in terms of assumptions and guarantees properties

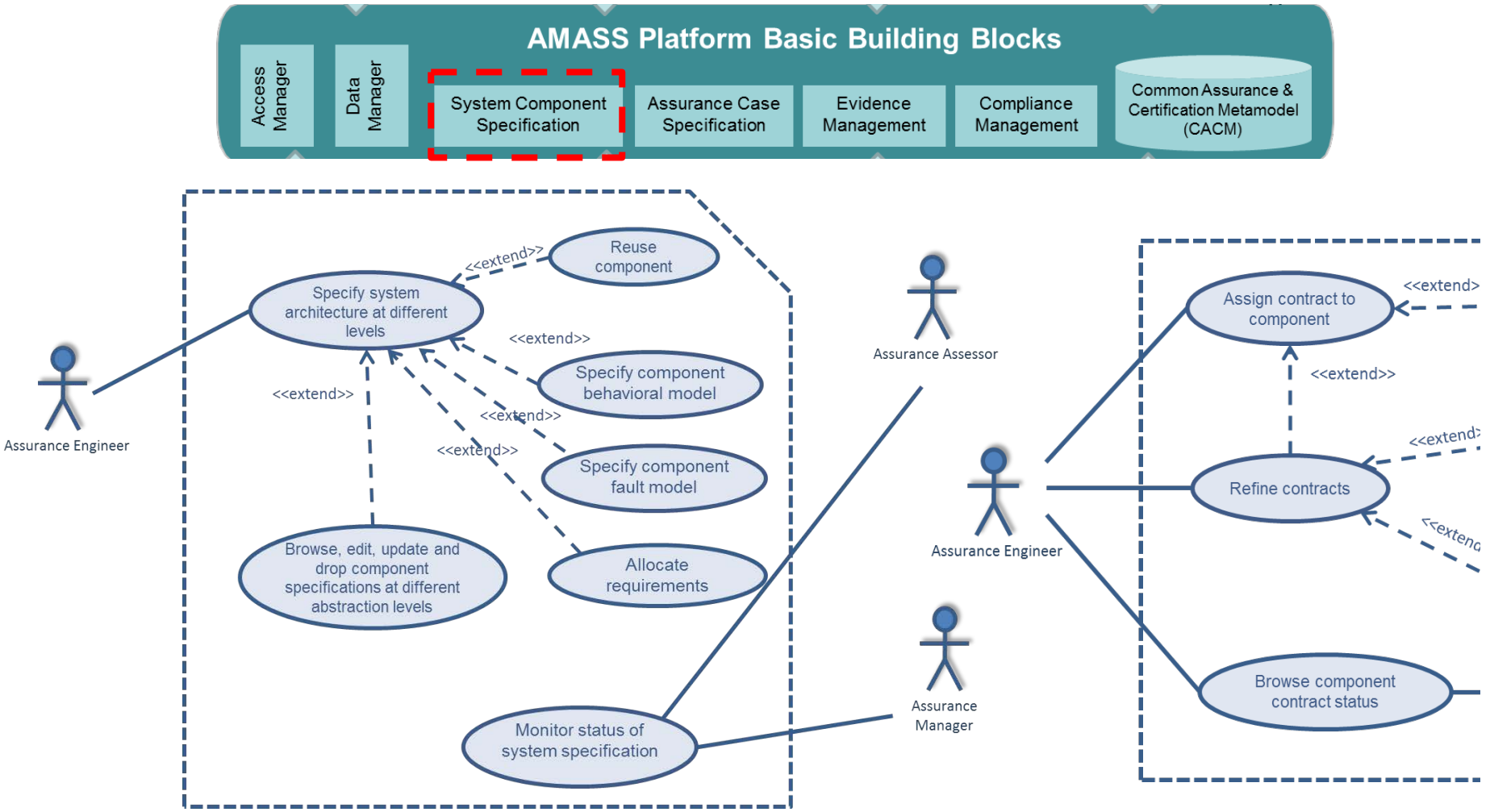
AiDA high-level logical architecture



The system architecture is used for model-driven engineering , supporting contract-based, pattern-based design, V&V

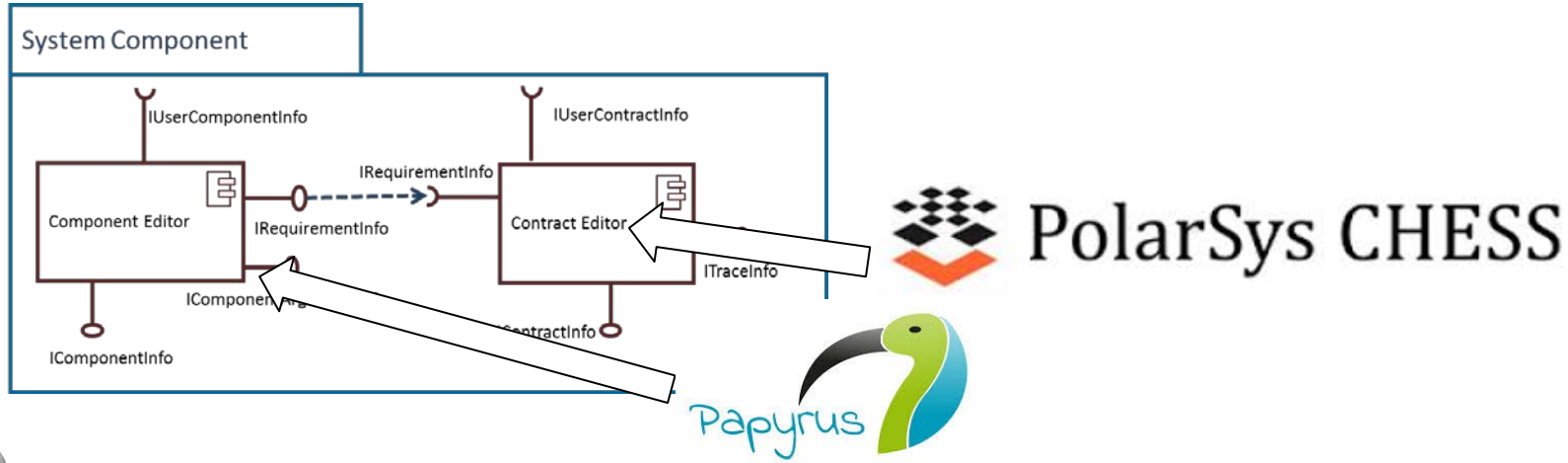


Architecture Driven Assurance Implementation

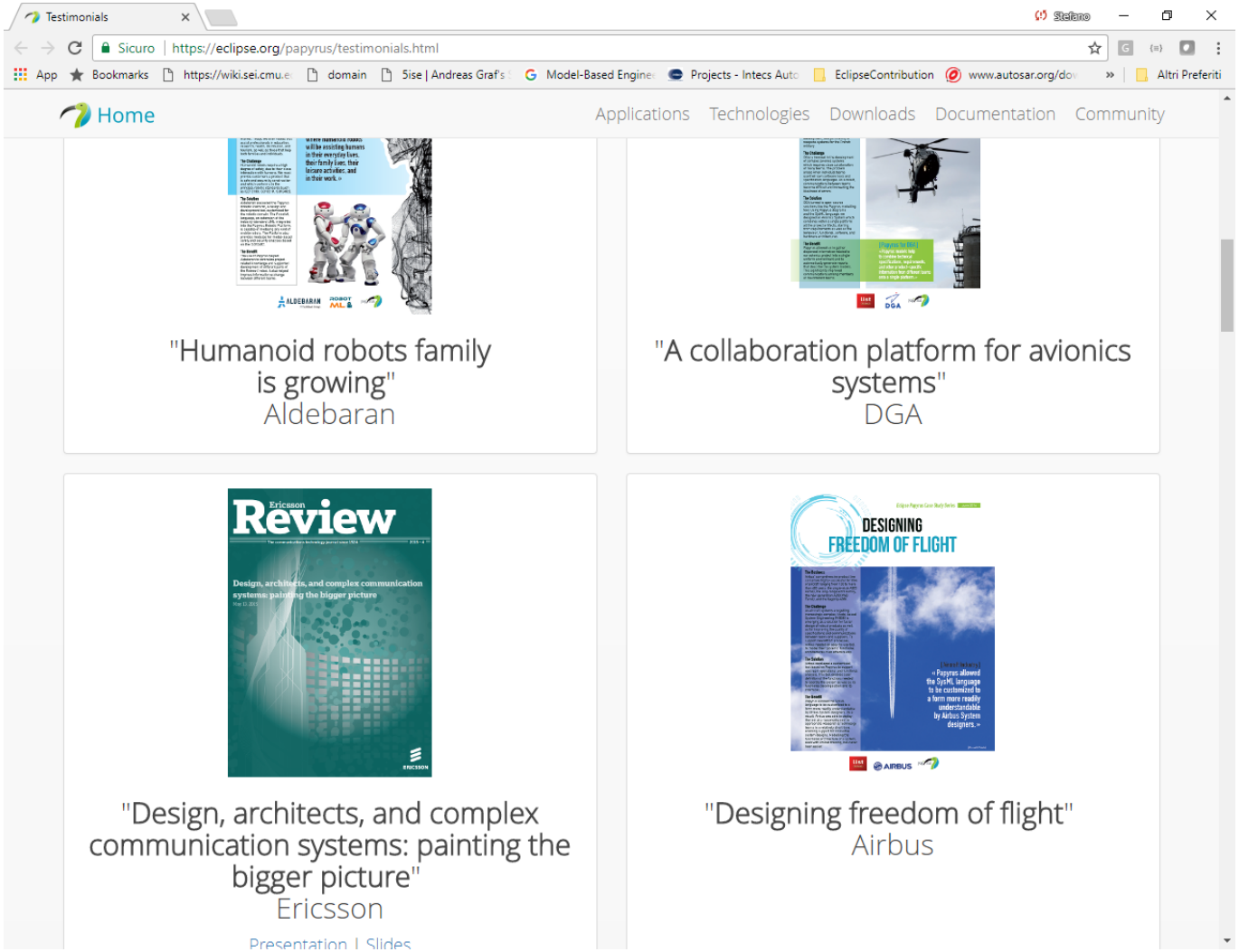


System Component Specification

- Polarsys Papyrus and CHESS projects as basic building blocks
  - Papyrus allows UML/SysML and DSL modelling facilities, integration with external models/tools (requirements import/export from different sources, import from Rhapsody…)
  - CHESS is methodology and toolset for model-based design, validation and implementation of high-critical software application (result of Artemis JU CHESS and CONCERTO projects)
    - Customizes UML/SysML/MARTE and Papyrus tool
    - Provides support for contract-based design
      - Originally created in the context of SafeCer project
      - Extended in AMASS to support AiDA meta-model



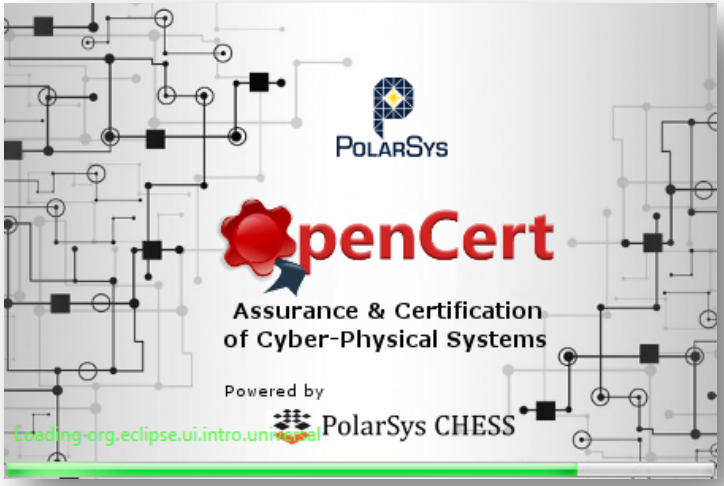
Papyrus Use Case Stories



"…partners have already built industrial tools based on Papyrus to support their domains and their customers."

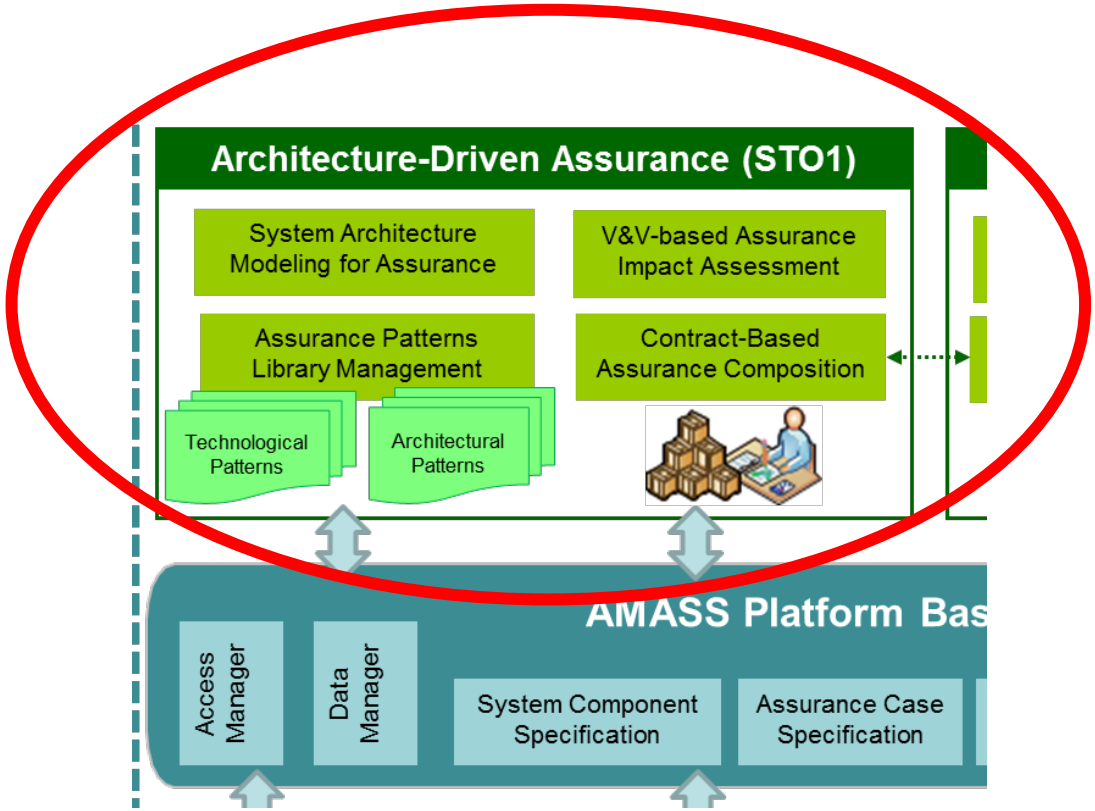
OpenCert: powered by CHESS

- CHESS has been extended to allow modelling traceability between contracts/formal properties, claims and evidences
- Demo…



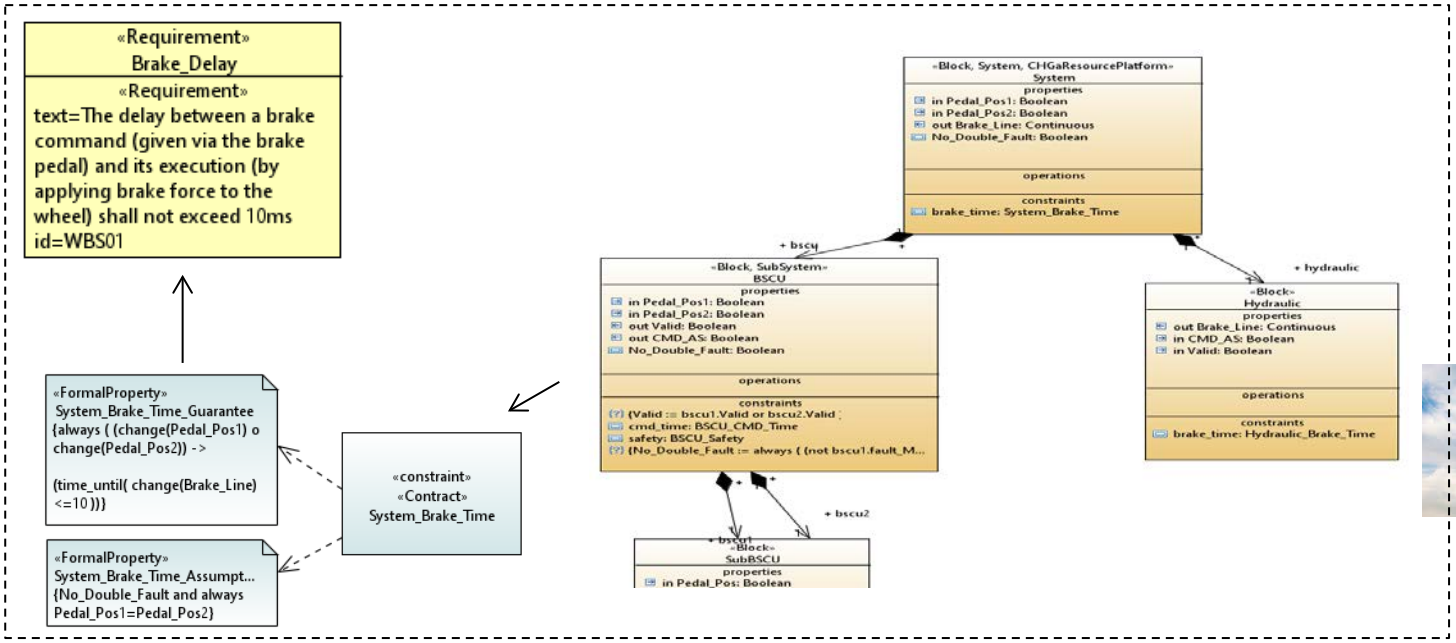
UC3-chess-opencert.mp4

Features for Architecture-Driven Assurance



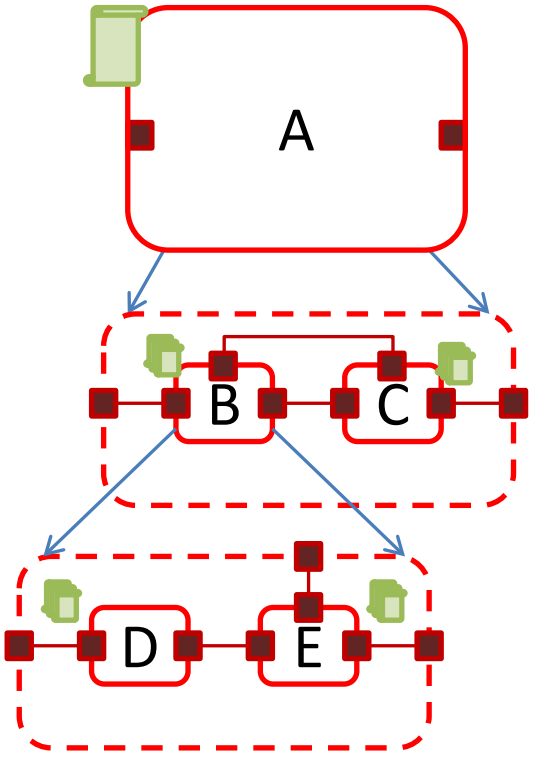
System Architecture Modelling for Assurance

- Functionalities that are focused on the modelling of the system architecture to support the system assurance



Contract-based design for Assurance

- Contract-Based Design for Assurance: functionalities that support the contract-based design of the system architecture, which provides additional arguments and evidence for the system assurance



Contract-based verification

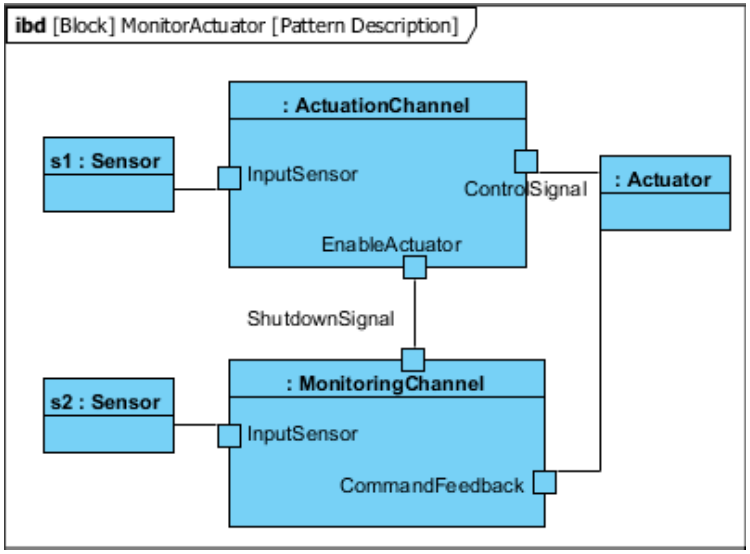
Components reuse that is supported by checks on the contracts.

Generation of assurance arguments from the contract specification and validation

Defining system boundaries is a difficult task…

Architectural Patterns for Assurance

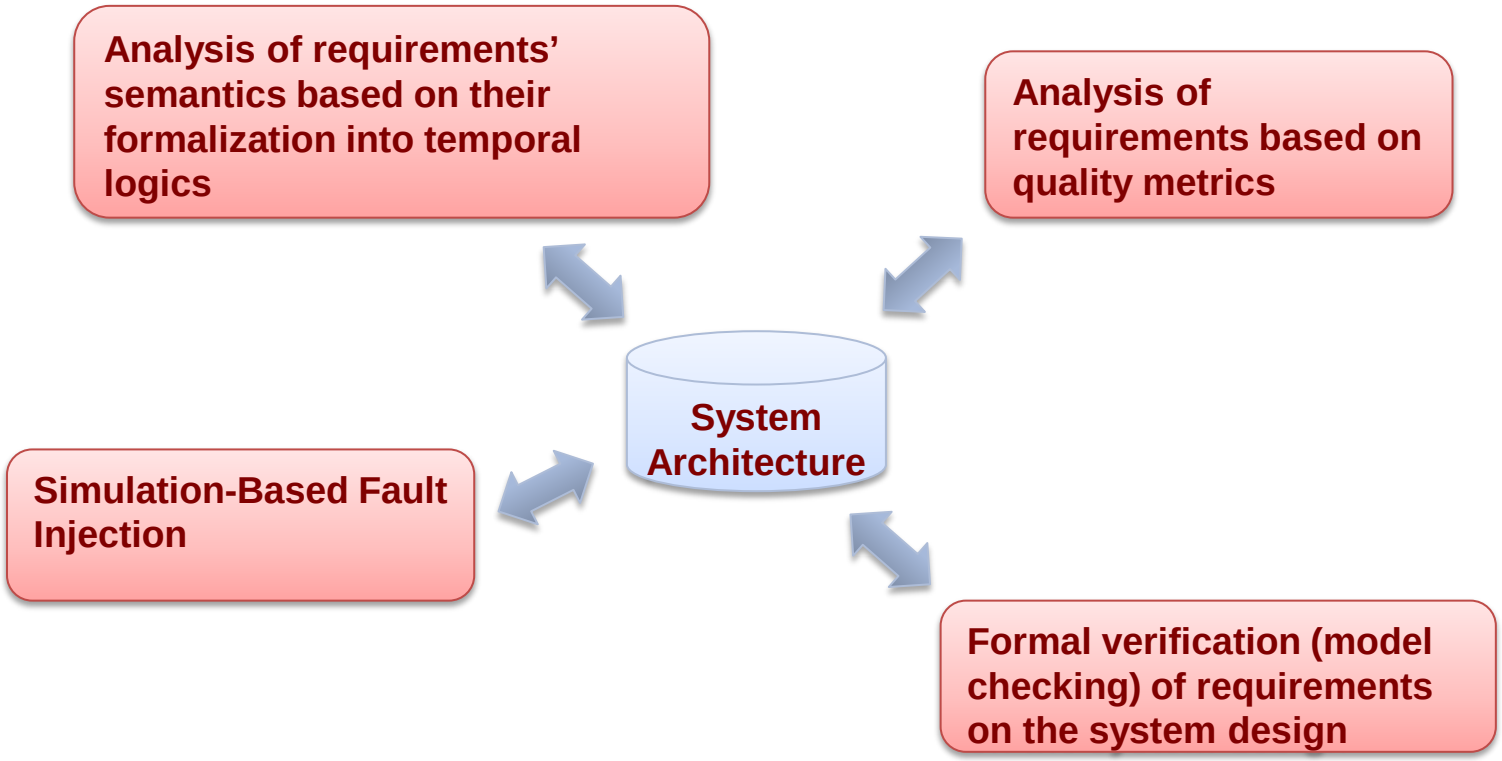
- Architectural Patterns for Assurance: functionalities that are focused on architectural patterns to support the system assurance
  - Management of a library of architectural patterns
  - Application of architectural patterns by using associated contracts



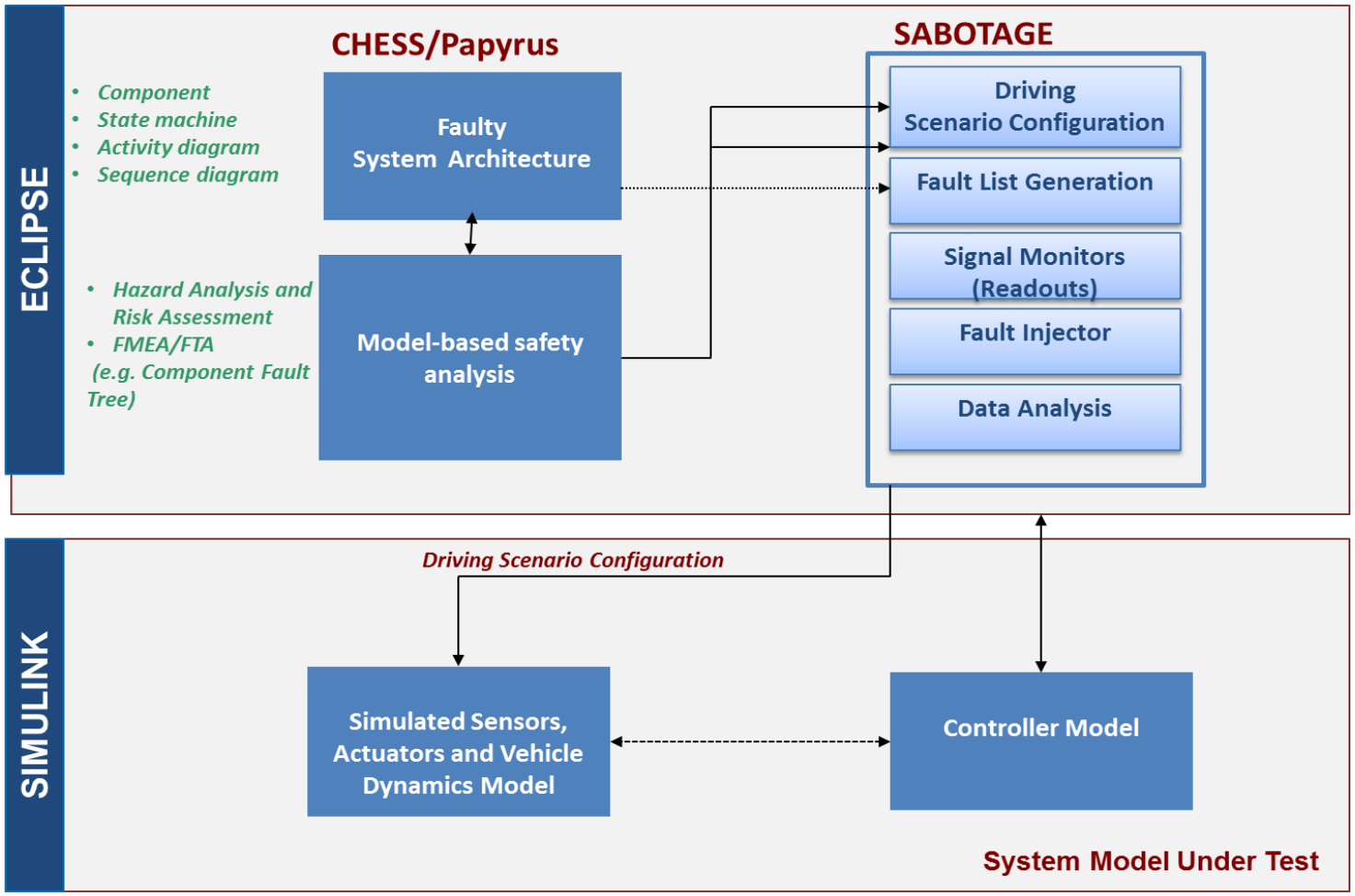
Pattern Name	
…	…
Context	define in which context the pattern is used. For example, define if the pattern is recommended for a specific safety-critical domain.
PatternAssumptions	the contract assumptions related to the design pattern.
PatternGuarantees	the contract guarantees related to the design pattern.

V&V-Based Assurance

- Functionalities that are focused on advanced analysis to enrich the evidence of the assurance case.



V&V ex1: Simulation-Based Fault Injection

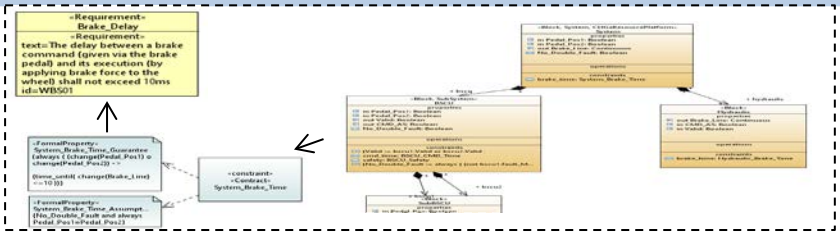


G. Juez et al, "Early Safety Assessment of Automotive Systems Using Sabotage Simulation-Based Fault Injection Framework", SAFECOMP 2017

G. Juez et al, “Safety Assessment of Automated Vehicle Functions by Simulation-based Fault Injection”, ICVES 2017

G. Juez et al “Fault Injection method for Safety and Controllability Evaluation of Automated Driving”, IEEE IV 2017

V&V ex2: Contract-based verification and analysis



MODELING \ ANALYSIS	V & V	Safety Assessment: Fault extension	Safety Assessment: Fault trees computation
Architecture decomposition & Contracts	• Automatic contract refinement verification **OCRA**	• Automatic fault extension **OCRA**	• Automatic hierarchical fault tree generation • Over-approximation **OCRA**
Behavioral Implementation (Leaf components & System)	• Automatic compositional verification **OCRA** • Automatic monolithic verification **nuXmv**	• Failure modes defined by the user • Generation of the extended system implementation **xSAP**	• Automatic flat fault tree generation **xSAP**
	$M \models \varphi$	$M \rightsquigarrow M_{[F]}$	$\delta(F) : M_{[F]} \not\models \varphi$

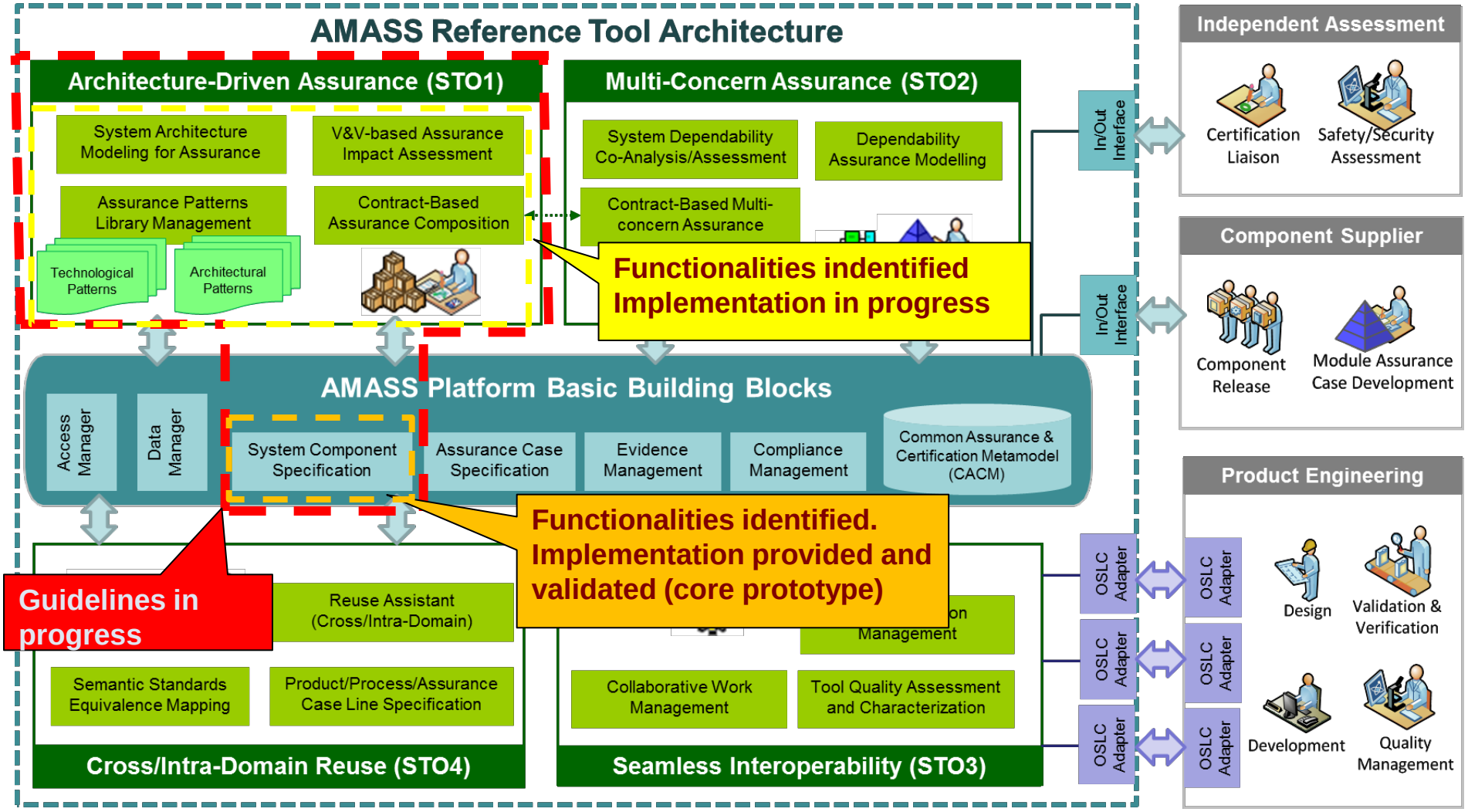
Semi-automatic Generation

M

Main Achievements

- Way forward has been identified with respect to the current state of the art and state of the practice
- Initial definition of the system component metamodel for assurance, with links between architectural and assurance case entities
- Basic building blocks for system architecture specification have been provided
- Initial definition of the conceptual level for architecture-driving assurance, as set of functionalities to be provided on top of the AMASS platform

AMASS Reference Tool Architecture, AiDA Scope: implementation status



Summary and future work

- Definition of techniques and functional architecture supporting AiDA
  - Assurance case fragments definition
- Definition of guidelines
- Support for the AMASS second prototype

Thank you for your attention!