Software Safety and Security, Assurance Cases and Model Management

Marsha Chechik

UNIVERSITY OF TORONTO

September, 2017

SEFM'17

A Brief and Partial Research History

Model-checking (Xchek) and formal specifications, para-consistent logics

Reasoning about incomplete and inconsistent systems

Modeling and reasoning about variability, product lines

Compliance of software-based systems

Mid-1990 | 2000s | 2010s | now | time

Static analysis of programs, state machine specifications

Software model-checking (Yasm, UFO)

Runtime analysis of web service interactions

Reuse (feature-level)

Reuse (model transformation level)

Software-based systems are at the core of modern society

And yet we have trouble producing systems that do not fail

Technology

Airbus A400M plane crash linked to software fault

By Leo Kelion
Technology desk editor

20 May 2015 | Technology

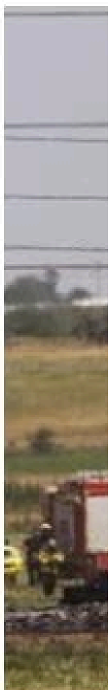
The A400M cargo plane crashed near Seville airport on 9 May

Techn

Airl
soft

By Leo K
Technol

20 Ma



The A

13

FDA: SOFTWARE FAILURES RESPONSIBLE FOR 24% OF ALL MEDICAL DEVICE RECALLS

by **Paul Roberts** June 20, 2012 , 9:05 pm

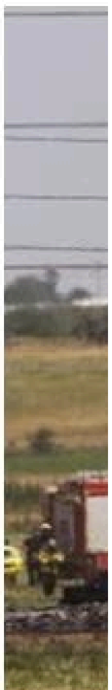
Software failures were behind 24 percent of all the medical device recalls in 2011, according to data from the U.S. Food and Drug Administration, which said it is gearing up its labs to spend more time analyzing the quality and security of software-based medical instruments and equipment.

Techn

Airl
sof

By Leo I
Technol

20 Ma



The A

13

by P

Sc
ac
u
m

Volvo recalls 59,000 cars over software fault

20 February 2016 | Europe

Sweden, Britain and Germany are the main markets affected

Swedish carmaker Volvo is recalling 59,000 cars across 40 markets over a fault that can temporarily shut down the engine.

Robotics

U.S. Wants Makers of Driverless Cars to Prove They Are Safe

The auto industry is beginning to get some clarity on the rules of the road for autonomous cars.

by Will Knight September 20, 2016

The U.S. government has issued its first rules for automated vehicles. They include a 15-point set of "safety assessment" guidelines for self-driving systems. These cover issues such as cybersecurity, black-box recordings to aid crash investigations, and potential ethical conundrums on the road.

"Standards are documented agreements containing technical **specifications** or other precise criteria to be used consistently as **rules**, **guidelines**, or **definitions** of characteristics, to ensure that materials, products, processes and services are fit for their purpose."

[ISO 1997]

Standards

Aimed to assure a particular property of a system in a particular domain

Properties:

Safety – does the system correctly handle threats?

Security – does the system mitigate being tampered with

Privacy – does the system appropriately handle

data of its users?

Standards

Aimed to assure a particular property of a system in a particular domain

Domains:

- Automotive
- Aerospace
- Nuclear
- Healthcare

DO-178B - Software Considerations in Airborne Systems and Equipment Certification

IEC62304 – Medical Device Software – Software Life Cycle Processes

SO/IEC 27000 Family - Information Security Management Systems

ISO/IEC 27000:2016

- Information technology
- Security techniques
- Information security management syster
- Overview and vocabulary

ISO/IEC 27001:2013

- Information technology
- Security techniques
- Information security management syster
- Requirements

ISO/IEC 27002:2013

- Information technology
- Security techniques
- Code of practice for information security controls

ISO/IEC 29100:2011 Privacy Framework

- specifies a common privacy terminology;
- defines the actors and their roles in processing personally identifiable information (PII);
- describes privacy safeguarding considerations; and
- provides references to known privacy principles for information technology.

ISO/IEC 29100 Privacy Guidelines

- 1. Consent and choice
- 2. Purpose legitimacy and specification
- 3. Collection limitation
- 4. Data minimization
- 5. Use, retention and disclosure limitation
- 6. Accuracy and quality
- 7. Openness, transparency and notice
- 8. Individual participation and access
- 9. Accountability
- 10. Information security
- 11. Privacy compliance

ISO/IEC 27018

Code of practice for protection of personally identifying information (PII) in public clouds

- establishes commonly accepted control objectives, controls and guidelines
  - for implementing measures to protect Personally Identifiable Information (PII)
  - in accordance with the privacy principles in ISO/IEC 29100
  - for the public cloud computing environment.

ISO26262 - Functional Safety of Road Vehicles

Standards are

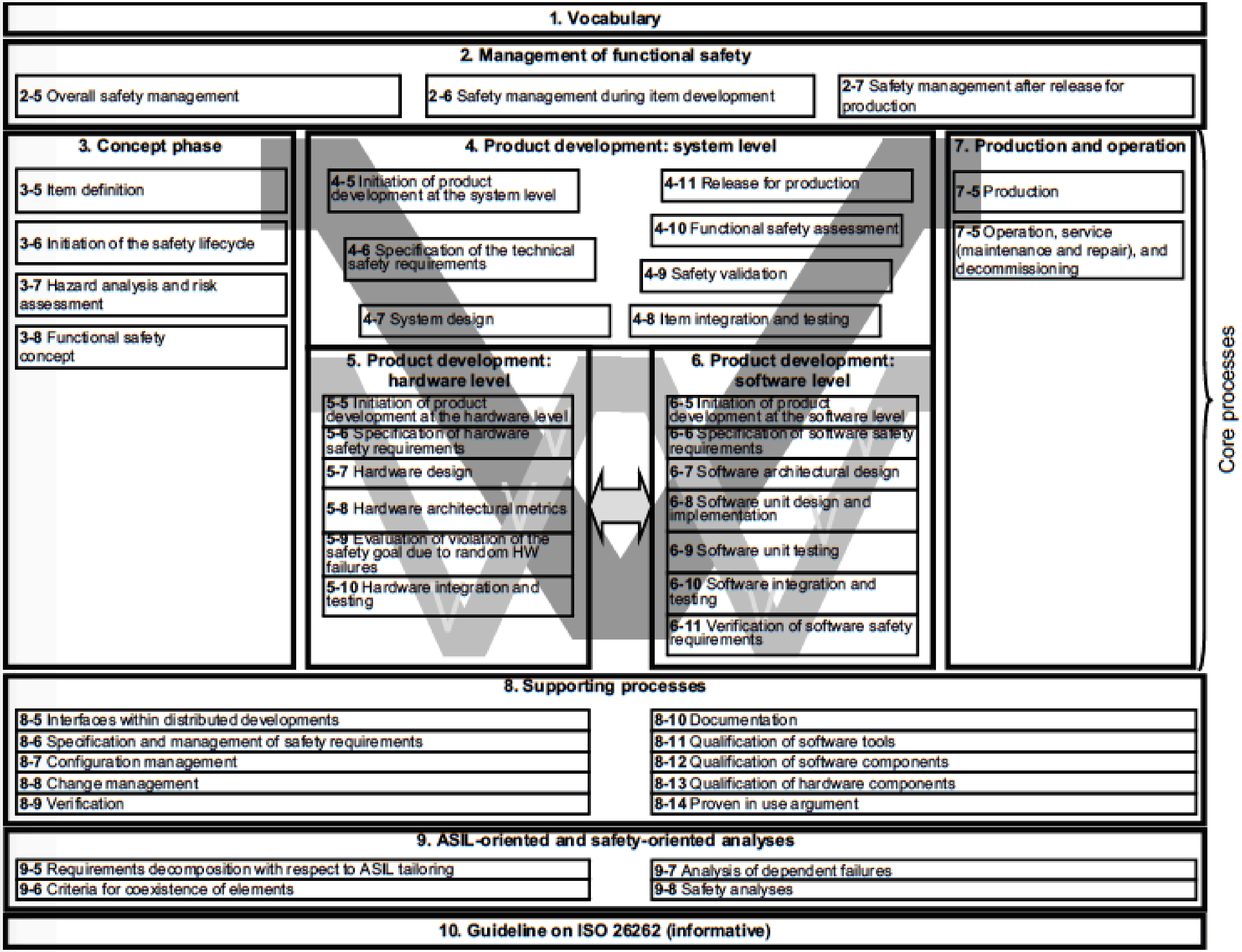
BIG *coMpleX*

ISO26262 - Functional Safety of Road Vehicles

Standard has 10 parts

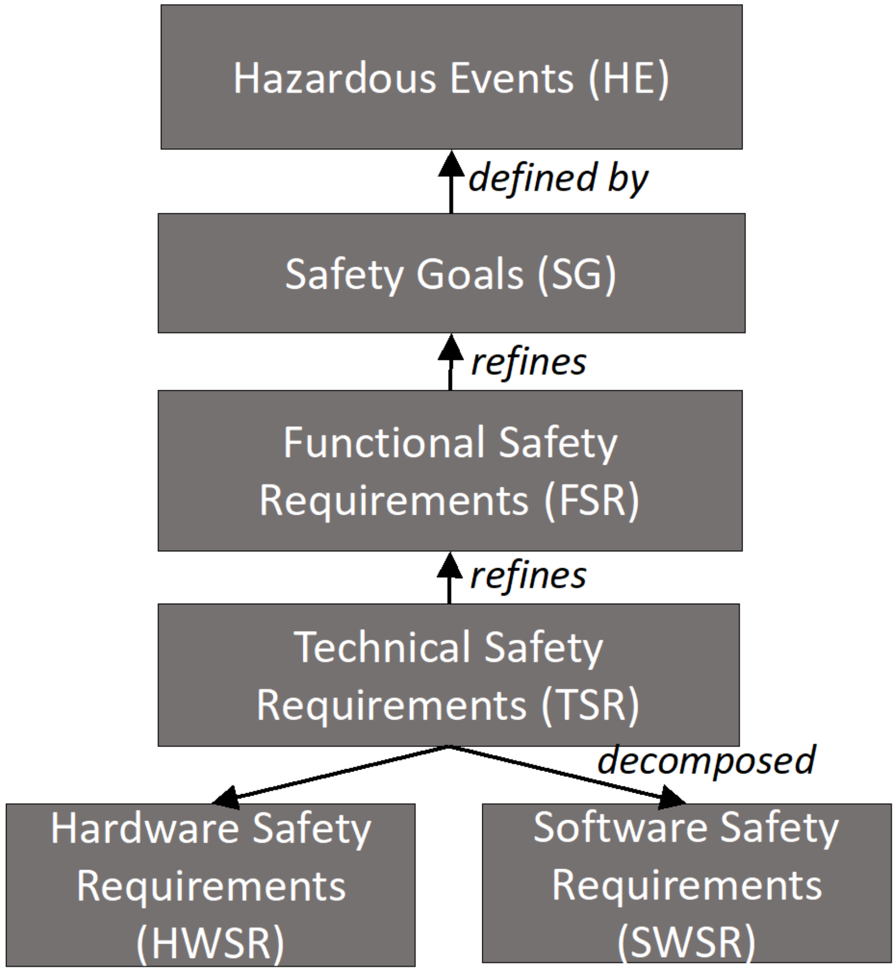
- Span across **~450 pages**
- Require the production of **~120 work products**

… that are the result of fulfilling a much larger number of **requirements** and **recommendations**


1. Vocabulary
2. Management of functional safety
2-5 Overall safety management
2-6 Safety management during item development
2-7 Safety management after release for production
3. Concept phase
3-5 Item definition
3-6 Initiation of the safety lifecycle
3-7 Hazard analysis and risk assessment
3-8 Functional safety concept
4. Product development: system level
4-5 Initiation of product development at the system level
4-6 Specification of the technical safety requirements
4-7 System design
4-8 Item integration and testing
4-9 Safety validation
4-10 Functional safety assessment
4-11 Release for production
5. Product development: hardware level
5-5 Initiation of product development at the hardware level
5-6 Specification of hardware safety requirements
5-7 Hardware design
5-8 Hardware architectural metrics
5-9 Evaluation of violation of the safety goal due to random HW failures
5-10 Hardware integration and testing
6. Product development: software level
6-5 Initiation of product development at the software level
6-6 Specification of software safety requirements
6-7 Software architectural design
6-8 Software unit design and implementation
6-9 Software unit testing
6-10 Software integration and testing
6-11 Verification of software safety requirements
7. Production and operation
7-5 Production
7-6 Operation, service (maintenance and repair), and decommissioning
Core processes
8. Supporting processes
8-5 Interfaces within distributed developments
8-6 Specification and management of safety requirements
8-7 Configuration management
8-8 Change management
8-9 Verification
8-10 Documentation
8-11 Qualification of software tools
8-12 Qualification of software components
8-13 Qualification of hardware components
8-14 Proven in use argument
9. ASIL-oriented and safety-oriented analyses
9-5 Requirements decomposition with respect to ASIL tailoring
9-6 Criteria for coexistence of elements
9-7 Analysis of dependent failures
9-8 Safety analyses
10. Guideline on ISO 26262 (informative)

But in essence, what standards recommend is pretty simple

ISO 26262 Recommendation



Identify obstacles to achieve your goal

Goals

Requirements

Compliance

What is it?

The extent to which software developers have acted in accordance with practices set down in the standard.

Why it is done?

Establish **consistency** between actual development process and normative models embedded in the standards.

How is it done?

An artifact, called an **assurance case**, is often required to demonstrate that a system meets the property set forth by the standard (e.g., Safety, Privacy, Security, etc.)

Assurance Process

1. Completely and correctly identify goals (for safety / security / privacy)
2. Collect sufficient evidence that you have adequately dealt with each of them

Assurance Case

- A.k.a. safety case, security case, privacy case, etc.
- An artifact that shows how each of the important **claims** about the system (e.g., its **safety/security/privacy** goals) can be argued for, ultimately from **evidence** obtained about the system
- Evidence can come in many forms:
  - test results
  - analyses
  - model checking results
  - expert opinion
  - etc.
- The argument is often informal
  - “sufficient”
  - “adequate”

…with some degree of confidence

Assurance Arguments

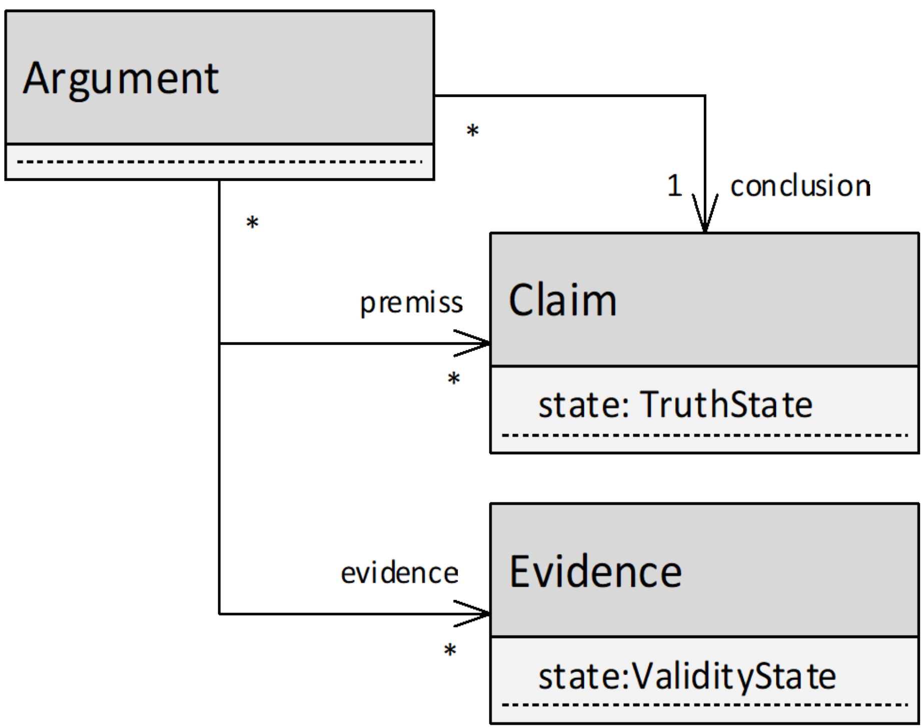
- Pragmatic and widely applicable
- Broaden applicability of formal methods
- Allow combining different types of evidence
- A nice connection with other engineering disciplines

- Informal (although rigorous)
- Expensive to produce
- Difficult to analyze / reuse

Assurance Case Modeling

Some approaches for modeling assurances cases:

GSN, CAE, KAOS-based, OMG SACM…



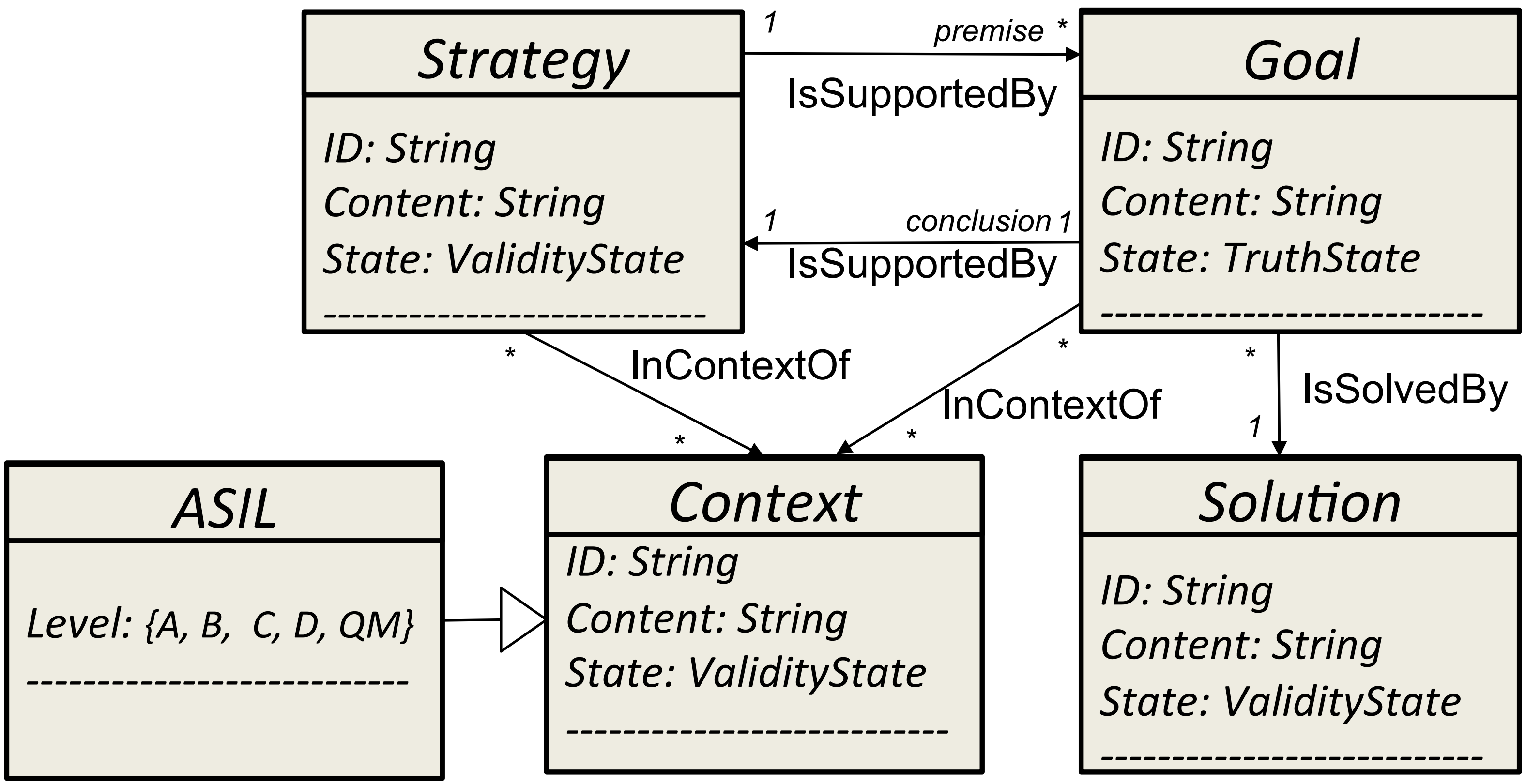
Generic Assurance Case Metamodel

Optionally:

- Dependency Relations
- Semantic Assumptions

Claims = goals
Evidence = solutions
Arguments = strategies

GSN – Goal Structuring Notation

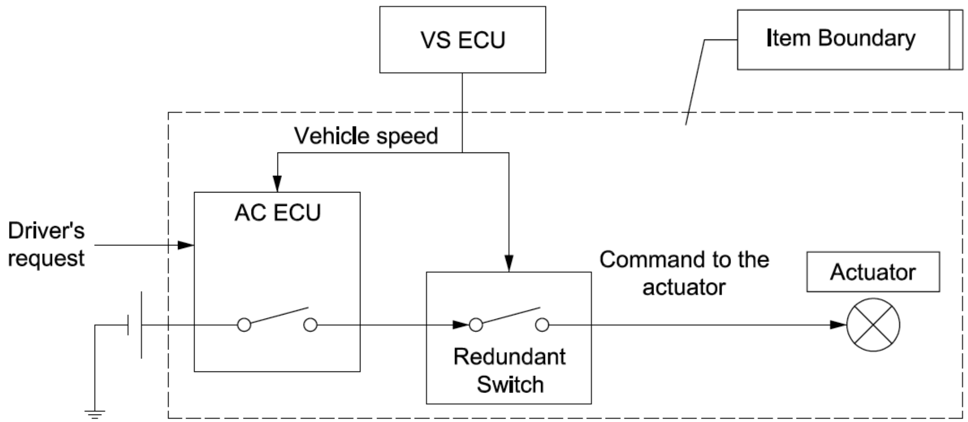


In this talk: assume software development is done using MDE

- "MDE" - Model Driven Engineering
- Models as first class citizens

  – Reduce accidental complexity by working at a higher level of abstraction
  – Code is automatically generated from models
  – Minimize development cost

Example: Power Sliding Door System

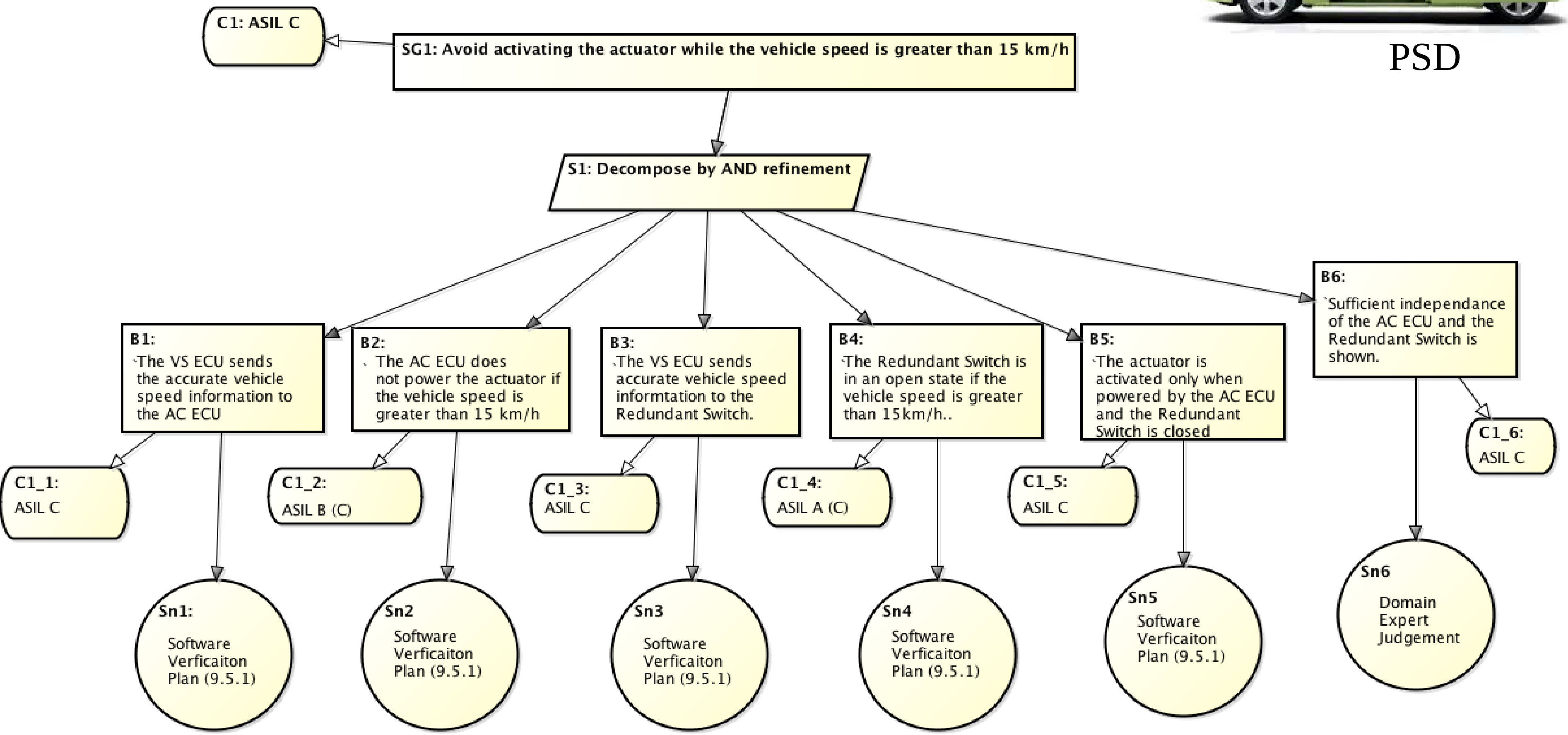


PSD

Safety goal SG1
Avoid activating the actuator when vehicle speed > 15 kph

Power Sliding Door Safety Case

PSD



Handling Assurance

– Informal (although rigorous)
– Expensive to produce
– Difficult to analyze / reuse

How to do automation over such informal artifacts?
– assessing compliance due to evolution
– compliance to multiple standards
– compliance of product lines

… and how to do this in a sound way?

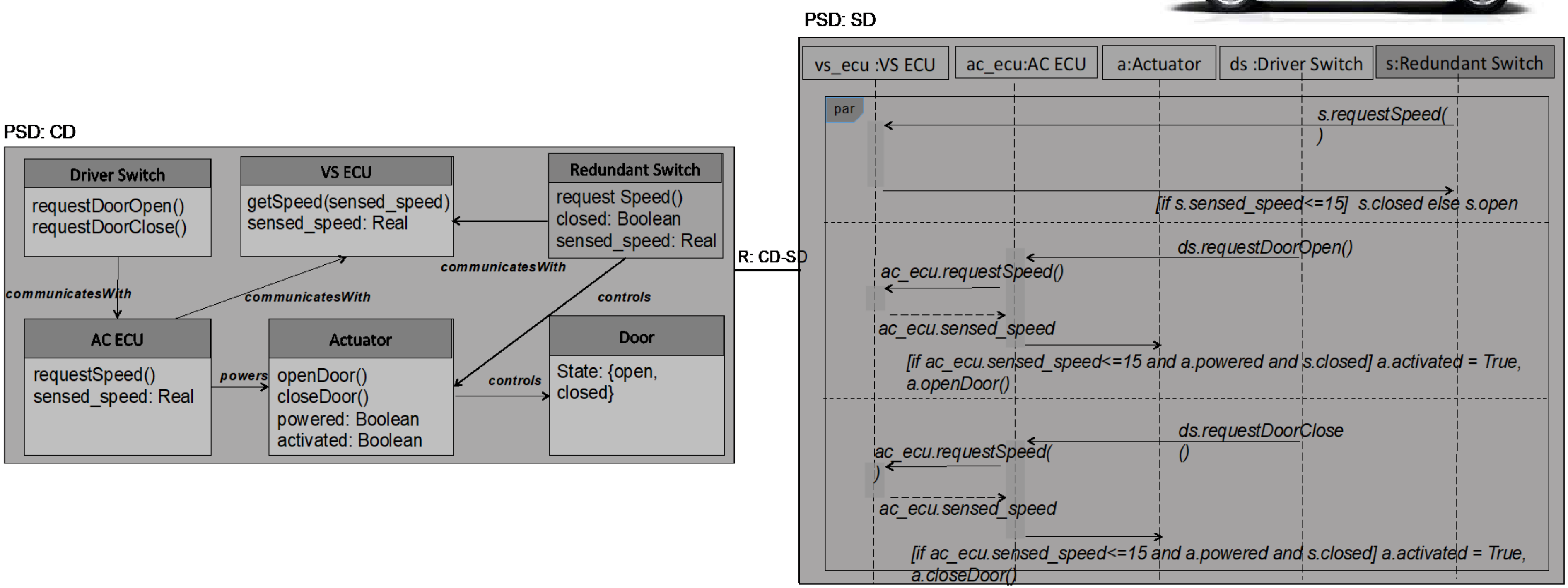
Will describe a particular solution in this space, using a model-management approach

Model Management (MM)

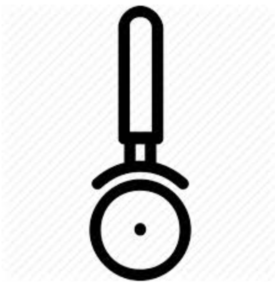
- High-level view in which entire **models** and their **relationships** can be manipulated using **operators** to achieve useful outcomes.

- **Megamodel:** a special type of model in which the elements represent models and the links between the elements represent relationships between the models.

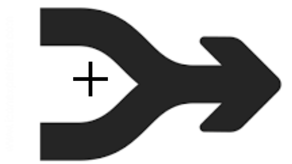
Example: Power Sliding Door System



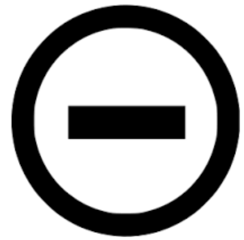
Some Model Management Operators



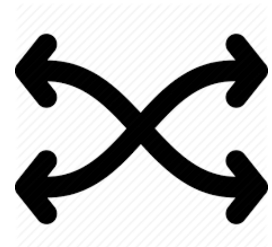
slice



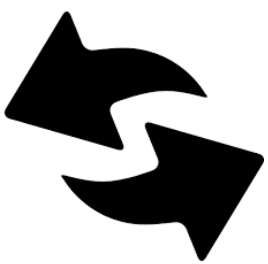
merge



diff



match



bidirectional MT

lift

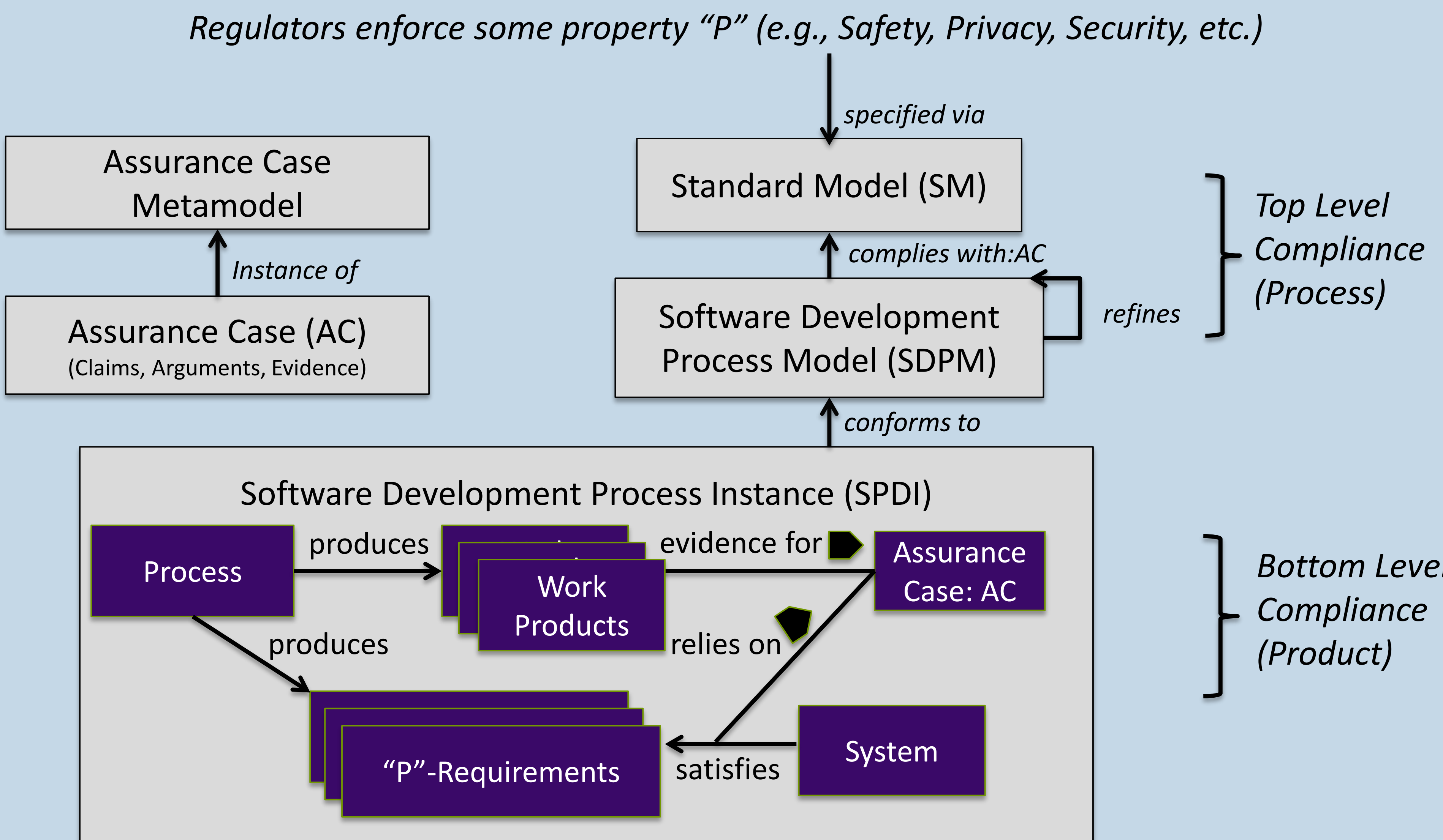
+ Megamodel Operators (Map, Filter, Reduce)

[MODELS'15]

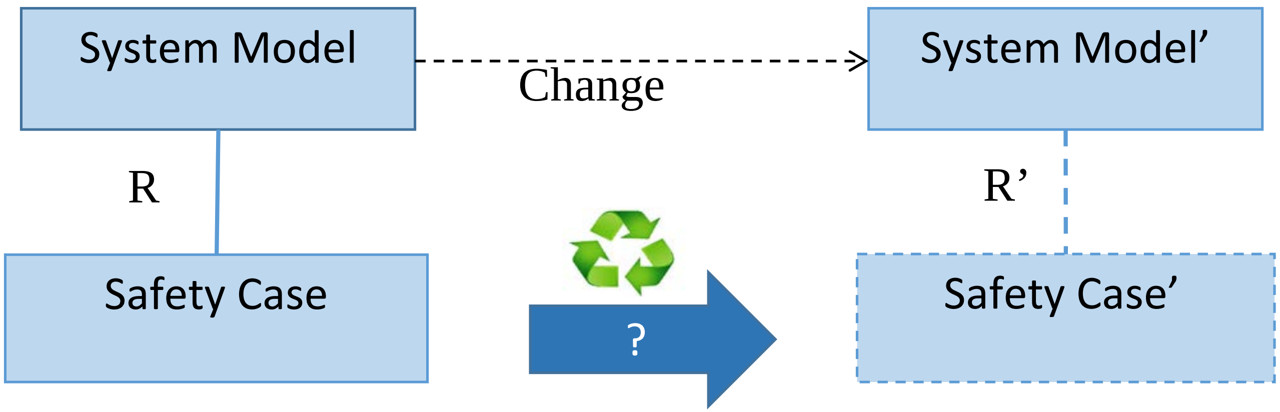
(Adapted) Model Management Toolbox

<u>Hypothesis:</u> Model Management Operators and Tools can be adapted to help structure, manage and reason about regulatory compliance.

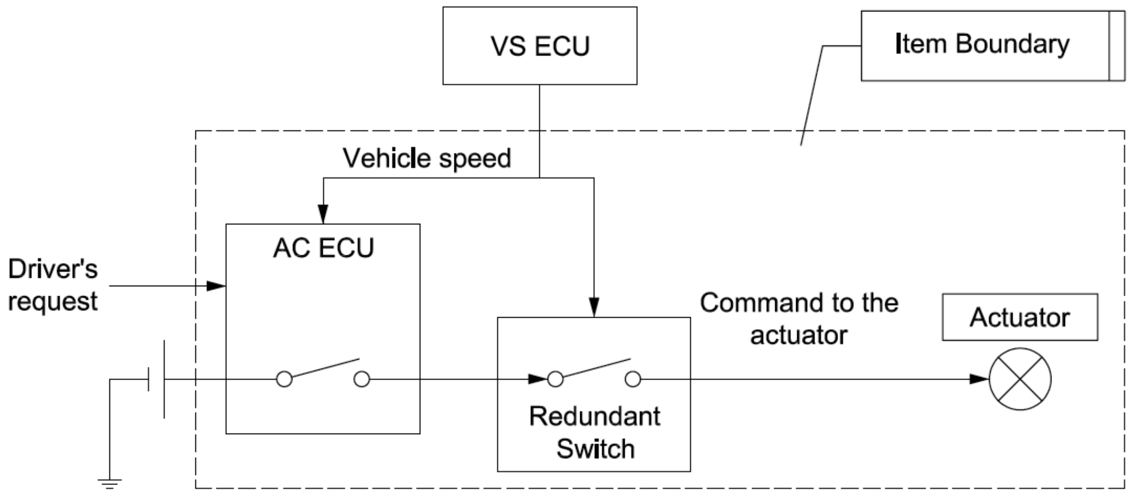
A General Model of Compliance



Problem: Safety Case and System Co-Evolution

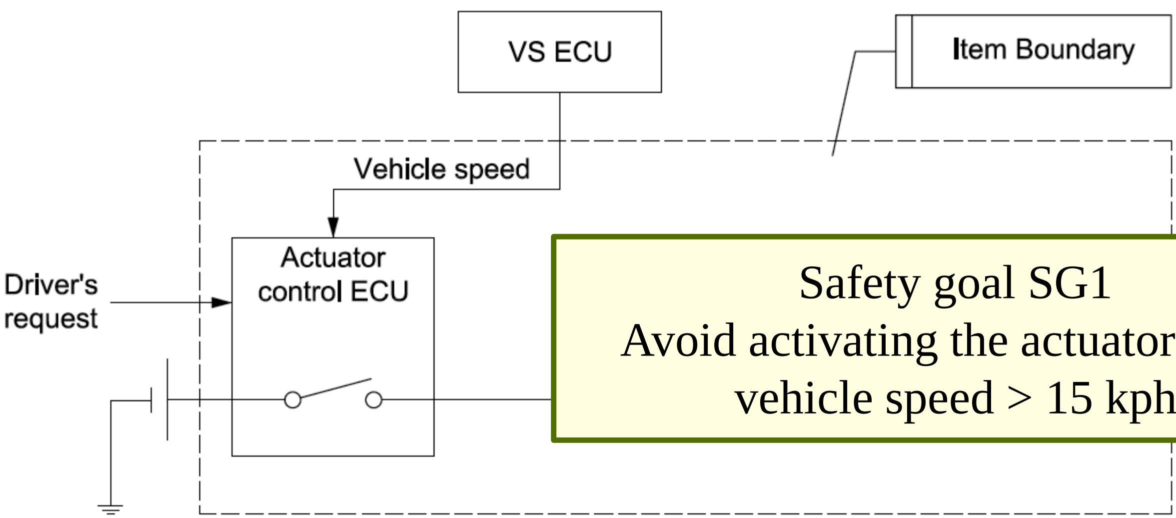


Example:
Removing Redundant Switch in PSD:



System S

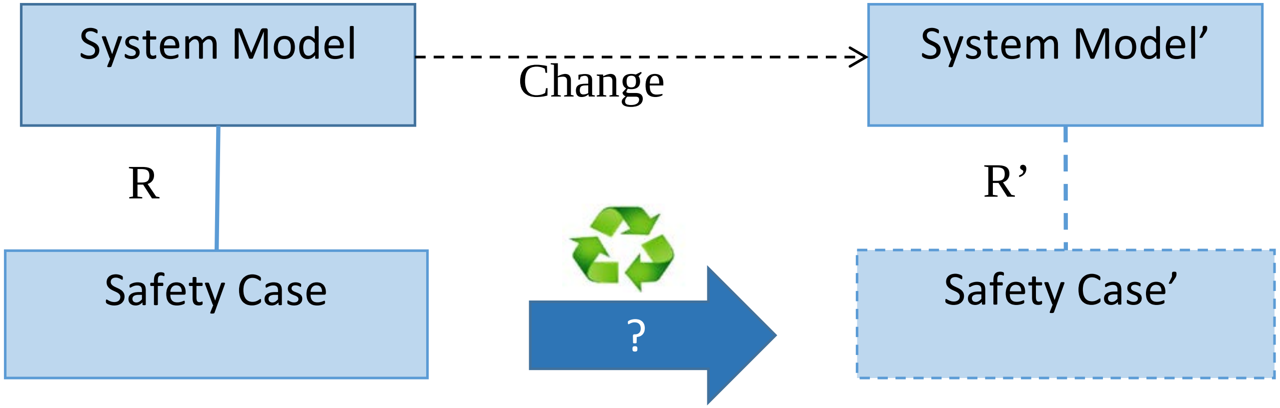
Δ: removal of redundant switch



l System S’

Safety goal SG1
Avoid activating the actuator when vehicle speed > 15 kph

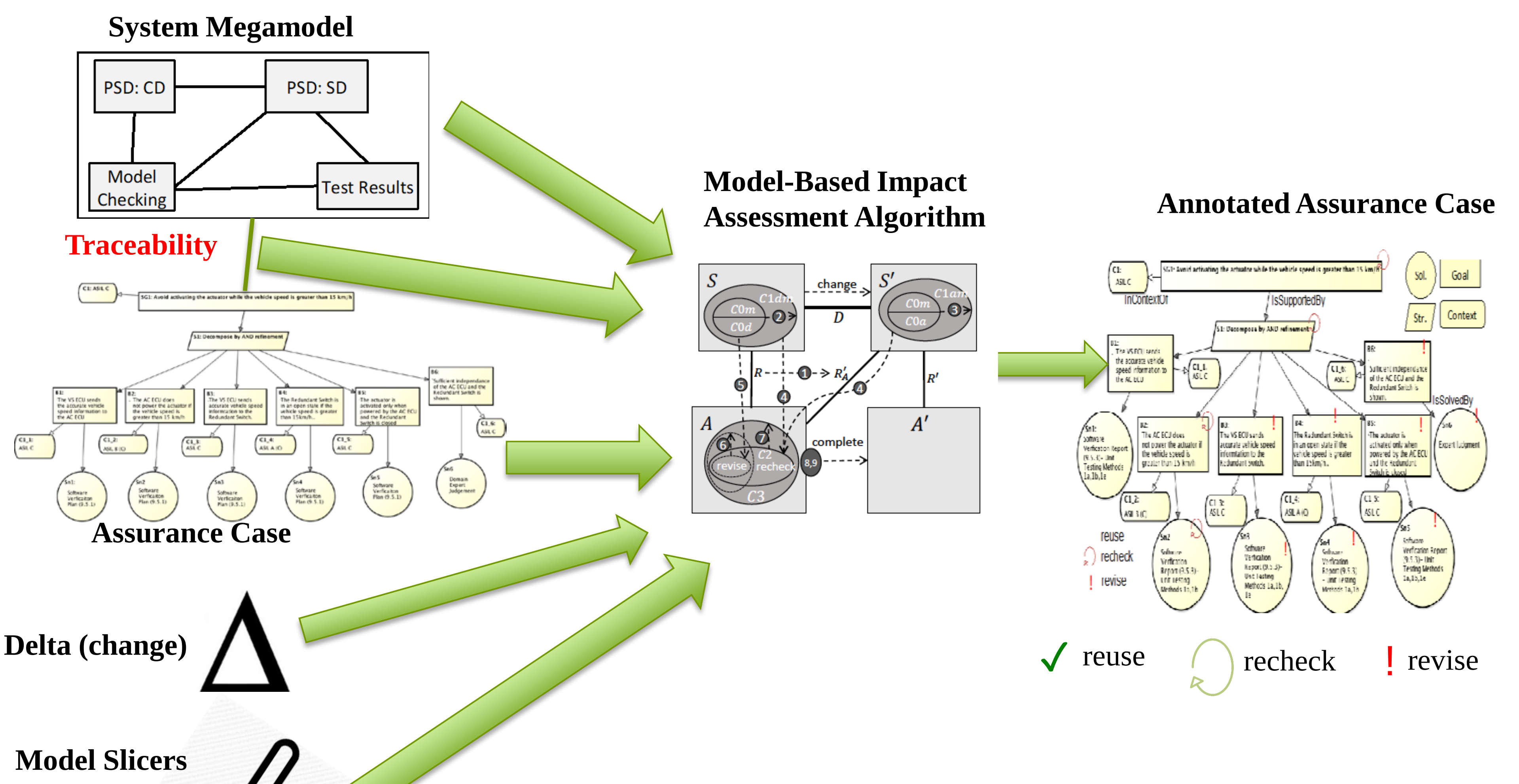
Problem: Safety Case and System Co-Evolution



How to maximize sound reuse of components of the original safety case?

First step: Impact assessment to identify how changes in the system affect the safety case.

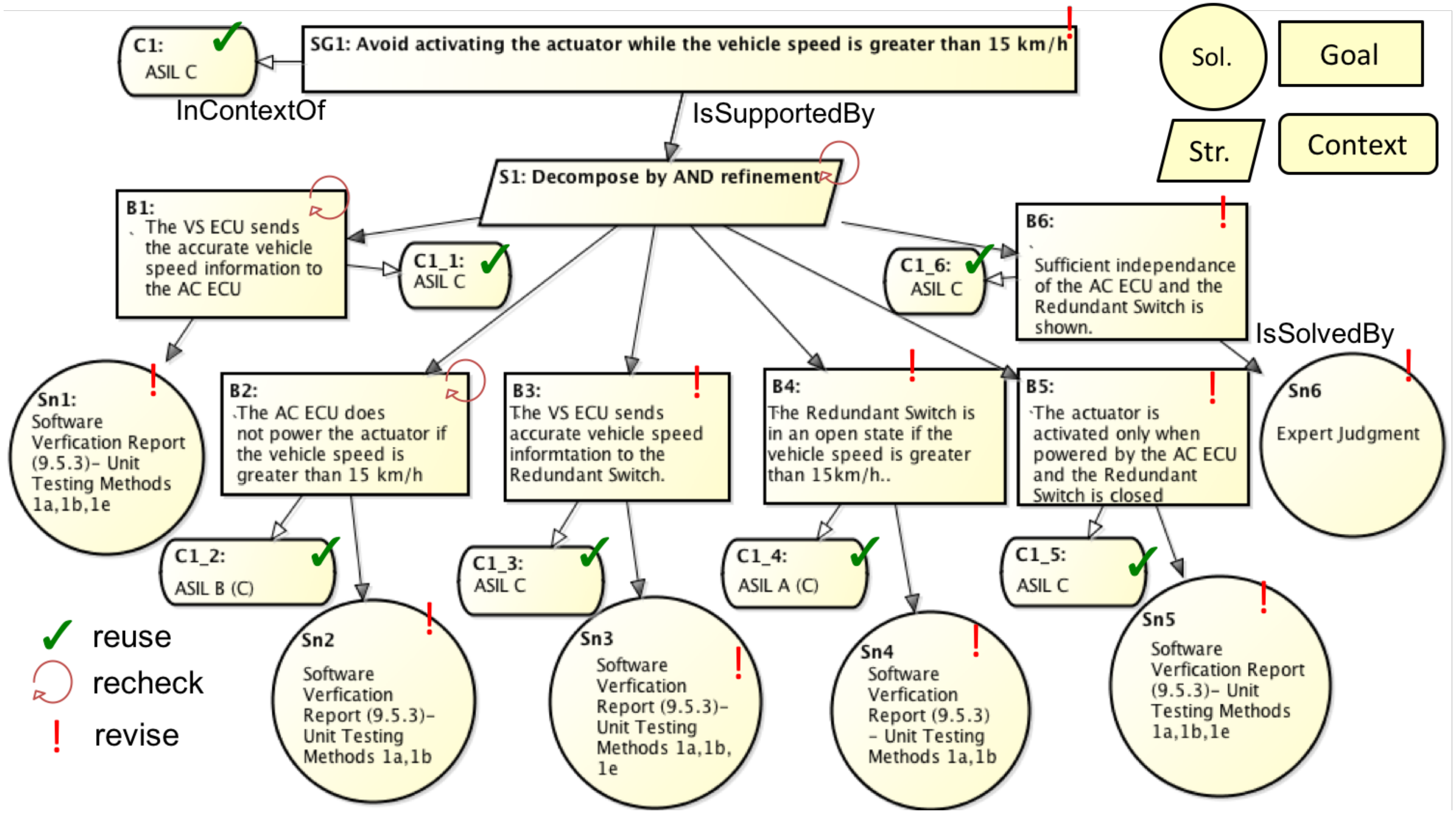
Solution: Model Based Impact Assessment



[MODELS16]: original approach

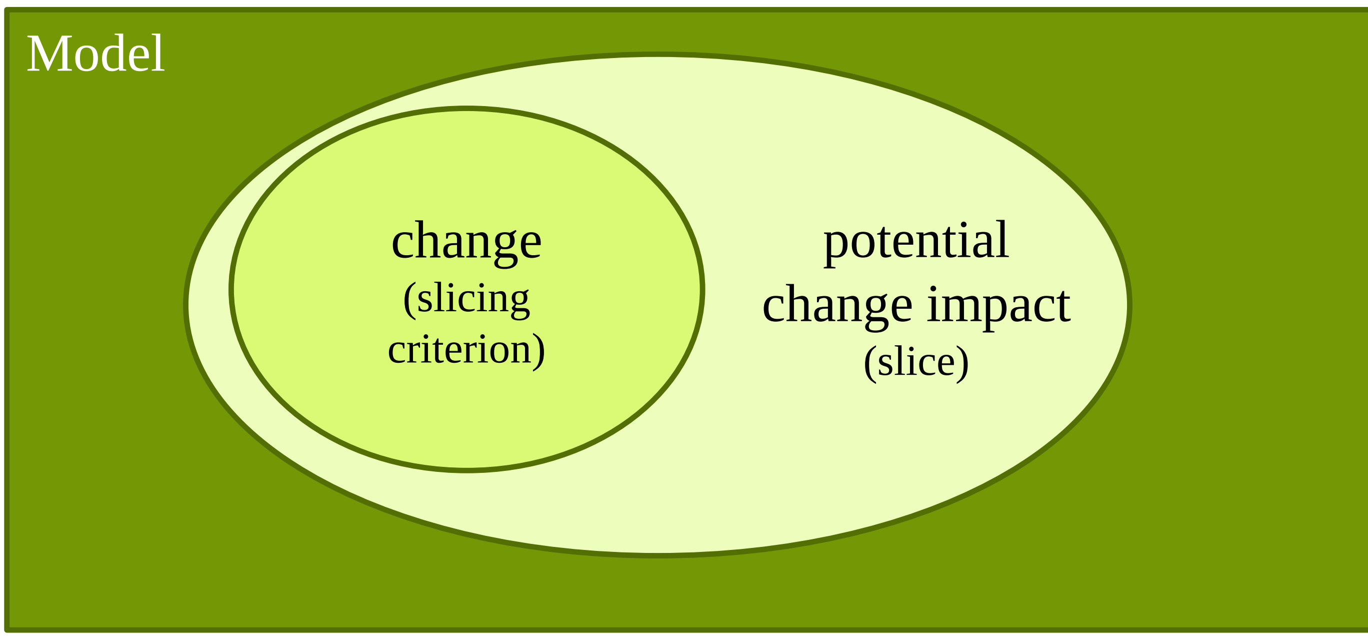
[SafeComp17]: improved approach, assurance case slicer, cost-savings analysis

Resulting Annotated Safety Case of PSD



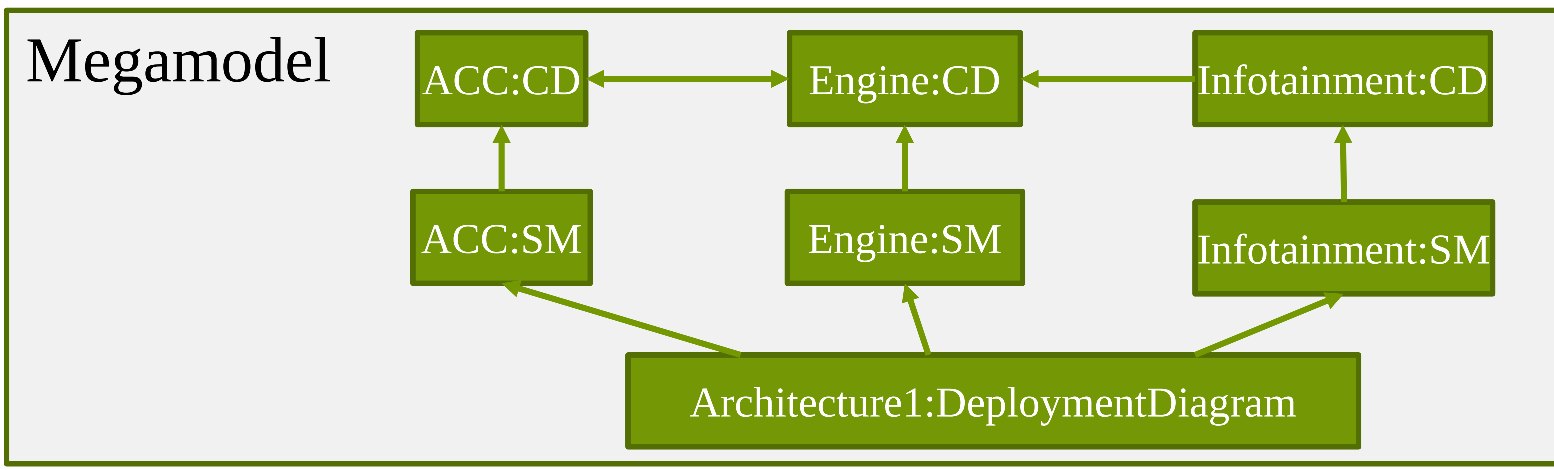
Model Slicing

- Model slicing to identify change impact is a key technique for supporting model evolution



Megamodel Slicing

- Slicing is well studied for individual models ..
- .. but not for heterogeneous collections of related models (megamodels) which are common in large projects



- megamodel slicing can be useful for identifying impact due to evolution across multiple models

Megamodel Slicing Algorithm 

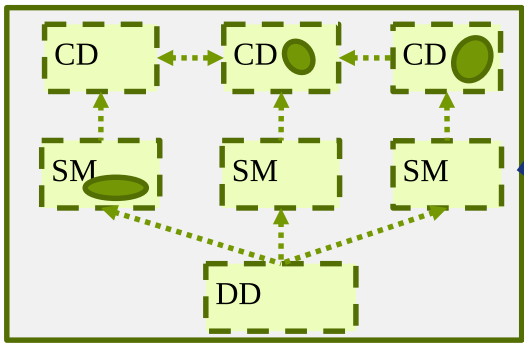
- ✓ operates on megamodels
- ✓ works with arbitrary model types (heterogenous)
- ✓ uses traceability relations to assess change impact

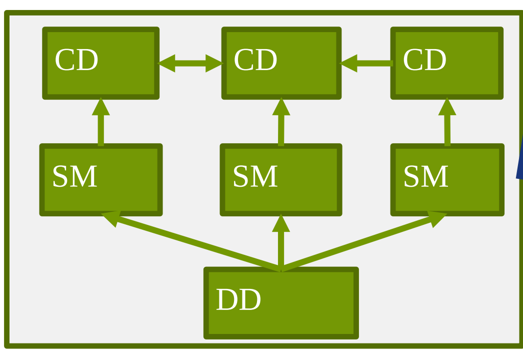
Assumptions:

1. (Slicers) There is a slicer available for each model type represented in the megamodel
2. (Dependencies) The relationships express all and only the inter-model dependencies

Megamodel Slicing Algorithm

criterion megamodel fragment



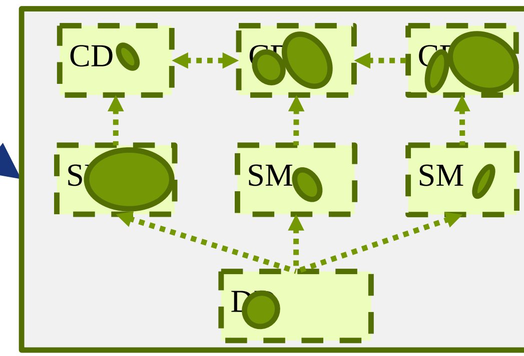


megamodel

Algorithm: Forward Megamodel Slice
Input: megamodel X, criterion megamodel fragment $S_c[X]$
Output: slice megamodel fragment $S[X]$

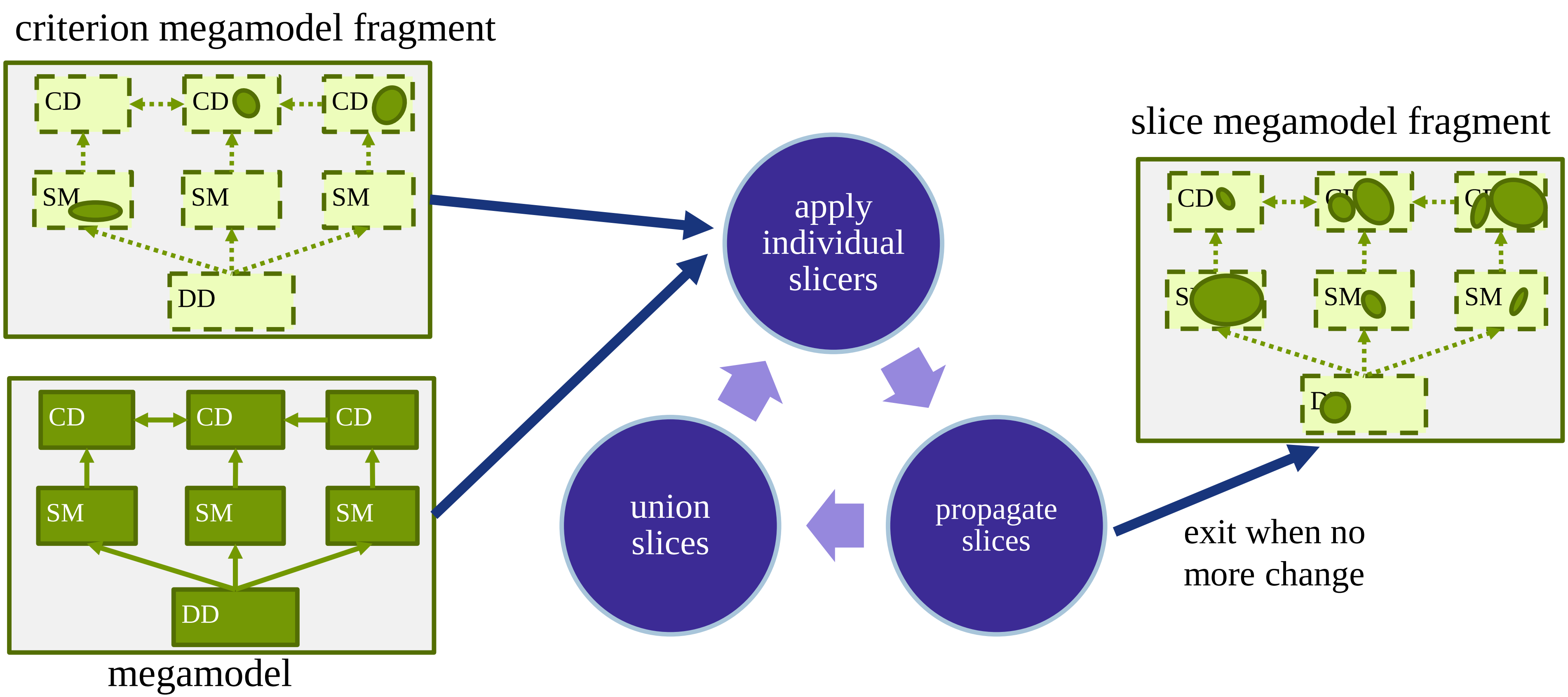
```
S[X] := S_c[X]
do {
    S'[X] := S[X]
    S_1[X] := ∅
    for (S[M] ∈ S[X]) {
        T := M.type
        S_1[M] := Slice_T(M, S[M])
        S_1[X] := Union(S_1[X], {S_1[M]})
    }
    S_2[X] := ∅
    for (S_1[M] ∈ S_1[X]) {
        for (R ∈ M.end) {
            M' := OppEnd(R, M)
            S_2[M'] := Trace(R, S_1[M])
            S_2[X] := Union(S_2[X], {S_2[M']})
        }
    }
    S[X] := Union(S_1[X], S_2[X])
} until (S[X] ⊑ S'[X])
return S[X]
```

slice megamodel fragment

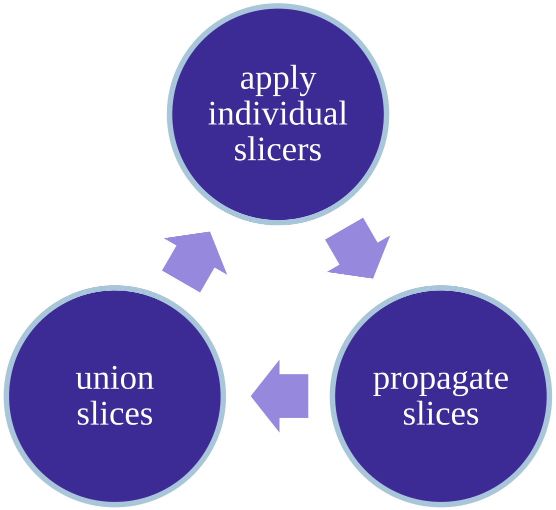


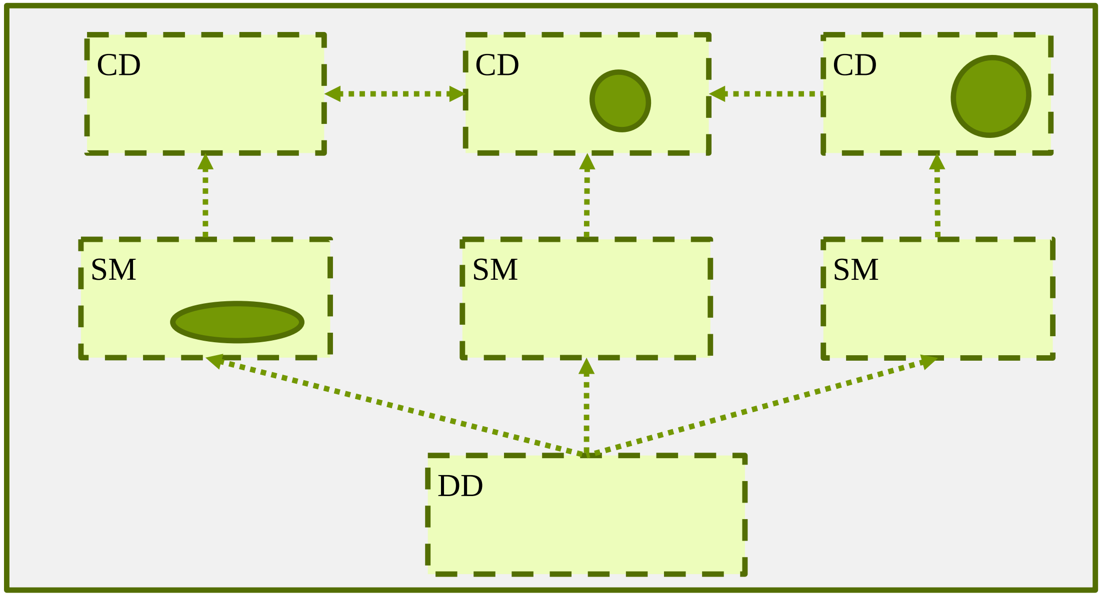
a slicer for each model type is assumed to be available

Slicing Algorithm

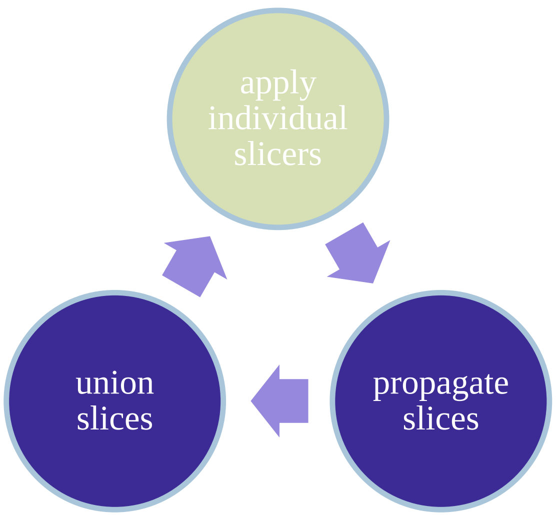


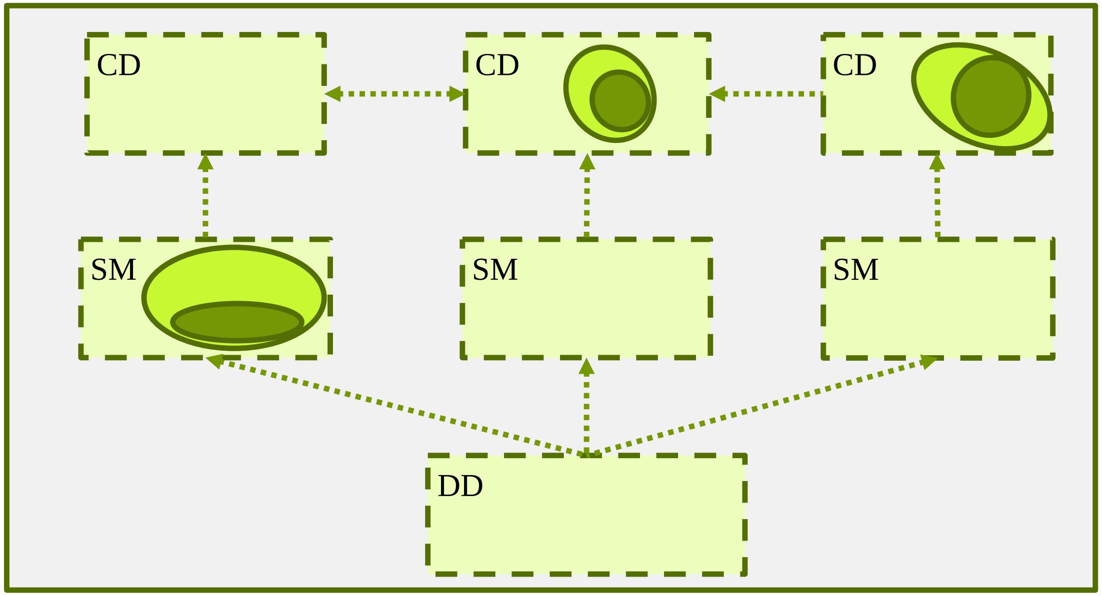
Example Run: Slicing Criterion



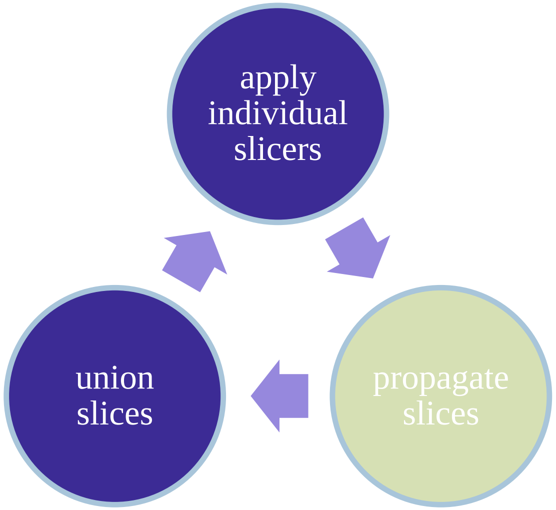


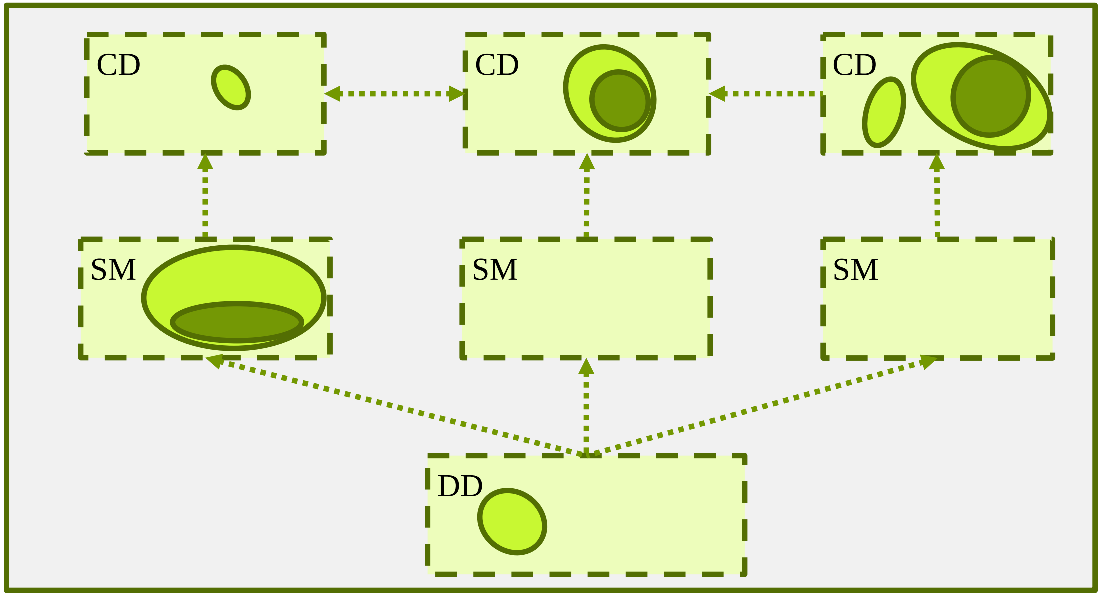
Example Run: 1st Iteration



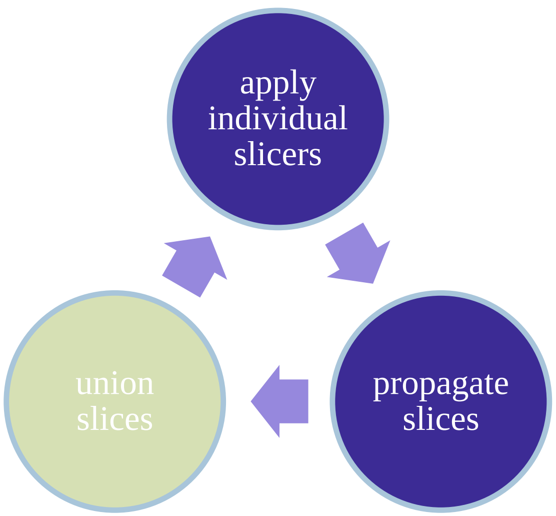


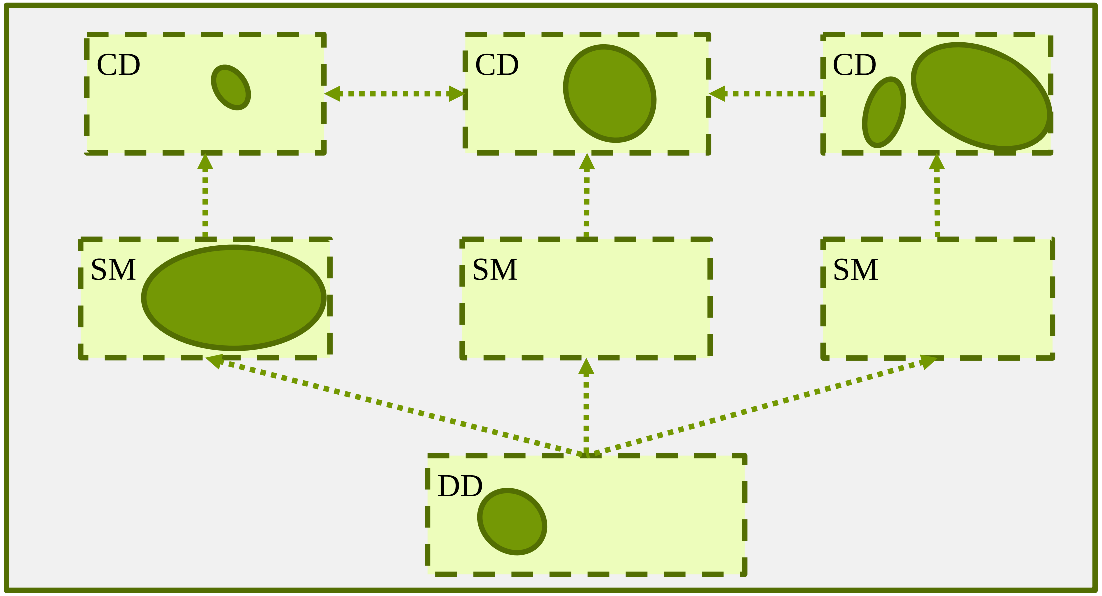
Example Run: 1st Iteration



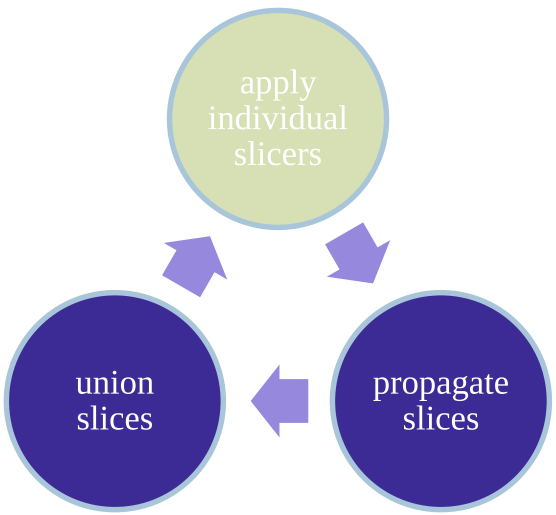


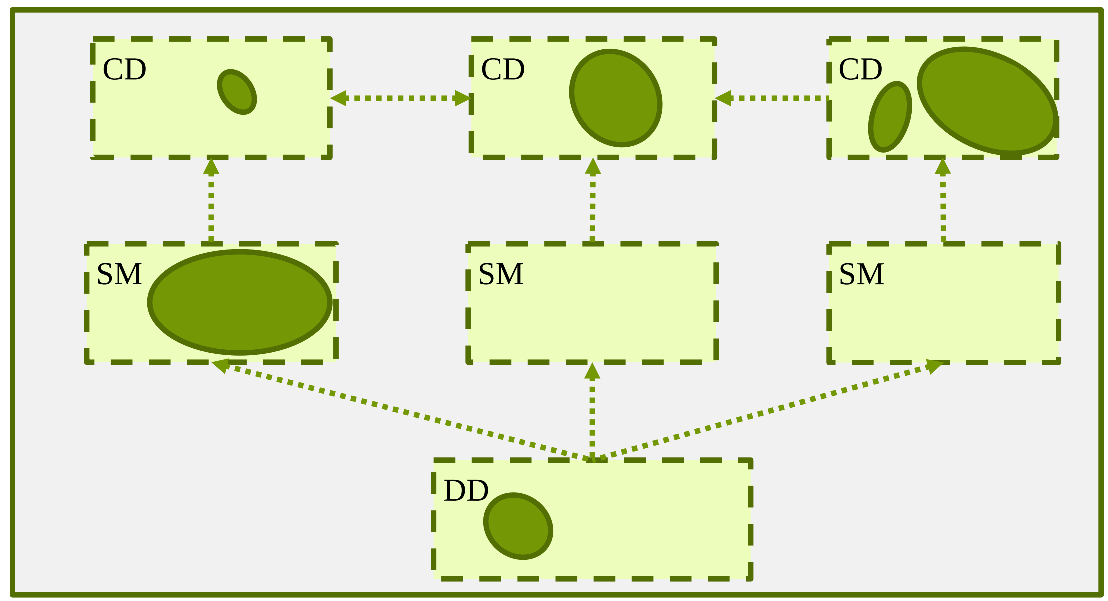
Example Run: 1st Iteration



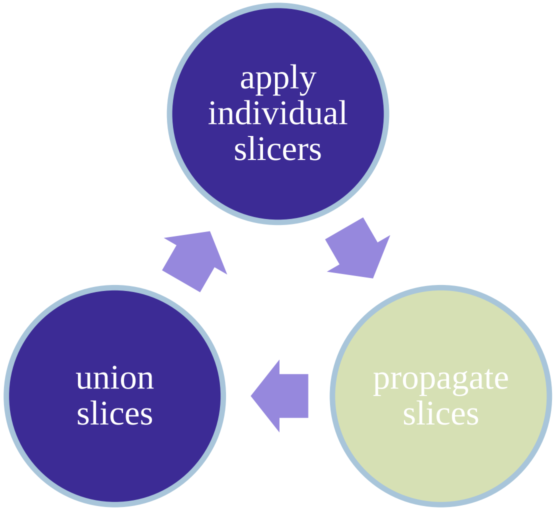


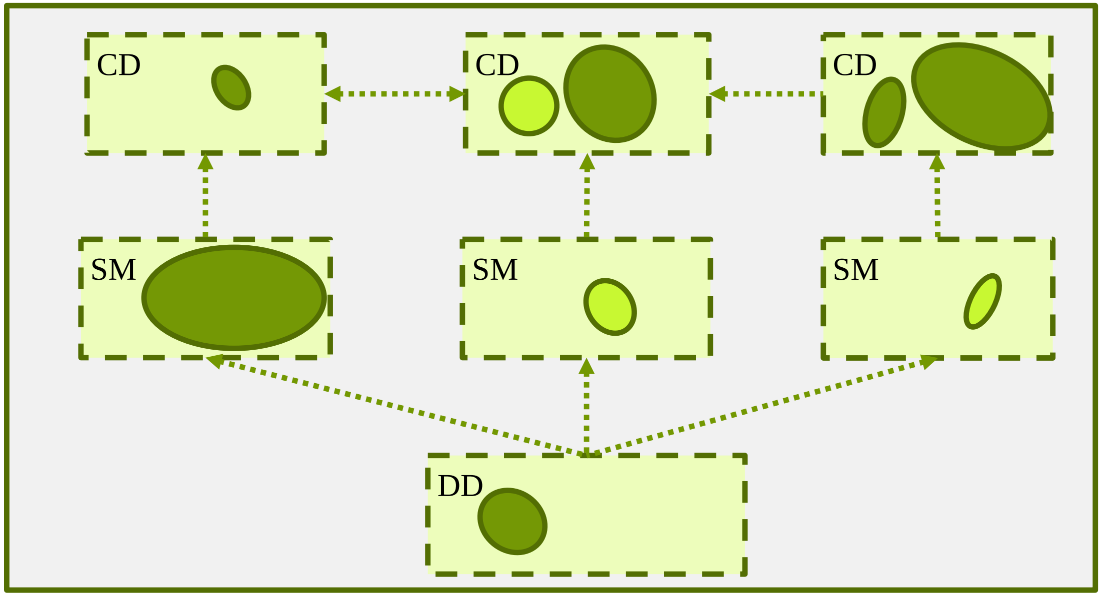
Example Run: 2nd Iteration



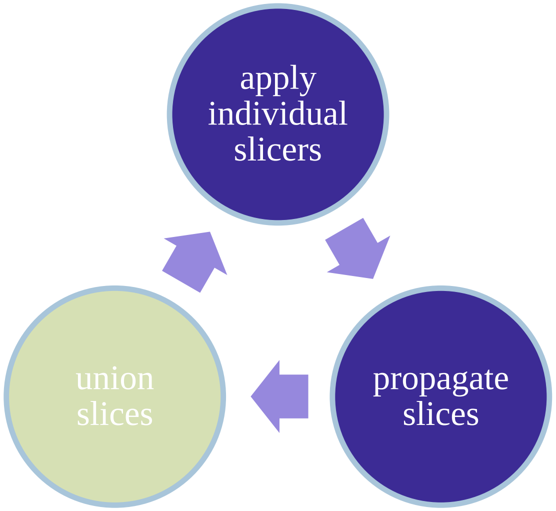


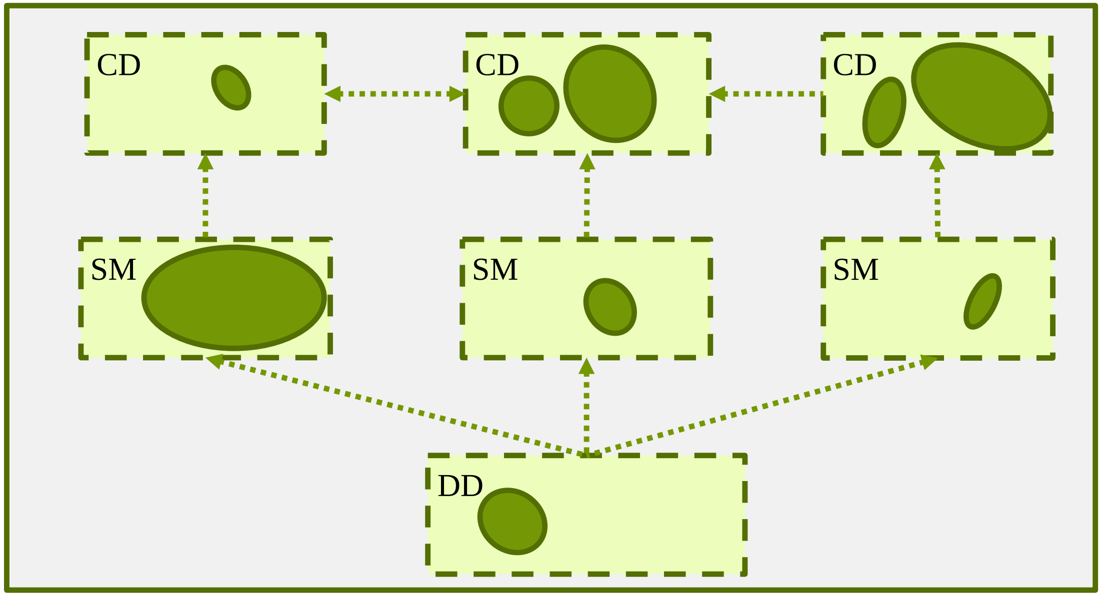
Example Run: 2nd Iteration



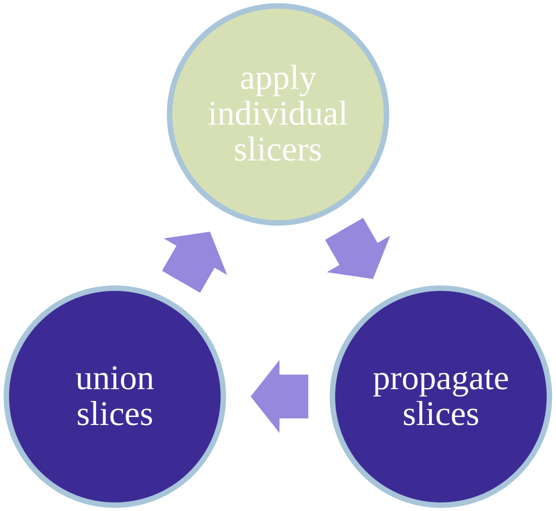


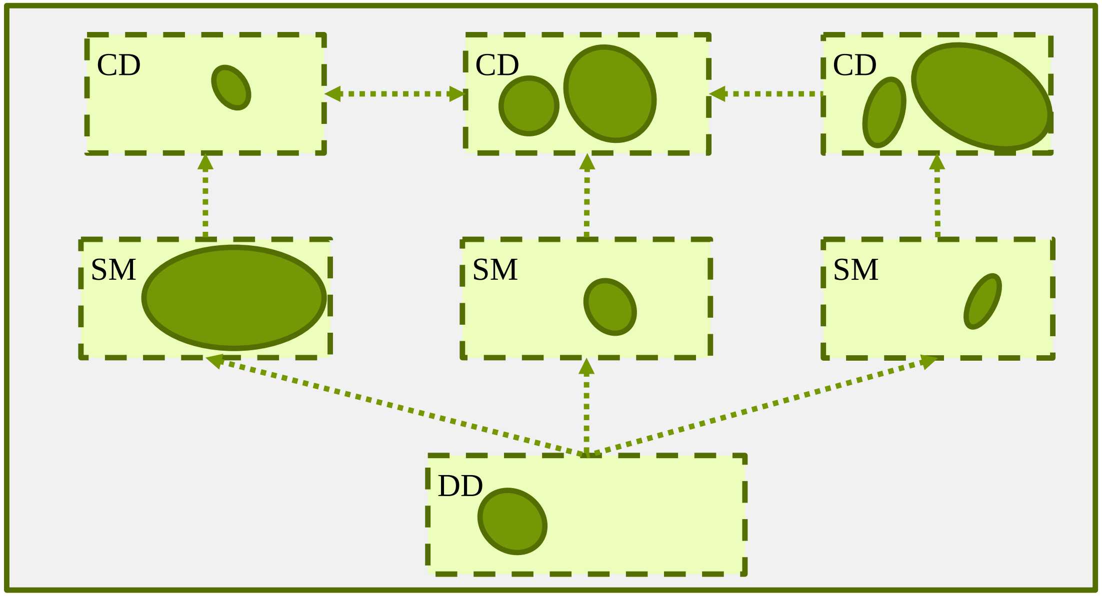
Example Run: 2nd Iteration



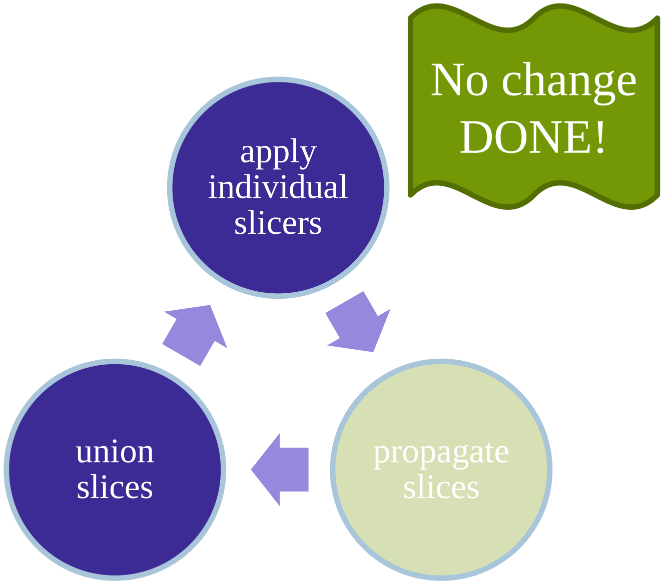


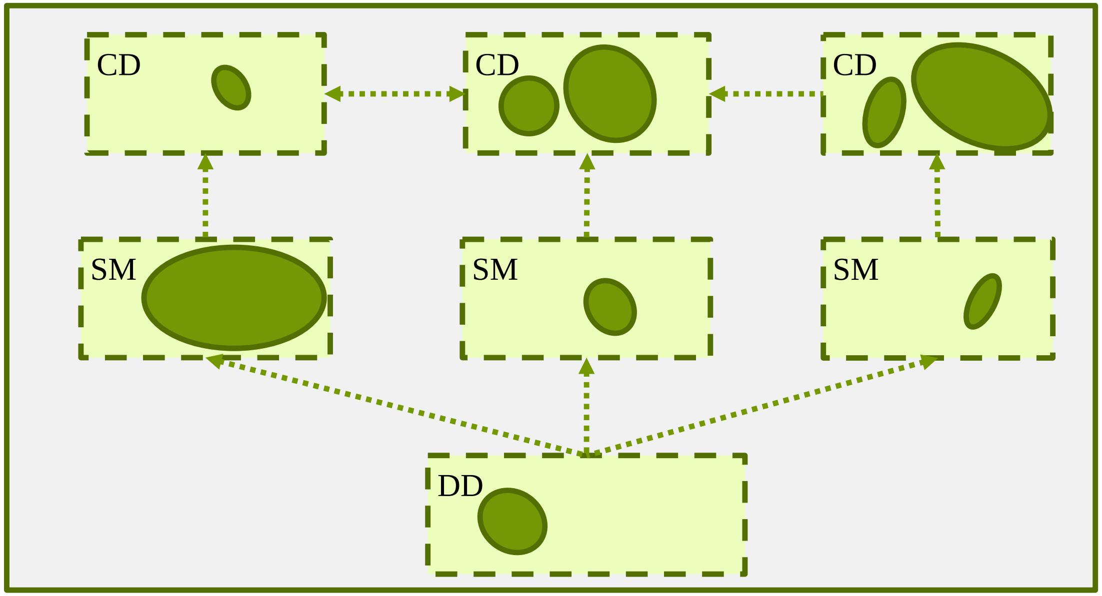
Example Run: 3rd Iteration



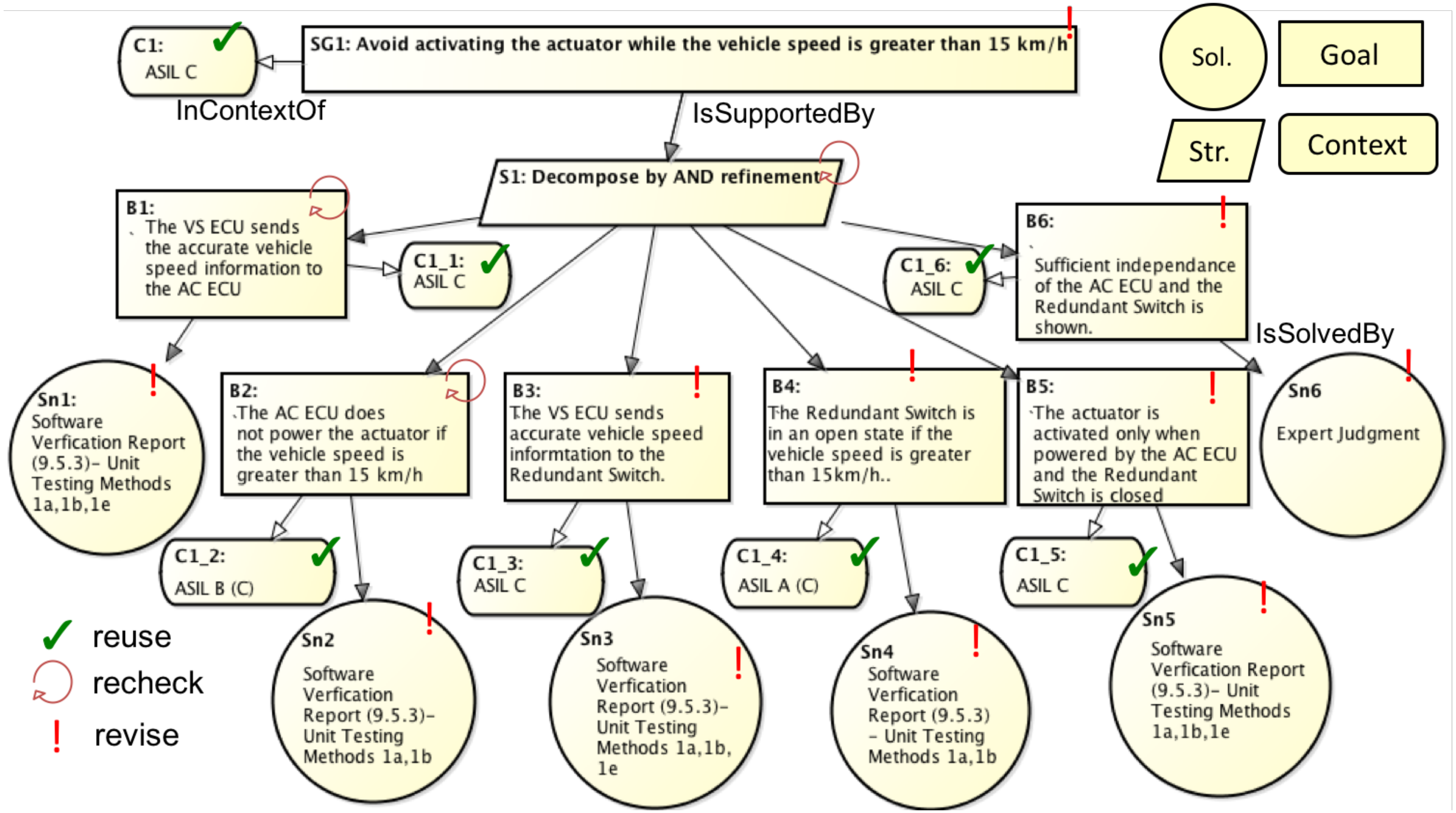


Example Run: 3rd Iteration





Resulting Annotated Safety Case of PSD



Improving the Precision of Impact Assessment

[SafeComp17]

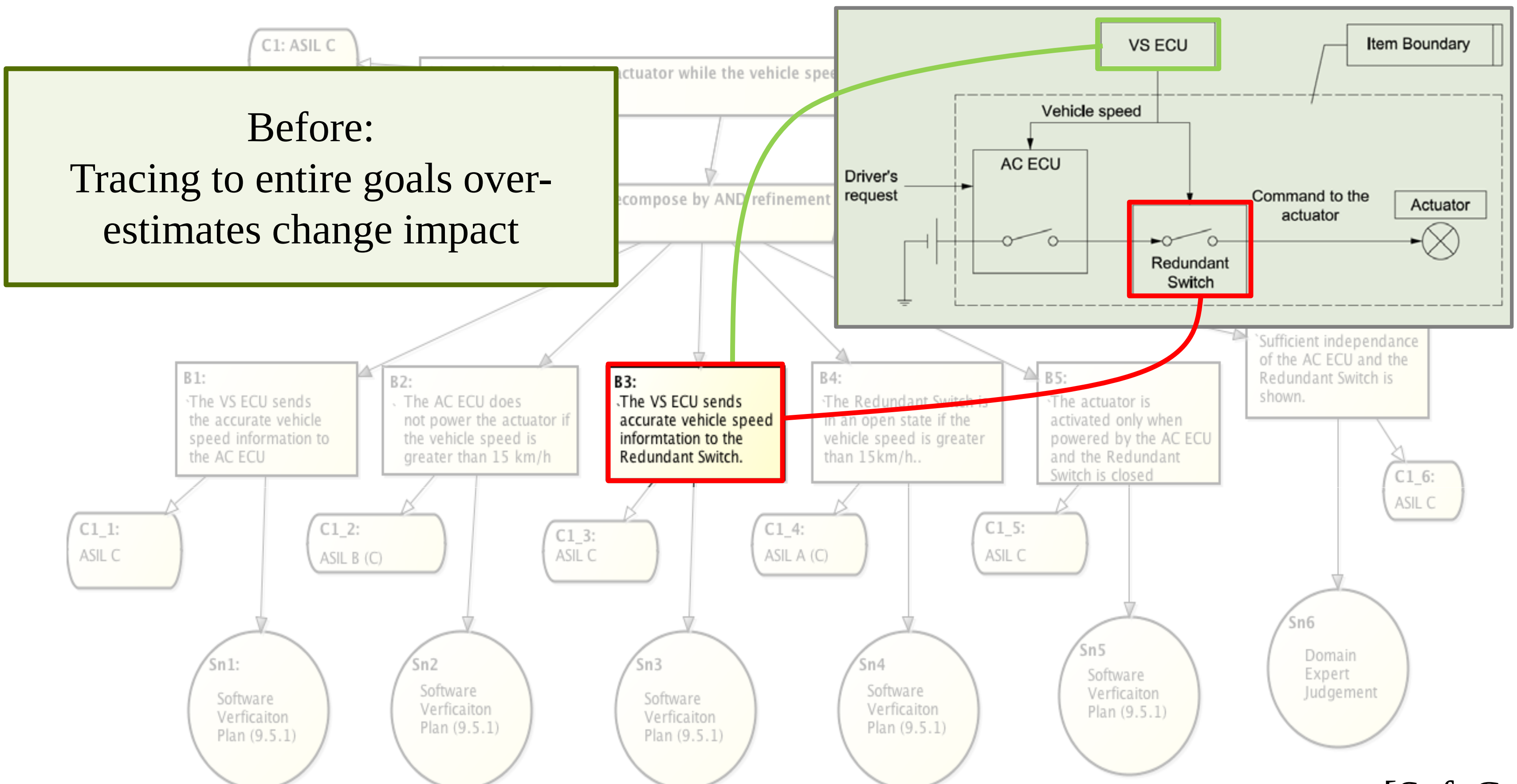
Six ways to improve precision

1. Increasing the granularity of traceability between the system and the assurance case
2. Identifying sensitivity of assurance case to system changes
3. Understanding semantics of strategies
4. Decoupling revision from rechecking
5. Realizing that strengthened solutions do not impact associated goals
6. Understanding standard-system and standard-safety case traceability (specific to safety standards)

Outcome:

fewer “false positives” in “revise” and “recheck” annotations

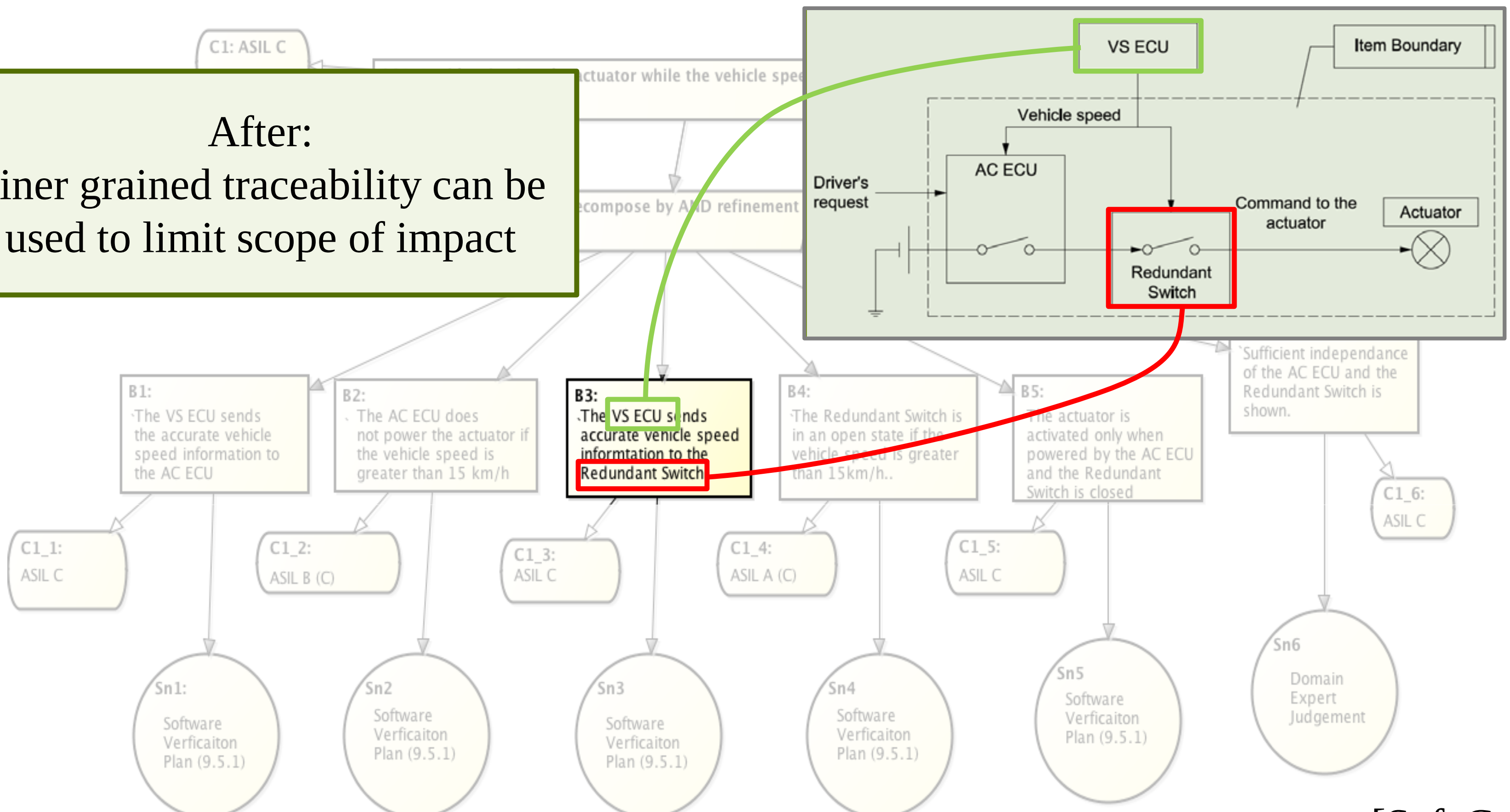
1: Increasing Granularity of Traceability between System and Assurance Case



[SafeComp17]

1: Increasing Granularity of Traceability between System and Assurance Case

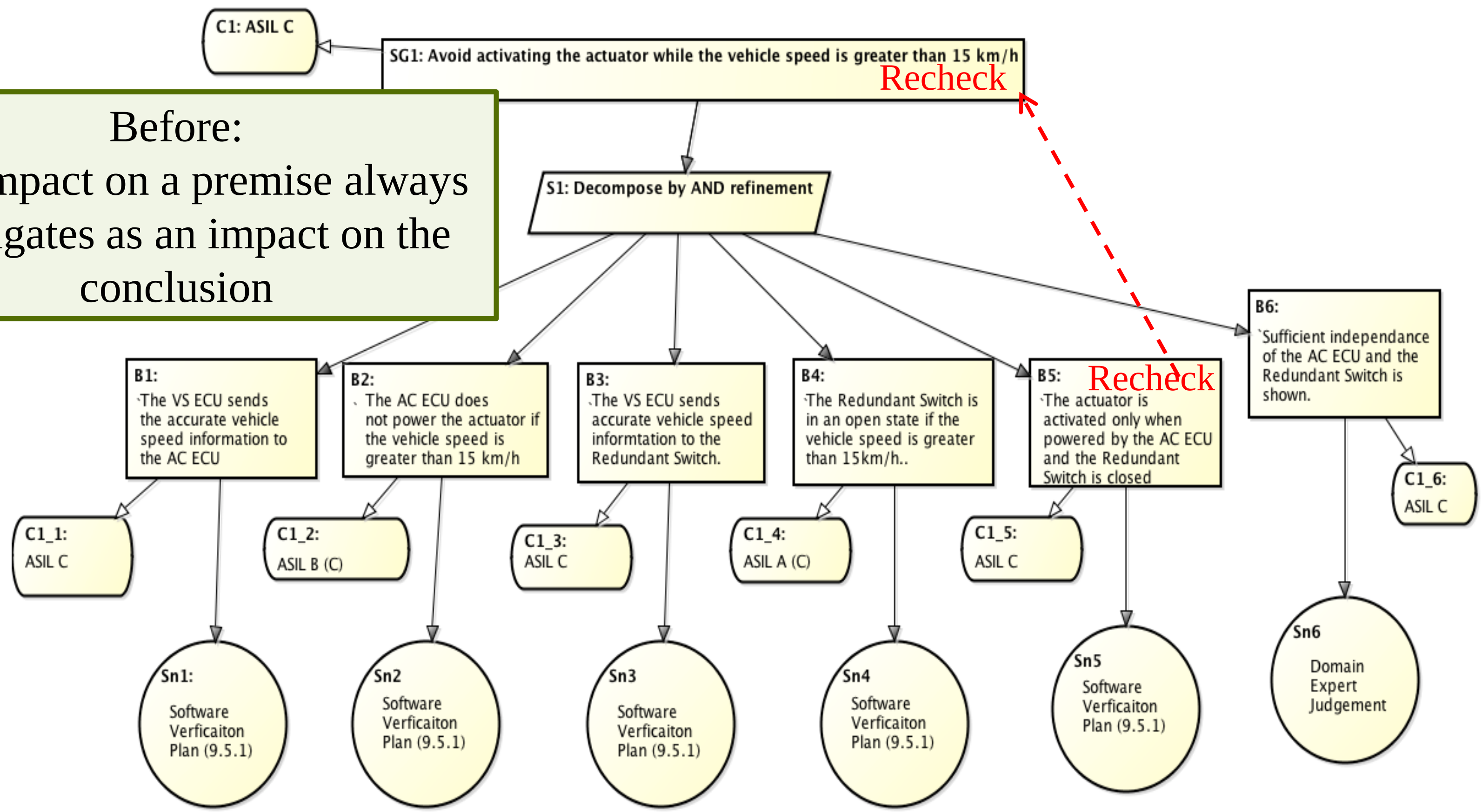
After:
Finer grained traceability can be used to limit scope of impact



[SafeComp17]

3: Understanding Semantics of Strategies

Before:
Any impact on a premise always propagates as an impact on the conclusion

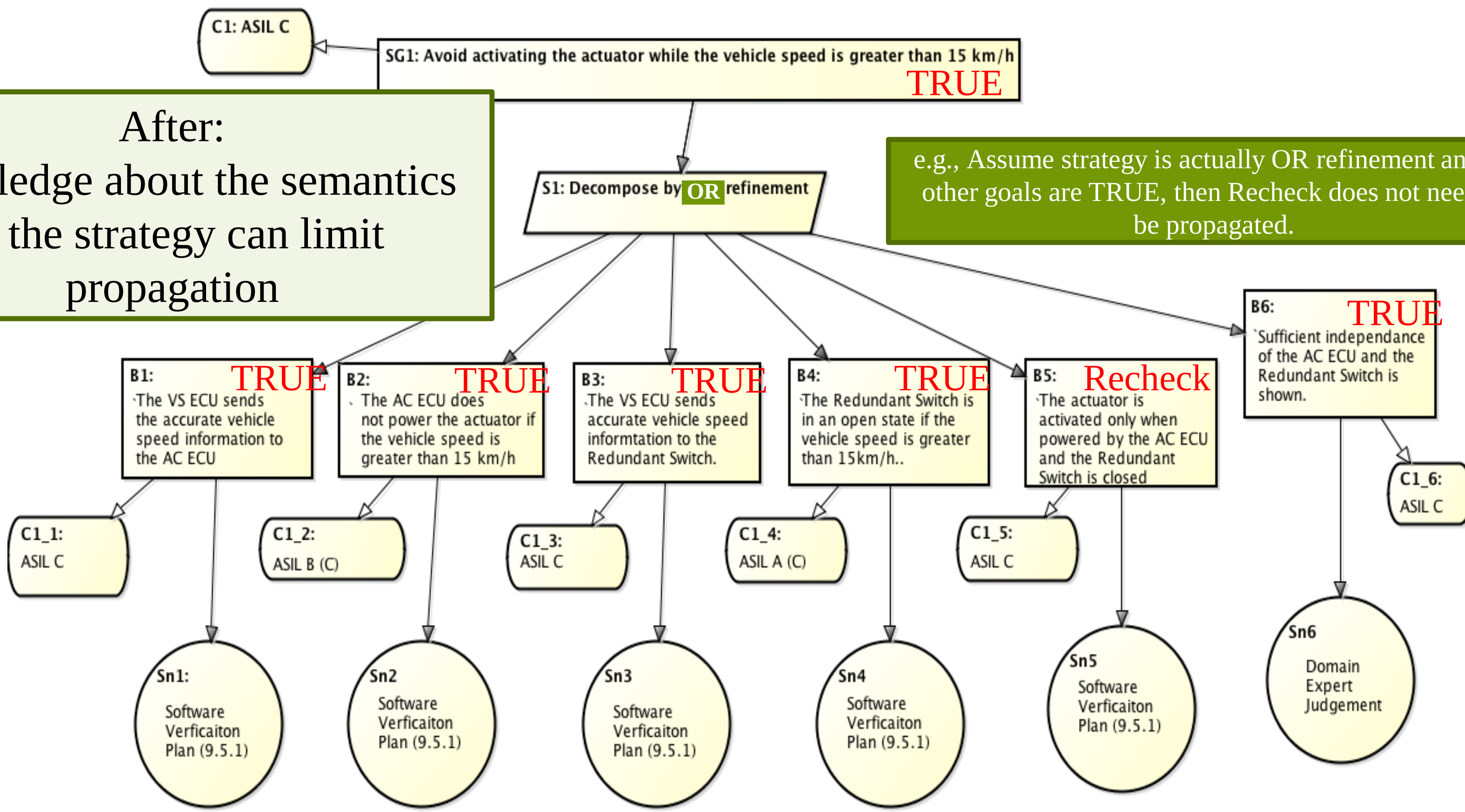


[SafeComp17]

3: Understanding Semantics of Strategies

After:
Knowledge about the semantics of the strategy can limit propagation

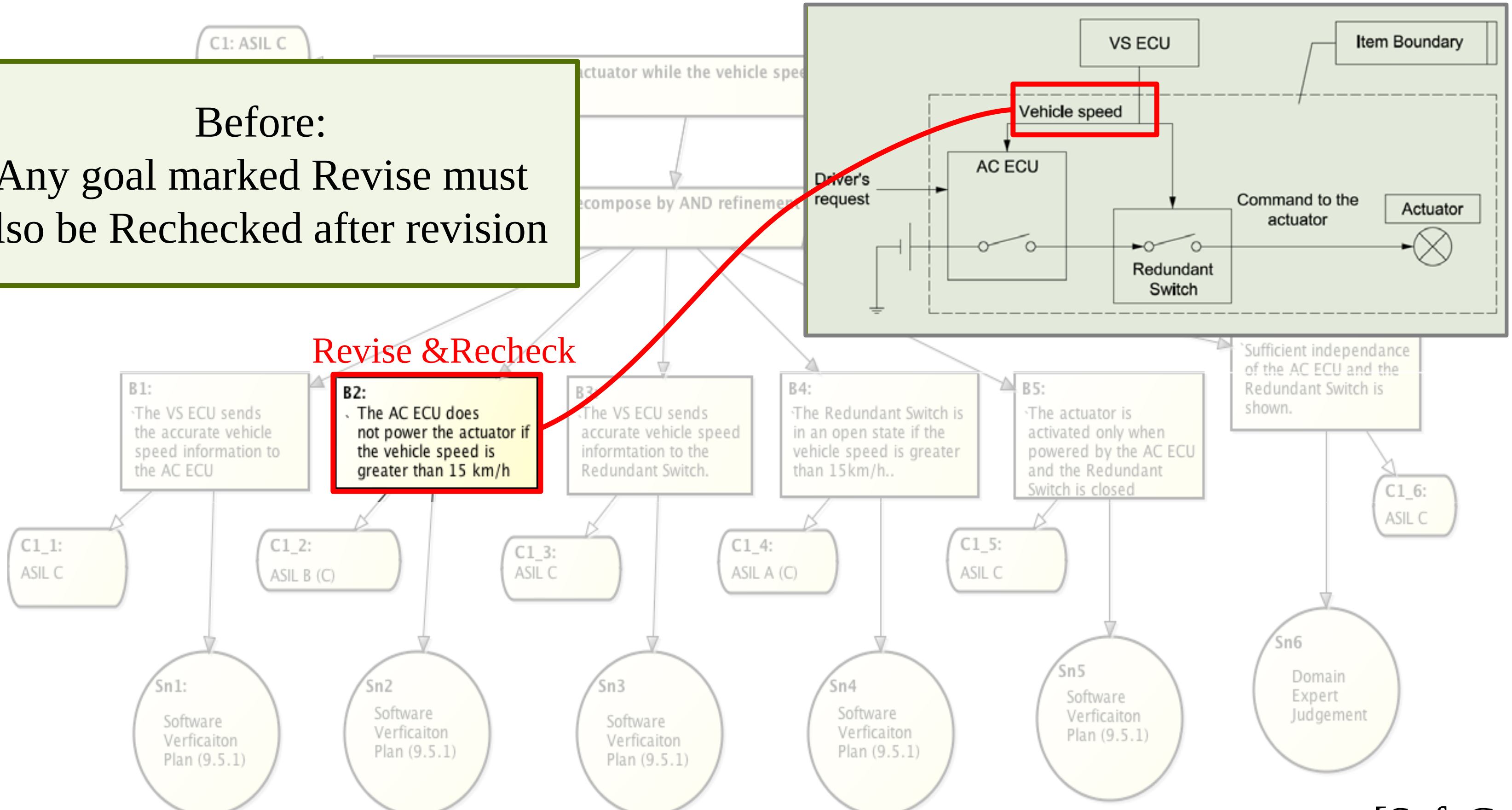
e.g., Assume strategy is actually OR refinement and all other goals are TRUE, then Recheck does not need to be propagated.



[SafeComp17]

4: Decoupling Revision from Rechecking

Before:
Any goal marked Revise must also be Rechecked after revision



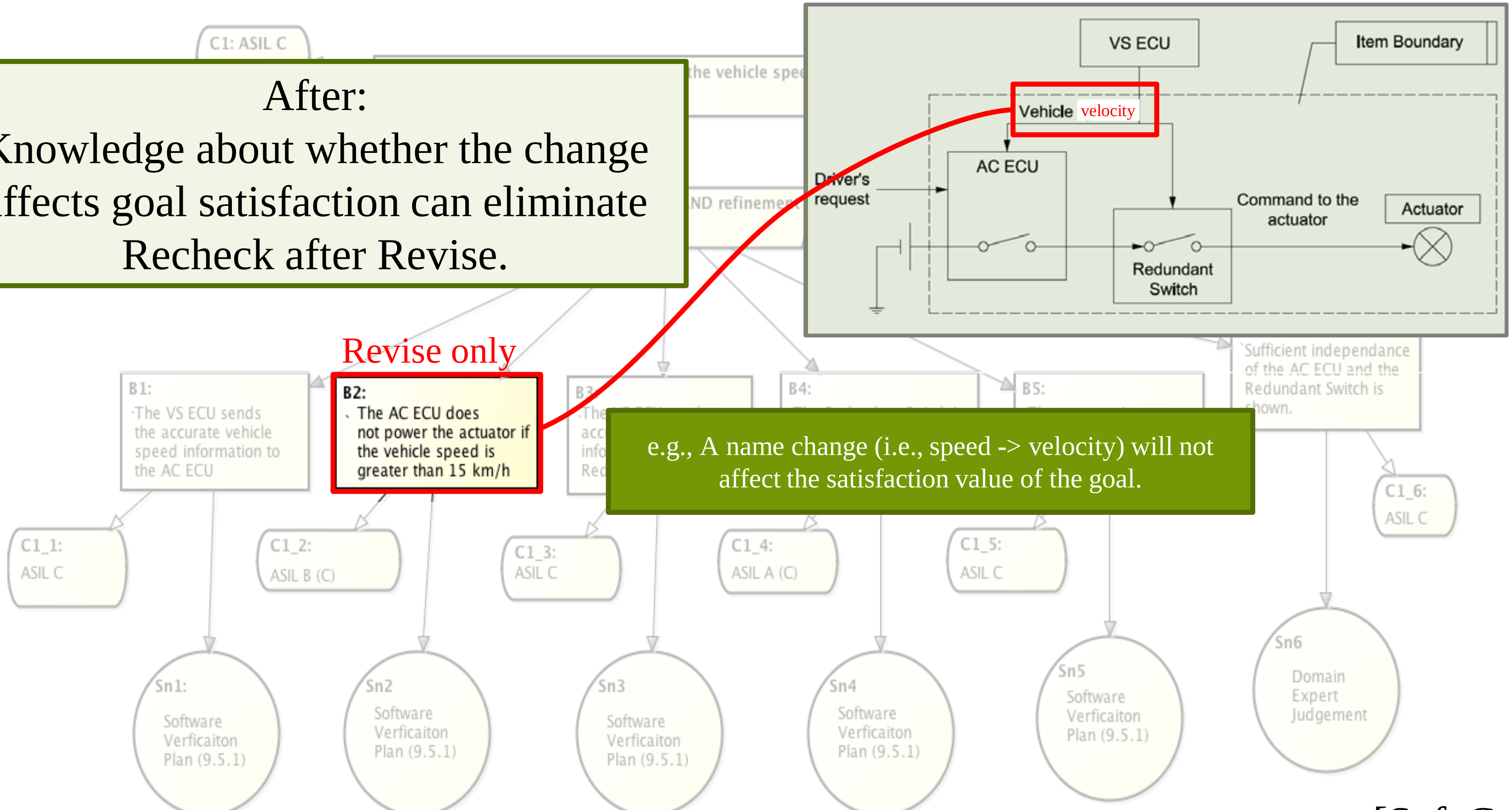
[SafeComp17]

4: Decoupling Revision from Rechecking

After:
Knowledge about whether the change affects goal satisfaction can eliminate Recheck after Revise.

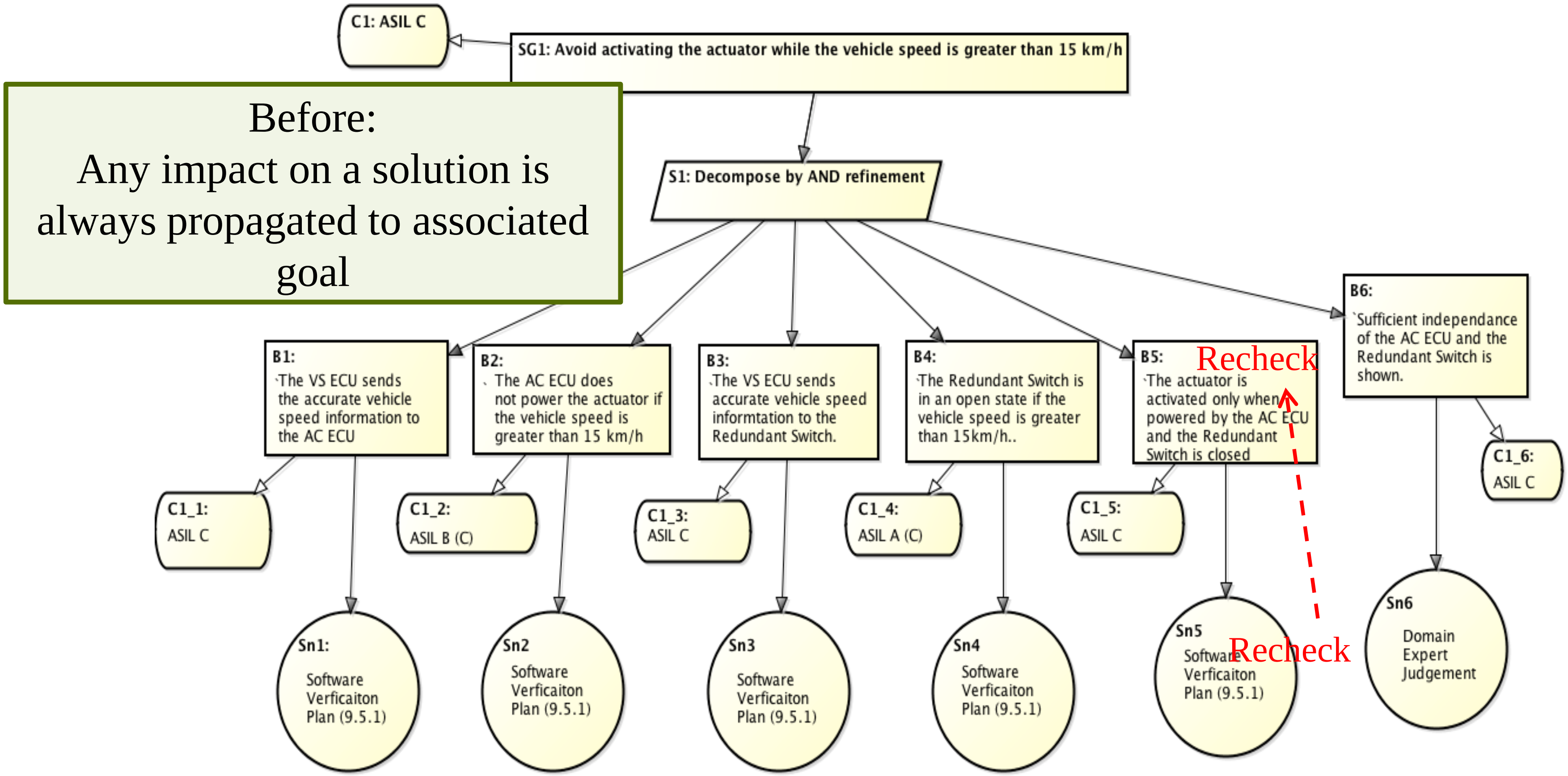
Revise only

e.g., A name change (i.e., speed -> velocity) will not affect the satisfaction value of the goal.



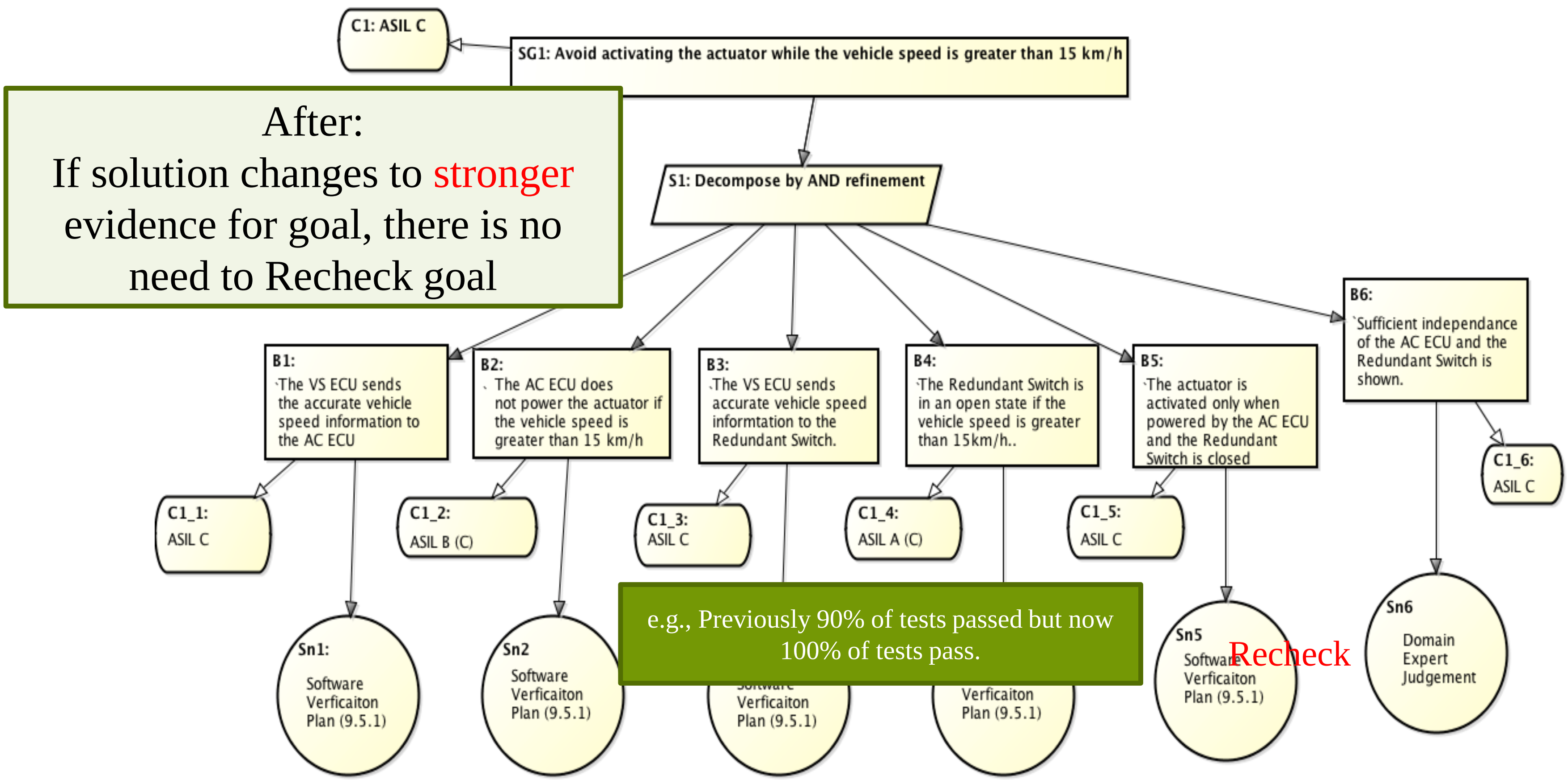
[SafeComp17]

5: Strengthened Solutions Do Not Impact Associated Goals



[SafeComp17]

5: Strengthened Solutions Do Not Impact Associated Goals



[SafeComp17]

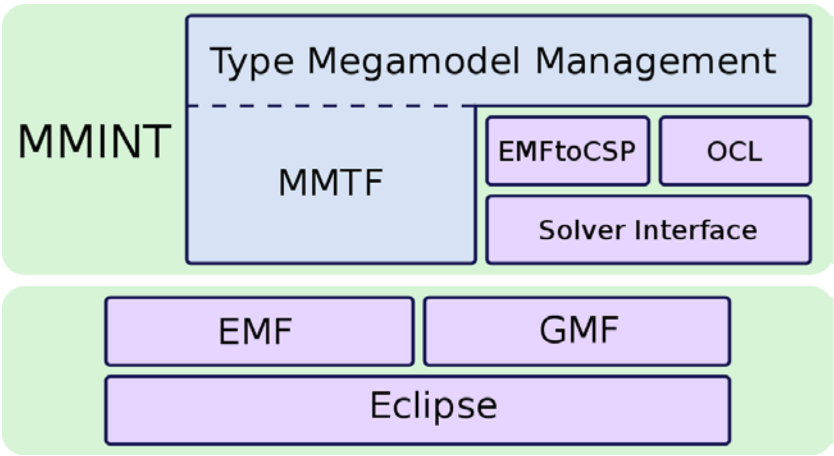
Soundness

- limited to claims of evolution due to atom **changes** and **deletions**
- **added** components required to be assessed by assurance engineer

Relative Efficiency

- an impact assessment approach is **more efficient** if it reports fewer “false positives"
- efficiency relies on the information the algorithm uses to determine impact (depth of knowledge about dependency relations in assurance case)

MMINT: Tool Support for Model-Driven Assurance Case Handling

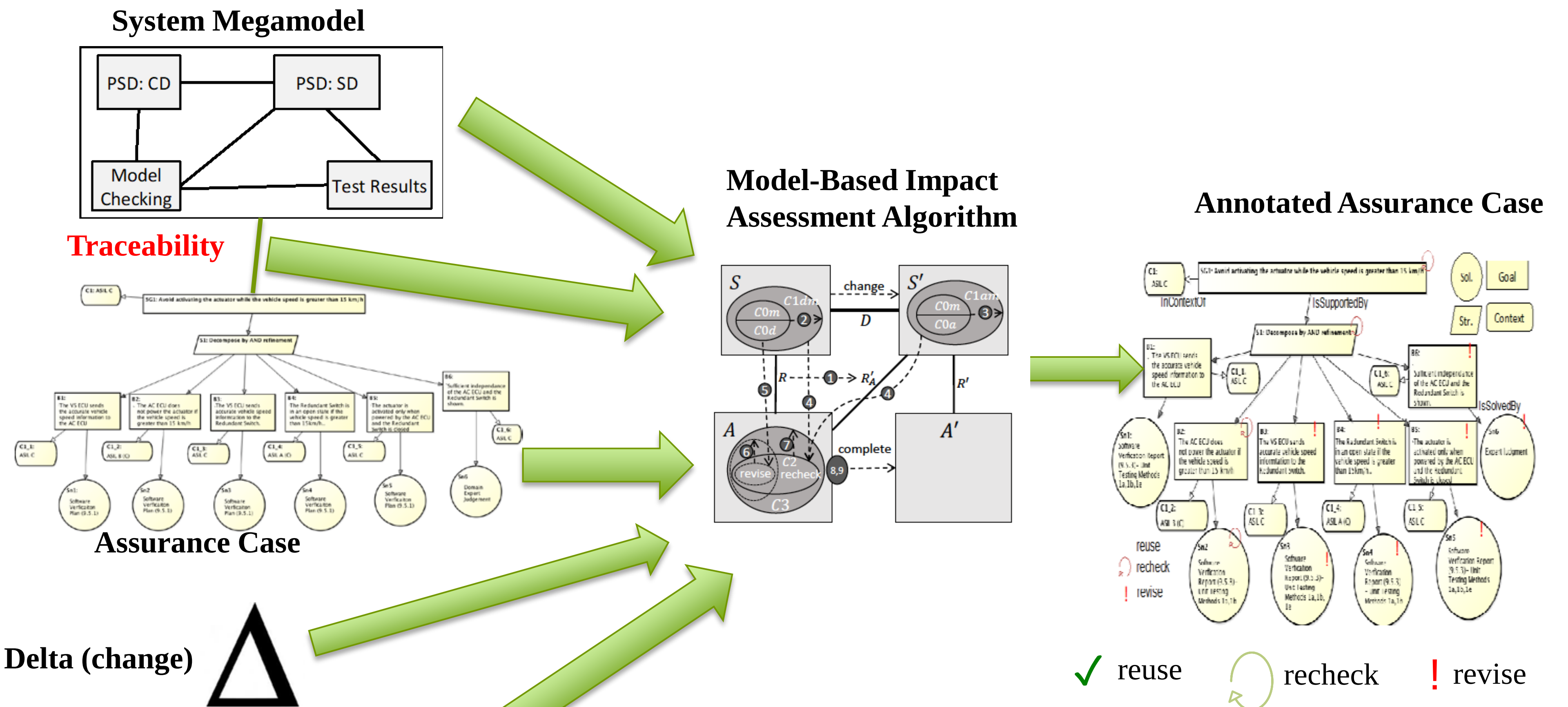


* https://github.com/adisandro/MMINT

Features:

- ✓ assurance cases as a model type
- ✓ model management operators for assurance cases (e.g., assurance case **slice**)
- ✓ explicit trace links between the assurance case and the standard/system
- ✓ heterogeneous megamodeling operators (e.g., megamodel slice) as model management workflows.

Summary: Model Based Impact Assessment to Support Assurance Case Reuse due to System Evolution



Model Slicers

[MODELS16]: original approach

[SafeComp17]: improved approach, assurance case slicer, cost-savings analysis

Summary: Handling Assurance

– Informal (although rigorous)
– Expensive to produce
– Difficult to analyze / reuse

How to do automation over such informal artifacts?
– assessing compliance due to evolution
– compliance to multiple standards
– compliance of product lines
… and how to do this in a sound way?

Handling Assurance

– Informal (although rigorous)
– Expensive to produce
– Difficult to analyze / reuse

How to do automation over such informal artifacts?

– assessing compliance due to evolution
– compliance to multiple standards
– compliance of product lines

… and how to do this in a sound way?

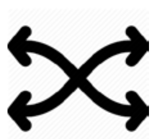
Compliance to Multiple Standards

Problem:

– How to reuse assurance work given a change to the standard or an introduction of a new standard?

Approach:

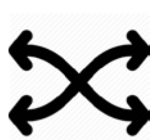
– Identify overlaps between the “old” standard and the new one



match

diff

– Reuse portions of assurance cases corresponding to these overlaps, merging with newly developed parts



match

merge

Handling Assurance

- Informal (although rigorous)
- Expensive to produce
- Difficult to analyze / reuse

How to do automation over such informal artifacts?

- assessing compliance due to evolution
- compliance to multiple standards
- compliance of product lines

… and how to do this in a sound way?

Compliance of Product Lines

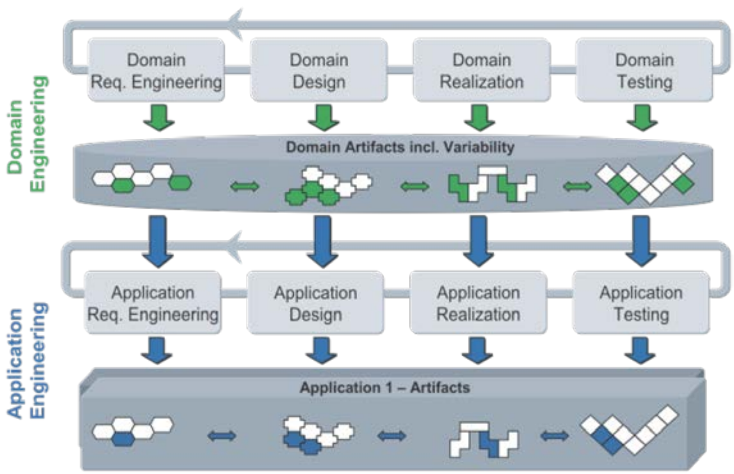
Product lines are essential in many domains: automotive, consumer electronics, aerospace

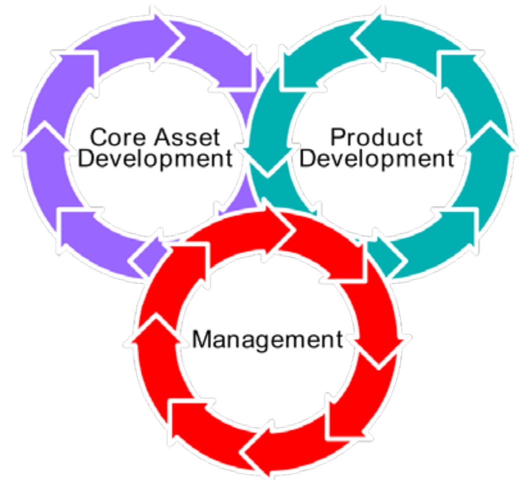
Creating and maintaining individual assurance cases for every similar but somewhat different product is very expensive

How to reuse assurance work from one product to another?

Compliance of Product Lines

SPLE - a discipline that promotes planned and predictive software reuse





Is there a meaningful notion of a product-line assurance case?

What evidence can be generated for every product and which is product specific?

- How to reuse product-specific evidence?
- How to determine cases for which generated evidence is most beneficial for reuse?

Summary: Handling Assurance

Assuring safety / security / privacy is a broad and complex problem

Assurance cases:

- Informal (although rigorous)
- Expensive to produce
- Difficult to analyze / reuse

Model management can be adapted to provide automation over such artifacts in a sound way

- assessing compliance due to evolution
- compliance to multiple standards
- compliance of product lines

Regulatory Compliance and SE

- Regulatory compliance = complex standards = safety-critical systems = all about process

Regulatory Compliance and SE

Certification is increasingly product-based

General approach of

- identifying notion of requirements w.r.t. a particular property of interest and
- building arguments that they are adequately addressed, as is done in assurance cases

… is applicable to a much broader class of systems

Argument for Sufficient Evidence

- A big unifier for SEFM!
  - A lot of software analysis methods
  - Different types of testing / model-checking / theorem-proving

Challenges

Current standards

- ❑ domain-specific
- ❑ concern-specific
- ❑ primarily assume that software is engineered

What about assurance cases / compliance?

- ❑ Domain independence vs. specificity
- ❑ Concern (safety / security / privacy) independence vs. specificity
- ❑ Designed vs. “learned” artifacts
  - E.g., self-driving cars

Recent Privacy Example (June 2017)

http://www.bbc.co.uk/news/business-40324983

Personal data on a connected car that you sold or rented …stays on it, without provisions for removal!!!!!

"The collection and use of data by Connected and Autonomous Vehicles (CAV) is not a matter of significant concern for consumers"

Report into Connected and Autonomous Vehicles (CAVs) commissioned by the UK's Society of Motor Manufacturers and Traders

"So the next time you hire a connected car it might be worth asking the rental provider what data removal options they provide and whether they can give you written proof that your personal data has been successfully and totally erased"

Need to certify self-driving vehicles …and even smart appliances!

Be careful of your fridge!!!!!

“Your personal data is as secure as the weakest link on your network”

https://www.theguardian.com/technology/2017/jul/24/smart-tvs-fridges-should-carry-security-rating-police-chief-says

Acknowledgements

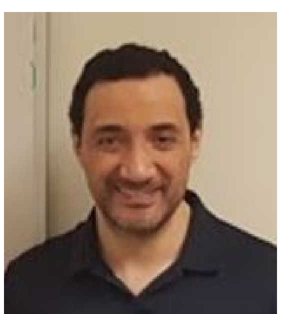
In Toronto

Sahar Kokaly

Rick Salay

Alessio Di Sandro

Nick Fung



Ramy Shahin

In McMaster

Mark Lawford

Tom Maibaum

Valentin Cassano

In General Motors:

Joe D'Ambrosio

Ramesh S

Related Work

Using Modeling for Compliance

Compliance Management Frameworks

[Hamou-Lhadj], [DLVara], [Habli2008]

Algorithms and Operators for Compliance

[Nejati], [Ghanavati]

Modeling Standards and Assurance Cases

[Kelly2004], [Luo] [Panesar-Walawege], [Ghanavati] [Bandur]

Model-based Approaches for Compliance

[Habli2010], [Gallina]

Safety Case Construction and Maintenance

[Kelly2001], [Li], [Jaradat]