Correctness and Completeness in Requirement Engineering

FUSE Seminar 2016-09-23

Anders Werneman - Qamcom

Jörgen Tryggvesson - Comentor

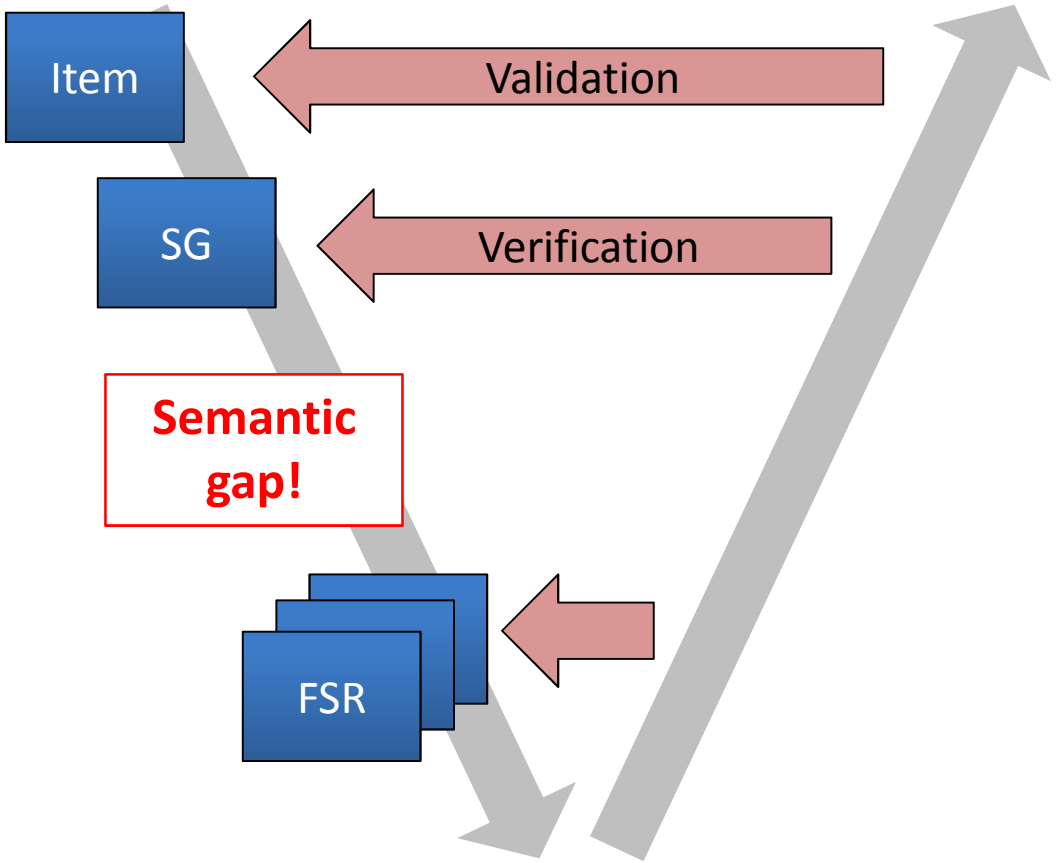
Refinement verification

- The main problem with all non-formal and semi-formal requirement refinement methods is to argue for completeness and correctness

- Legacy, reuse and inheritance are limitations for introducing new processes and methods, it can not be expected to start everything from scratch

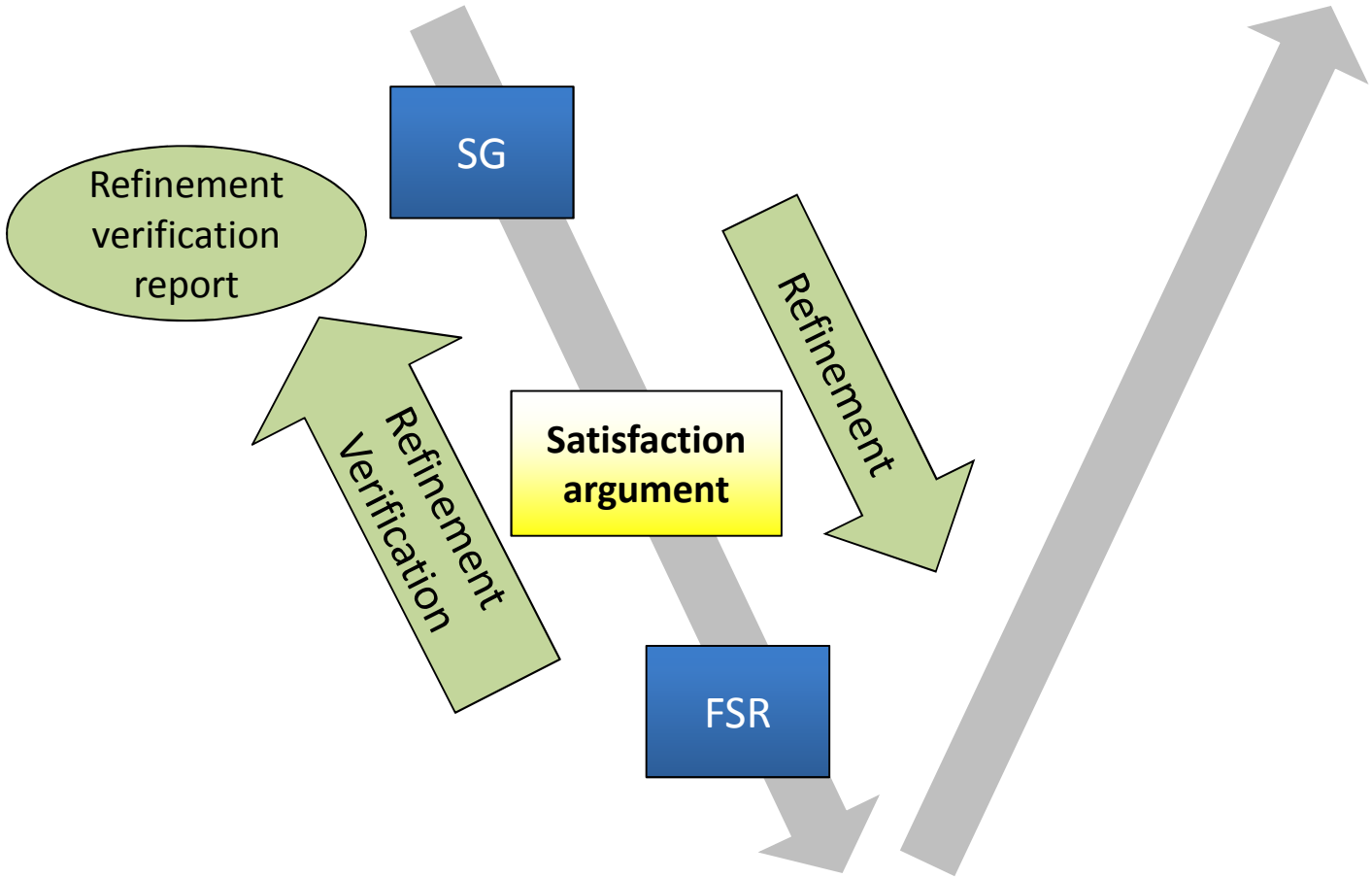
- Refinement verification with satisfaction arguments[1] is one way to address this problem

The Problem

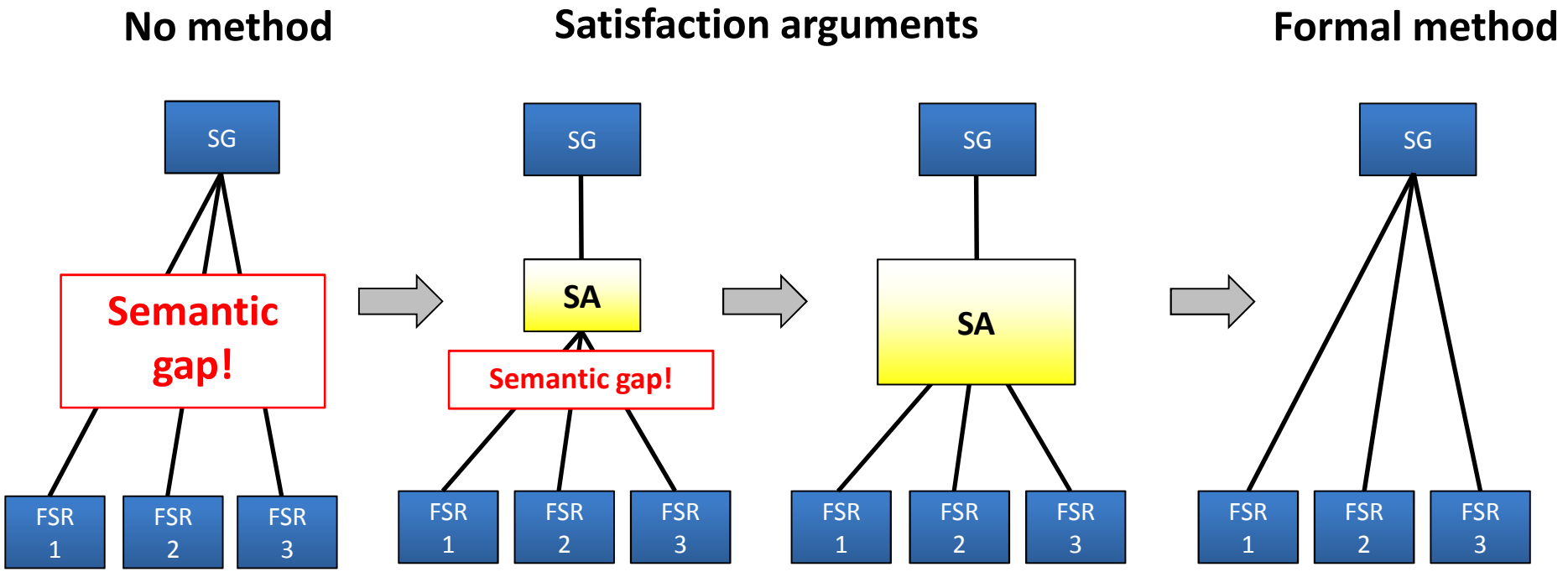


semcon KTH SP COMENTOR qamcom VOLVO FUSE

The Idea


SG
Refinement verification report
Refinement
Satisfaction argument
Refinement Verification
FSR

Satisfaction arguments



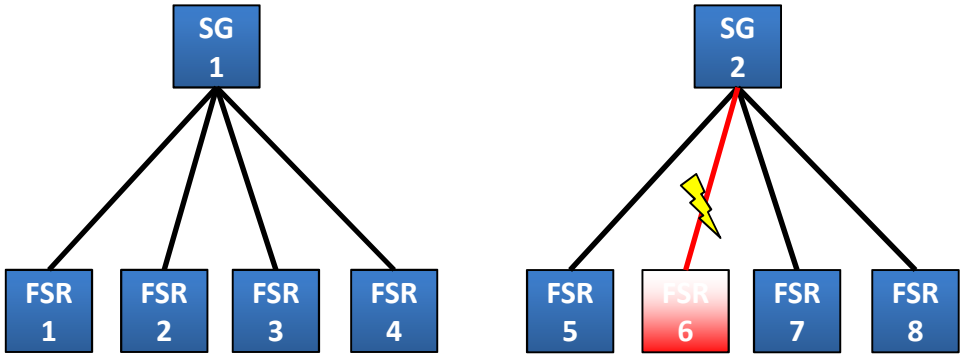
Research question: How to show that the SG to FSR breakdown is correct and complete?

Note: SG to FSR with a formal method has no semantic GAP

semcon KTH SP COMENTOR qamcom VOLVO FUSE

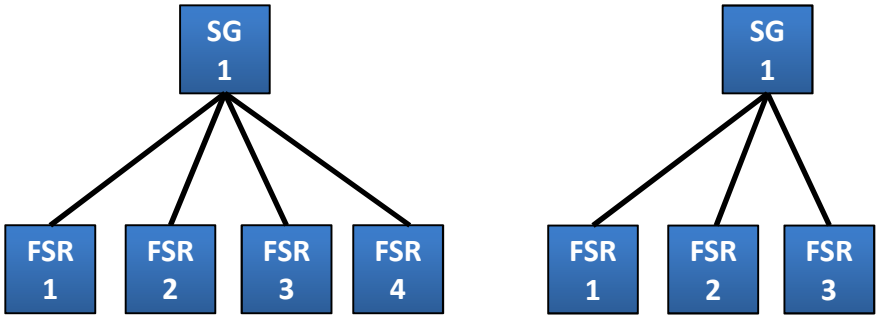
Correctness and Completeness

- Correctness is shown by a logical argumentation from one requirement level to another. The argumentation can be inductive or deductive.



Correct **Incorrect**

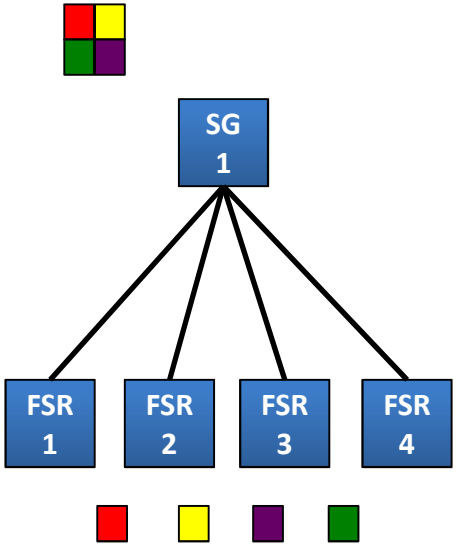
- Completeness is not an absolute measure and must reside in a context. A set of requirements can only be complete in relation to some defined domain, i.e. a set of defining properties.



Complete **Incomplete** (but correct)

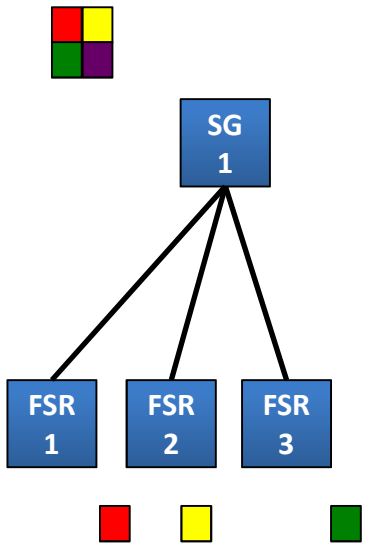
Domains and Defining properties

Defining properties (Domain)



Complete

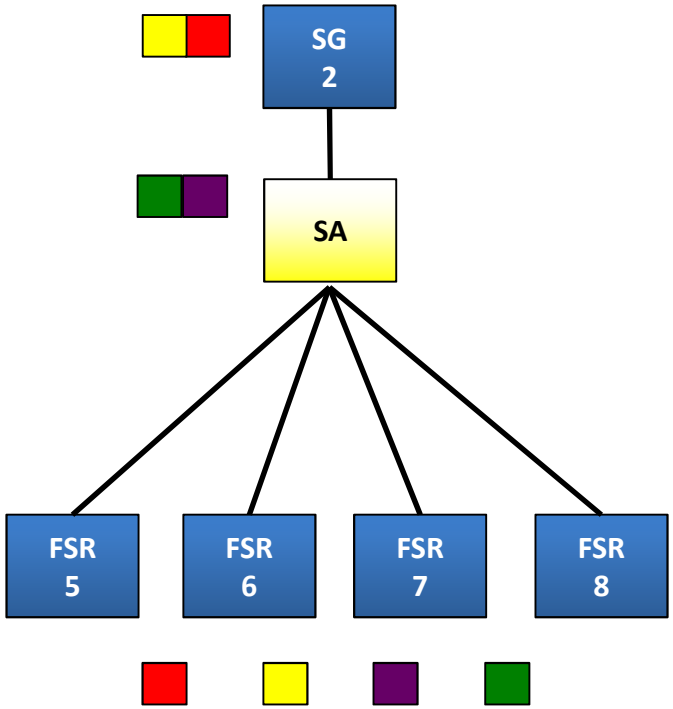
Does span the complete domain



Incomplete

Does <u>not</u> span the complete domain

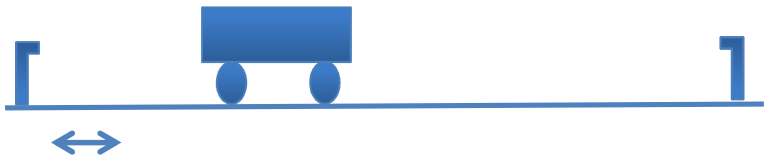
Defining properties and satisfaction arguments



Complete

Example: Shuttle

The problem



RS1.1: The shuttle shall not hit any end stop.

Correct ?
Complete ?

RS2.1: The shuttle shall have max speed < v_{max} (8 m/s).

RS2.2: The shuttle shall release the brakes when distance d_0 (5 m) from the end stop is passed.

The solution (work flow)

1. Specify satisfaction arguments (SA)
   - Assumptions
   - Domain knowledge
2. Verify correctness
   - Analysis
   - Update SA
3. Verify completeness
   - Defining properties
   - Update SA

Example: Shuttle

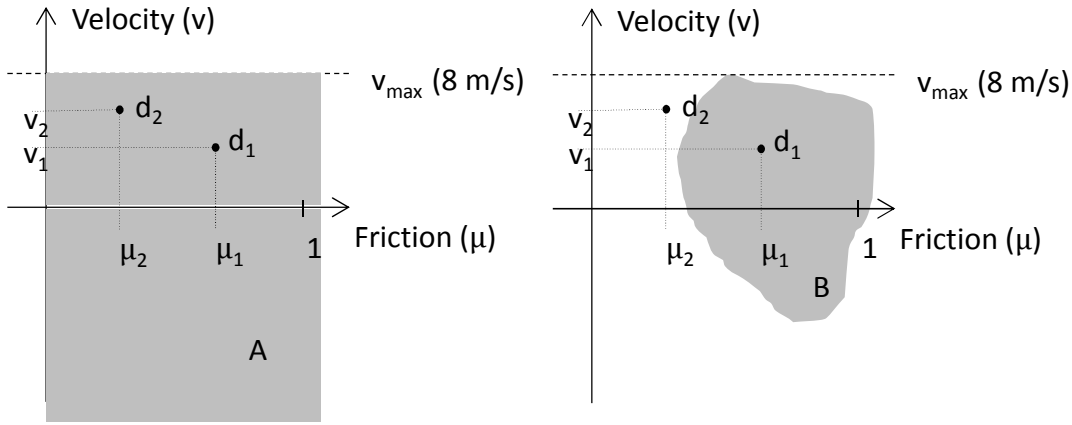
Original problem is address by requirements on
- Speed
- Position (stop distance)

Work…

Our selection of **defining properties** for this problem:
- Velocity
- Position (stop distance)
- Friction

More work…

Note: In this case SA are assumptions about indoor/outdoor operation, floor conditions, domain knowledge about the physical constraints and characteristics of the shuttle system etc.
The analysis part is based on the law of physics.



Conclusion:
The original requirement set is not complete w r t our choice of defining properties .

⊙ Stop distance (d)

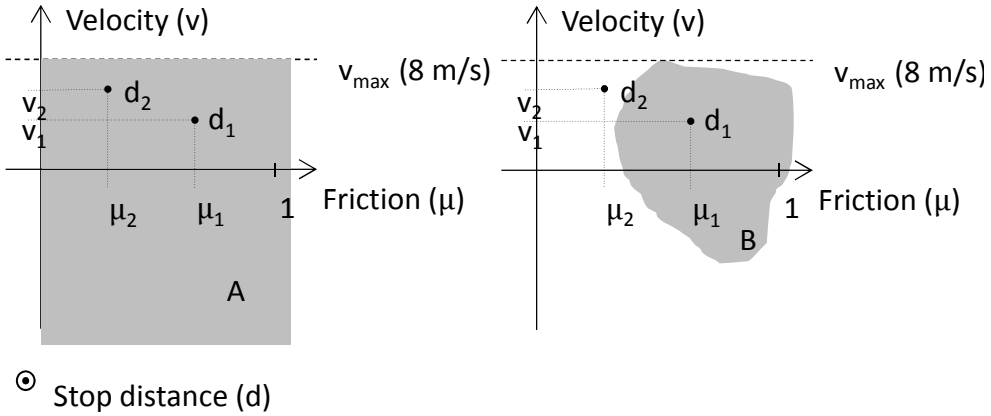
Example: Shuttle

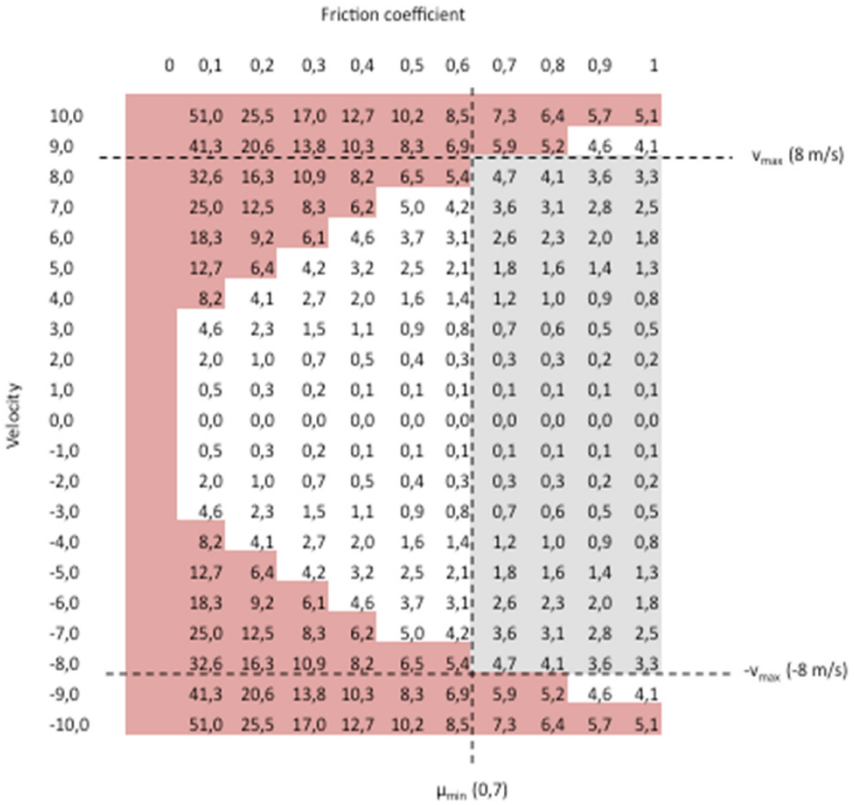
Updated requirement set

RS2.1: The shuttle shall have max speed ***|vmax| ≤ 8 m/s.***

RS2.2: The shuttle shall release the brakes when distance d0 (5 m) from the end stop is passed.

RS2.3: The shuttle shall have tires that ensure a friction coefficient μ ≥ 0,7 against the floor.





Conclusion:
Requirement set is complete (and correct) w r t our choice of defining properties => The high level requirement is always fulfilled.

Conclusions and Next step

- Conclusion
  - Satisfaction arguments is a known method for refinement, described in the literature. FUSE's idea is to use satisfaction arguments for refinement verification, i.e. argue for both correctness and completeness.
  - Completeness is shown w r t a set of defining properties, i.e. a domain. The domain gives the scope for stating completeness.
  - Formal method may seem better, but is in many cases not feasible due to legacy and reuse. Refinement verification with satisfaction arguments can handle this.

- Next step
  - Formalize the method
  - Scale the method to bigger, more realistic systems
  - Validate the method in a tool
  - Investigate interaction with other methods, e.g. Contract theory

Notes and references

[1] Satisfaction arguments are mentioned in the literature by for example J. Dick (Telelogic UK) in the article “Rich Traceability” and by K. Attwood, T. Kelly and J. McDermid in the paper: “The use of Satisfaction Arguments for Traceability in Requirements Reuse for System Families: Position Paper”.