McMaster University

Inspiring Innovation and Discovery

Formal Methods for Certification of Control Software and Potential for their Successful Application in Autonomous (Robot) Control

Alan Wassyng

McMaster Centre for Software Certification

McSCert

The Quest for Autonomy

- Autonomy has been, and continues to be a continuous quest




time



McSCert

Why the increased focus?

- What we want to make autonomous is changing!
- And, the “completeness” of the autonomy is changing
- Robot fetches clothes puts them in the washer then transfers to dryer then puts them away

McSCert

Challenges for Auto Laundry

- May have to traverse stairs while carrying a load
- Have to sort compatible items into a load
- Set appropriate washing program
- Sort which items can go in the dryer
- …

McSCert

Challenges for Auto Laundry

- May have to traverse stairs while carrying a load
  - Do we really need to?
- Have to sort compatible items into a load
  - Do we really need to?
- Set appropriate washing program
- Sort which items can go in the dryer
- …

McSCert

Challenges for Auto Laundry

- May have to traverse stairs while carrying a load
  - Do we really need to? We have choices!
- Have to sort compatible items into a load
  - Do we really need to? We have choices!
- Set appropriate washing program
- Sort which items can go in the dryer
- …

McSCert

Questions re Auto Laundry

- Do we have to use ML?
- Would it help to use ML?
- Do we have to use adaptive techniques?
- Would it help to use adaptive techniques?
- What if we build infrastructure?

- So, maybe if we put sorted laundry into bins next to the washer dryer, then Auto Laundry takes over and does the rest – is that complete enough?

McSCert

What is the implication for autonomous vehicles

- The obvious one is to provide infrastructure that aids progress and safety (but will add cyber-security challenges)

McSCert

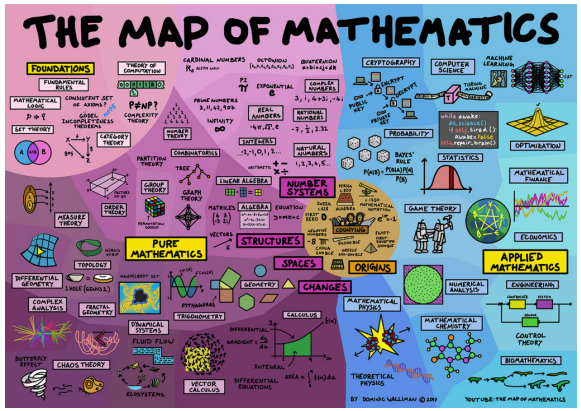
What is the implication for autonomous vehicles

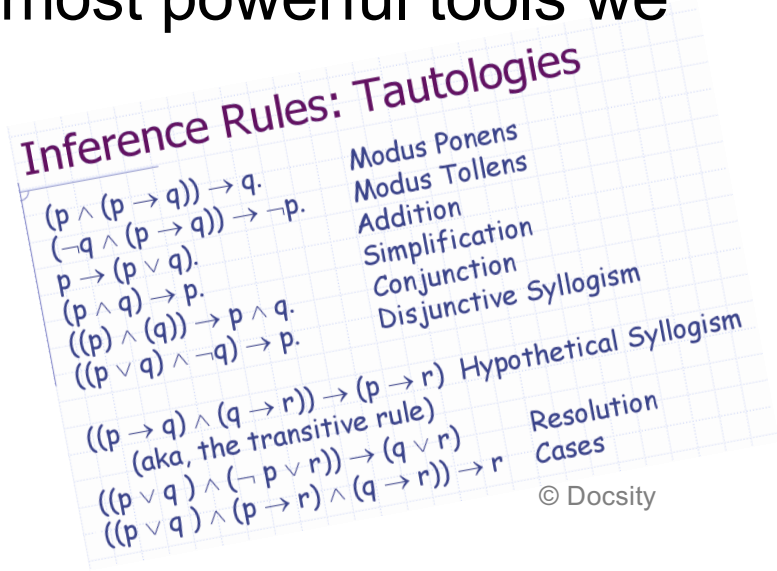
- The obvious one is to provide infrastructure that aids progress and safety (but will add cyber-security challenges)
- What happens when your autonomous vehicle has to go where there is no infrastructure? (In-complete)

McSCert

Why Formality? Really!

- Mathematics is one of the most powerful tools we have at our disposal





- Mathematics is precise – and sometimes ☺ unambiguous
  - The notation and language is unambiguous, but sometimes we have misunderstandings about the model it is built on

- *Use mathematics to guide us – not as the end goal!*

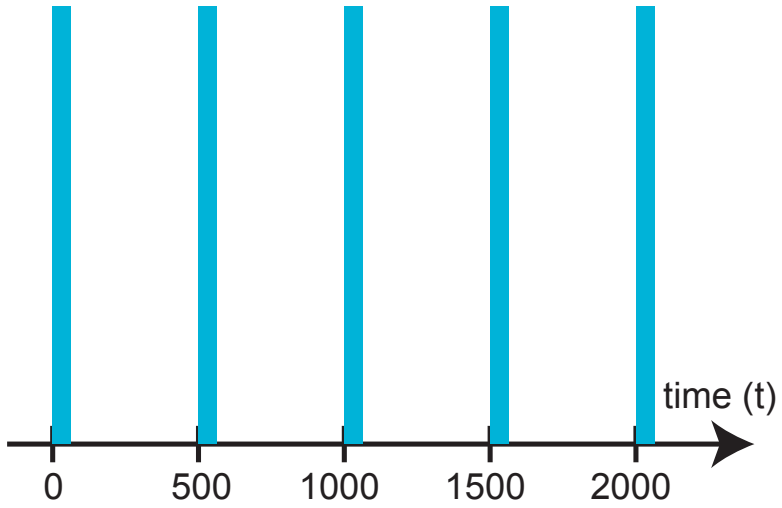
McSCert

Somethings to Keep in Mind

- When we use mathematics we can still be wrong
- When we use mathematics we can still be ambiguous
- When we use mathematics we can still be incomplete
- The really bad news – correct, non-ambiguous, complete mathematics may still be misunderstood
- And sometimes (quite often) we fool ourselves into thinking something is precisely defined when we use mathematics to arrive at numbers to rank something

Example 1

- t mod 500 ms used to describe a periodic event – transmitting a communications packet, for example



- What about with tolerances in the time?
- t mod (500ms±100ms) = 0 used to specify when to send a communications packet

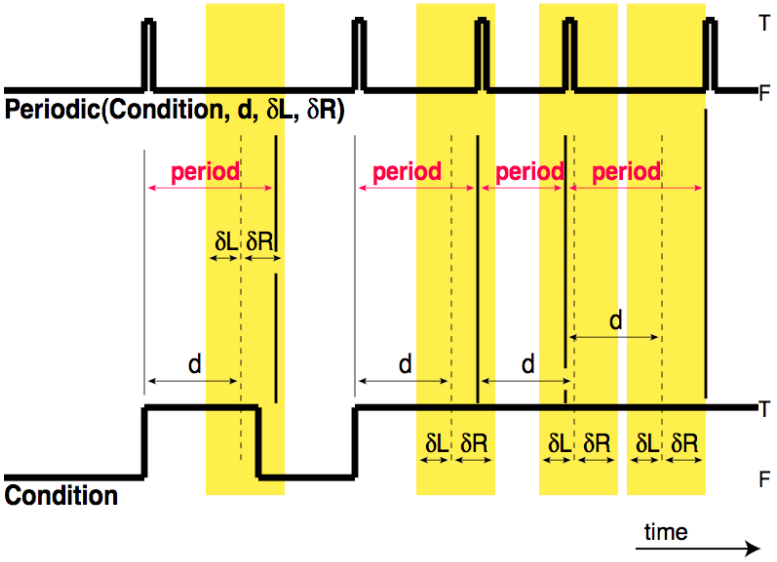
McSCert

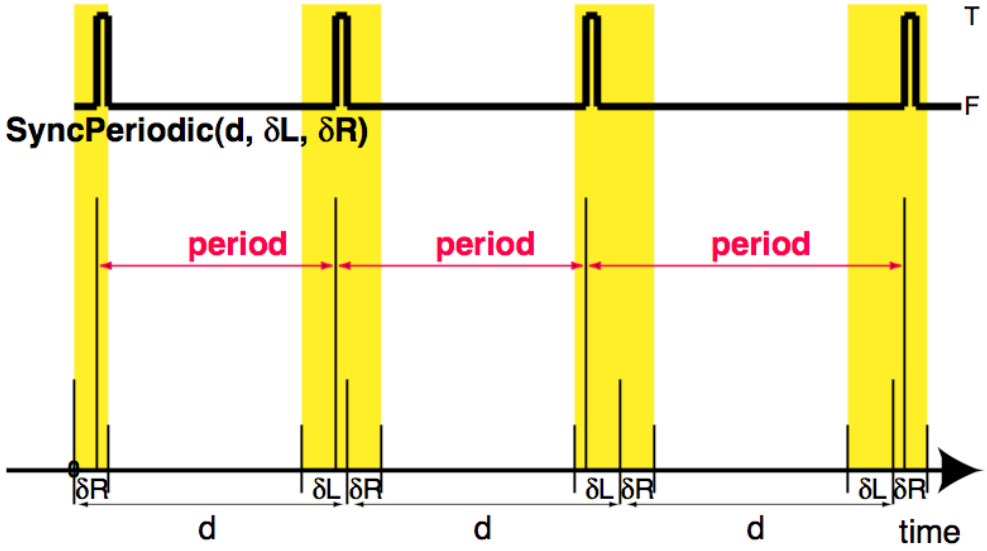
Example 1

- t mod (500ms±100ms) = 0 used to specify when to send a communications packet
  400-600, 800-1200, 1200-1800, 1600-2400, 2000-3000, …
  (complete overlap)

Example 1

- t mod (500ms±100ms) = 0 used to specify when to send a communications packet
  400-600, 800-1200, 1200-1800, 1600-2400, 2000-3000, …
  complete overlap
  - Even if we modify this to better specify the intent of the above spec – it would be wrong!
  - What we wanted was a periodic function but not one synchronized with an external clock





So This Is What We Needed

Periodic(Condition: bool, d: $\mathbb{R}^{>0}$, δL, δR: $\mathbb{R}^{\geq 0}$): bool
period(Periodic$_{-1}$: bool): [d-δL, d+δR]
previous_pulse_time(Condition: bool): $\mathbb{R}^{\geq 0}$
Initially:
period = any value in [0, δR]; previous_pulse_time$_{-1}$ = 0; Periodic$_{-1}$ = False

	period
Periodic$_{-1}$ = True	Any value in [d-δL, d+δR]
Periodic$_{-1}$ = False	No Change

			Periodic	**previous_pulse_time**
Condition = True	Condition$_{-1}$ = False		True	t_{now}
	Condition$_{-1}$ = True	$t_{now} \geq$ previous_pulse_time$_{-1}$ + period	True	t_{now}
		$t_{now} <$ previous_pulse_time$_{-1}$ + period	False	No Change
Condition = False			False	No Change

[Wassyng2005]

Which Is Very Different From

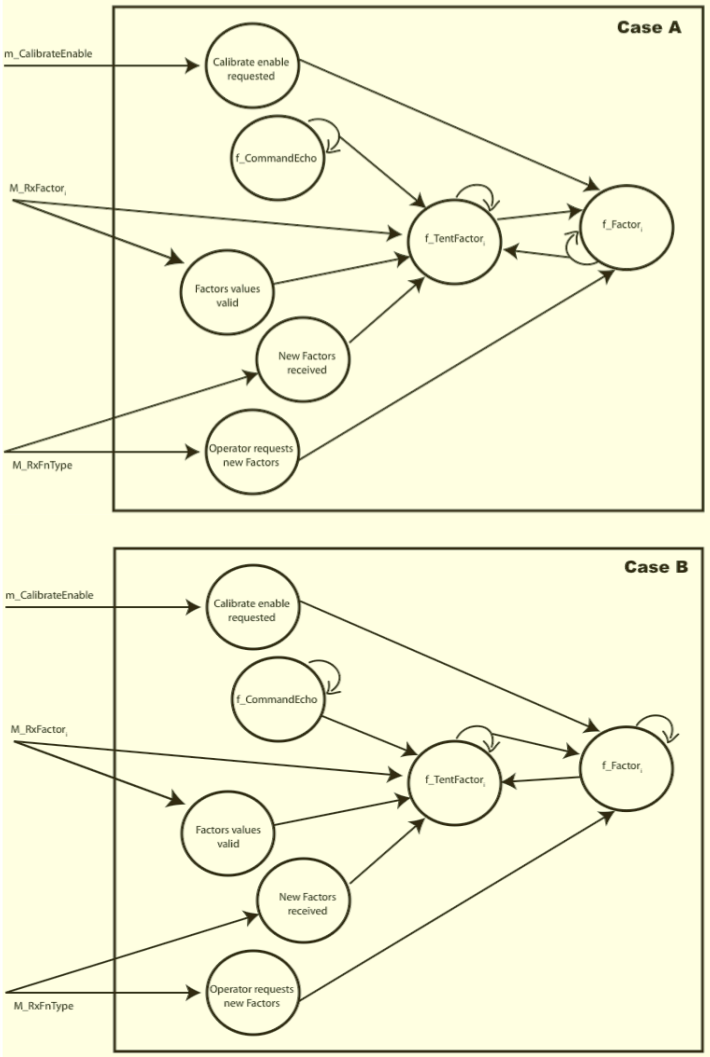
SyncPeriodic (d: $\mathbb{R}^{>0}$, δL, δR: $\mathbb{R}^{\geq 0}$): bool
n: $\mathbb{N}$
Δ: $\mathbb{R}$
Initially:
    n = 0
    Δ = any value in [0, δR]
    $SyncPeriodic_{-1}$ = False

	Δ	**n**
$SyncPeriodic_{-1}$ = True	Any value in [-δL, δR]	n+1
$SyncPeriodic_{-1}$ = False	No Change	No Change

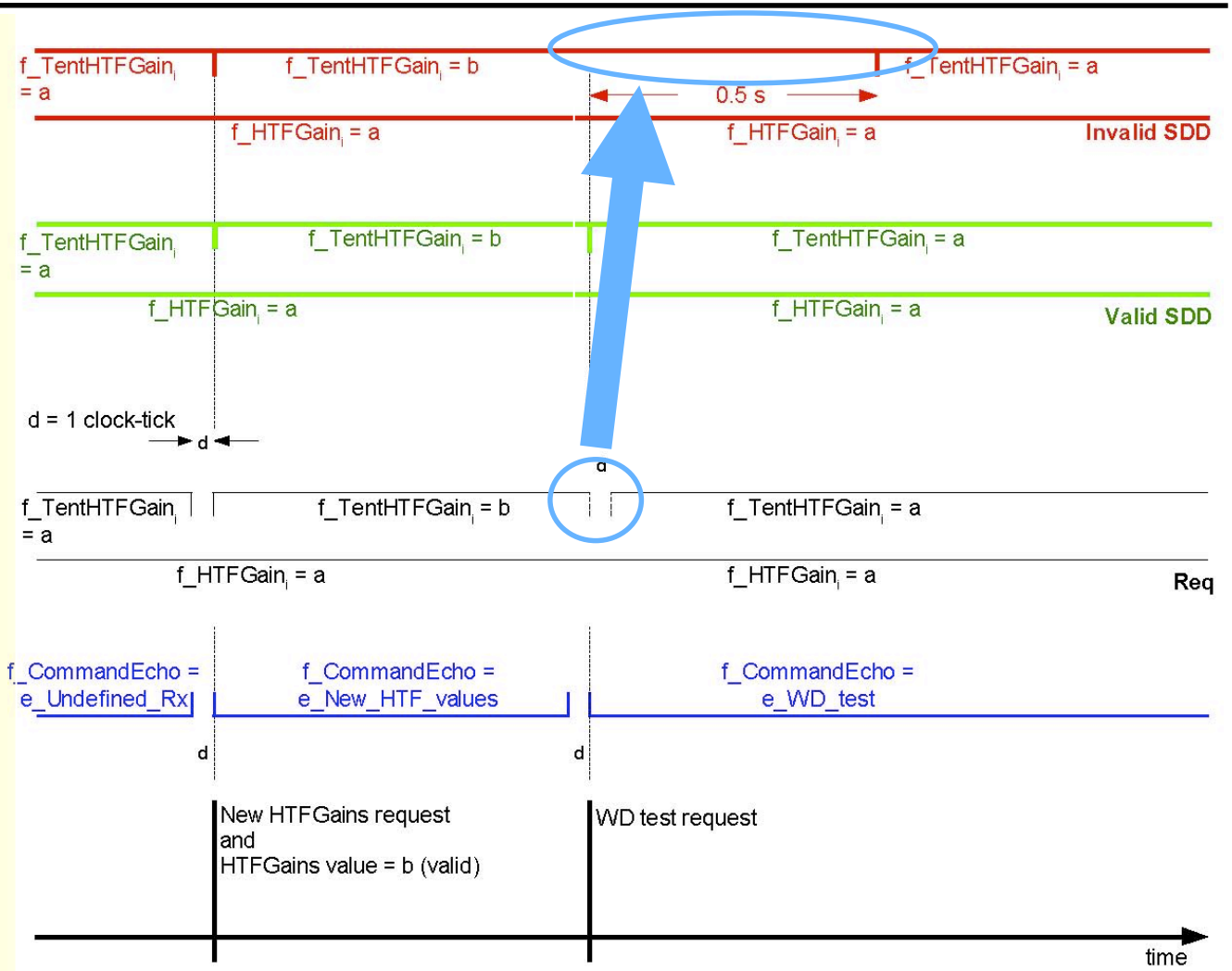
	SyncPeriodic
$t_{now} \geq n \cdot d + \Delta$	True
$t_{now} < n \cdot d + \Delta$	False

[Wassyng2005]

Example 2



Requirements changed from Case A to Case B



Requirements model is FSM with arbitrarily small clock tick. Design model is multiple FSMs with different clock ticks – and the designers did not understand the difference, so they simply implemented every transition in the requirements

McSCert

A Great Reason to Use Math

- The world is turning to model driven engineering in a big way – for good reasons
- To develop effective models we need mathematics
- This has become one of the great motivators for formal approaches
- There are also powerful methods and tools based on mathematics that help in various aspects of the development life-cycle – that are much more practical than we used to see not so long ago
- Building the mathematics into tools can make it possible for more people to take advantage of these powerful methods

McSCert

Practically Formal Methods & Tools

- Practically
  - Have to be usable on real world problems
  - Have to be usable by the people solving those real world problems
- Formal
  - Based on sound mathematics
  - Powerful methods that can be more capable and more efficient than ad hoc solutions
- Not necessarily 100% formal (Practical**ly** …)
  - I think the most important point of all

How Do We Build Safe Systems?

- Safety is an example here – we could ask the same about Security or Dependability
- Safety is a good property to discuss – because it presents many problems, and it is a primary goal of many critical systems
- DISCLAIMER
  - My apologies, but I am not an expert in robotics.
  - However, I believe the approaches we have developed and are working on are directly applicable to robotics
  - We have certainly used them and focused on safety-critical cyber physical systems

Definition(s) of Safety

NOT risk based

- Safety is the absence of accidents, where an accident is an event involving an unplanned and unacceptable loss [Leveson2012]
- Freedom from conditions that can cause death, injury, occupational illness, damage to or loss of equipment or property, or damage to the environment

[MIL-STD-882E]

risk based

- Freedom from unacceptable risk [IEC61508] & others
- Absence of unreasonable risk [ISO26262]
- 'safety' means the protection of people and the environment against radiation risks, and the safety of facilities and activities that give rise to radiation risks

[IAEA Safety Glossary]

Safety Integrity Levels

- SILs also differ from domain to domain [Joannou2014]

Industry sector	Standard	Integrity level or equivalent term	Factors used to determine integrity level
General	ISO/IEC 15026	Integrity level	Severity and likelihood
	ISO/IEC 61508	Safety integrity level	Severity and likelihood
Nuclear	IEC 61226	Category	Severity
Aviation	DO-178C	Software level	Severity
Defense	MIL-STD-882E	Software criticality index	Severity
Automotive	ISO 26262	Automotive safety integrity level	Severity and likelihood
Medical devices	IEC 62304	Software safety class	Severity, possibly mitigated through hardware risk control measures
Security	ISO/IEC 15408	Evaluation assurance level	Severity and likelihood
	ISO/IEC 27034	Level of trust	Severity and likelihood

- *We are trying to use formal methods to solve problems where the problems may not be uniformly defined*

Motivation for Formal Methods for Certification

- What follows is a walk through planning for developing safe systems
- Certification should follow naturally
- Eventually ☺ we will get to why formal methods help

McSCert

The Heart of a Safety Argument

Claim: the requirements are correct & complete (includes all "safety requirements")

Claim: the implementation complies with its requirements (within some stated tolerance)

If we could do everything perfectly, this may be enough (Ha!)

The Heart of a Safety Argument

Claim: the requirements are correct & complete (includes all "safety requirements")

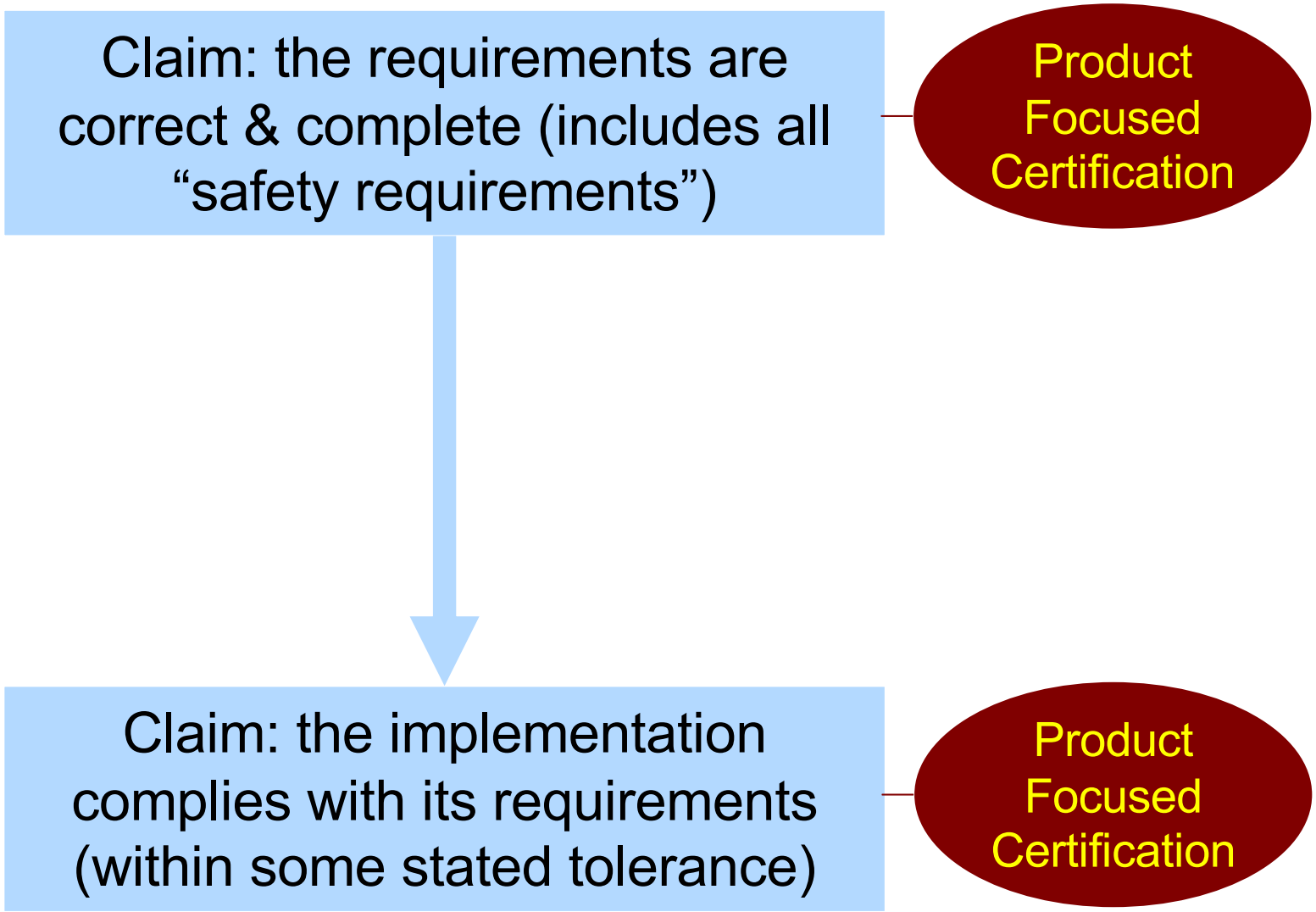
Claim: the implementation complies with its requirements (within some stated tolerance)

If we could do everything perfectly, this may be enough (Ha!)

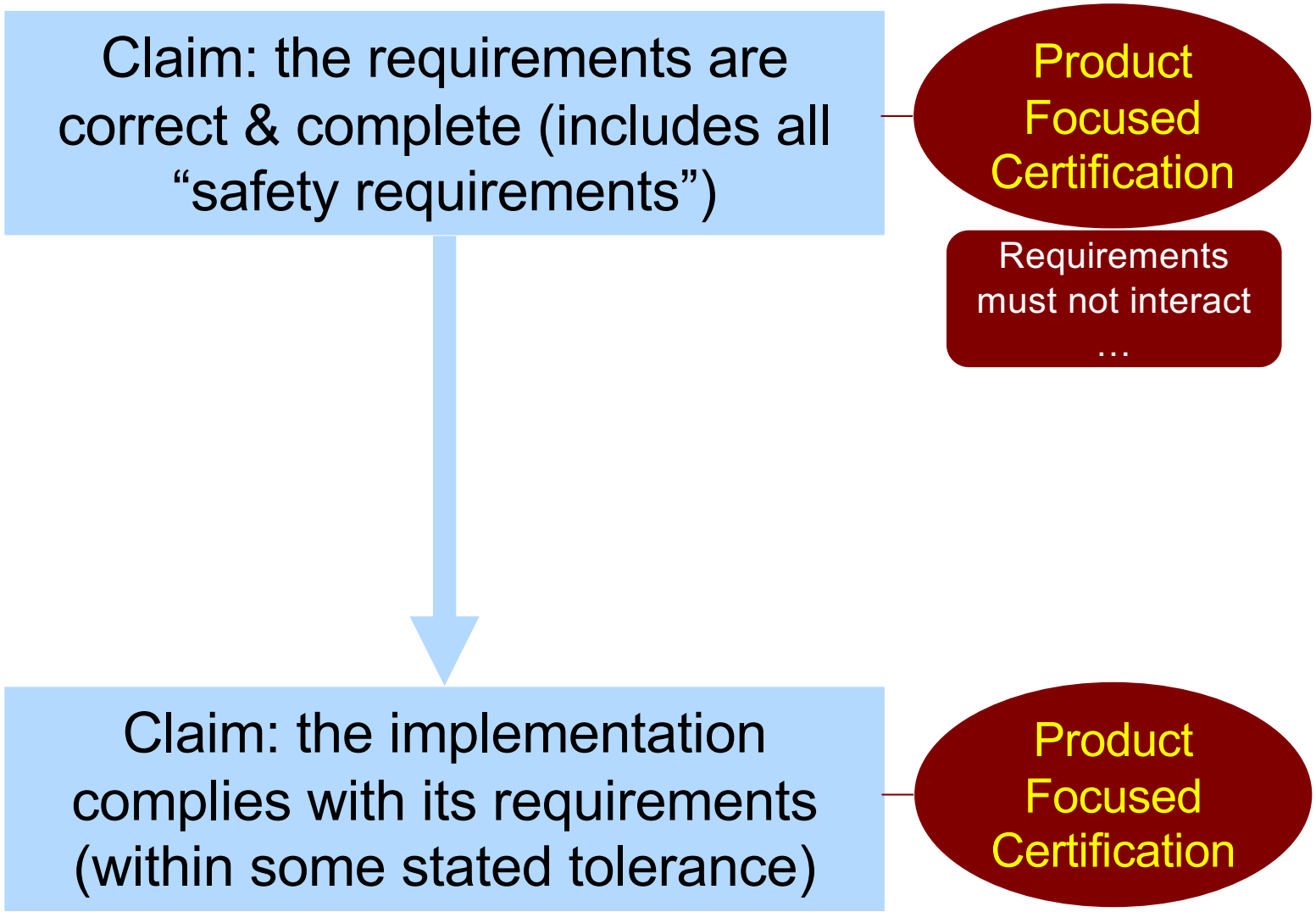
NOTE: This is not close to being equivalent to what some advocate: *Check that all hazards are eliminated or mitigated in the implementation*.

1. We need to check interactions between requirements (safety & functional)
2. I feel very strongly that we need to confirm the system delivers what it is expected to deliver – or else users find workarounds that adversely affect the safety we have built into the system

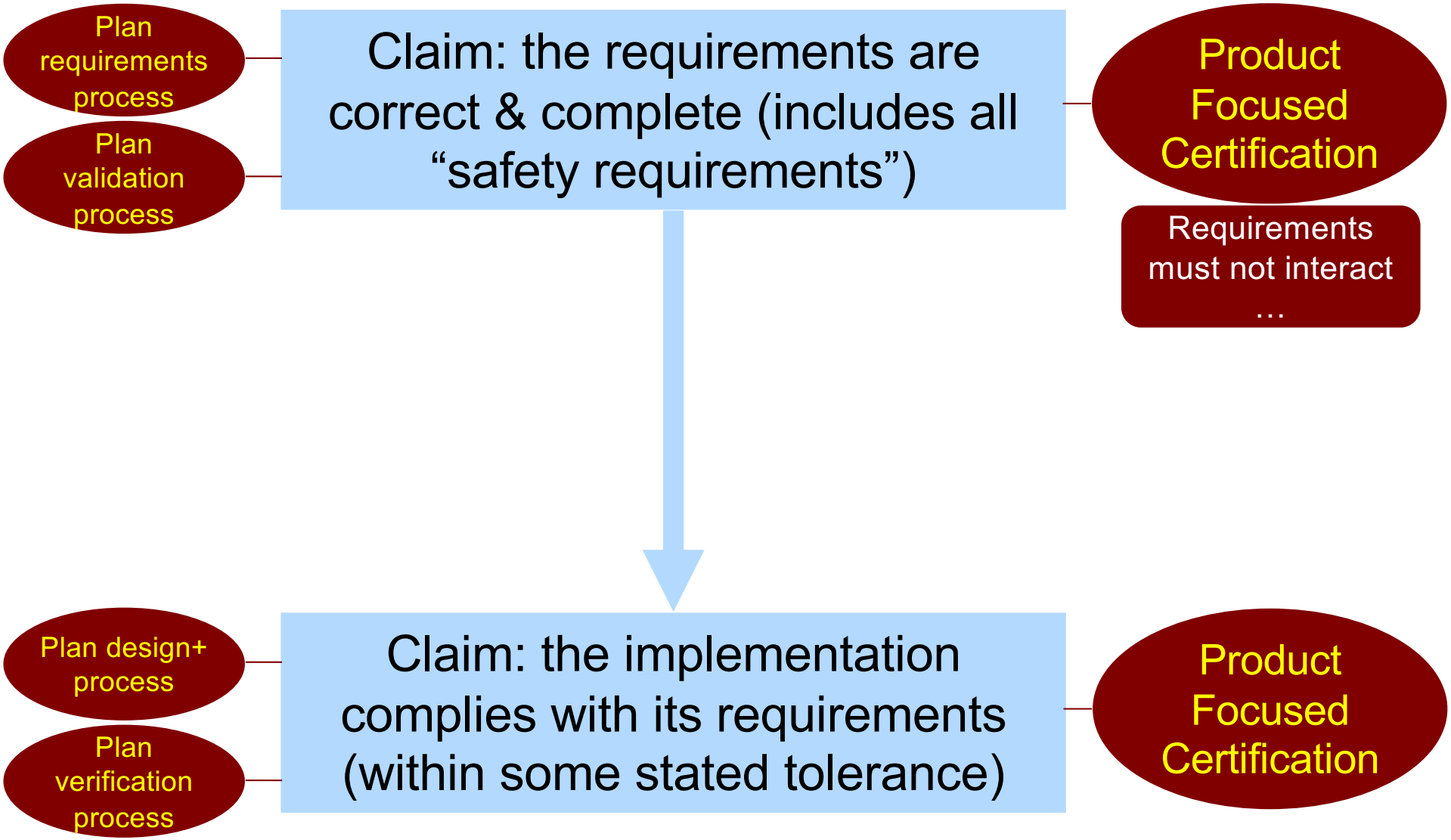
The Heart of a Safety Argument



The Heart of a Safety Argument

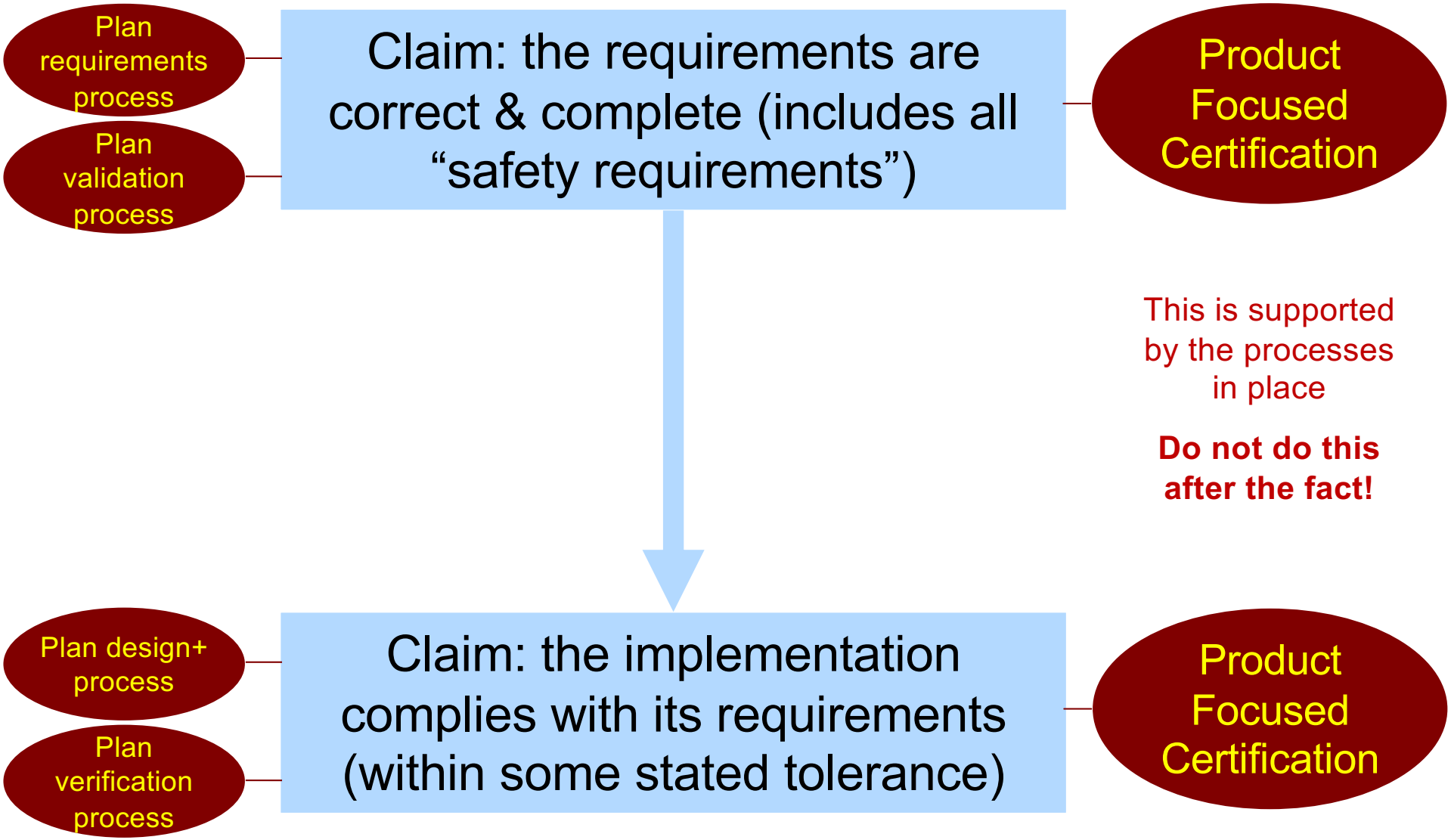


The Heart of a Safety Argument



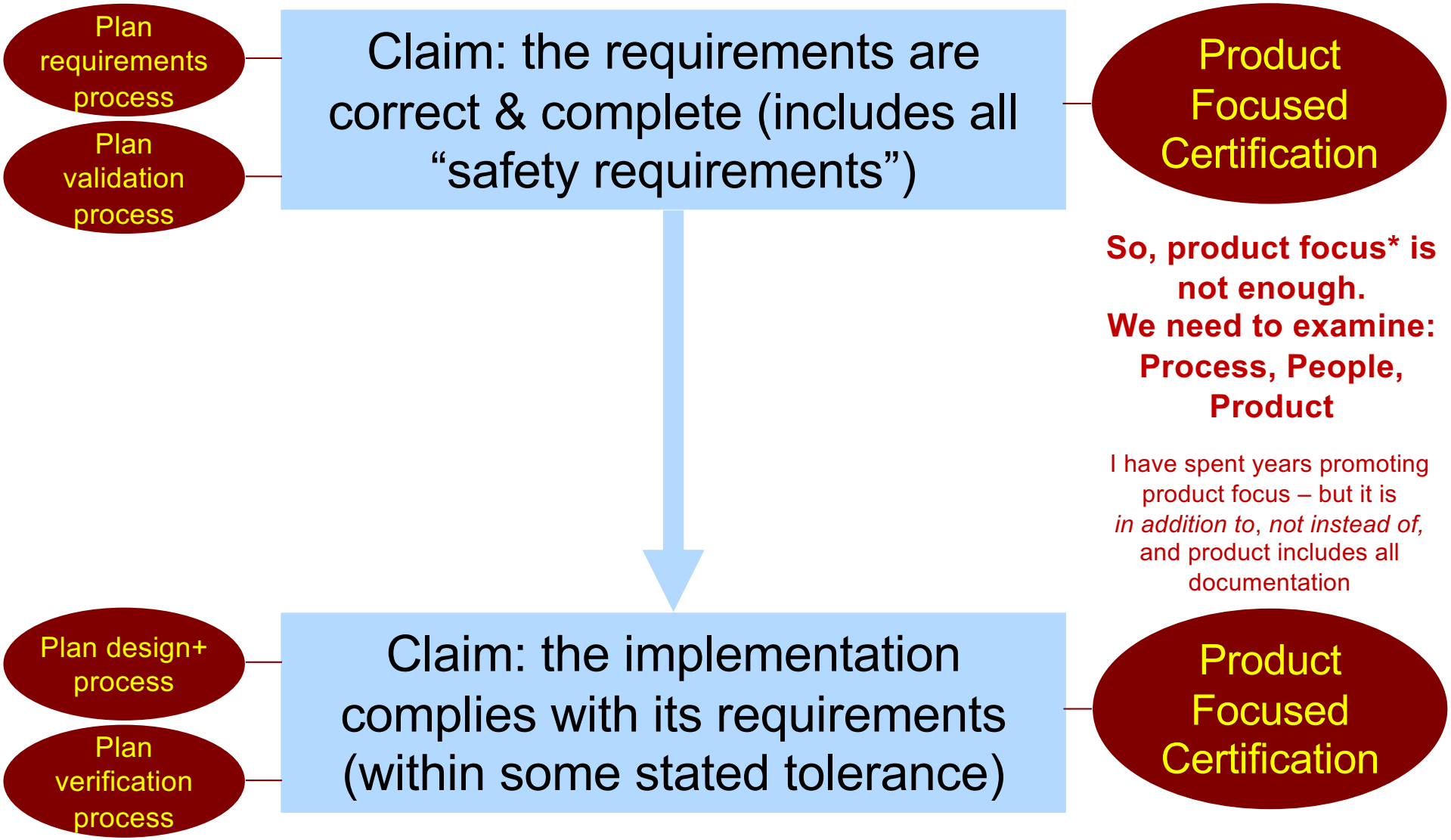
And we want to plan that the system will be safe **before** we start developing it

The Heart of a Safety Argument



And we want to plan that the system will be safe **before** we start developing it

The Heart of a Safety Argument



The Heart of a Safety Argument

Plan requirements process

Plan validation process

Claim: the requirements are correct & complete (includes all "safety requirements")

Product Focused Certification

How can we build confidence in the certification?

So, product focus* is not enough.
We need to examine: Process, People, Product

I have spent years promoting product focus – but it is *in addition to*, *not instead of,* and product includes all documentation

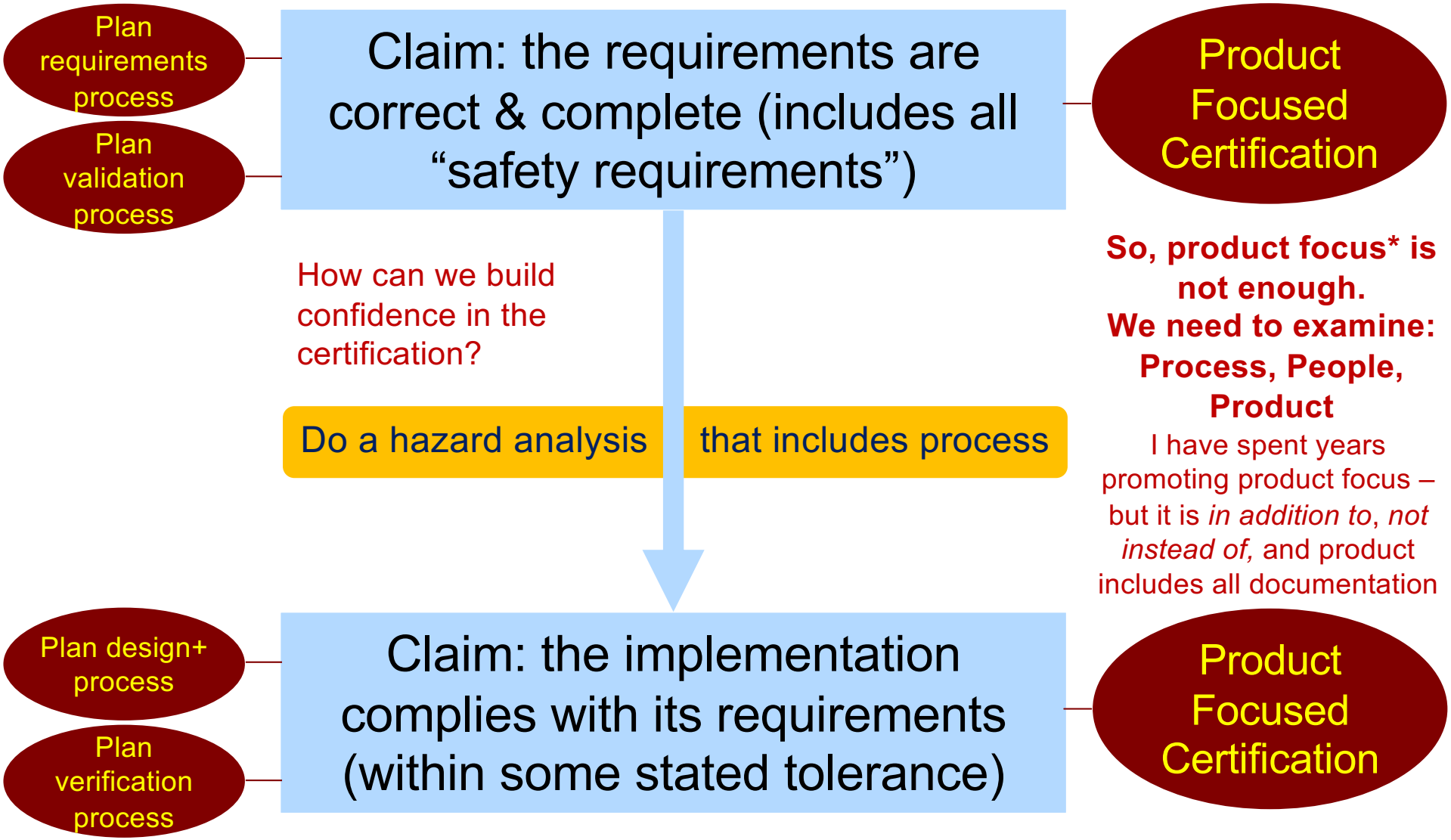
Plan design+ process

Plan verification process

Claim: the implementation complies with its requirements (within some stated tolerance)

Product Focused Certification

The Heart of a Safety Argument



So, product focus* is not enough.
We need to examine: Process, People, Product

I have spent years promoting product focus – but it is *in addition to*, *not instead of,* and product includes all documentation

The Heart of a Safety Argument

McSCert

Plan requirements process

Plan validation process

Claim: the requirements are correct & complete (includes all "safety requirements")

Product Focused Certification

How can we build confidence in the certification?

What will the hazard analysis show?

Do a hazard analysis that includes process

Single points of failure ... Major risks …

Plan design+ process

Plan verification process

Claim: the implementation complies with its requirements (within some stated tolerance)

Product Focused Certification

So, product focus* is not enough.
We need to examine: Process, People, Product

I have spent years promoting product focus – but it is *in addition to*, *not instead of,* and product includes all documentation

The Heart of a Safety Argument

McSCert

* Plan requirements process

Plan validation process

Expert opinion?

Claim: the requirements are correct & complete (includes all "safety requirements")

Product Focused Certification

How can we build confidence in the certification?

What will the hazard analysis show?

Do a hazard analysis that includes process

Single points of failure … Major risks …

So, product focus* is not enough. We need to examine: Process, People, Product

I have spent years promoting product focus – but it is *in addition to*, *not instead of,* and product includes all documentation

*Process followed?

Plan design+ process

Plan verification process

Testing only?

Claim: the implementation complies with its requirements (within some stated tolerance)

Product Focused Certification

The Heart of a Safety Argument

McSCert

* Plan requirements process

Plan validation process

Expert opinion?

Add more experts?

Claim: the requirements are correct & complete (includes all "safety requirements")

Product Focused Certification

How can we build confidence in the certification?

What will the hazard analysis show?

Do a hazard analysis that includes process

Single points of failure … Major risks …

So, product focus* is not enough.
We need to examine: Process, People, Product

I have spent years promoting product focus – but it is *in addition to*, *not instead of,* and product includes all documentation

*Process followed?

Plan design+ process

Plan verification process

Testing only?

Claim: the implementation complies with its requirements (within some stated tolerance)

Product Focused Certification

Add mathematical verification? (or correct by construction with checks)

yes – please ☺

The Heart of a Safety Argument

McSCert

* Plan requirements process

Plan validation process

Claim: the requirements are correct & complete (includes all "safety requirements")

Product Focused Certification

Expert opinion?

Add more experts?

Add mathematical analysis?
Or simulation?

How can we build confidence in the certification?

What will the hazard analysis show?

Do a hazard analysis that includes process

Single points of failure … Major risks …

So, product focus* is not enough.
We need to examine: Process, People, Product

I have spent years promoting product focus – but it is *in addition to*, *not instead of*, and product includes all documentation

*Process followed?

Plan design+ process

Plan verification process

Claim: the implementation complies with its requirements (within some stated tolerance)

Product Focused Certification

Testing only?

Add mathematical verification? (or correct by construction with checks)

yes – please ☺

McSCert

Conclusions

(not the conclusion of the talk – sorry)

- If we want to perform mathematical verifications as well as testing – we need formal requirements
  - A huge task for some systems
- If we want mathematical analysis or simulations of our requirements – we need formal requirements
  - There are different notations for formality
  - And, maybe not all requirements need to be formal
    - Because maybe we can separate off a “protective system”
- So – that’s how practically formal enters the picture

Two Observations

1

- Validation
  - Most important result we should work on!
  - My impression – done very poorly most of the time – because it is so difficult
  - (Look at what John Rushby found re aviation accidents attributed to software ☺)

Claim: the requirements are correct & complete (includes all "safety requirements")

- (Math) Verification
  - Lots of good work on effective practical verification methods – and people talking at this school have good advice for you!

Claim: the implementation complies with its requirements (within some stated tolerance)

2

- Physically and logically separating control and safety is incredibly powerful – the "protective system" can be much simpler than the (very) complex control system
  - Example: Darlington NGS Shutdown System – each of the two is about 35K LOC, control system about 1M LOC
  - We may be able to produce formal requirements, etc

McSCert

Model Driven Engineering

- MDE is already changing how we develop software intensive systems
- It will increasingly affect certification of those systems
- It promises many benefits – correctness by construction for a start (checking is much easier than verifying development steps)
- We will need to certify the tools
- The tool chain will include certification specific tools
- And, at the moment, validation is still a huge problem
  - SMT solvers seem to hold out more hope in this area than model checking or SAT solvers

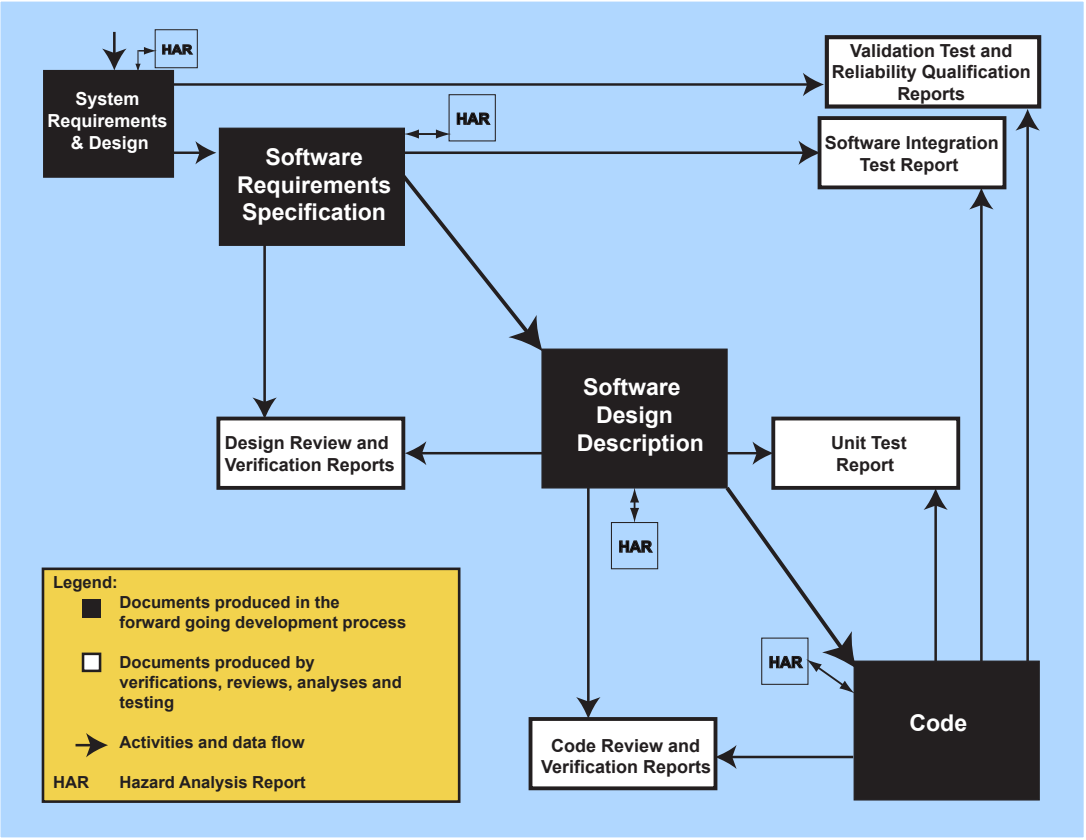
McSCert

Now Let's Focus on Certification

- What did we do before we used "modern" assurance cases?
- Why have I had a problem with GSN-like assurance cases?
- Why I think GSN-like assurance cases still work - but still think we can do much better
- Why will practically formal approaches to assurance cases improve them – a lot?

Why We (think we) Need Assurance Cases

- One way of convincing a regulator that the system is safe



So – regulators:

- Pre-approved approximately 20 process documents
- Audited development to be in compliance with processes
- Audited output of the processes
- Witnessed in-plant testing

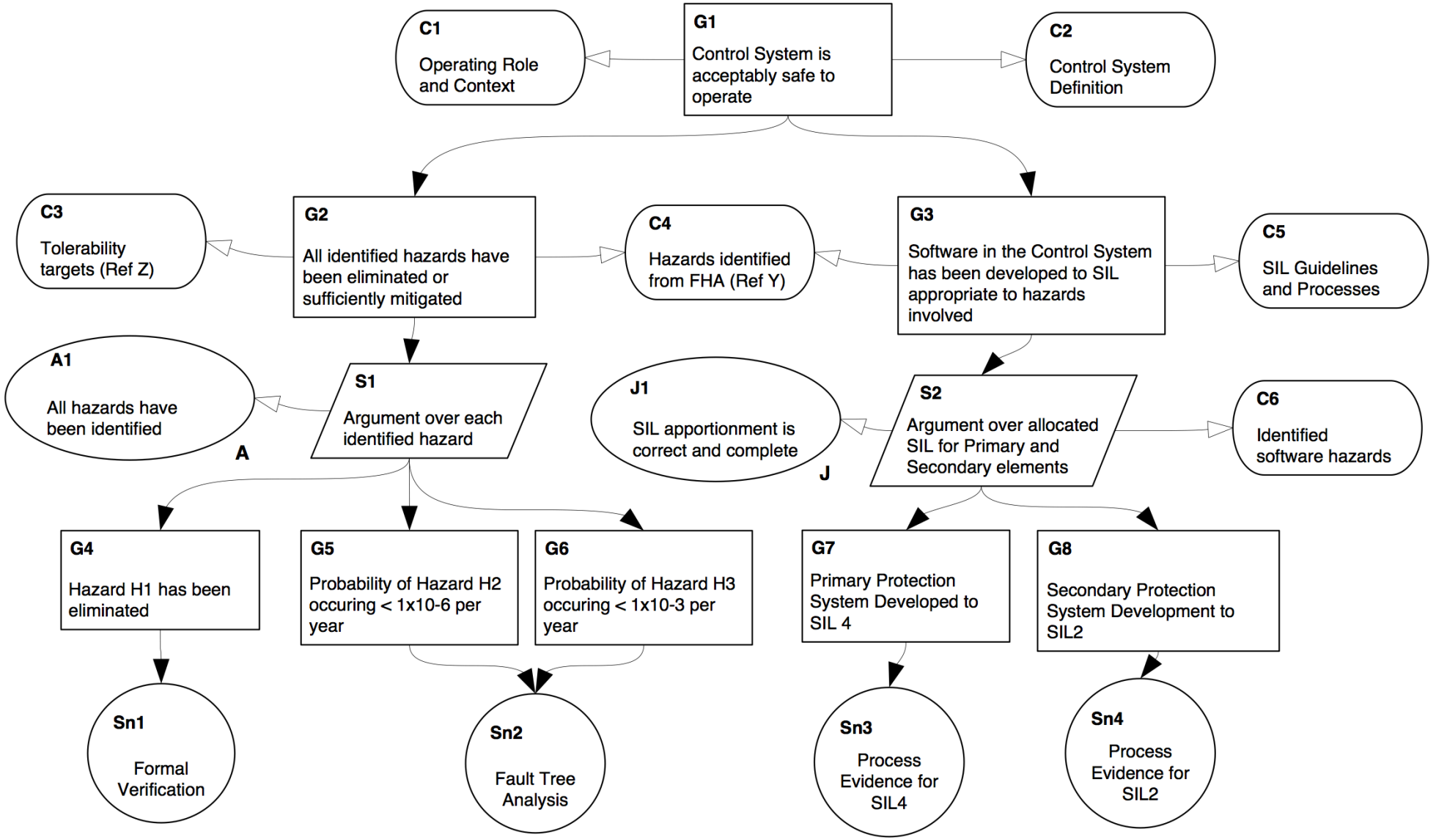
- And then give the regulators all the documents you have created
  - but the regulator then has to *discover* why we thought the system is safe

McSCert

Assurance (Safety) Cases

- The promise
  - Present an explicit argument grounded in (product, process, people) evidence that demonstrates the product is safe!
- Common practice
  - Goal Structuring Notation (GSN) is the most popular notation for modern assurance cases [Kelly1998]
  - At its core it presents a decomposition of a goal (claim) into sub-goals (sub-claims) and eventually evidence (called solutions in GSN)

GSN (2011)



GSN COMMUNITY STANDARD VERSION 1 - 2011 [GSN2011]

McSCert

GSN Benefits

- Benefits
  - I think GSN and similar notations like CAE (Claims Arguments and Evidence, developed by Adelard) have had a very positive effect in producing safe(r) systems
    - It is appealingly intuitive – I have seen practicing engineers (and students) pick-up the basic idea in a GSN assurance case as fast as you would ever want them to … sometimes faster
    - Its primary success (I think) is to get people to look critically at the "argument" and ask important and usually overlooked questions – and this is invaluable
    - It has motivated many more developers and certifiers to consider seriously what we need to demonstrate that a system is safe (or secure or dependable etc)

McSCert

GSN Shortcomings

- A personal view
- Cases tend to have an ad hoc structure dictated by experience & preference
  - Patterns help but they are not the “solution”
- Cross-cutting concerns abound
- There is an element of confirmation bias
- Provides a false sense of confidence (no matter how we “measure” confidence) since we think our reasoning is rigorous (most never claim formal) – but it is not
- Safety impact analysis is difficult to impossible
  - In general, effective traceability is tough

[Wassyng2011]

McSCert

Formal Approaches

- There are a variety of formal approaches to assurance (safety) cases, and I think that all of the ones I know about involve GSN …
- Work done by Ewen Denney involves not only safety case construction, but also formal approaches to the evidence – see https://ti.arc.nasa.gov/profile/edenney/
- I believe you will already have been told about SACM 2.0 [SACM2.0]
- Since one of my important goals is to develop assurance cases that facilitate incremental safety assurance, I feel that these approaches are not what I want in the future – even though they represent excellent work
  - They do not overcome my objections to the lack of a safety argument

Basis for a New Approach

- Last year we published an early version of the framework in MoDELS [Diskin2018]

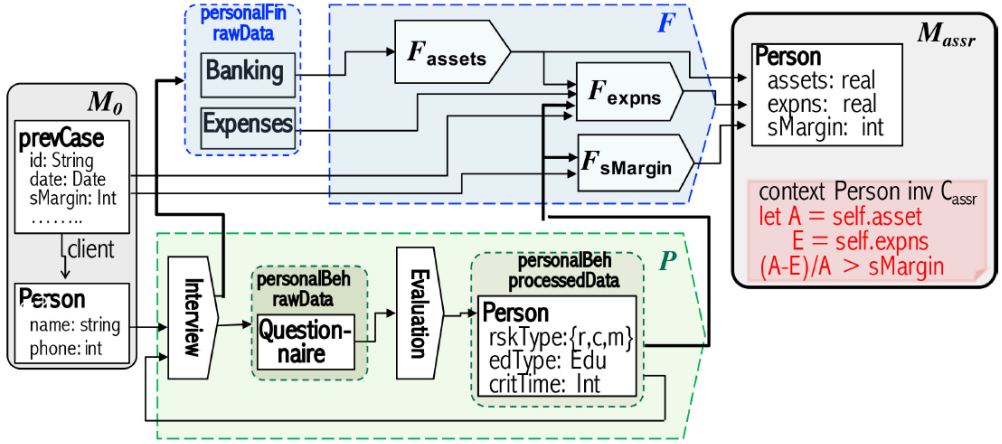
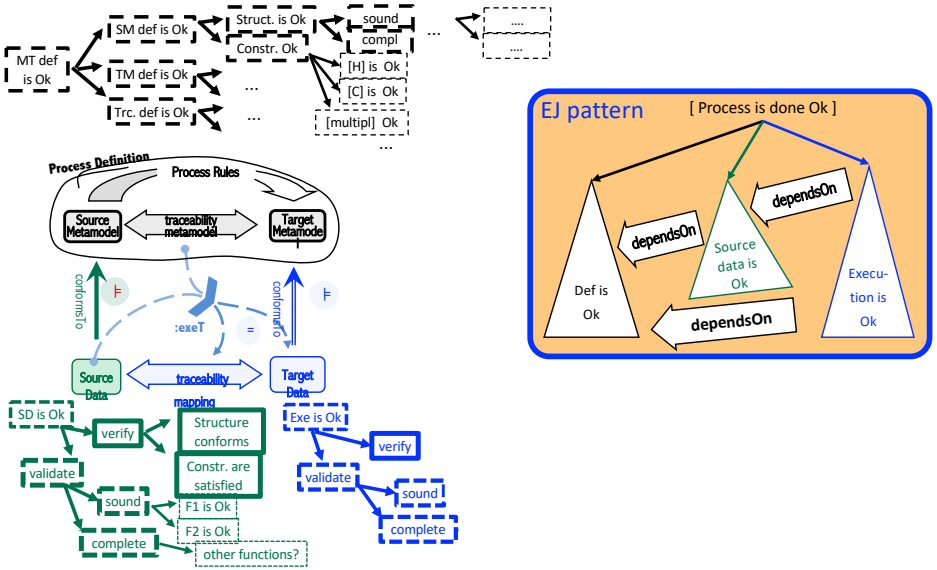


Figure 3: The process in detail



These “assurance steps” are suggested by the mathematical structure - no longer ad hoc.

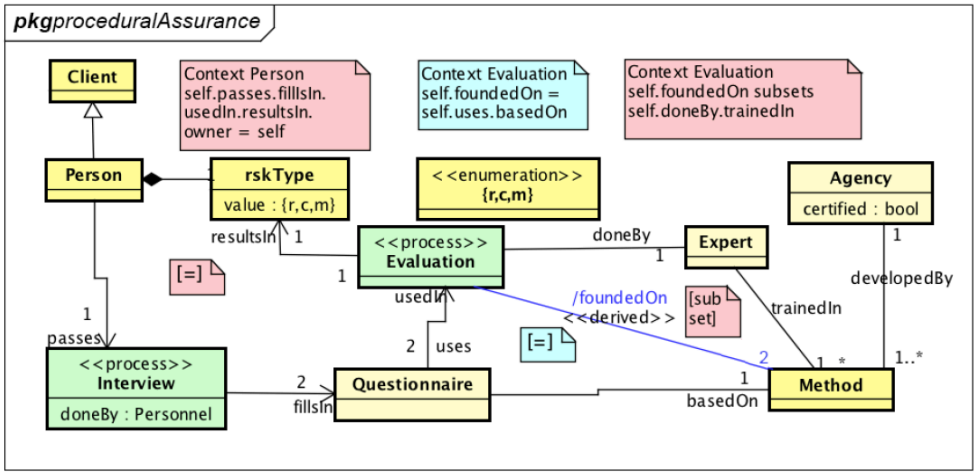
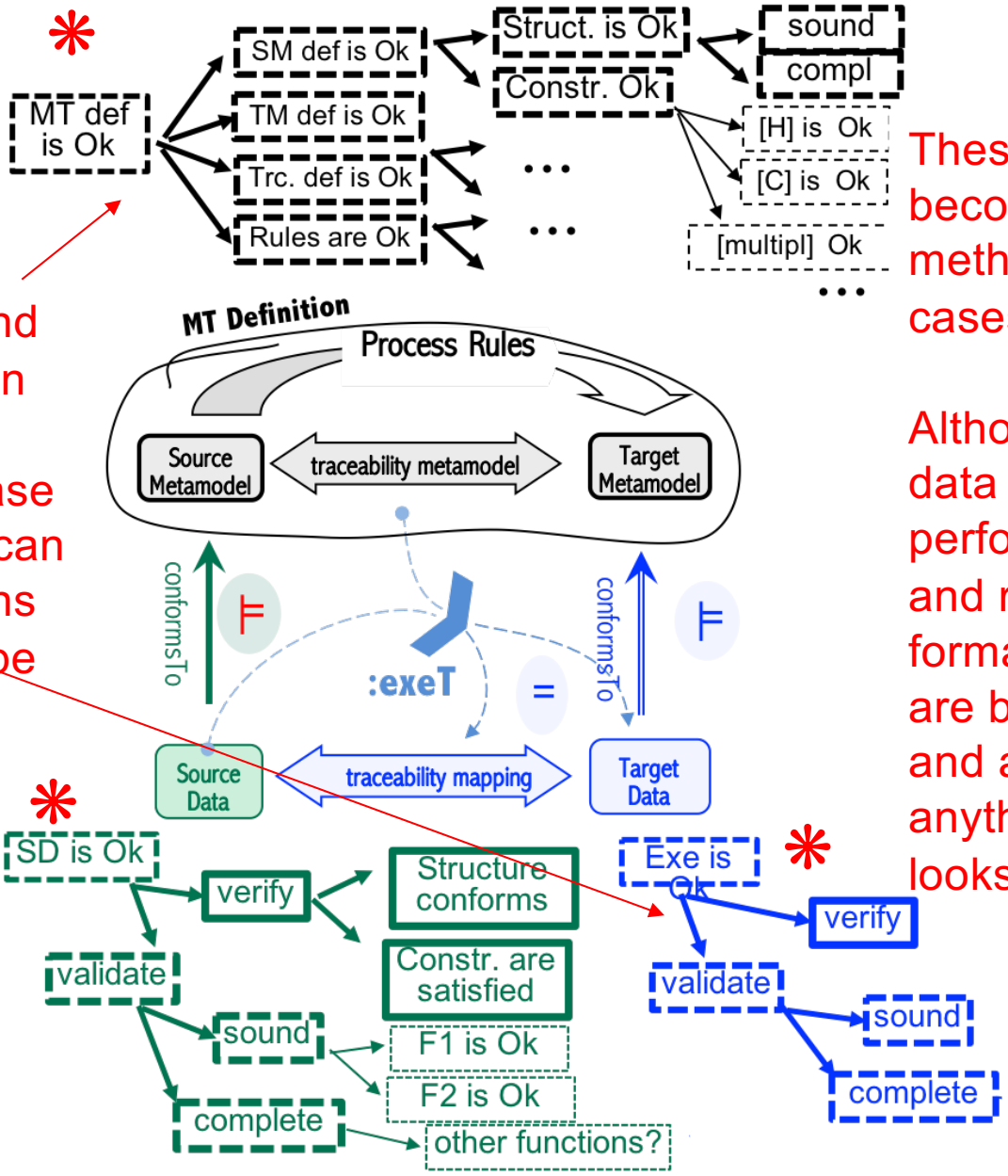


Figure 4: Workflow and metamodels together

Basis for a New Approach

In order to "assure" the refinement and transformation steps in the assurance case process, we can identify actions that have to be performed



These ✳ "assurance steps" become the basis of our new method for constructing assurance cases.

Although we cannot model the data flow in complete detail, nor perform the model transformations and refinements completely formally, these assurance steps are based on formal reasoning and are no longer ad hoc where anything is allowed as long as it looks reasonable.

Practically formal!

McSCert

Next Talk

- Will provide more detail of how we started with (GSN-like) Assurance Case Templates [Wassyng2016] – and have moved into this new transformation based formal approach

McSCert

Takeaways

- Formality must not be an end in itself
- Sometimes “formal” is not quite possible
  - But the approach based on a (hypothetical) formal method can be much better than ad hoc approaches we may be tempted to use
- There are powerful verification techniques that are scalable – use them where possible
- Explore ways of doing better validation
  - Formal requirements help
    - I really like tabular expressions for this (did not discuss them)
- We have to develop sound approaches to incremental safety assurance – rigorous/formal I think is necessary by the nature of the problem

McSCert

References

[Diskin2018] Zinovy Diskin, Tom Maibaum, Alan Wassyng, Stephen Wynn-Williams, Mark Lawford, "Assurance via model transformations and their hierarchical refinement," MoDELS 2018: 426-436.

[MIL-STD-882E] DoD, "MIL-STD-882E - DEPARTMENT OF DEFENSE STANDARD PRACTICE – SYSTEM SAFETY." 2012.

[GSN2011] GSN Community, GSN Community Standard, Std., Rev. Ver. 1, 2011. [Online]. Available: http://www.goalstructuringnotation.info/documents/GSN Standard.pdf

[IAEA Safety Glossary] International Atomic Energy Agency, Ed., IAEA safety glossary: terminology used in nuclear safety and radiation protection, 2007 ed. Vienna, Austria: International Atomic Energy Agency, 2007.

[IEC 61508] IEC, "IEC 61508-4 - Functional safety of electrical/electronic/programmable electronic safety related systems - Part 4: Definitions and abbreviations." 2010.

[ISO 26262] ISO, "ISO 26262-1 - Road vehicles - Functional safety - Part 1:Vocabulary." 2011.

[Kelly1998] T. Kelly, "Arguing safety – a systematic approach to managing safety cases," Ph.D. dissertation, University of York, September 1998.

[Leveson2012] N. G. Leveson, Engineering a Safer World: Systems Thinking Applied to Safety. The MIT Press, 2012.

[SACM2.0] Obtainable from: https://www.omg.org/spec/SACM/About-SACM/

[Wassyng2005] A. Wassyng, M. Lawford, & X. Hu. "Timing tolerances in safety-critical software." In International Symposium on Formal Methods (pp. 157-172). Springer, 2005.

[Wassyng2011] A. Wassyng, T. Maibaum, M. Lawford, and H. Bherer, "Software certification: Is there a case against safety cases?" in Foundations of Computer Software. Modeling, Development, and Verification of Adaptive Systems, LNCS, R. Calinescu and E. Jackson, Eds. Springer, 2011, vol. 6662, pp. 206–227.

[Wassyng2016] A. Wassyng, P. Joannou, M. Lawford, T. Maibaum, N.K. Singh, "New Standards for Trustworthy Cyber-Physical Systems." A. Romanovsky, F. Ishikawa, Trustworthy Cyber-Physical Systems Engineering, CRC Press, 2016, 341-371.