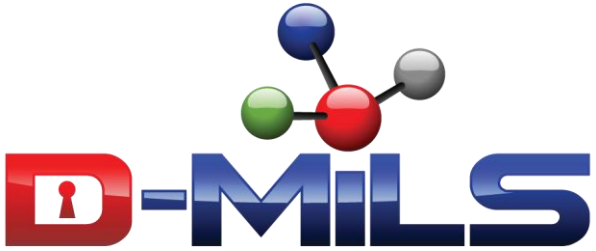


Developing Assurance Cases for D-MILS Systems

Richard Hawkins, Tim Kelly, Ibrahim Habli

Department of Computer Science,

The University of York

Overview

- Motivation
- Assurance Case Patterns for D-MILS systems
- Pattern Instantiation

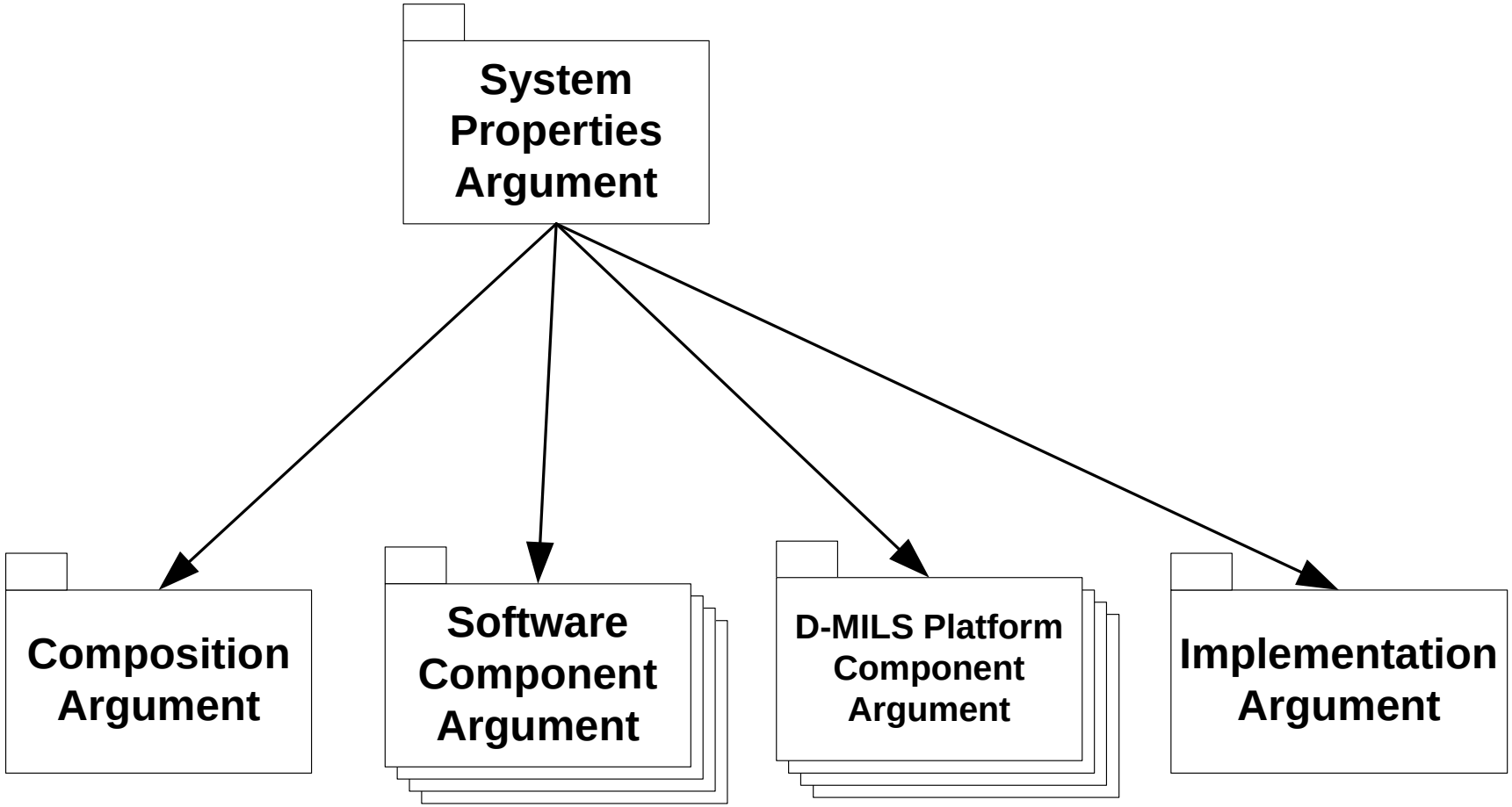
D-MILS Assurance Case

- Aims
  - To use a D-MILS system a system developer must be able to convince others it is secure, safe etc. (1, 3)
    - Assurance cases allow this, but we must help and guide people to do this well
  - Minimise assurance cost and effort for D-MILS systems (2, 3)
  - Ensure the highest levels of assurance can be demonstrated (1)
    - Understand what is required in a D-MILS assurance case
  - Support the objectives of DMILS (2)
    - E.g. compositionality of independently developed components.
- 3 fundamental elements:
  1. D-MILS Assurance Case Patterns
  2. Modular Approach
  3. Automated extraction of instantiation information directly from models wherever possible (e.g. MILS-AADL)

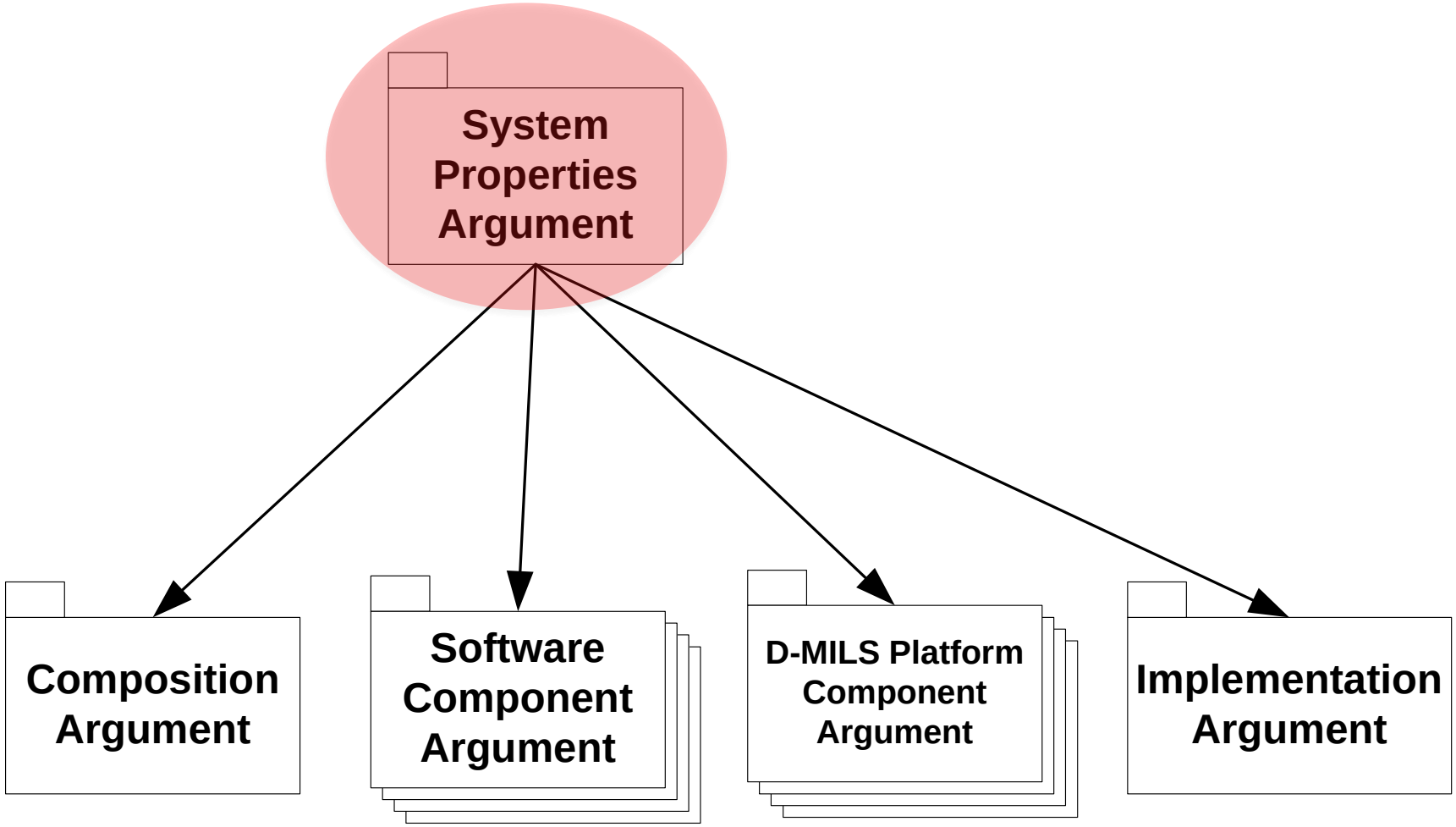
Modular Assurance Case

- Want to usefully organise and partition what must be done to create an assurance case
- Assurance case must align with the compositional approach of D-MILS
- Modular assurance cases allow us to compose large assurance cases from separate but interconnected modules of argument and evidence
  - each assurance case module reasons about one aspect of the overall case
- Dependencies captured by inter-module references ("away goals") to claims in other modules
  - assurance case modules can be developed independently by different organisations

Modular Structure of D-MILS Assurance Case



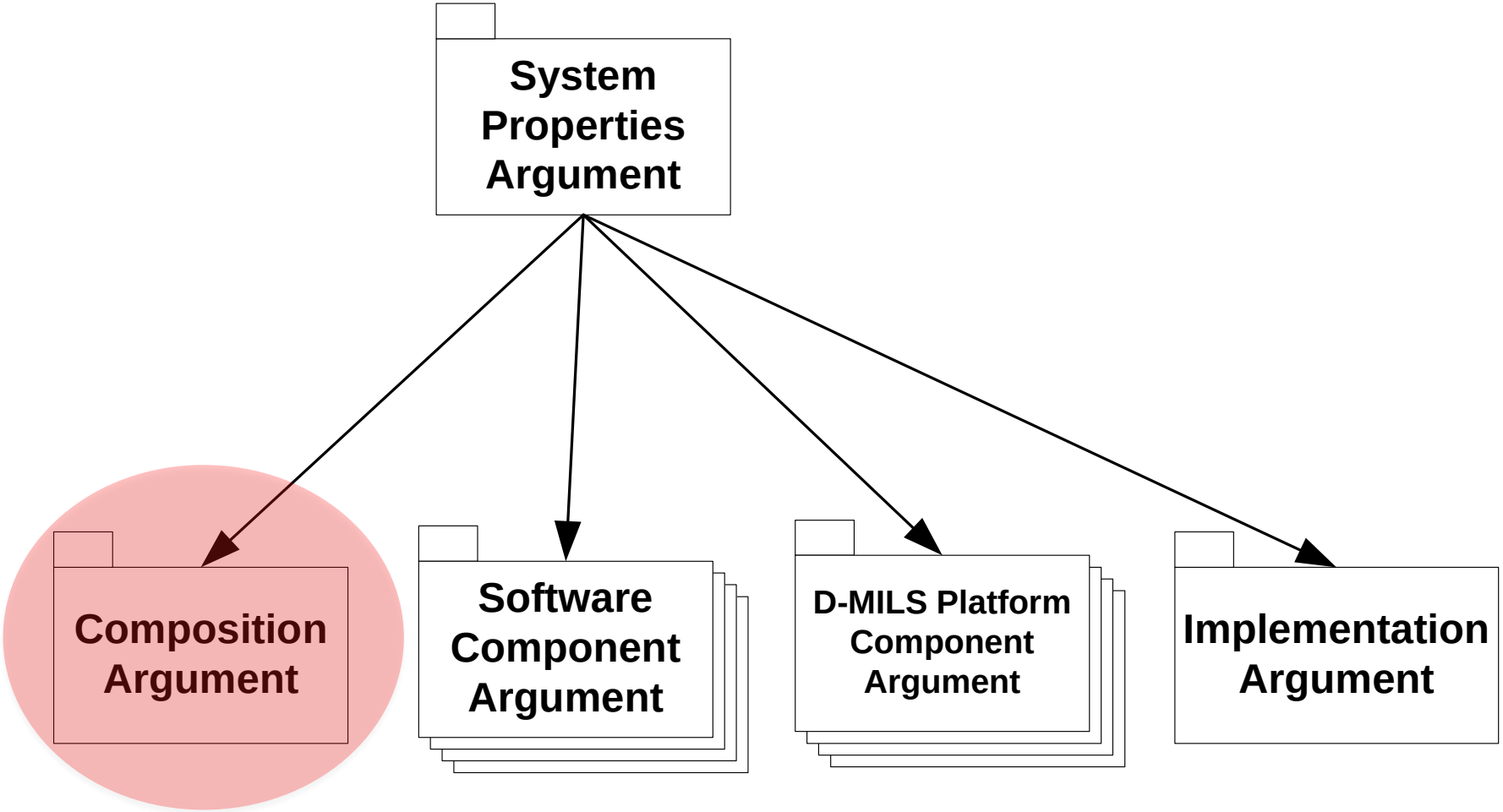
Modular Structure of D-MILS Assurance Case



System Properties Argument Pattern

- Assurance Guarantees
  - System security/safety properties (informal) are enforced
    - Those properties are complete & correct w.r.t. hazards, threats etc
  - Formal system properties (OCRA contracts)are satisfied in the MILS-AADL model
    - Formal properties are equivalent to informal system properties
- Assurance Dependencies (away goals in other modules)
  - Compositional verification proves properties in the model
  - The MILS-AADL model is faithfully implemented
  - Trusted software components implement their specification
  - D-MILS platform guarantees required properties

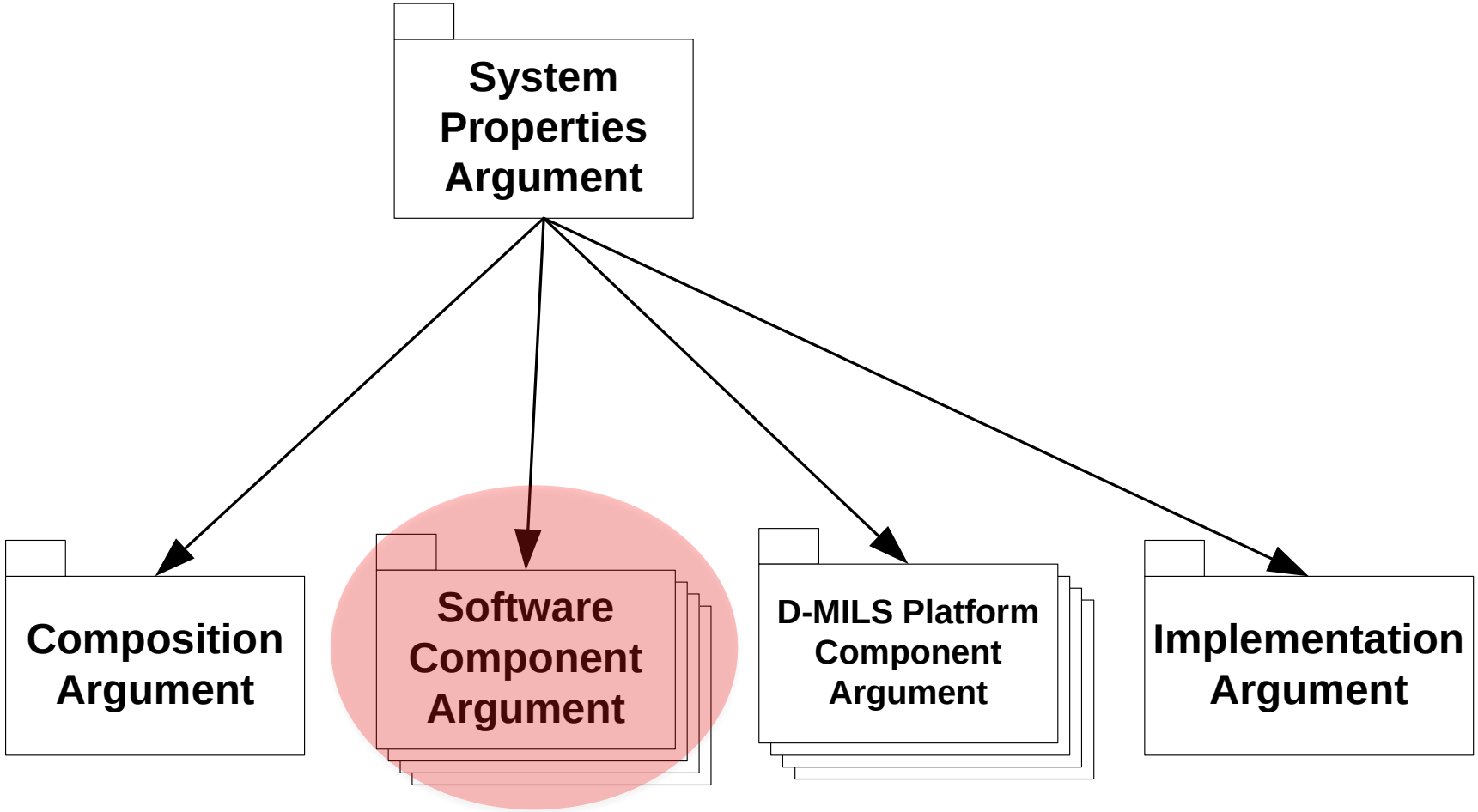
Modular Structure of D-MILS Assurance Case



Composition Argument Pattern

- Assurance Guarantees
  - MILS-AADL model satisfies each formal property
    - Refinement and model checking
  - The formal verification results are trustworthy
    - Translations between formal notations
- Assurance Dependencies (away goals in other modules)
  - Trusted components satisfy MILS-AADL implementation specification
  - Assumptions of system property contracts are satisfied

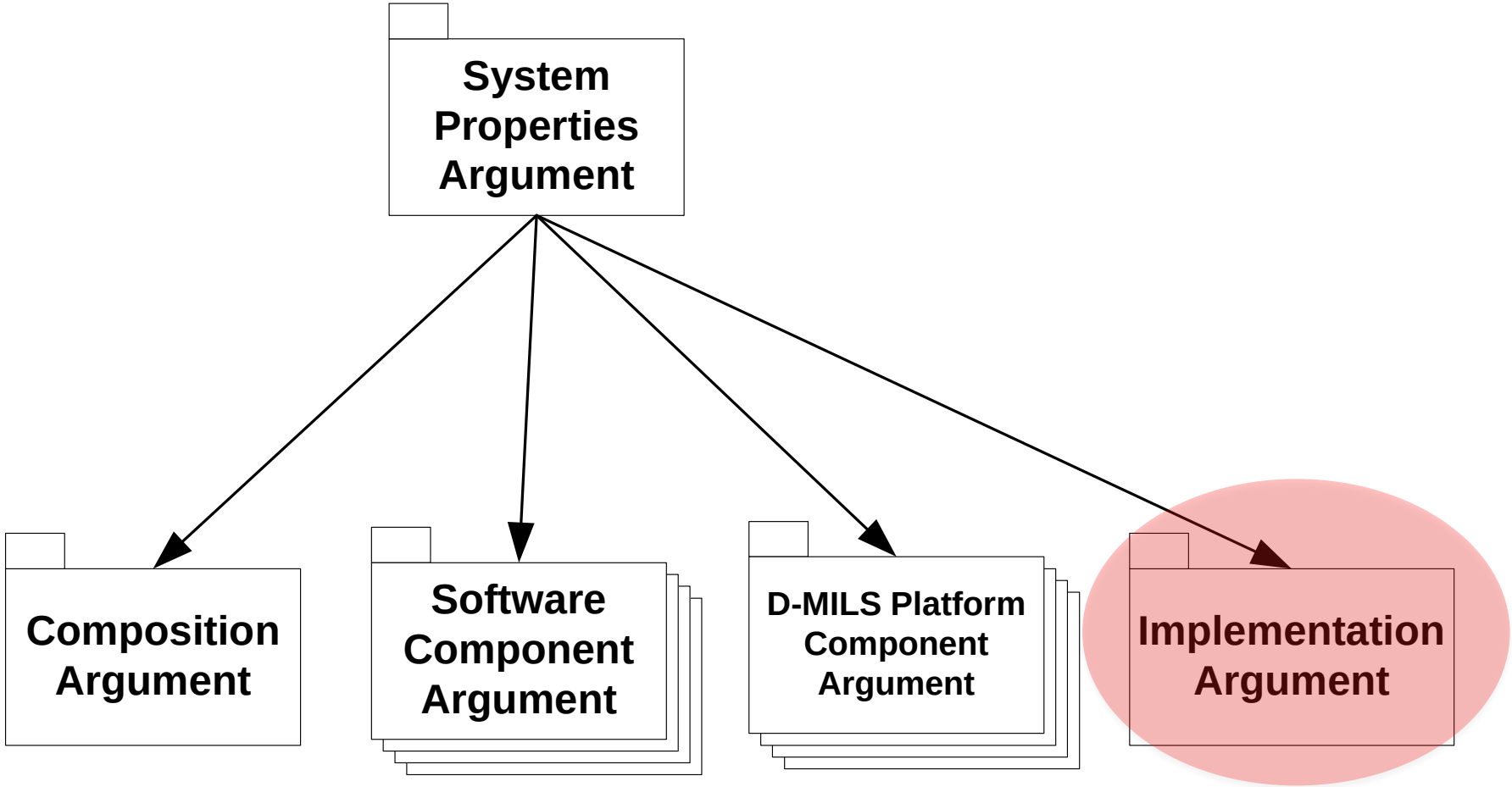
Modular Structure of D-MILS Assurance Case



Software Component Argument Pattern

- Separate module created for each trusted component
- Assurance Guarantees
  - Implementation of software component satisfies MILS-AADL implementation specification
    - deliberately avoid constraining the assurance methods (or standard) adopted by third-party providers
      - Domain / application specific
- Assurance Dependencies (away goals in other modules)
  - Identified by assurance case of software component

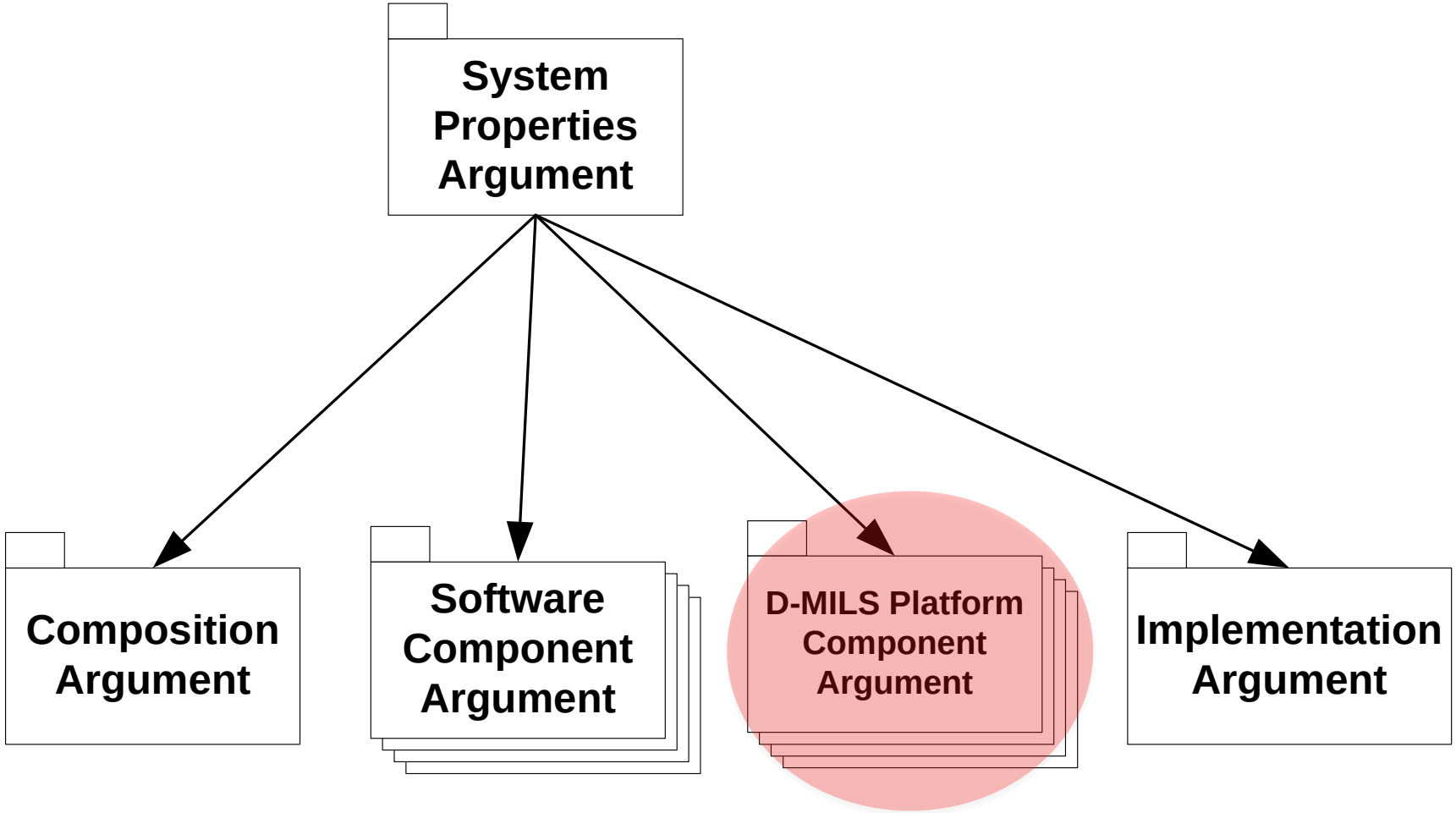
Modular Structure of D-MILS Assurance Case



Implementation Argument Pattern

- Assurance Guarantees
  - MILS-AADL model is faithfully implemented
  - Generated configuration is correct w.r.t. MILS-AADL model
    - The inputs to the configuration compiler are correct
      - Policy and platform description etc.
    - Configuration compiler tool performs correctly
    - Configuration is well-formed and satisfy all constraints
  - Target-specific configurations are syntactically and semantically correct w.r.t. the configuration
- Assurance Dependencies (away goals in other modules)
  - Target-specific configurations are realised by the D-MILS platform components

Modular Structure of D-MILS Assurance Case



D-MILS Platform Argument Pattern

- Assurance Guarantees
  - D-MILS platform guarantees the required properties
    - Inter-nodal communication occurs only as defined in MILS-AADL model
      - MILS networking system (MNS) controls network communication
    - Intra-nodal interference occurs only as defined in MILS-AADL model
      - MILS seperation kernel controls access to shared memory
    - Vulnerabilities and threats are mitigated
- Assurance Dependencies (away goals in other modules)
  - Target-specific configurations are correctly interpreted by the D-MILS platform components
  - Separate module created for each D-MILS platform component
    - SK, TTE switches, MNS…

Pattern Instantiation

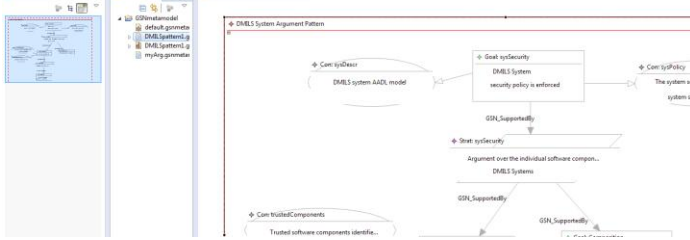
- Creation of assurance case must not be burden to adoption of D-MILS approach
- Argument patterns essentially define information requirements
  - to instantiate the assurance claims, provide evidence and make instantiation choices
- Possible to manually obtain this to instantiate pattern
  - But, repetitive and mechanistic in nature, time-consuming and prone to human error
- However, if you have the right models should be possible to largely automatically generate the assurance case directly from the models
  - We have developed a novel approach that achieves this

Automated Instantiation

Required System Models (e.g. AADL)

```
system CryptoController
  features
    inframe: in data port Frame;
    outframe: out data port Frame;
end CryptoController;
system implementation CryptoController.Imp
  subcomponents
    red: node Splitter.Imp accesses channels;
    bypass: node Bypass.Imp accesses channels;
    crypto: node Crypto.Imp accesses channels;
    black: node Merger.Imp accesses channels;
    channels: network CryptoNet.Imp;
  flows
    port inframe -> red.frame;
    port red.header -> bypass.inheader;
    port red.payload -> crypto.inpayload;
    port bypass.outheader -> black.header;
    port crypto.outpayload -> black.payload;
    port black.frame -> outframe;
end CryptoController.Imp;

--
-- Splitter component for decomposing frames into header
--
node Splitter
  features
    frame: in data port Frame;
    header: out data port Header;
    payload: out data port Payload;
end Splitter;
node implementation Splitter.Imp
  flows
    port fst(frame) -> header;
    port snd(frame) -> payload;
end Splitter.Imp;
```



Visualised in GSN Editor

DMILS Argument Patterns (GSNML files)

```
<?xml version="1.0" encoding="UTF-8"?>
<gsnmetamodel:Case xmi:version="2.0" xmlns:xmi="http://www.omg.org/XMI" xmlns:xsi="http://www.w3.
  <contains xsi:type="gsnmetamodel:GSN_Module" id="DMILS System Argument Pattern">
    <ArgumentElements xsi:type="gsnmetamodel:GSN_Goal" id="Goal: sysSecurity">
      <contents xsi:type="gsnmetamodel:Literal" literal="security policy is enforced"/>
      <contents xsi:type="gsnmetamodel:Role" role="DMILS System"/>
    </ArgumentElements>
    <ArgumentElements xsi:type="gsnmetamodel:GSN_ContextAsReference" id="Con: sysPolicy">
      <contents xsi:type="gsnmetamodel:Literal" literal="The system security policy is"/>
      <contents xsi:type="gsnmetamodel:Role" role="system security policy"/>
    </ArgumentElements>
    <ArgumentElements xsi:type="gsnmetamodel:GSN_InContextOf" hasSource="//@contains.0/@ArgumentE
    <ArgumentElements xsi:type="gsnmetamodel:GSN_ContextAsReference" id="Con: sysDescr">
      <contents xsi:type="gsnmetamodel:Role" role="DMILS system AADL model"/>
    </ArgumentElements>
    <ArgumentElements xsi:type="gsnmetamodel:GSN_InContextOf" hasSource="//@contains.0/@ArgumentE
    <ArgumentElements xsi:type="gsnmetamodel:GSN_Strategy" id="Strat: sysSecurity">
      <contents xsi:type="gsnmetamodel:Literal" literal="Argument over the individual software co
      <contents xsi:type="gsnmetamodel:Role" role="DMILS Systems"/>
    </ArgumentElements>
    <ArgumentElements xsi:type="gsnmetamodel:GSN_SupportedBy" hasSource="//@contains.0/@ArgumentE
    <ArgumentElements xsi:type="gsnmetamodel:GSN_Goal" id="Goal: components">
      <contents xsi:type="gsnmetamodel:Literal" literal="Trusted software components behave accor
    </ArgumentElements>
    <ArgumentElements xsi:type="gsnmetamodel:GSN_SupportedBy" hasSource="//@contains.0/@ArgumentE
    <ArgumentElements xsi:type="gsnmetamodel:GSN_ContextAsReference" id="Con: trustedComponents">
      <contents xsi:type="gsnmetamodel:Literal" literal="Trusted software components identified i
      <contents xsi:type="gsnmetamodel:Role" role="software components"/>
    </ArgumentElements>
    <ArgumentElements xsi:type="gsnmetamodel:GSN_InContextOf" hasSource="//@contains.0/@ArgumentE
    <ArgumentElements xsi:type="gsnmetamodel:GSN_ContextAsReference" id="Con: policyArchitecture"
      <contents xsi:type="gsnmetamodel:Literal" literal="The policy architecture is described by
    </ArgumentElements>
    <ArgumentElements xsi:type="gsnmetamodel:GSN_InContextOf" hasSource="//@contains.0/@ArgumentE
    <ArgumentElements xsi:type="gsnmetamodel:GSN_Goal" id="Goal: Composition">
      <contents xsi:type="gsnmetamodel:Literal" literal="The composition of the DMILS system guar
    </ArgumentElements>
```

epsilon

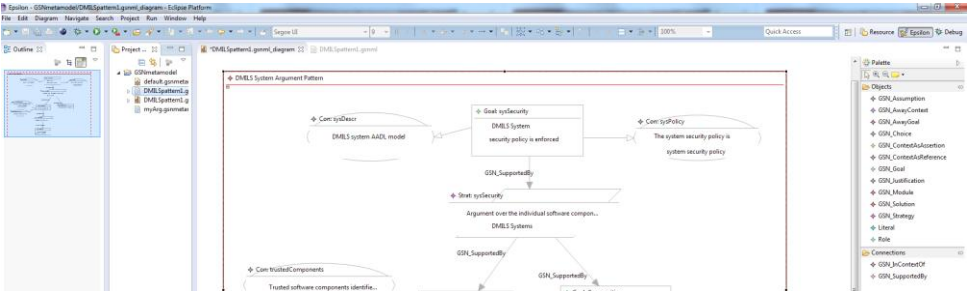
Instantiation Program (eol)

Generates

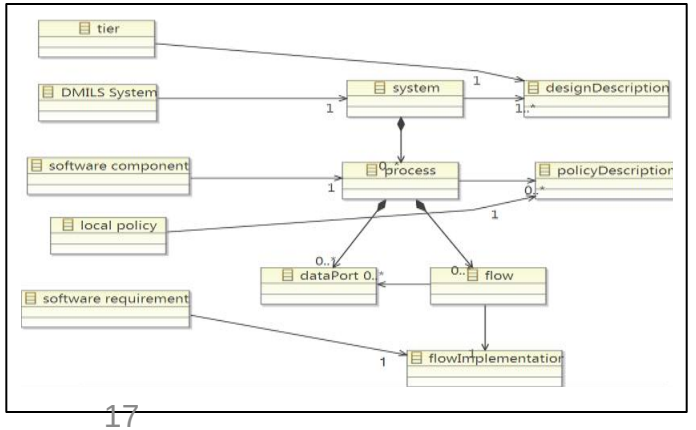
```
<?xml version="1.0" encoding="UTF-8"?>
<gsnmetamodel:Case xmi:version="2.0" xmlns:xmi="http://www.omg.org/XMI" xmlns:xsi="http://www.w3.
  <contains xsi:type="gsnmetamodel:GSN_Module" id="DMILS System Argument Pattern">
    <ArgumentElements xsi:type="gsnmetamodel:GSN_Goal" id="Goal: sysSecurity">
      <contents xsi:type="gsnmetamodel:Literal" literal="security policy is enforced"/>
      <contents xsi:type="gsnmetamodel:Role" role="DMILS System"/>
    </ArgumentElements>
    <ArgumentElements xsi:type="gsnmetamodel:GSN_ContextAsReference" id="Con: sysPolicy">
      <contents xsi:type="gsnmetamodel:Literal" literal="The system security policy is"/>
      <contents xsi:type="gsnmetamodel:Role" role="system security policy"/>
    </ArgumentElements>
    <ArgumentElements xsi:type="gsnmetamodel:GSN_InContextOf" hasSource="//@contains.0/@ArgumentE
    <ArgumentElements xsi:type="gsnmetamodel:GSN_ContextAsReference" id="Con: sysDescr">
      <contents xsi:type="gsnmetamodel:Role" role="DMILS system AADL model"/>
    </ArgumentElements>
    <ArgumentElements xsi:type="gsnmetamodel:GSN_InContextOf" hasSource="//@contains.0/@ArgumentE
    <ArgumentElements xsi:type="gsnmetamodel:GSN_Strategy" id="Strat: sysSecurity">
      <contents xsi:type="gsnmetamodel:Literal" literal="Argument over the individual software co
      <contents xsi:type="gsnmetamodel:Role" role="DMILS Systems"/>
    </ArgumentElements>
    <ArgumentElements xsi:type="gsnmetamodel:GSN_SupportedBy" hasSource="//@contains.0/@ArgumentE
    <ArgumentElements xsi:type="gsnmetamodel:GSN_Goal" id="Goal: components">
      <contents xsi:type="gsnmetamodel:Literal" literal="Trusted software components behave accor
    </ArgumentElements>
    <ArgumentElements xsi:type="gsnmetamodel:GSN_SupportedBy" hasSource="//@contains.0/@ArgumentE
    <ArgumentElements xsi:type="gsnmetamodel:GSN_ContextAsReference" id="Con: trustedComponents">
      <contents xsi:type="gsnmetamodel:Literal" literal="Trusted software components identified i
      <contents xsi:type="gsnmetamodel:Role" role="software components"/>
    </ArgumentElements>
    <ArgumentElements xsi:type="gsnmetamodel:GSN_InContextOf" hasSource="//@contains.0/@ArgumentE
    <ArgumentElements xsi:type="gsnmetamodel:GSN_ContextAsReference" id="Con: policyArchitecture"
      <contents xsi:type="gsnmetamodel:Literal" literal="The policy architecture is described by
    </ArgumentElements>
    <ArgumentElements xsi:type="gsnmetamodel:GSN_InContextOf" hasSource="//@contains.0/@ArgumentE
    <ArgumentElements xsi:type="gsnmetamodel:GSN_Goal" id="Goal: Composition">
      <contents xsi:type="gsnmetamodel:Literal" literal="The composition of the DMILS system guar
    </ArgumentElements>
```

DMILS Assurance Argument (GSNML)

Creates



DMILS GSN Editor



Weaving Model (Ecore)

Conclusions

- Our model-based approach to creating D-MILS assurance case provides:
  - Reduced time and effort in creation
  - Increased consistency in instantiation
  - Consistency between argument and system models
  - Validation and feedback
- More straightforward for D-MILS system developers to generate a rigorous assurance case for their systems