Safety-related pre-conditions for a sustainable automated mobility

Mobility as a Service

Eric Verhulst

www.altreonic.com

Abstract

The dream of autonomous traffic poses a serious trust and safety challenge. Each vehicle becomes a component in the transport system and must be error-resilient while the dynamic constraints are very high. The current car's architecture is not in a position to provide this (ARRL-3) but a future electric or hybrid car might be able to (ARRL-5) and when integrated in a larger feedback-driven transport system it becomes an ARRL-7 component. ARRL stands for Assured Reliability and Resilience Level and is a novel criterion that defines safety in a wider context. Altreonic is researching these topics and working on developing a scalable and modular vehicle concept that is capable of meeting these requirements.

Altreonic profile

- History goes back to 1989 (Eonic Systems)
  - Specialised in parallel RTOS (T800, C40, C6x, 2016x, TS102, G4, …)
  - Used from 1 CPU to 1600 DSPs (sonar, radar) to 12000 nodes (heterogeneous (sensing + 3D deconvolution))
  - Virtuoso RTOS in use on ROSETTA
  - Acquired by Wind River Systems in 2001
- Altreonic: created as new spin-off in 2008 after R&D
  - Unified systems engineering methodology
  - Formalised when possible => **OpenComRTOS Designer**
  - **GoedelWorks**: from early requirements to implementation
  - Focus on **trustworthy scalable embedded system**s
    - Safety, Security, Usability, Privacy
    - Unique “Open Technology License” model

Rosetta: rendez-vous after 10 years

- The law of Newton (F=ma) is the common factor with autonomous driving
- The rest is trustworthy engineering

Problem statement

- (Terrestrial) Mobility and Transport are reaching a choking point
- Major issues:
  - Scalability (density)
  - Performance (to get from A to B)
  - Transport means confused with transport modes
  - Connection points are bottlenecks
- The Quality Of Service is declining
- Safety is a growing concern
- Energy efficiency is a must

Roadmap thinking: MaaS

- Mobility and transport are not vehicle markets but service markets => **Mobility as a Service**
- Fundamental requirement: door-to-door
- Mobility = vehicle + infrastructure + process
- Holistic approach needed:
  - What is a mobility/transport unit?
  - If automation is the answer, how can it be safe, cost-efficient yet what people need?
  - What means scalable?
- Not solving the issue is an economic disaster
- Solving the issue is an competitive advantage

A mobility pod as a unit fo transportation

- Cfr. Packet switching (as done in telecom, internet)
- Bottom-up view: from small vehicles for individual use that change mode by becoming tightly or loosely coupled with each other.
- Vehicles scale up into modes (now covered by separate transport means e.g. bicyle, car, bus, train, ...), with each mode allowing more autonomous, faster and safer transport, provided the road infrastructure is adapted.
- Enabling technology is the use of hybrid or electric redundant drive systems that allow to use such vehicles as fault-tolerant transport components.
- Separate people's mobility from good's transport

Consequences are far reaching

- This project encompasses many domains: energy-efficient mobility, road infrastructure, safety and systems engineering, regulation, legal, …
- Fault-tolerance is a necessary pre-condition to achieve safety levels
  - Not covered by the current automotive standards.
- Enabler for other domains: the challenge is high!

Mobility and safety across domains

- **Automotive**:
  - 1,2 million people killed/year: **daily event**
  - Cars get better, but people get killed: safer? QoS?
- **Aviation**:
  - 500 people killed/year: **a rare event**
  - Planes get better, cheaper, safer, energy-efficient
- Railway, telecommunications, medical, …
  - Similar examples
- **What sets them apart?**

Systems Engineering vs. Safety Engineering

- System = holistic
- Real goal is **"Trustworthy Systems"**
  - Cfr. Felix Baumgartner almost did not do it because he didn't trust his safe jumpsuit
- TRUST = by the user or stakeholders
  - Achieving intended Functionality
  - Safety & Security & Usability & Privacy
  - Meeting non-functional objectives
    - Cost, energy, volume, maintainability, scalability, Manufacturability,..
  - Quality of Service is multi-criteria property

User expects a guaranteed "QoS"

from a "Trustworthy System"

Safety and certification

- **Safety** can be defined to be the **control of *recognized hazards*** to achieve an ***acceptable level of risk***.
  - Safety is general property of a system, not 100% assured
  - It is complex but there are moral liabilities
- Certification: In depth review => safe to operate
  - "Conformity assessment" (for automotive)
  - Not a technical requirement: confidence, legal
- **Evidence makes the difference:**
  - Evidence is a **coherent** collection of **information** that relying on a number of **process artifacts** linked together by their **dependencies and sufficient structured arguments** provides an **acceptable proof** that a specific system goal has been reached.

Categorisation of Safety Risks

Category	Consequence upon failure	Typical SIL/ASIL
Catastrophic	Loss of multiple lives	4 / D
Critical	Loss of a single life	3 / C
Marginal	Major injuries to one or more persons	2 / B
Negliglible	Minor injuries at worst or material damage	1 / A
No consequence	No damages, user dissatisfaction	0

- SIL $\cong$ f (**probability of occurrence, severity, controllability**)
  - As determined by HARA
  - SIL goals ≅ Risk Reduction Factor
- Criteria and classification are open to interpretation

Problems with SIL definition

- Poor harmonization of definition across the different standards bodies which utilize SIL=> Reuse?
- Process-oriented metrics for derivation of SIL
- SIL level determines architecture (system specific)
- Estimation of SIL based on **reliability estimates**
  - System complexity, particularly in software systems, makes SIL estimation difficult if not impossible
  - based on probabilities that are very hard if not impossible to measure and estimate
  - Reliability of software (discrete domain) is not statistical!:
  - The **law of Murphy still applies**:
    - The next instant can be catastrophic

New definition: start from the component up

ARRL: Assured Reliability and Resilience Level

ARRL 0	it might work (use as is)
ARRL 1	works as tested, but no guarantee
ARRL 2	works correctly, IF no fault occurs, guaranteed no errors in implementation) => formal evidence
ARRL 3	ARRL 2 + goes to fail-safe or reduced operational mode upon fault (requires monitoring + redundancy) - fault behavior is predictable as well as next state
ARRL 4	ARRL 3 + tolerates one major failure and is fault tolerant (fault behavior predictable and transparent for the external world). Transient faults are masked out
ARRL 5	The component is using heterogeneous sub-components to handle residual common mode failures

ARRL: what does it mean?

- **Assured:**
  - There is verified, trustworthy **evidence**
  - Process related and architecture related
- **Reliability:**
  - In absence of faults, MTBF is >> life-time: **QA aspects**
- **Resilience:**
  - The fault behaviour is predicted: **trustworthy behaviour**
  - Capability to continue to provide core function
- **Level**: ARRL is **normative**
  - Components can be classified: **contract**

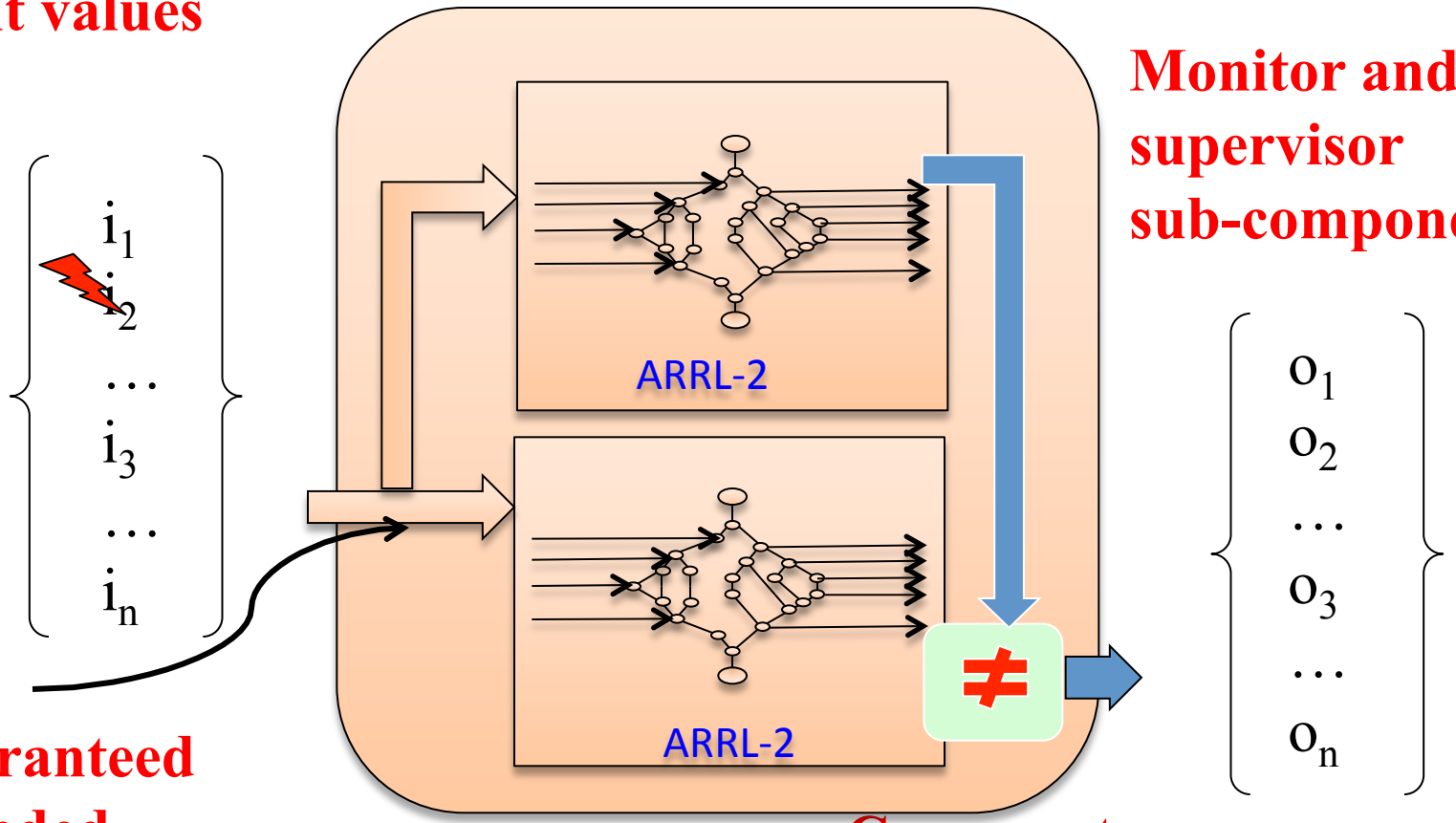
Consequences

- If a system/component has a fault, it drops into a degraded mode => lower ARRL
  - ARRL3 is the operational mode after an ARRL4 failure
    - Functionality is preserved
    - Assurance level is lowered
- SIL not affected and domain independent
  - System + environment + operator defines SIL
- ARRL is a **normative criterion**:
  - Fault behavior is made explicit: verifiable
  - Cfr. IP-norm (comes with a predefined test procedure)

ARRL-3

Induced fault

Unanticipated input values



Monitor and supervisor sub-component

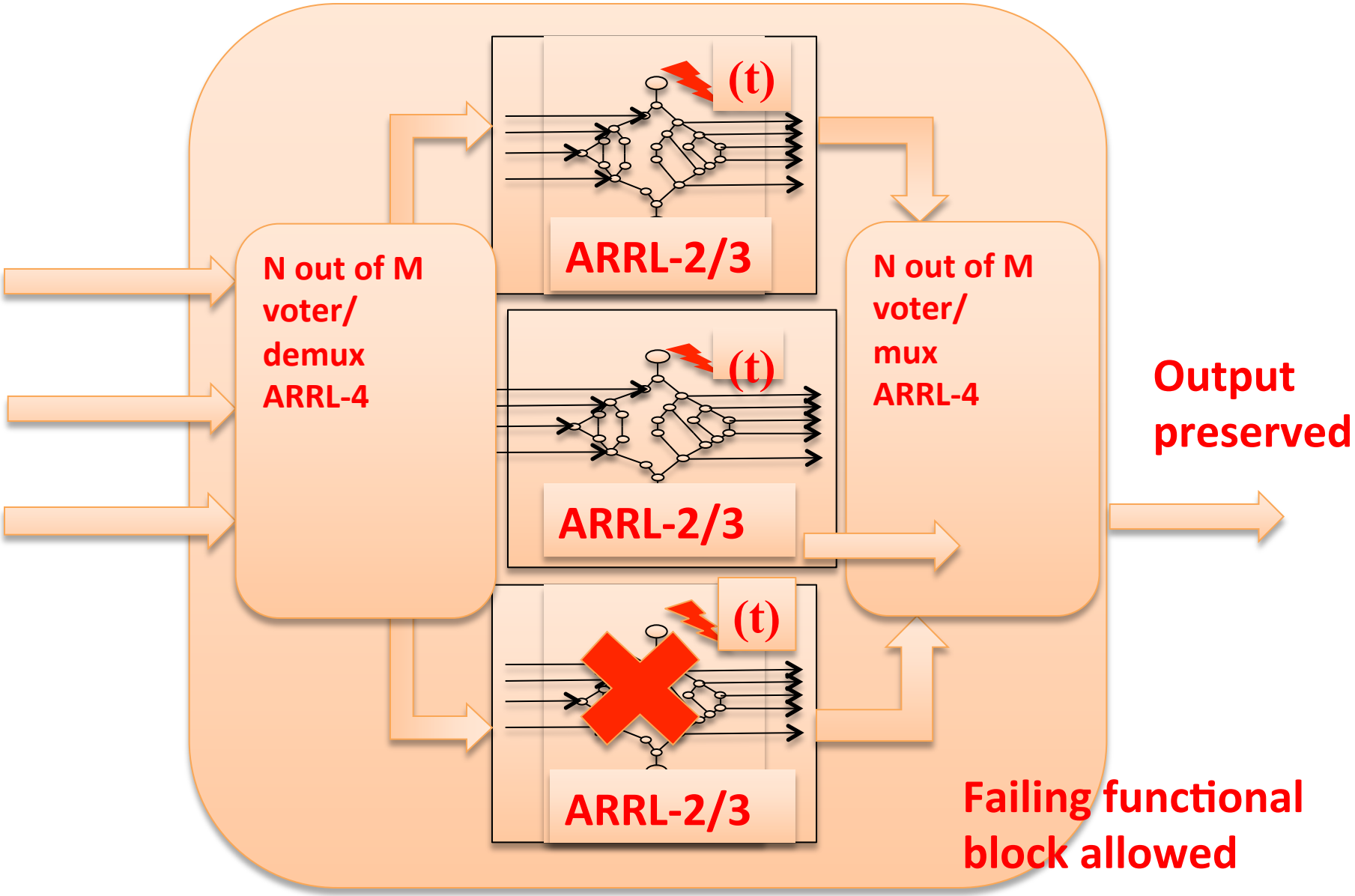
Guaranteed bounded

Comparator

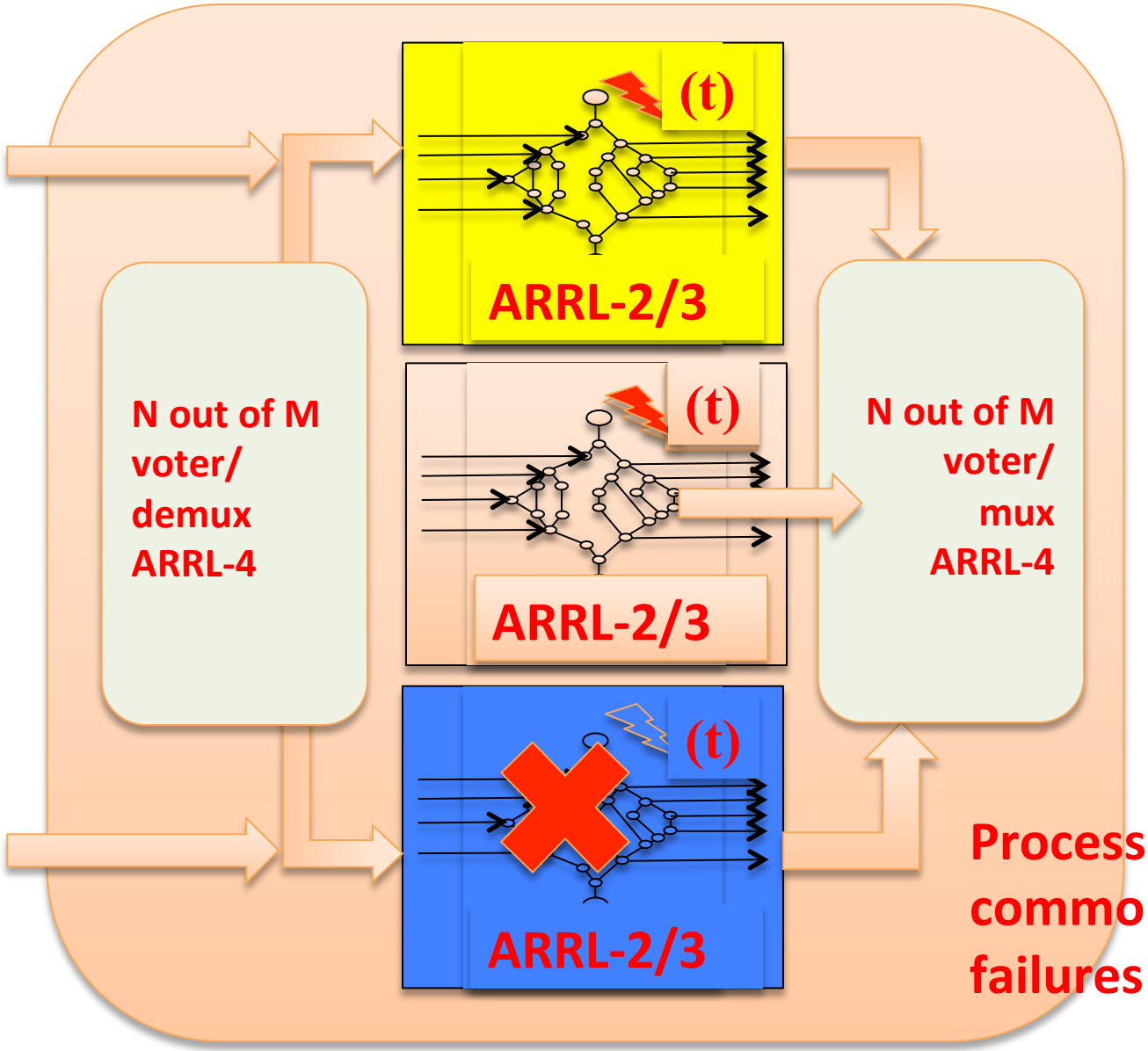
Fail safe output (but not correct)

Common mode failures possible

ARRL-4: voting



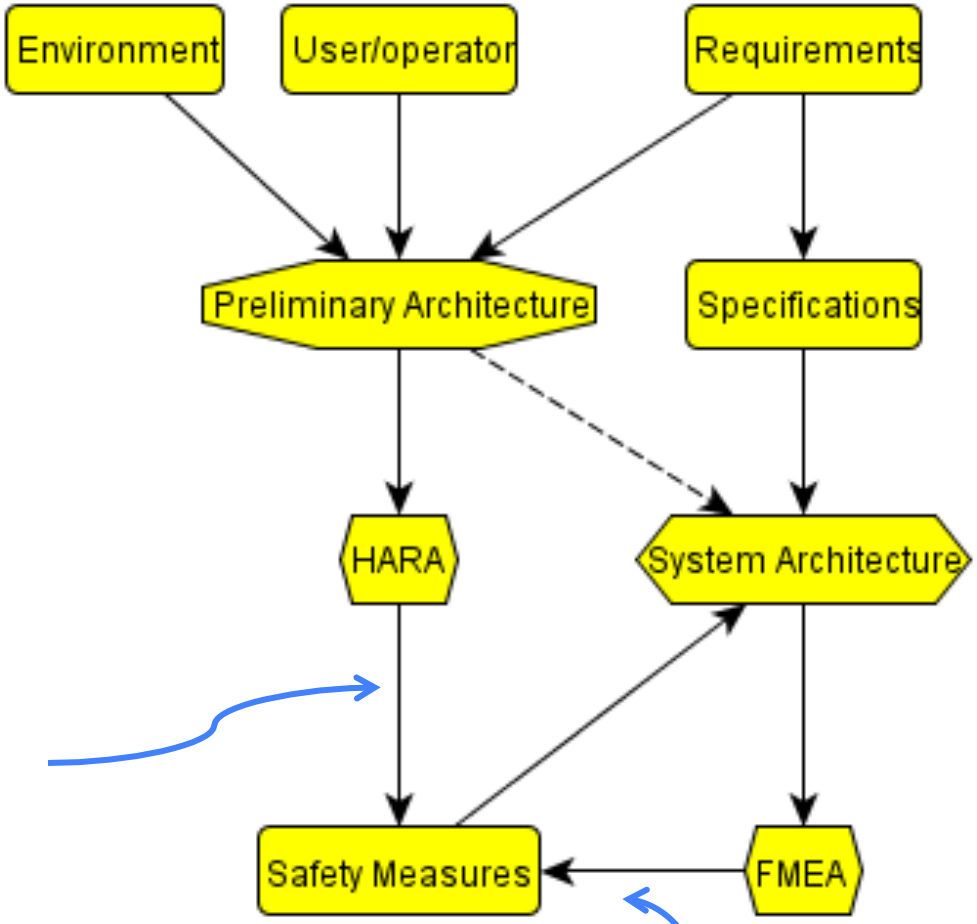
ARRL-5: diversity (against common mode failures



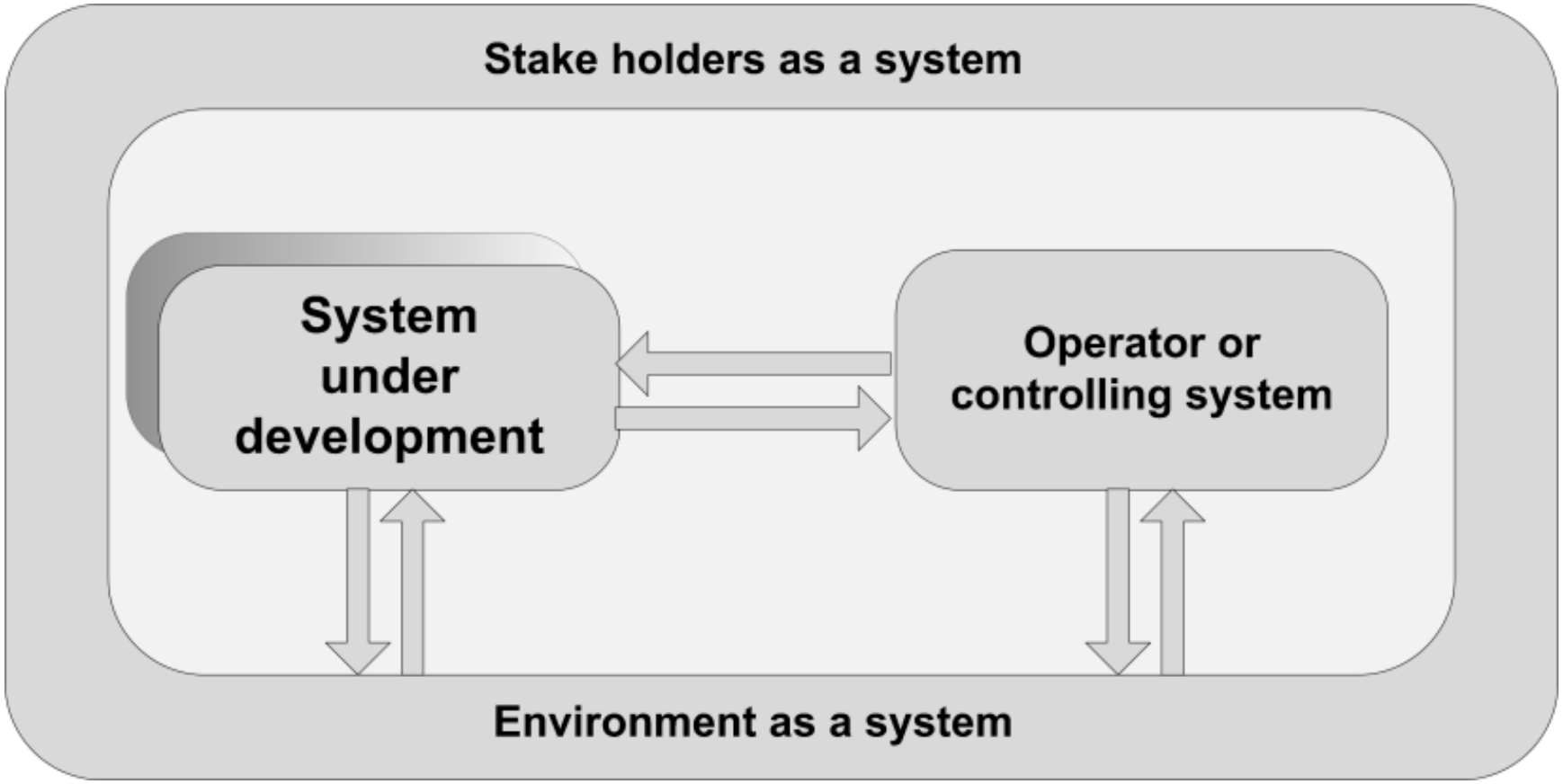
Output preserved

Process related common mode failures minimised

SIL and ARRL are complementary



A system is never alone



What means “anti-fragile”?

- New term quoted by Taleb
- An anti-fragile system gets “**better**” after being exposed to “**stressors**”
  - Better: we need a **metric** => QoS?
  - Stressors: cfr. **hazard, faults**, …
  - The issue in safety: **rare events** (improbable a priori, certain post factum) (Taleb’s “black swan”)
- **What does it mean** in the context of safety/ systems engineering? Isn’t ARRL-5 not the top level?

ARRL-6 and ARRL-7 (inherits ARRL-5)

ARRL 3	ARRL 2 + goes to fail-safe or reduced operational mode upon fault (requires monitoring + redundancy) - fault behavior is predictable as well as next state
ARRL 4	ARRL 3 + tolerates one major failure and is fault tolerant (fault behavior predictable and transparent for the external world). Transient faults are masked out
ARRL 5	The component is using heterogeneous sub-components to handle residual common mode failures
ARRL 6	The component (subsystem) is monitored and a process is in place that maintains the system's functionality
ARRL 7	The component (subsystem) is part of a system of systems and a process is in place that includes continuous monitoring and improvement supervised by an independent regulatory body

Preconditions for anti-fragility

- **Extensive domain knowledge**: experience
- **Openness**: shared critical information
- **Feedback loops** at several levels between large number of stakeholders
- Independent **supervision**: guidance
- Core components are **ARRL-4 or -5**
- **The system is the domain**
- **Service matters more** than the component

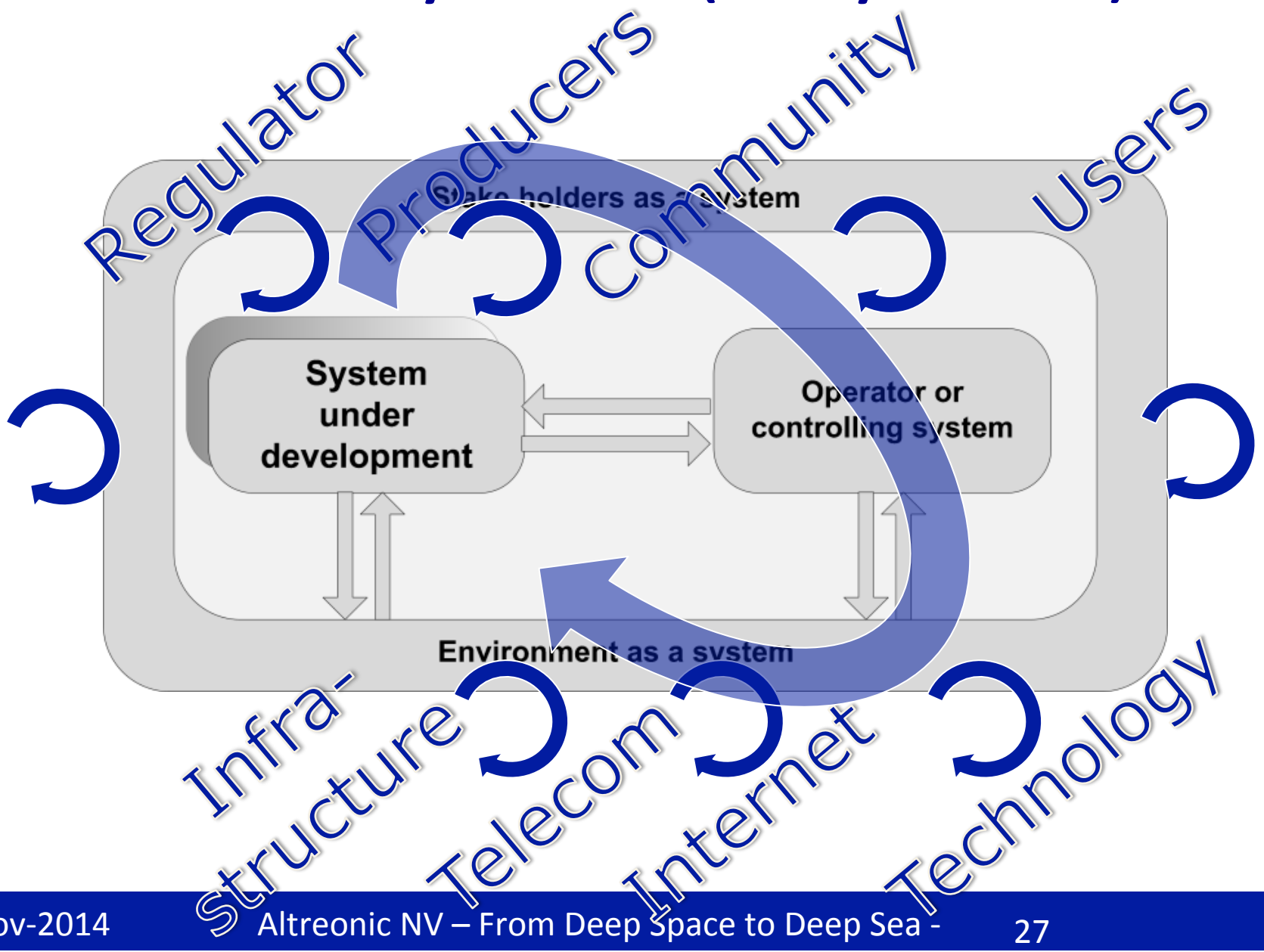
Assessment in terms of ARRL

- **Automotive**:
  - Vehicle is an **ARRL-3 system**
  - Upon fault, presumed to go the fail-safe state
  - No black box, no records, …
  - Automotive is **a collection of vehicles**
- **Aviation**:
  - Planes are **ARRL-5**
  - Upon fault, redundancy takes over
  - Black box, central database,
  - Preventive maintenance
  - Aviation is **an eco-system providing a Service**

ARRL benefits and challenges

- Trustworthy = system + evidence
  - ARRL defines a contract on the component
- Challenge:
  - Complete coverage: state space is enormous
  - Still needs to take into account environment
- Benefits:
  - Reuse of components: plug and play
  - Environment still to be taken into account
- A must for a system with many evolving components: life-cycle engineering

Extended systems (of systems) view

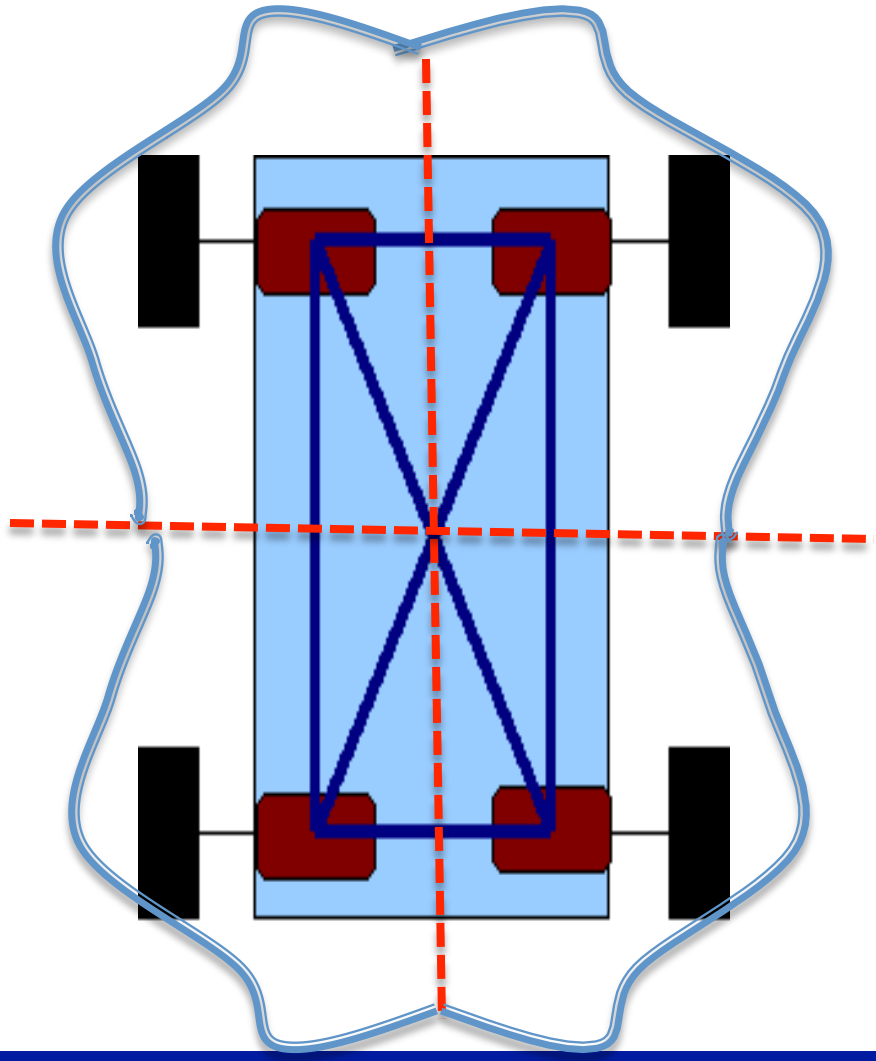


Autonomous traffic

- Self-driving cars are the future? Cfr. Google car as Proof of Concept
- Systems engineering challenge much higher than flying airplanes
- Huge impact: socio-economic “black swan”
- Pre-conditions:
  - Vehicles become ARRL-5
  - System = traffic, includes road infrastructure
  - Standardisation (vehicles communicate)
  - Continuous improvement process
- Hence: needs ARRL-7

System architecture

- Combine reusable units to increase
  - Economy of scale
  - Redundancy
- Unit =
  - Propulsion
  - Energy
  - Control
  - Sensors



Conclusions

- ARRL concept allows compositional safety engineering with reuse of components/subsystems
- More complex systems can be safer
- A unified ARRL aware process pattern can unify systems and safety engineering standards
- ARRL-6 and ARRL-7 introduce a system that include a feedback loop process during development but also during operation
- **Mission is to provide a service => MaaS**
- **ANTIFRAGILE = ARRL-7: life-cycle engineering**
- **Pre-condition for trustworthy autonomous driving**

Status

- Project proposals under Horizon 2020
- Using formal hybrid logics and simulation modelling
- We work bottom-up
- Formalised and incremental engineering
- First research prototype developed

Contact:

Eric.Verhulst @ altreonic.com

www.altreonic.com