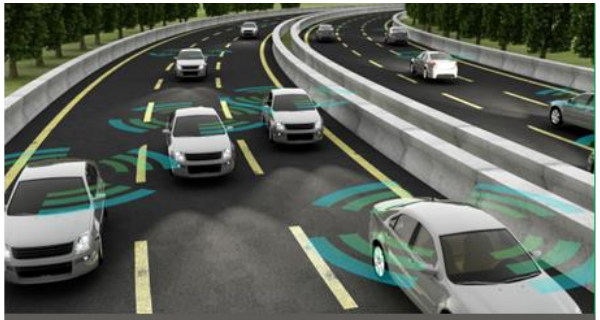
Critical Systems Labs
Innovating Safely

Safety Integrity Levels for Artificial Intelligence

WAISE 2023 – Toulouse

Critical Systems Labs Inc.

Laure Millet, Ph.D. & Simon Diemert, M.Sc., P.Eng.



Provided expertise for obtaining compliance with ISO 26262

Integration of safety and cybersecurity in safety critical systems

Performed safety risk assessment of ship helicopter operating limits

Identified and analyzed safety of advanced technology in driverless trains

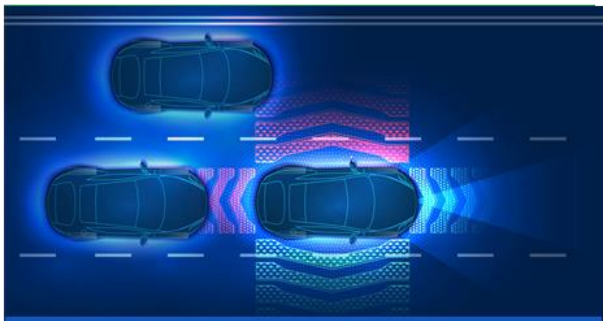
Assisted with hardware and software requirements to obtain FDA approval

Performed systems safety analysis of control systems in critical infrastructure

Analyzed safety critical SW in jet engines using formal (mathematical) methods

Provided system safety design expertise for warship and military submarine navigation systems

Technical review of critical protection systems in CERN Large Hadron Collider



Provided expertise on the safety design of ADAS and autonomous vehicles

Presenter

Laure Millet is a Software and Systems Engineer at Critical Systems Labs Inc. She has extensive experience in safety assurance across a wide range of technical domains including aerospace, automotive, defense, medical and rail. She is often involved in client projects that involve unique challenges in managing safety risk associated with emergent technology such as the use of Machine Learning in autonomous vehicles. Laure has received a doctorate in Computer Science from Pierre and Marie Curie University (Paris, France).

Risk Ranking

RISK ASSESSMENT MATRIX				
SEVERITY / PROBABILITY	Catastrophic (1)	Critical (2)	Marginal (3)	Negligible (4)
Frequent (A)	High	High	Serious	Medium
Probable (B)	High	High	Serious	Medium
Occasional (C)	High	Serious	Medium	Low
Remote (D)	Serious	Medium	Medium	Low
Improbable (E)	Medium	Medium	Medium	Low
Eliminated (F)	Eliminated			

MIL STD 882E

- What is the probability of failure for a system whose behaviour is mainly determined by AI or ML?

Level of Rigour Approach in ISO 26262

Severity class	Exposure class	Controllability class		
		C1	C2	C3
S1	E1	QM	QM	QM
	E2	QM	QM	QM
	E3	QM	QM	A
	E4	QM	A	B
S2	E1	QM	QM	QM
	E2	QM	QM	A
	E3	QM	A	B
	E4	A	B	C
S3	E1	QM	QM	A[a]
	E2	QM	A	B
	E3	A		
	E4	E		

Table 7 — Methods for software unit verification

	Methods	ASIL A	ASIL B	ASIL C	ASIL D
1a	Walk-through[a]	++	+	o	o
1b	Pair-programming[a]	+	+	+	+
1c	Inspection[a]	+	++	++	++
1d	Semi-formal verification	+	+	++	++
1e	Formal verification	o	o	+	+
1f	Control flow analysis[b, c]	+	+	++	++
1g	Data flow analysis[b, c]	+	+	++	++
1h	Static code analysis[d]	++	++	++	++
1i	Static analyses based on abstract interpretation[e]	+	+	+	+
1j	Requirements-based test[f]	++	++	++	++
1k	Interface test[g]	++	++	++	++

Confidence is increased by performing increasingly demanding engineering activities.

Level of Rigour in MIL STD 882E

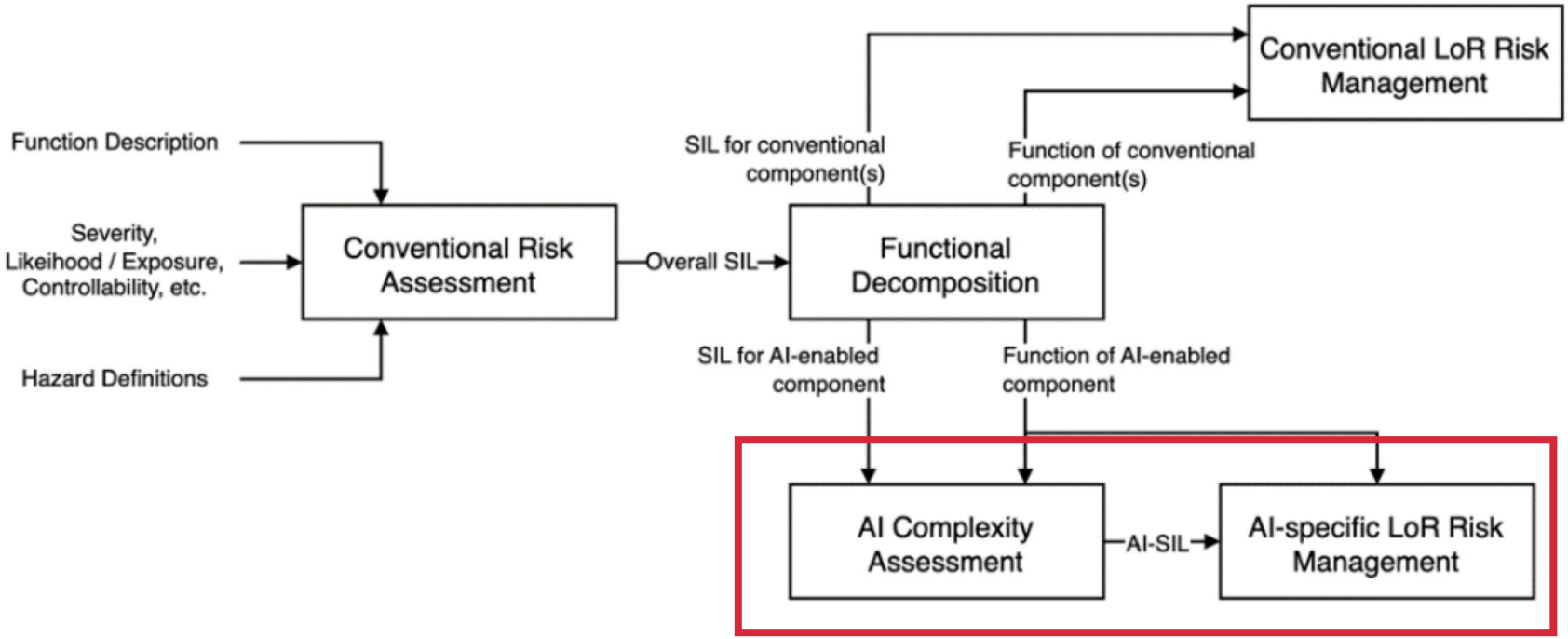
SOFTWARE CONTROL CATEGORIES		
Level	**Name**	**Description**
1	Autonomous (AT)	• Software functionality that exercises autonomous control authority over potentially safety-significant hardware systems, subsystems, or components without the possibility of predetermined safe detection and intervention by a control entity to preclude the occurrence of a mishap or hazard. *(This definition includes complex system/software functionality with multiple subsystems, interacting parallel processors, multiple interfaces, and safety-critical functions that are time critical.)*
2	Semi-Autonomous (SAT)	• Software functionality that exercises control authority over potentially safety-significant hardware systems, subsystems, or components, allowing time for predetermined safe detection and intervention by independent safety mechanisms to mitigate or control the mishap or hazard. *(This definition includes the control of moderately complex system/software functionality, no parallel processing, or few interfaces, but other safety systems/mechanisms can partially mitigate. System and software fault detection and annunciation notifies the control entity of the need for required safety actions.)* • Software item that displays safety-significant information requiring immediate operator entity to execute a predetermined action for mitigation or control over a mishap or hazard. Software exception, failure, fault, or delay will allow, or fail to prevent, mishap occurrence. *(This definition assumes that the safety-critical display information may be time-critical, but the time available does not exceed the time required for adequate control entity response and hazard control.)*
3	Redundant Fault Tolerant (RFT)	• Software functionality that issues commands over safety-significant hardware systems, subsystems, or components requiring a control entity to complete the command function. The system detection and functional reaction includes redundant, independent fault tolerant mechanisms for each defined hazardous condition. *(This definition assumes that there is adequate fault detection, annunciation, tolerance, and system recovery to prevent the hazard occurrence if software fails, malfunctions, or degrades. There are redundant sources of safety-significant information, and mitigating functionality can respond within any time-critical period.)* • Software that generates information of a safety-critical nature used to make critical decisions. The system includes several redundant, independent fault tolerant mechanisms for each hazardous condition, detection and display.
4	Influential	• Software generates information of a safety-related nature used to make decisions by the operator, but does not require operator action to avoid a mishap.
5	No Safety Impact (NSI)	• Software functionality that does not possess command or control authority over safety-significant hardware systems, subsystems, or components and does not provide safety-significant information. Software does not provide safety-significant or time sensitive data or information that requires control entity interaction. Software does not transport or resolve communication of safety-significant or time sensitive data.

SOFTWARE SAFETY CRITICALITY MATRIX				
	SEVERITY CATEGORY			
SOFTWARE CONTROL CATEGORY	Catastrophic (1)	Critical (2)	Marginal (3)	Negligible (4)
1	SwCI 1	SwCI 1	SwCI 3	SwCI 4
2	SwCI 1	SwCI 2	SwCI 3	SwCI 4
3	SwCI 2	SwCI 3	SwCI 4	SwCI 4
4	SwCI 3	SwCI 4	SwCI 4	SwCI 4
5	SwCI 5	SwCI 5	SwCI 5	SwCI 5

Risk Ranking of AI-autonomy

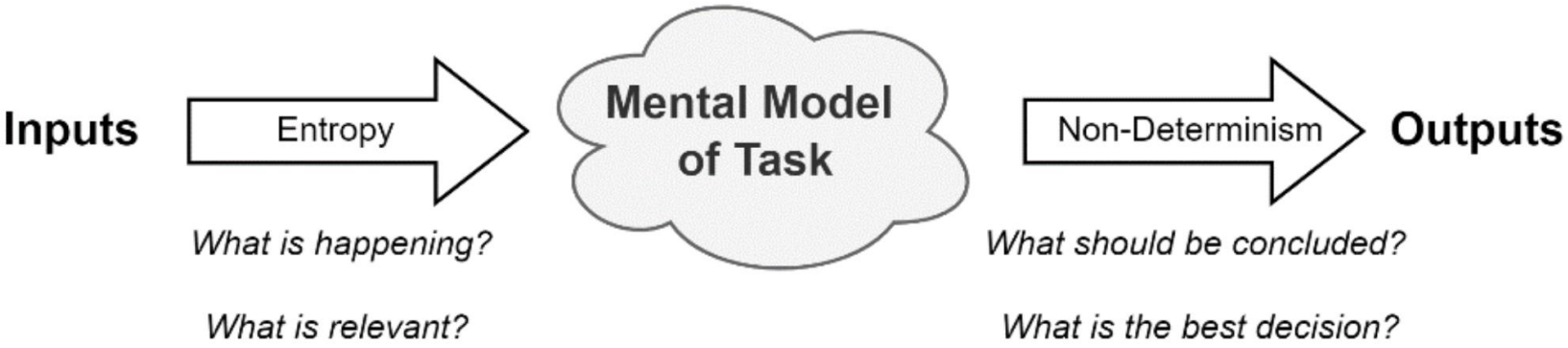
- Use a systematic approach to rank complexity associated with a **task** being performed by an AI-based component.
- Combine with **level of risk** as determined by conventional methods.
- Use the result to select **level of rigour activities** to gain confidence in the AI-based component.

Proposed Method at a Glance



Our contribution

Classifying Task Complexity



Rank task complexity based on "human expert" standard, independent of technology performing the task.

Example – Input Entropy

Example – Out Non-Determinism

Input Entropy

- **E1 (Low)** – Low input dimensionality with low variability in each dimension. Independent experts derive (nearly) identical mental models.
- **E2 (Medium)** – Moderate number of input dimensions with a high variability in each dimension. Independent experts might derive different mental models of the situation but, after discussion, could agree.
- **E3 (High)** – Many input dimensions where each dimension has high variability. Independent human experts might derive different mental models of the situation and, even after discussion, might disagree on the correctness.

Output Non-Determinism

- **N1 (Low)** – Given perfect knowledge of the situation, experts performing this task will produce the same result.
- **N2 (Medium)** – Given perfect knowledge of the situation, experts performing this task might produce different results, but likely agree that other's conclusions are acceptable.
- **N3 (High)** – Given perfect knowledge of the situation, experts performing this task might produce different results, and disagree on the acceptability of each other's result or the level of confidence of the results.

Selecting the AI-SIL

		Non-Determinism		
SIL	**Entropy**	**N1** (low)	**N2** (med)	**N3** (high)
SIL 1	**E1** (low)	1	1	1
	E2 (med)	1	1	1
	E3 (high)	1	1	2
SIL 2	**E1** (low)	1	2	2
	E2 (med)	2	2	2
	E3 (high)	2	2	3
SIL 3	**E1** (low)	2	3	3
	E2 (med)	3	3	3
	E3 (high)	3	3	4
SIL 4	**E1** (low)	3	4	4
	E2 (med)	4	4	4
	E3 (high)	4	4	4

Example – Vehicle Occupancy Detection

Task: turn off the interior lights in a vehicle when it is no longer occupied, the input is the ambient volume measured by a single microphone.

SIL: this application would likely be a SIL 1 system, as a turning the lights off is expected to be low risk.

Input Entropy: Single simple sensor → E1

Output Non-Determinism: Output is a binary value and it is unlikely that experts would disagree on if the vehicle is occupied or not → D1.

Combine: AI-SIL 1

Example – Vehicle Path Planning

Task: Generates a vehicle trajectory that 1) progress towards the objective is achieved, 2) driving rules are obeyed, and 3) no collisions occur.

SIL: Suppose the AI-based component performing this task is assigned SIL 4.

Input Entropy: There is a significant amount of variability in the environment including the diversity of object types and behaviours, driving rules, and environmental conditions → E3.

Output Non-Determinism: Many trajectories that could be produced that trade off progress and safety → N3.

Combine: AI-SIL 4

Level of Rigour from AI-SILs

For illustration (not a complete proposal)

Example of LoR Activities for AI-SIL 1

- **Focus on AI component performance.**
- Specify and verify performance requirements using well-known metrics (e.g., TP, TN, FP, FN rates).
- Monitor post-deployment to ensure “in-field” performance satisfies requirements.

		Non-Determinism		
SIL	**Entropy**	**N1** (low)	**N2** (med)	**N3** (high)
SIL 1	**E1** (low)	1	1	1
	E2 (med)	1	1	1
	E3 (high)	1	1	2
SIL 2	**E1** (low)	1	2	2
	E2 (med)	2	2	2
	E3 (high)	2	2	3
SIL 3	**E1** (low)	2	3	3
	E2 (med)	3	3	3
	E3 (high)	3	3	4
SIL 4	**E1** (low)	3	4	4
	E2 (med)	4	4	4
	E3 (high)	4	4	4

Example of LoR Activities for AI-SIL 2

- **Follow a systematic AI/ML development process.**
- Define operational domain using a structured method/representation.
- Requirements for data volume, data quality, robustness, performance, etc.
- Rationalize and document detailed design choices (e.g., number of layers in NN, activation functions, etc.).

		Non-Determinism		
SIL	**Entropy**	**N1** (low)	**N2** (med)	**N3** (high)
SIL 1	**E1** (low)	1	1	1
	E2 (med)	1	1	1
	E3 (high)	1	1	2
SIL 2	**E1** (low)	1	2	2
	E2 (med)	2	2	2
	E3 (high)	2	2	3
SIL 3	**E1** (low)	2	3	3
	E2 (med)	3	3	3
	E3 (high)	3	3	4
SIL 4	**E1** (low)	3	4	4
	E2 (med)	4	4	4
	E3 (high)	4	4	4

Example of LoR Activities for AI-SIL 3

- **Increased V&V rigour.**
- Independent review(s) of engineering artifacts.
- Require more sophisticated V&V methods, such as combinatorial testing, surprise adequacy, or metamorphic testing.
- Require coverage assessments over domain.
- Might make more use of simulation technologies.

		Non-Determinism		
SIL	**Entropy**	**N1** (low)	**N2** (med)	**N3** (high)
SIL 1	**E1** (low)	1	1	1
	E2 (med)	1	1	1
	E3 (high)	1	1	2
SIL 2	**E1** (low)	1	2	2
	E2 (med)	2	2	2
	E3 (high)	2	2	3
SIL 3	**E1** (low)	2	3	3
	E2 (med)	3	3	3
	E3 (high)	3	3	4
SIL 4	**E1** (low)	3	4	4
	E2 (med)	4	4	4
	E3 (high)	4	4	4

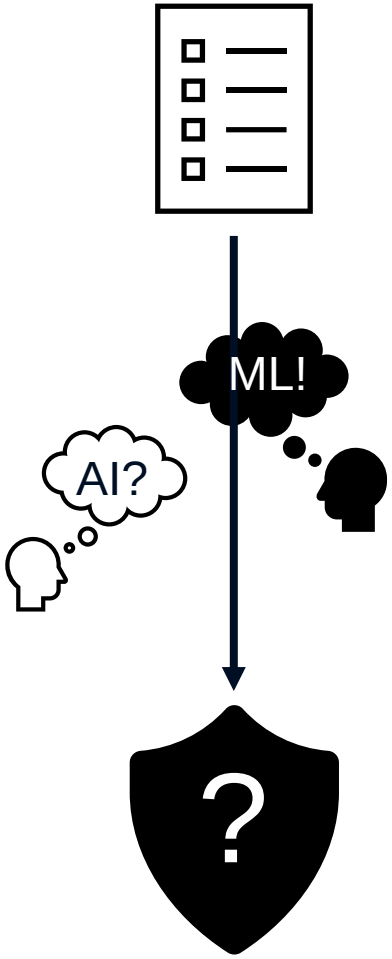
Example of LoR Activities for AI-SIL 4

- **Avoid AI-SIL 4 tasks.**
- Revise functional decomposition of system.
- Reduce task complexity (e.g., narrow operational domain, reduce degrees of freedom for output).
- Proceed with caution (consider formal verification).

		Non-Determinism		
SIL	**Entropy**	**N1** (low)	**N2** (med)	**N3** (high)
SIL 1	**E1** (low)	1	1	1
	E2 (med)	1	1	1
	E3 (high)	1	1	2
SIL 2	**E1** (low)	1	2	2
	E2 (med)	2	2	2
	E3 (high)	2	2	3
SIL 3	**E1** (low)	2	3	3
	E2 (med)	3	3	3
	E3 (high)	3	3	4
SIL 4	**E1** (low)	3	4	4
	E2 (med)	4	4	4
	E3 (high)	4	4	4

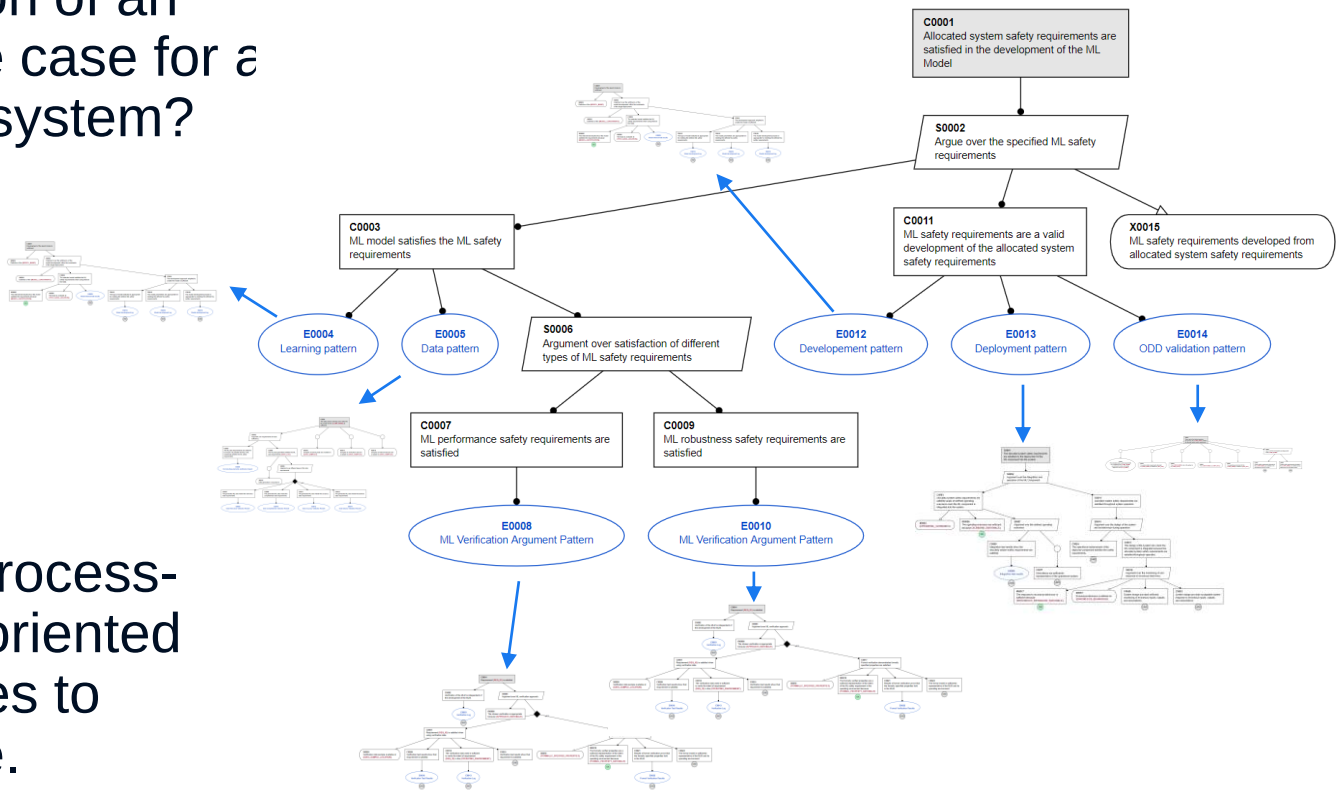
Next Steps – Validation of Method

- **Repeatability** – Do different users arrive at similar results for the same system?
- **Generalizability** – Does it apply in multiple domains to different tasks?
- **Useful** – Does this actually help structure risk management for AI-based systems?



Next Steps – Assurance Cases

- Can AI-SILs support the creation of an assurance case for a AI-based system?



- Balance process- and goal-oriented approaches to assurance.

Summary

- Use a systematic approach to rank complexity associated with a **task** being performed by an AI-based component.
- Combine with **level of risk** as determined by conventional methods.
- Use the result to select **level of rigour activities** to gain confidence in the AI-based component.

Laure Millet, Ph.D.
VP Research
laure.millet@cslabs.com