Altreonic

The long road from proof of concept to real-world autonomous systems

Eric Verhulst, Altreonic NV

eric.verhulst@altreonic.com

Content

- Who’s Altreonic?
- When is a system autonomous?
- When is a system trustworthy?
- Safety engineering standards
- ARRL criterion (1 to 7)
- Autonomous = antifragile = ARRL8?
- Conclusion

Altreonic profile

- 30 years aero-space-defense (Eonic Systems NV)
  - Specialised in parallel Real-Time Operating System
  - Used by ESA (**Virtuoso RTOS** on Rosetta mission)
    - Not an autonomous system: pre-programmed!
- Altreonic NV focuses on trustworthy embedded systems:
  - **Meta-model for systems engineering**
    - http://www.altreonic.com/content/altreonic-approach-systems-engineering-goedelworks
  - **VirtuosoNext Designer**, formally developed distributed RTOS with programming tools
  - **GoedelWorks**, a portal based environment to support Software Engineering, with embedded certification
- Competitive advantage for the novel e-vehicle **KURT**

Autonomous systems

1. **Government**.
   self-governing; independent; subject to its own laws only.
   pertaining to an autonomy, or a self-governing community.
2. **Having autonomy**; not subject to control from outside;
   independent: a subsidiary that functions as an autonomous unit.
3. **Of a vehicle:**
   Navigated and maneuvered by a computer, without a need for human control or intervention under normal road conditions
4. **Biology**
   - existing and functioning as an independent organism.
   - growing naturally or spontaneously, without cultivation.

NOTE: Autonomous =/= Automated =/= "pre-programmed"

Src:
http://www.dictionary.com/
https://en.wikipedia.org/wiki/Autonomous_car

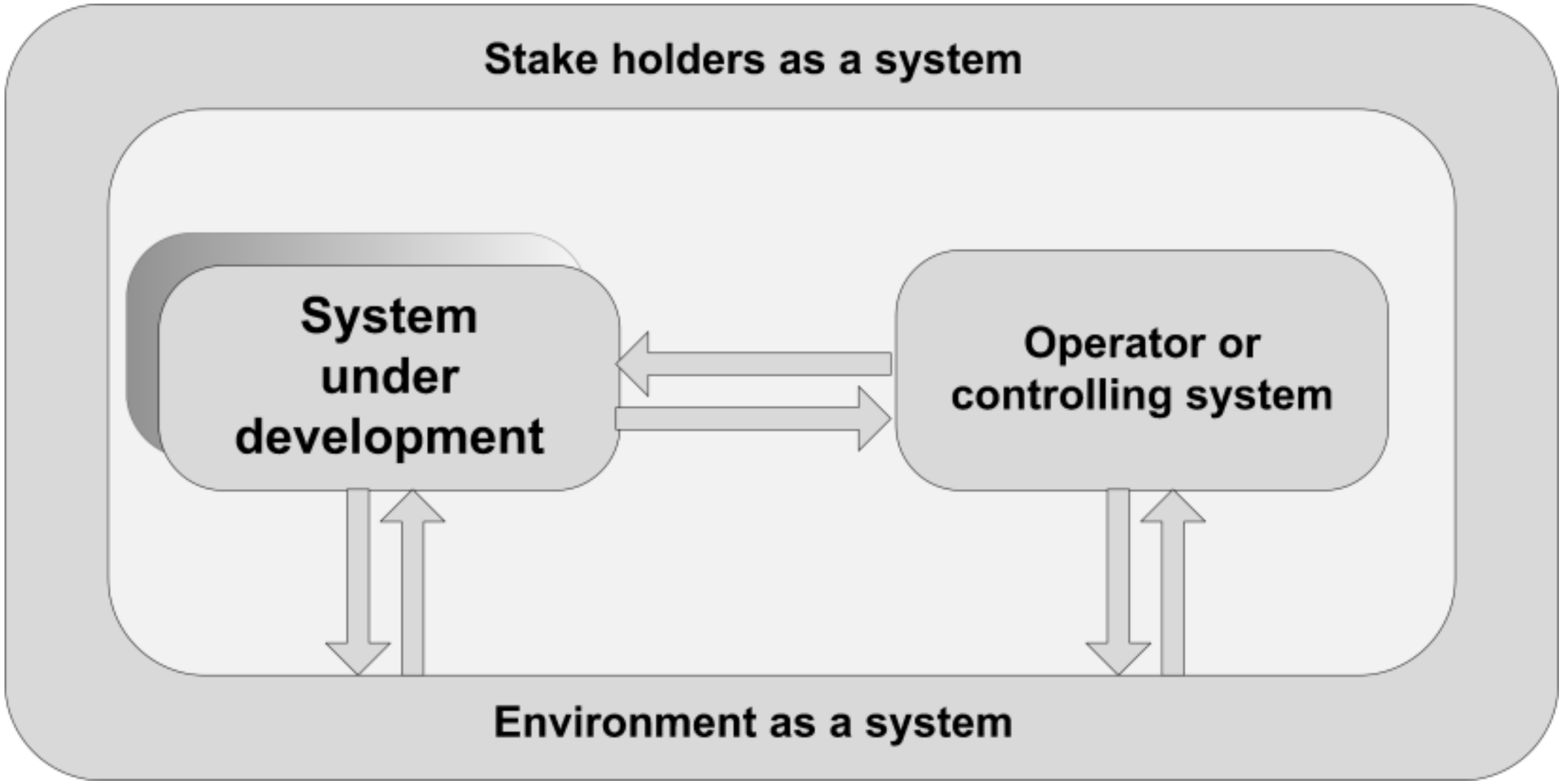
Tesla's game breaking move

- We are excited to announce that, as of today (19 October 2016), all Tesla vehicles produced in our factory – including Model 3 – will have the hardware needed for full self-driving capability at a safety level substantially greater than that of a human driver. Eight surround cameras provide 360 degree visibility around the car at up to 250 meters of range. Twelve updated ultrasonic sensors complement this vision, allowing for detection of both hard and soft objects at nearly twice the distance of the prior system. A forward-facing radar with enhanced processing provides additional data about the world on a redundant wavelength, capable of seeing through heavy rain, fog, dust and even the car ahead.
- To make sense of all of this data, a new onboard computer with more than 40 times the computing power of the previous generation runs the new Tesla-developed neural net for vision, sonar and radar processing software. Together, this system provides a view of the world that a driver alone cannot access, seeing in every direction simultaneously and on wavelengths that go far beyond the human senses.
- True or False?

What are the pre-conditions?

- The behavior is based on “internal” rules
- The behavior is not “pre-programmed” or due to an external actor
- Questions?
  - What guides its behavior?
  - What corrects its behavior?
  - What’s the use?
  - Does it really take the human out of the loop?
  - Define “normal road conditions”

A system is never alone



REQ & SPC

- REQ1: Navigated and maneuvered by a computer, without a need for human control or intervention under normal road conditions
- REQ2: Safe under all conditions
- SPC1:
  - Safe in hazardous conditions
  - Safe in the presence of faults

  SPC1.1: act correctly in less than 100 milliseconds

An embedded system also has hierarchy

Environment	Hostile by definition	Hazards injection faults and hazards
Human-Machine-Interface	Interface between operator and machine in an environment	Ergonomics!
Application software	SW: large state space, dynamic, Never error-free	Specification errors, timing issues, algorithms, illegal paths,
Drivers	HW: connection with the external world and system As Reliable as tested	Protocol errors, bit flips, missing data
Firmware/ (RT)OS	SW State machine, High Reliability (formal)	Fails in a few cycles
Electronics, memory	HW State machine, Very High Reliability (formal)	Fails in 1 clock pulse (< 20 nanoseconds)
Power & energy	Heat, energy density	System Hazard
Mechanical	Structure, mass, forces	Graceful degradation

What is a trustworthy system?

- System = holistic
- Real goal is "**Trustworthy Systems**"
  - Baumgartner didn't trust his safe jumpsuit
- TRUST = by the user or stakeholders
  - Achieving intended Functionality ALLWAYS!
  - Safety & Security & Usability & Privacy & ….ALLWAYS!
  - Meeting non-functional objectives
    - Cost, energy, volume, maintainability, scalability, Manufacturability,..
- User expects guaranteed "QoS" (multi-criteria) from a "Trustworthy system" under all circumstances
- Goal of autonomous = better QoS
- Standards focus on safety? Why?

Two example domains

- **Automotive**:
  - 1,2 millon people killed/year: **daily event**
  - 10 x get hurt for life
  - Cars get better: safer? QoS? Root cause?
- **Aviation**:
  - 500 people killed/year: **a rare event**
  - Planes get better, cheaper, safer, energy-efficient
- Railway, telecommunications, medical, …
  - Similar examples
- **What sets them apart?**

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

- SIL= f (**probability of occurrence, severity, controllability**)
  - Using (system) HARA (Hazard and Risk Analysis)
  - SIL goals ≅ Risk Reduction Factor
- Criteria and classification are open to interpretation

Safety as a goal across domains

Domain					
General (IEC-61508) **Programmable electronics**	(SIL0)	SIL1	SIL2	SIL3	SIL4
Automotive (26262)	ASIL-A	ASIL-B	ASIL-C	ASIL-D	-
Avionics (DO-178/254)	DAL-E	DAL-D	DAL-C	DAL-B	DAL-A
Railway (CENELEC 50126/128/129)	(SIL0)	SIL1	SIL2	SIL3	SIL4

Risk reduction factors depend
on domain and usage pattern!

Detailed analysis reveals **only partial mapping!**

Problems with SIL definition

- Poor harmonization of definition across the different standards bodies which utilize SIL=> Reuse?
- Automative SIL biased by single engine design!
- Process-oriented metrics for derivation of SIL
- SIL level determines architecture (system specific)
- Estimation of SIL based on **reliability estimates**
  - System complexity + based on probabilities that are very hard if not impossible to measure and estimate: reliability of software (discrete domain) is not statistical!:
  - Reliability of software = reliability of the process followed
  - The **law of Murphy still applies**:
    - The next instant can be catastrophic (Concorde as example)

ARRL: trust has hierarchy

- **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

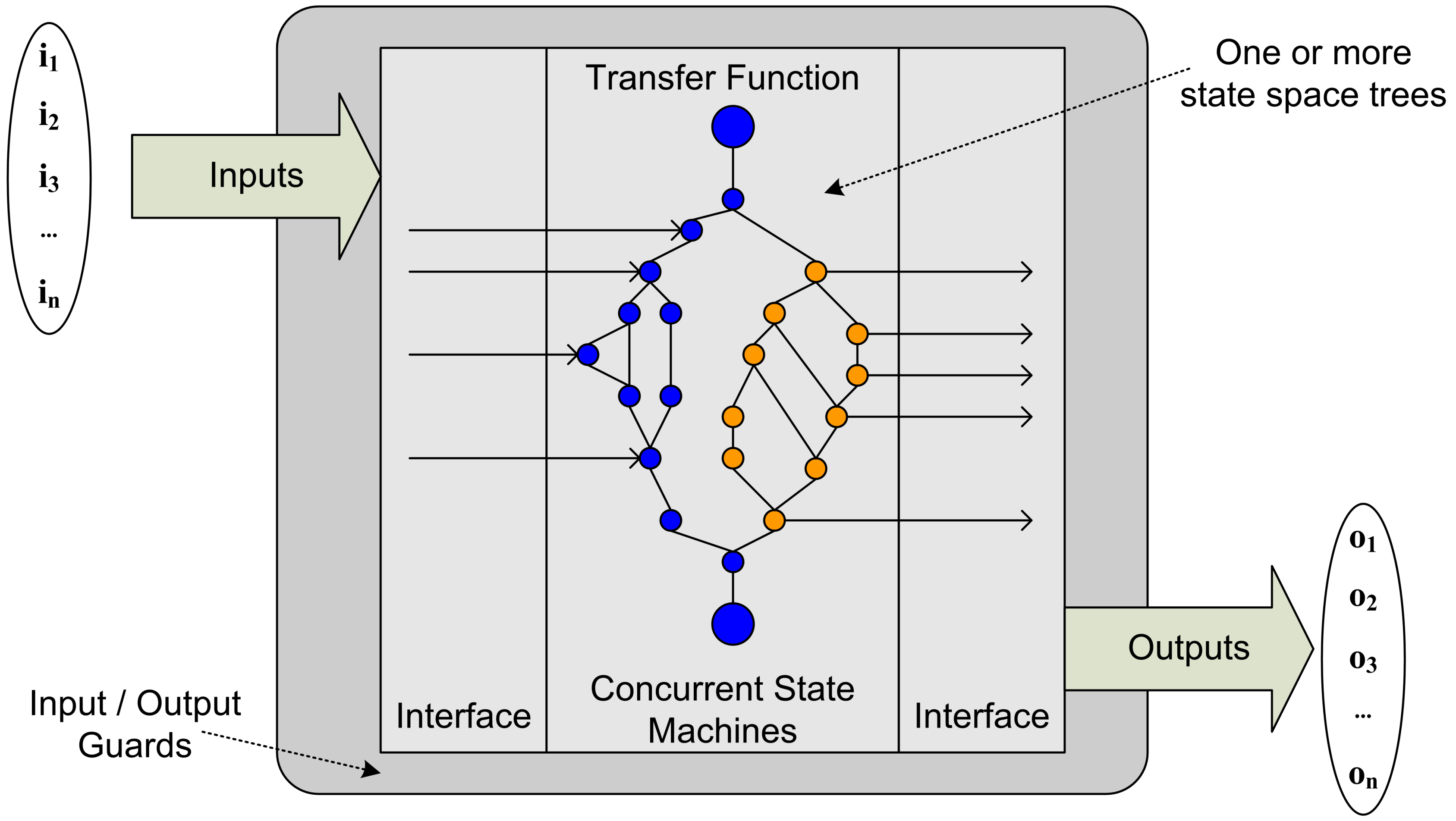
ARRL: what does it mean?

- **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)

Architectural component view (discrete domain)



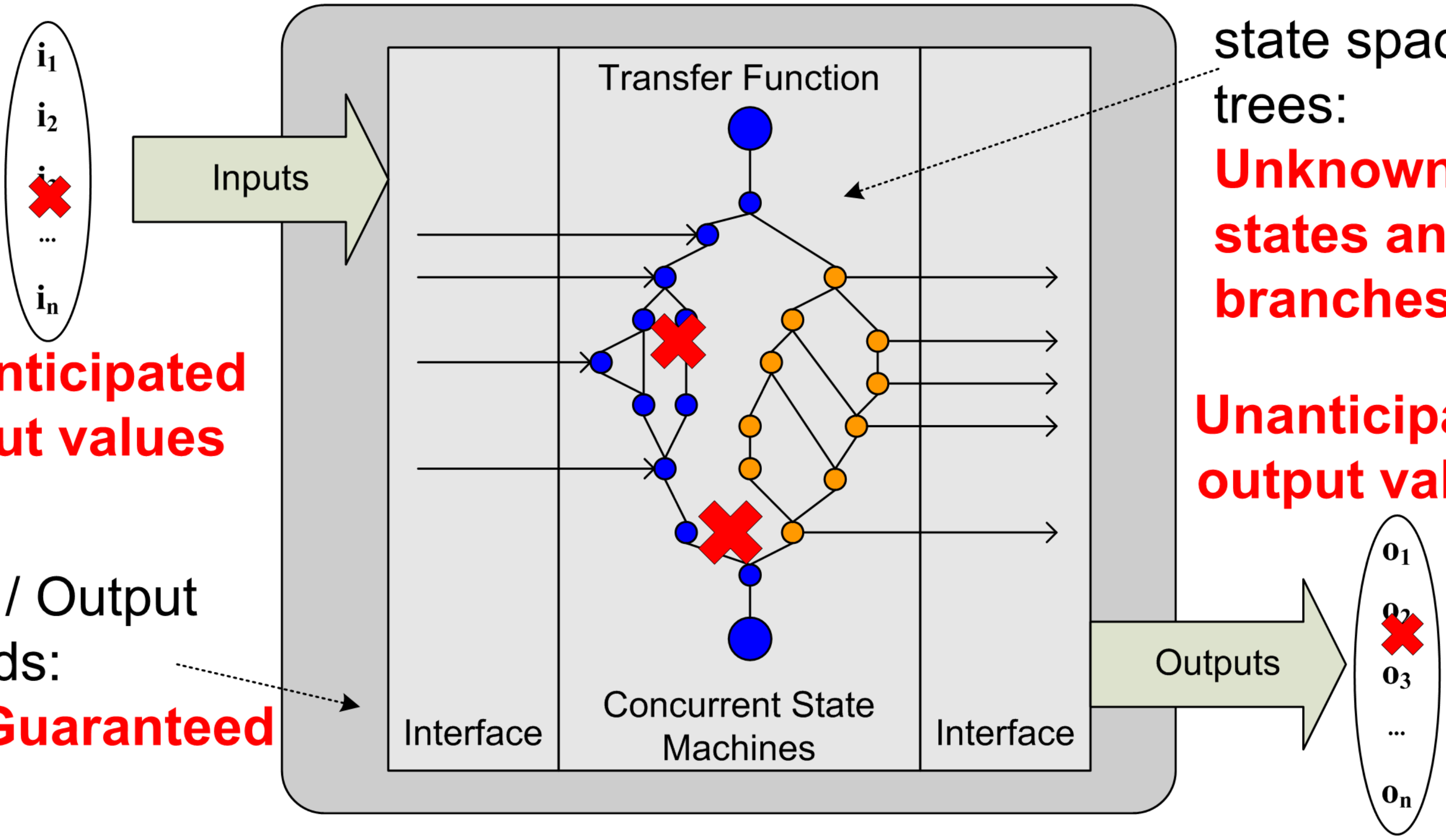
ARRL-1

One or more state space trees:
Unknowns states and branches

Unanticipated input values

Unanticipated output values

Input / Output Guards:
Not Guaranteed



Gaps/Risks due to erroneous specifications and incomplete testing

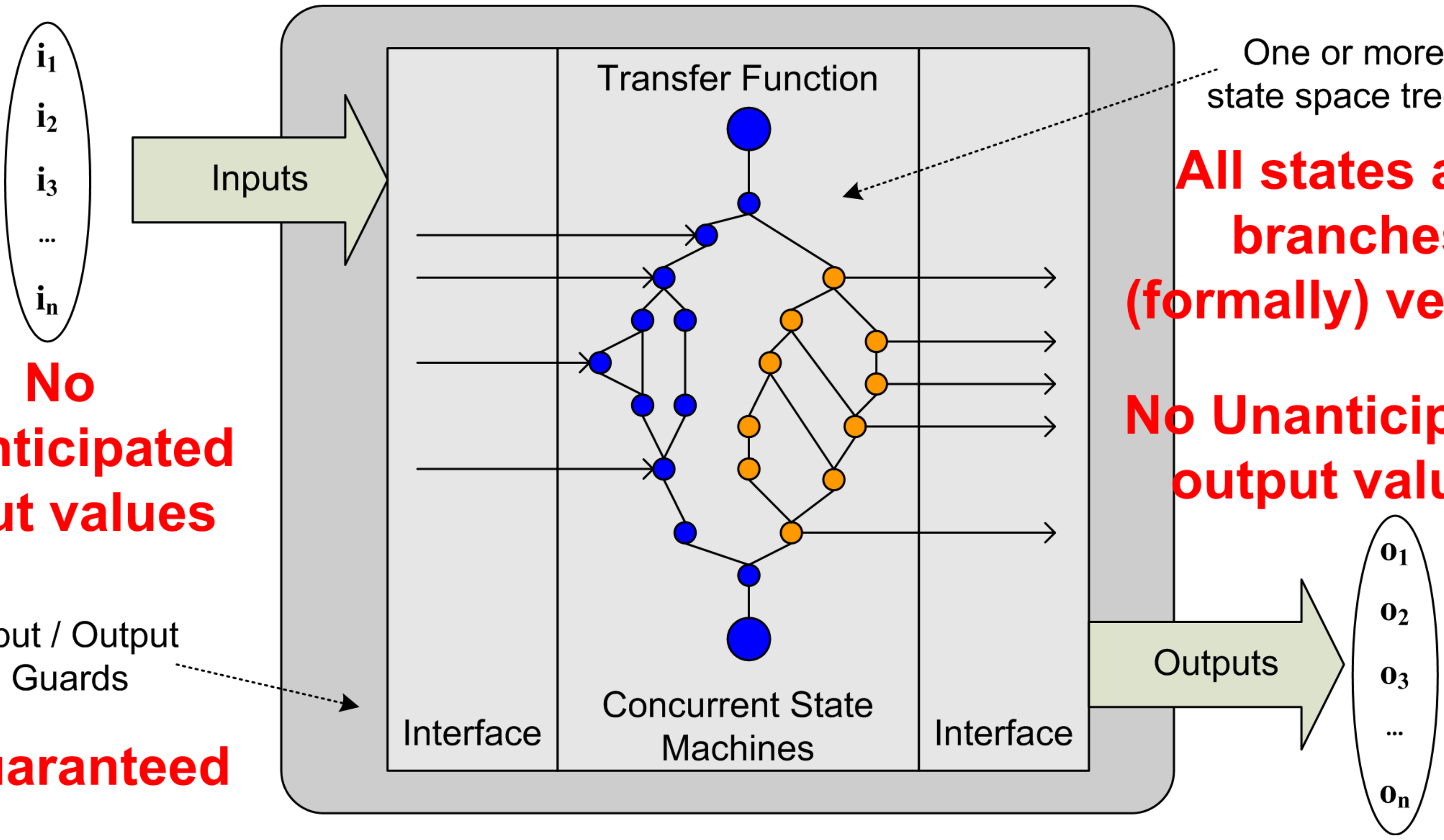
ARRL-0/1

- ARRL-0: "**use as is**"
  - No verified contract: no assurance
  - Still needs a specification
  - Assumes QA at production
- ARRL-1: ARRL-0 + "**works as tested**"
  - Scope of assurance limited to test cases
  - Evidence = verified test reports
  - Absence of errors not assured
- **not really usable for safety critical systems**

ARRL-2

- ARRL-2: ARRL-1 + formal evidence for all specified properties (if no fault): logically correct
- Hardware:
  - Design verification
  - Extensive testing, burn-in, etc.
- Software:
  - Formal evidence:
    - Use of FM, proven in use, …
- Process requirements:
  - Rigorous development, verification, validation, review, …
  - Stress testing to confirm corner cases are handled

ARRL-2



No unanticipated input values

All states and branches (formally) verified

No Unanticipated output values

Guaranteed

Normal Case specifications correct, implementation logically correct

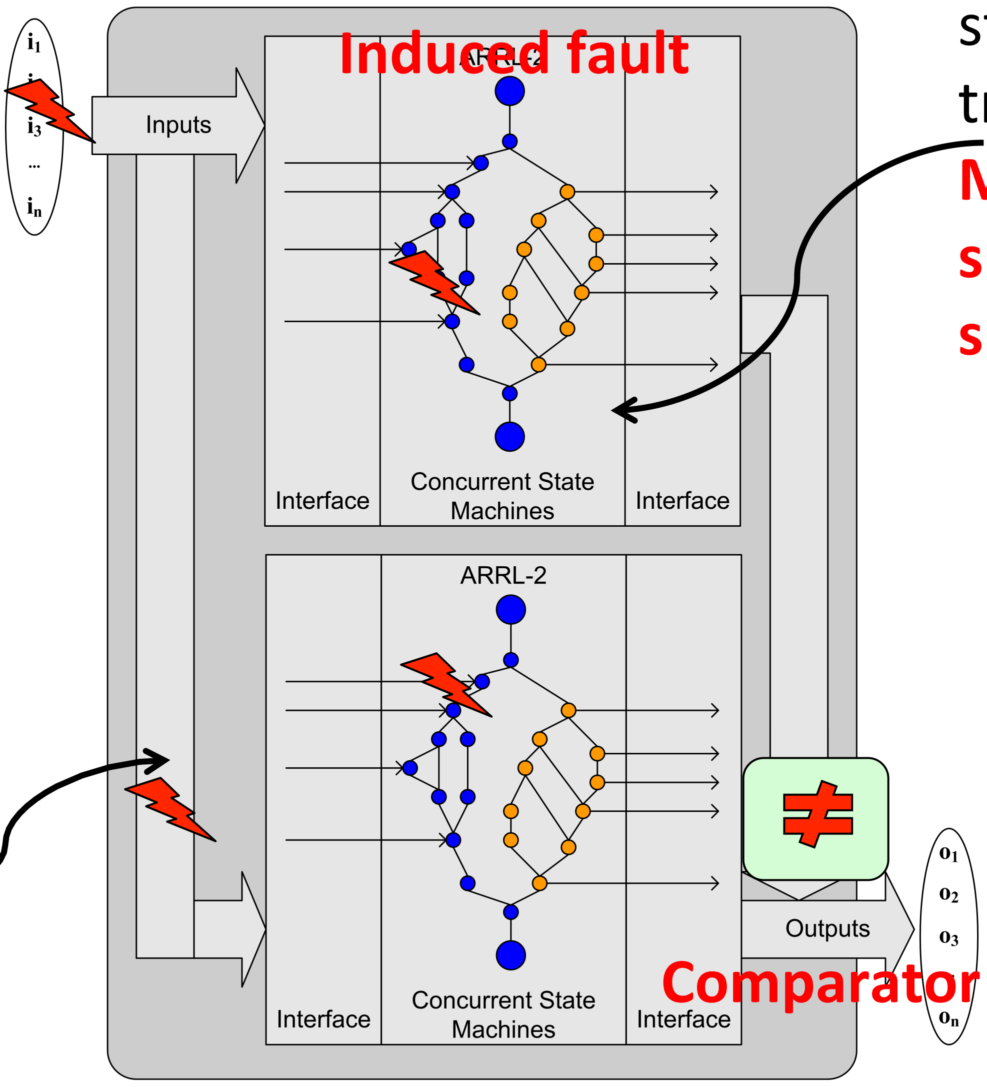
ARRL-3

- ARRL-3: ARRL-2 + fail-safe mode upon fault
- All possible fault cases are part of specification
- Fault behavior predictable upon fault
- Fault: at micro-level (bit level state)
- Features:
  - Monitoring and redundancy for degraded mode
  - Prevent error propagation, incl. externally
  - Isolate fault area internally
  - Easier with modular architecture
  - Keeps correct functionality if possible
  - HW/SW co-design

ARRL-3

Unanticipated input values

One or more state space trees:
Monitor and supervisor sub-component



Input/output guards:
Guaranteed bounded

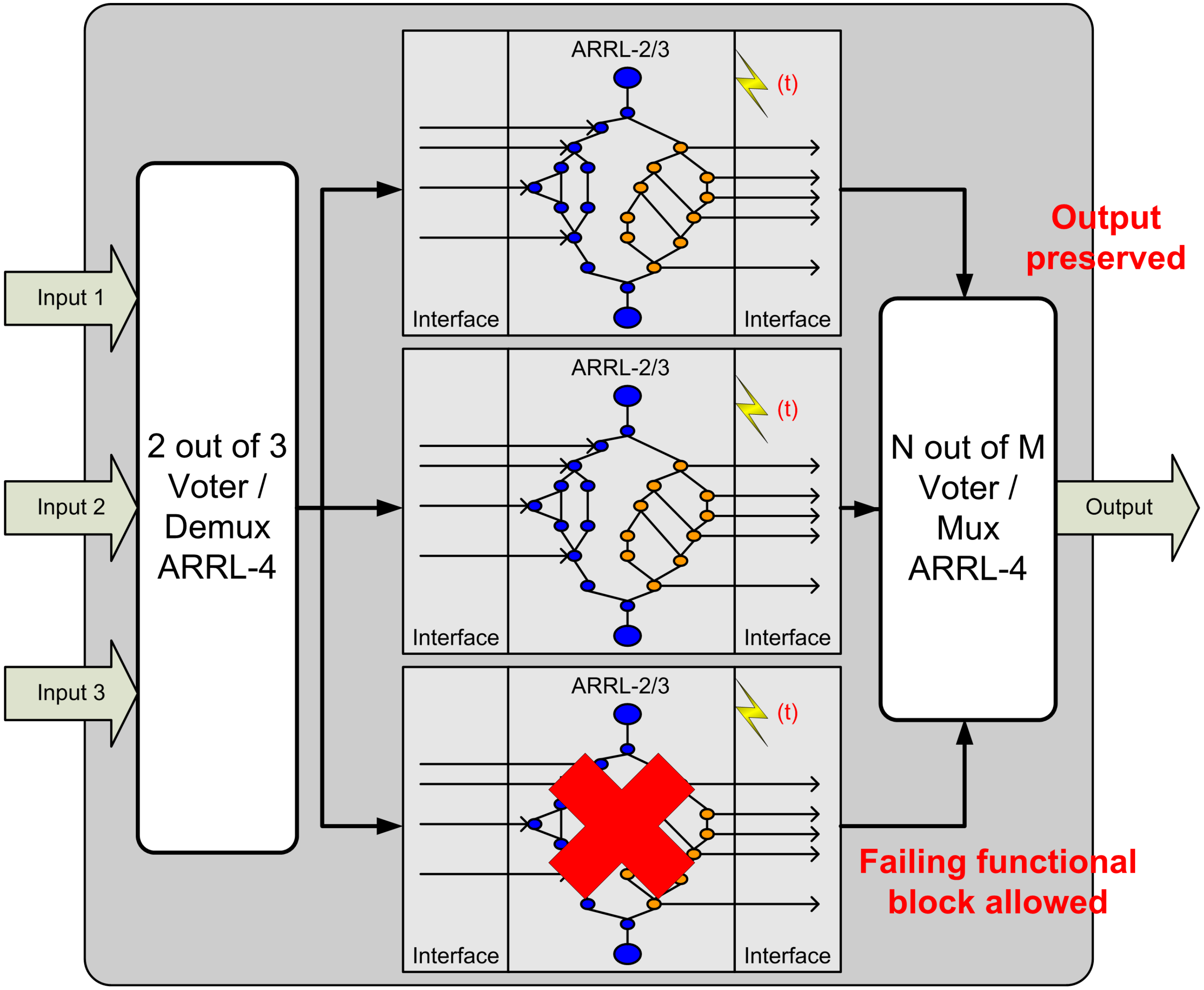
Fail safe output (can be NULL

Common mode failures possible

ARRL-4

- ARRL4: ARRL-3 + fault tolerance
- Fault: at macro-level (functional block)
  - What is the unit of failure?
- Requires macro-level redundancy + voting
- Interconnect needs to be ARRL-4 as well

ARRL-4


Input 1
Input 2
Input 3
2 out of 3 Voter / Demux ARRL-4
ARRL-2/3
(t)
Interface
Interface
ARRL-2/3
(t)
Interface
Interface
ARRL-2/3
(t)
Interface
Interface
N out of M Voter / Mux ARRL-4
Output
Output preserved
Failing functional block allowed

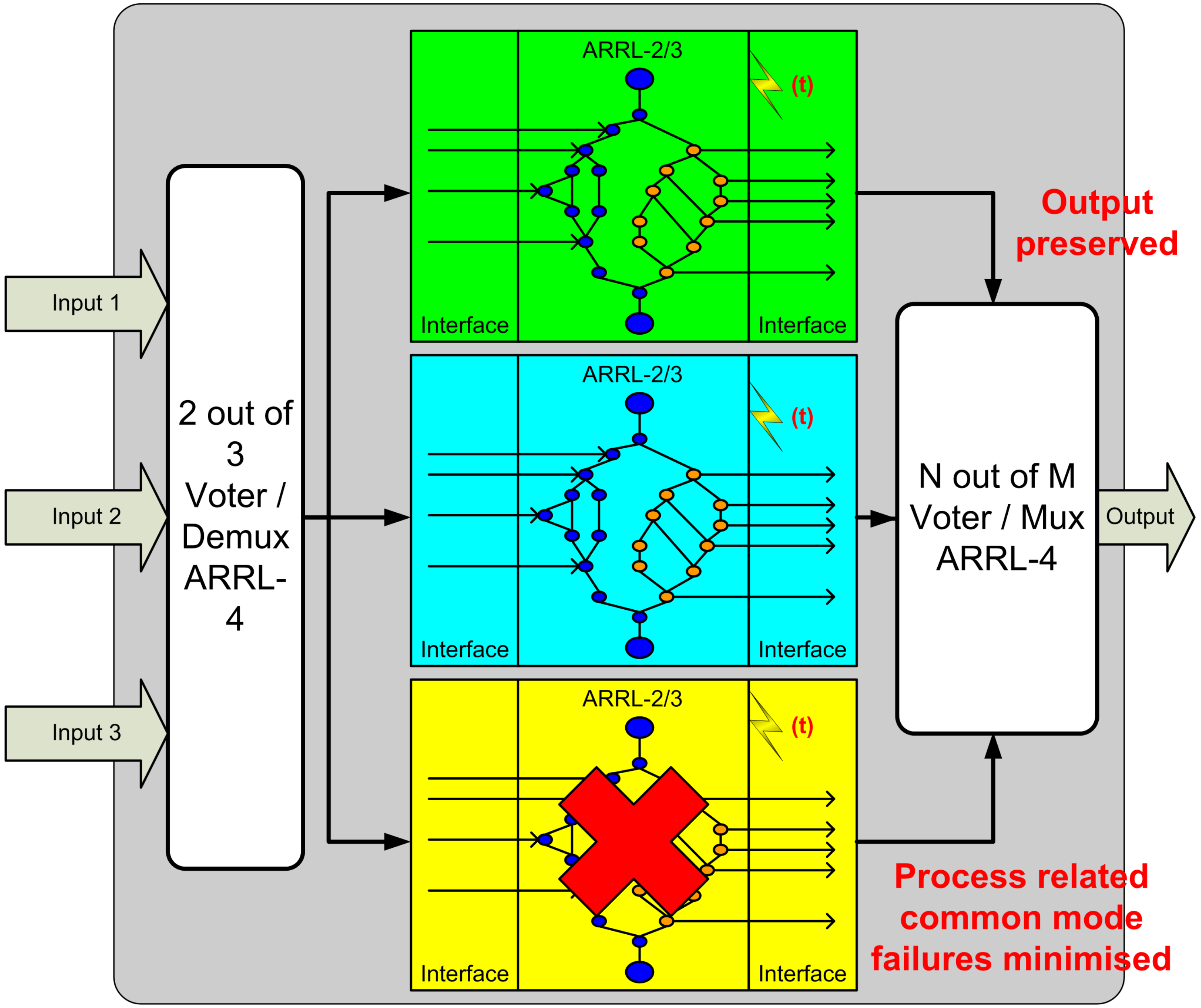
Common mode failures => ARRL-5

- ARRL-4 assumes independence of faults in each redundant channel
- Covers only a subset of the common mode failures
- Less visible are e.g. common misunderstanding of requirements, translation tool errors, time dependent faults => require asynchronous operation and **diversity/heterogeneous** solutions
- Hence we can define an ARRL-5 as well

ARRL-5

- ARRL5: ARRL-4 + design diversity
- Focus is on common mode failure at design level
- Requires rigorous interface specification
- Best use asynchronous interactions
- Can still affect real-time capabilities

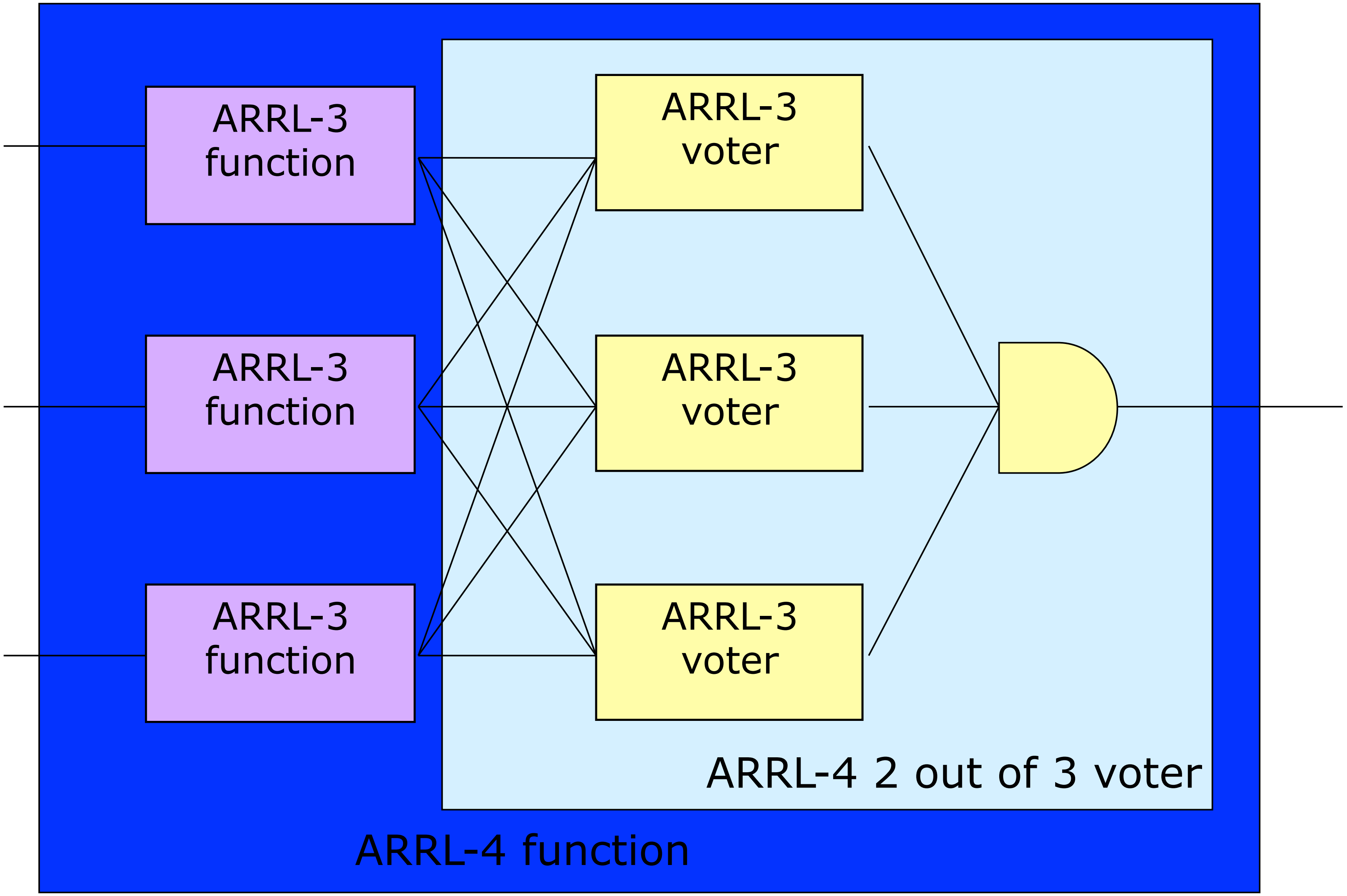
ARRL-5


Input 1
Input 2
Input 3
2 out of 3 Voter / Demux ARRL-4
ARRL-2/3
(t)
Interface
Interface
ARRL-2/3
(t)
Interface
Interface
ARRL-2/3
(t)
Interface
Interface
N out of M Voter / Mux ARRL-4
Output
Output preserved
Process related common mode failures minimised

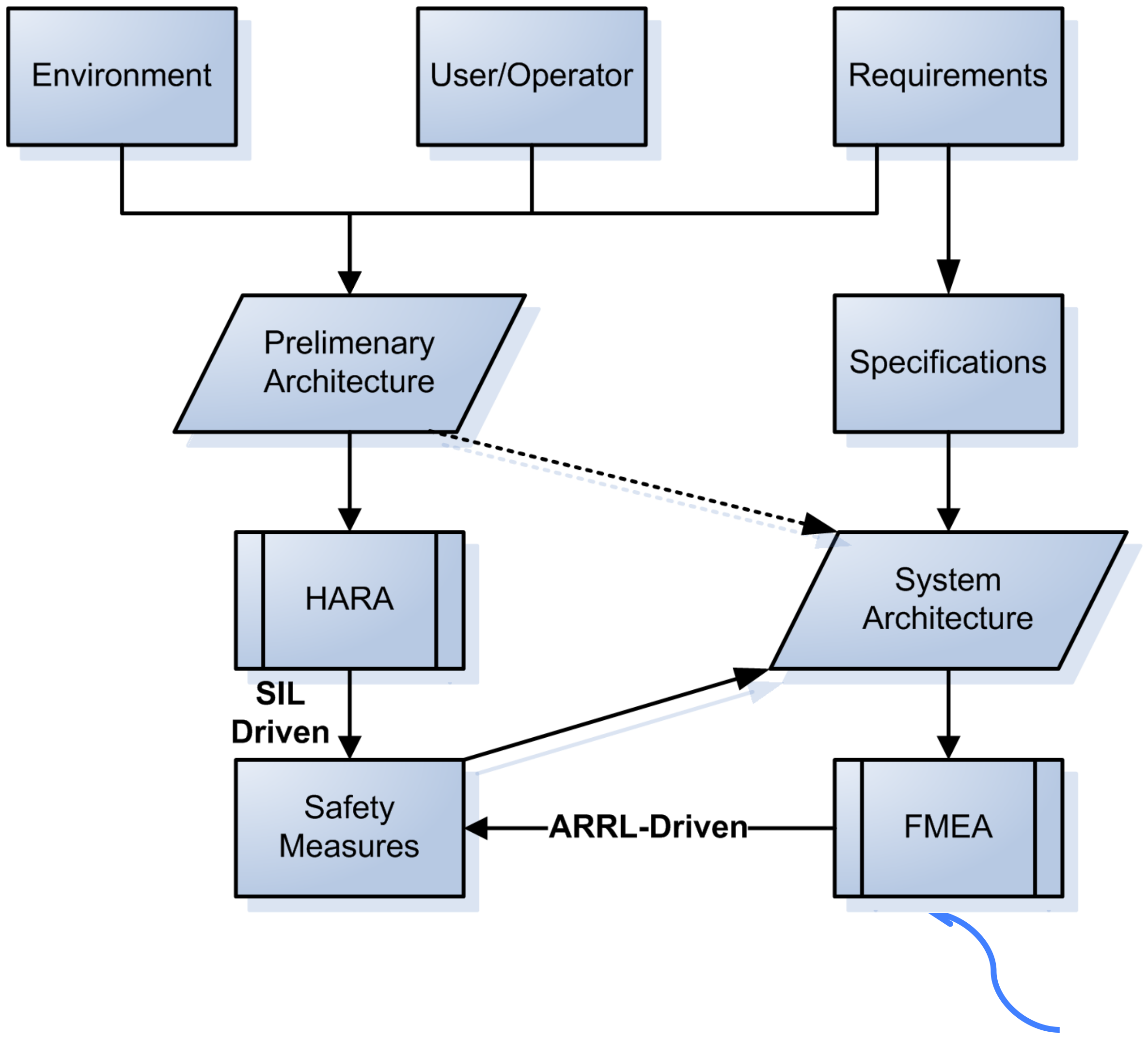
Composition rule:

- **A system can only reach a certain SIL level if all it components are at least of the same ARRL level.**
  - This is a necessary condition, not a sufficient condition
  - Redundancy can compose ARRL 4 components out of ARRL 3 components (needs an ARRL 4 voter)
  - ARRL3 component can use ARRL 2 components (>2)
  - In line with architectural recommendations based on SIL levels
- Consequences:
  - Interfaces and interactions also need ARRL level!
  - Error propagation is to be prevented => partitioning architecture (e.g. distributed, concurrent)
  - Using ARRL-3/4 components means that the system becomes resilient: run-away situations leading to critical states are contained.

Generic example



SIL and ARRL are complementary



SIL driven

ARRL driven

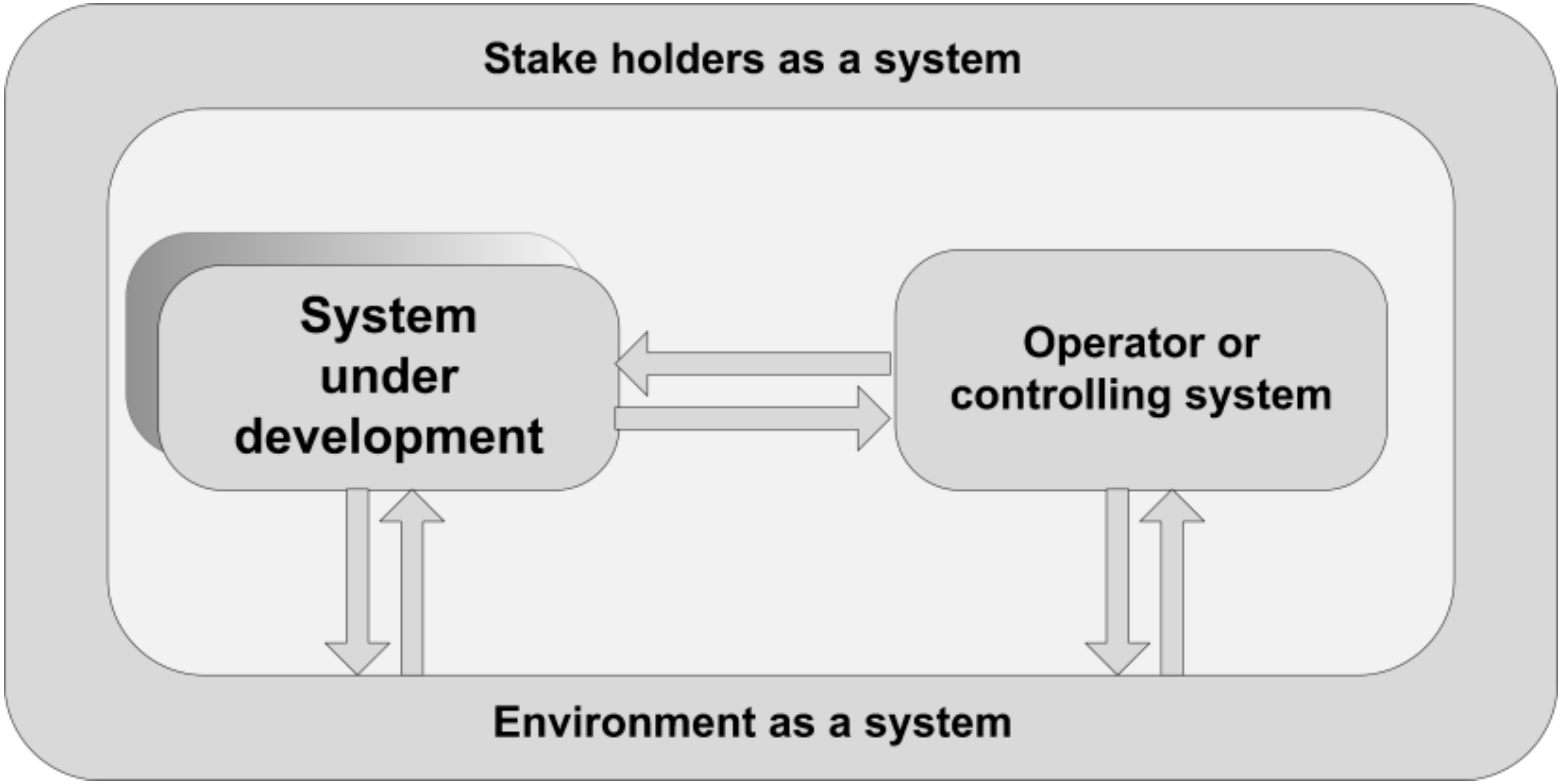
Role of Formal models

- Formal evidence is wide:
  - Use of formal models at design time
  - Use of formal verification post implementation
  - Evidence of rigorous process
    - Document - Test – Verify – Validate – Review – Confirm - …
  - Proven in use (weaker argument)
  - Stress testing (weaker argument)
- Formal methods increase confidence
- GoedelWorks follows a formal "meta"-model

From preprogrammed to autonomous

- ARRL-0 to 5 covers pre-programmed system:
  - Assumes complete knowledge
  - Assumes a static system (design once)
  - And hope for the best
  - Still rather expensive: price for trust
- Can we know the future?
- What if the assumptions were wrong?
  - Real-world testing: people can die or get hurt

A system is never alone



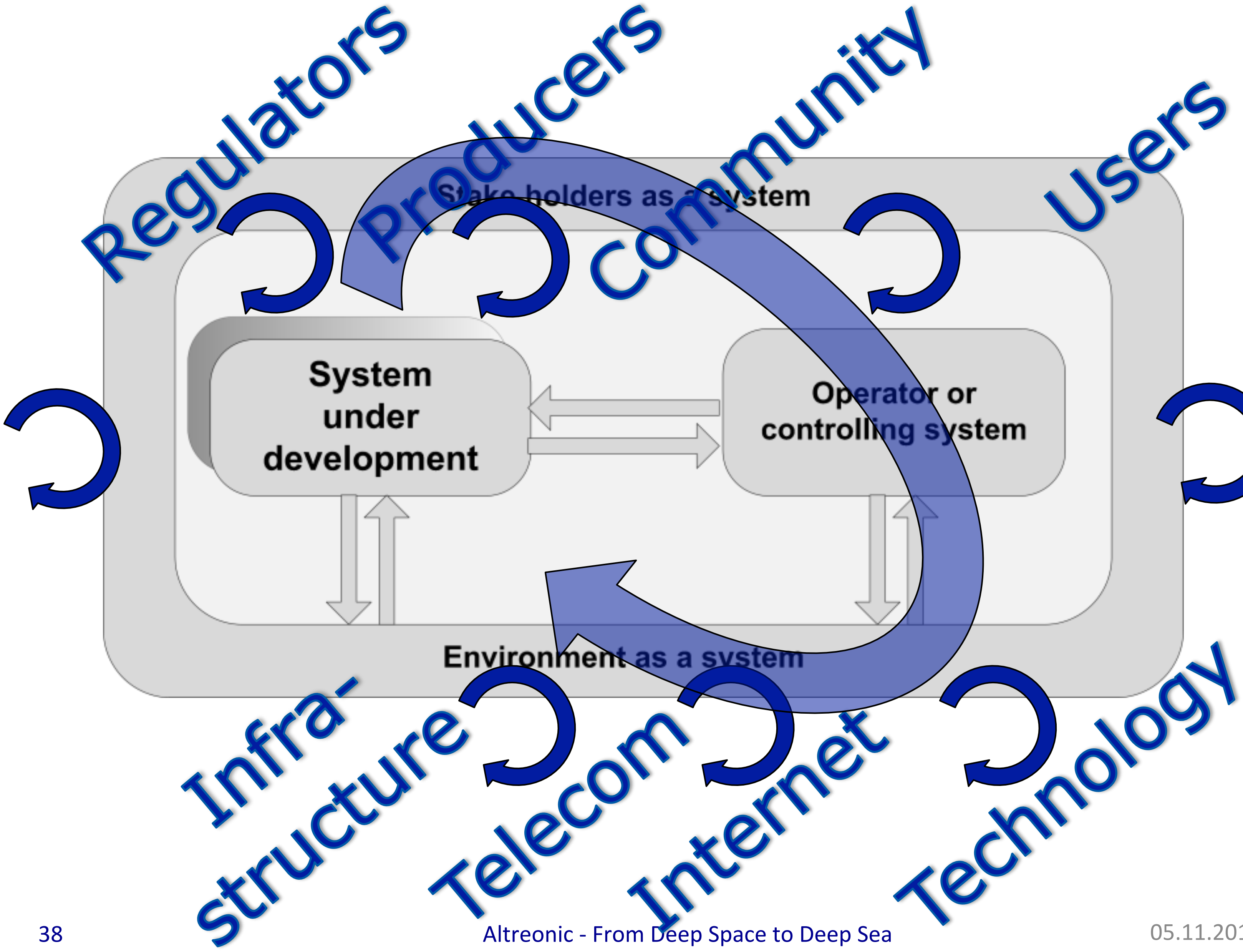
What if:

- The environment is not fully predictable?
- The operator is not fully predictable?

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” events)
- **What does it mean** in the context of safety/ systems engineering?
- Isn’t ARRL-5 the top level?

Extended systems (of systems) view



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

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**

Autonomous traffic

- Self-driving cars are the future?
- Systems engineering challenge much higher than flying airplanes (50 millisec vs. 2 minutes)
- 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 at least ARRL-7

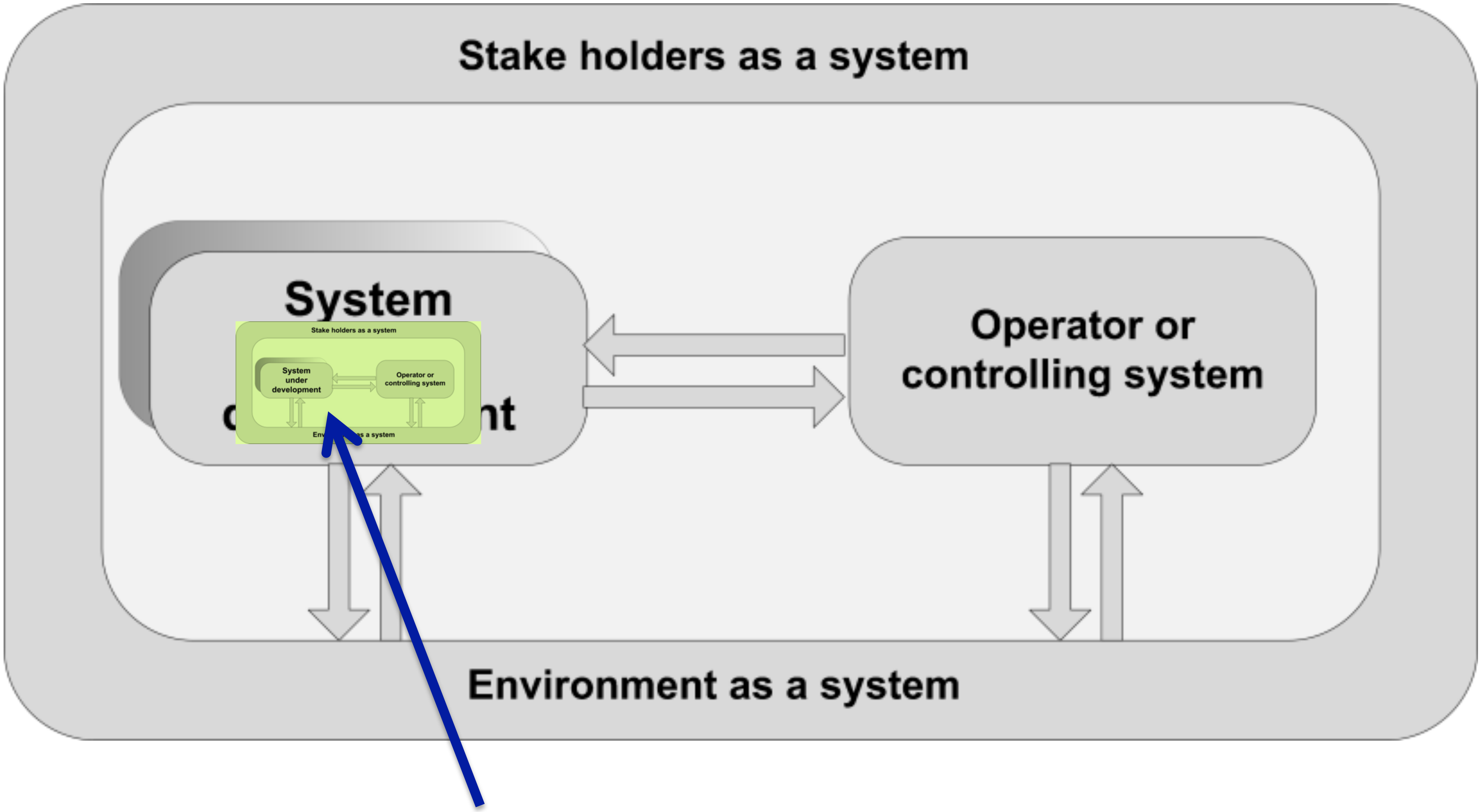
SAE automated vehicle classifications:

- **Level 0**: Automated system has no vehicle control, but may issue warnings.
- **Level 1**: Driver must be ready to take control at any time. Automated system may include features such as Adaptive Cruise Control (ACC), Parking Assistance with automated steering, and Lane Keeping Assistance (LKA) Type II in any combination.
- **Level 2**: The driver is obliged to detect objects and events and respond if the automated system fails to respond properly. The automated system executes accelerating, braking, and steering. The automated system can deactivate immediately upon takeover by the driver.

_____________________ **this is where we are today** _________________________________

- **Level 3**: Within known, limited (?) environments (e.g. freeways), the drivers can safely turn their attention away from driving tasks. => *always? For how long?*
- **Level 4**: The automated system can control the vehicle in all but a few environments such as severe weather. The driver must enable the automated system only when it is safe to do so. When enabled, driver attention is not required. => *how to keep the driver trained?*
- **Level 5**: Other than setting the destination and starting the system, no human intervention is required. The automatic system can drive to any location where it is legal to drive. => *define: legal to drive*

Autonomous system

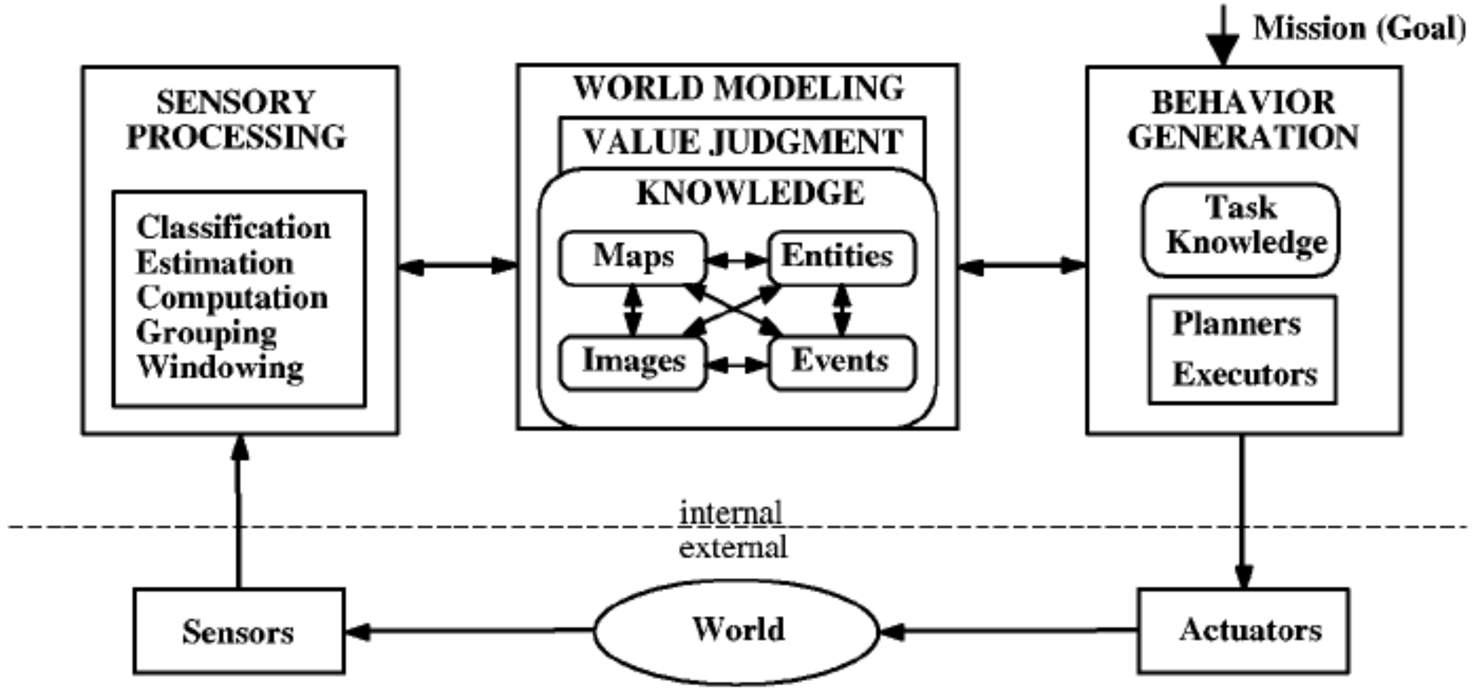

Stake holders as a system
System
Operator or controlling system
Environment as a system
Stake holders as a system
System under development
Operator or controlling system

System contains a model of itself and its environment: must also be ARRL-7

Beyond ARRL-7

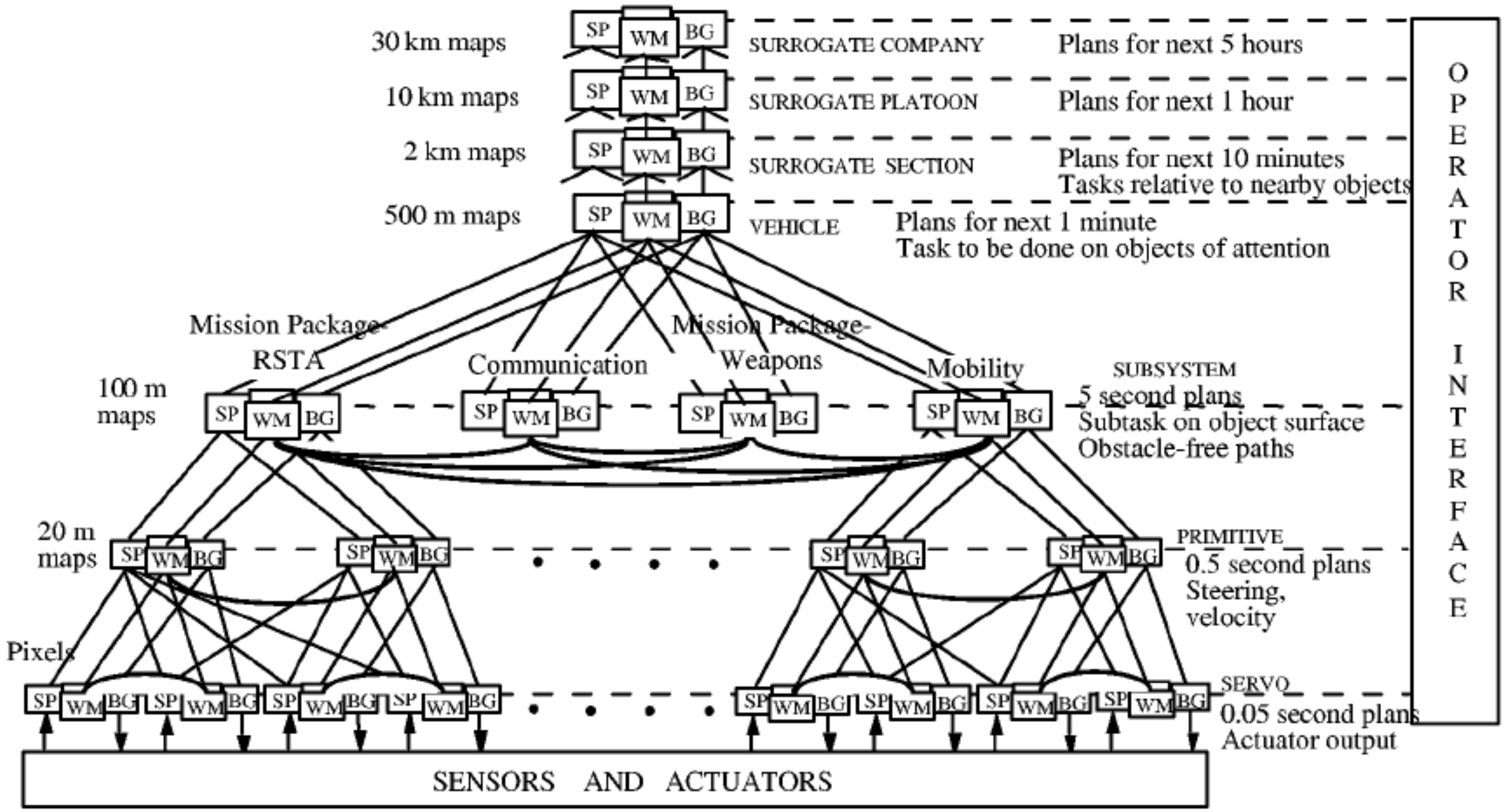
- Not all systems are engineered by humans
- Biological systems:
  - Survivability (selection) and adaption
  - Build-in mechanism (long term feedback loops)
  - ARRL-8 ? => autonomous + self-adaptive systems
  - Inheritance of ARRL-7 ?
- Genetic engineering:
  - Directed selection and adaptation
  - ARRL-9? Or ARRL-7 with bio-components?

Inspiration from DARPA (2002)



- 4D/RCS a reference model architecture for unmanned vehicle systems (NIST)
- RCS: Real-time Control System

Planning: from 5 hours to 50 millisec



Is AI needed?

- AI (or Knowledge Based Systems):
  - Software with symbolic and abstract entities
  - Rule based systems
  - Idea: emulate human thinking
  - Infinite space. Not really verifiable (ARRL-1)
- AI = the way some people think, humans think
- Should computers behave like humans?
- Human perception: sensors + preprocessing, "hard-wired" reactions, associative look-ups
- Homo economicus vs. safe robot?
- Should the computer be ethical (cfr. Asimov)

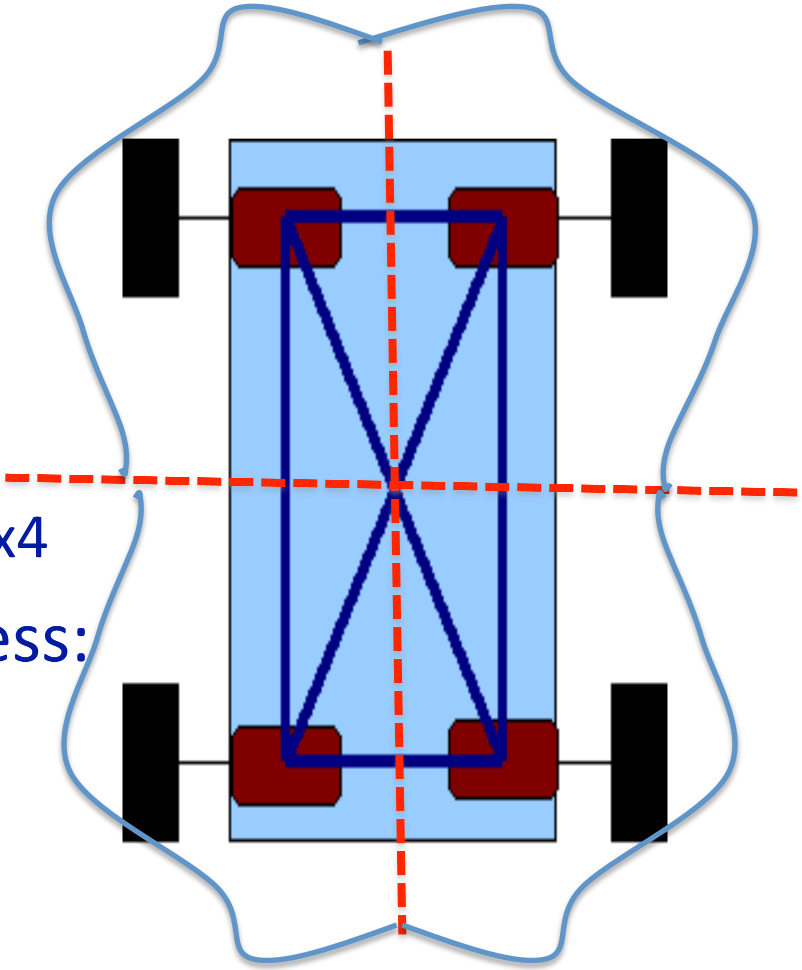
Once we get the SW right, shrink the HW

A trunk full

100 x 160 x 50 mm

KURT: Modular and Redundant architecture

- Combine reusable units
  - Economy of scale (COG!)
  - Redundancy (fault tolerant)
- Propulsion Unit =
  - Battery (LiOn, MNH, …) + motor
  - Suspension + wheels
  - Vectoring steer/drive by wire/4x4
- Smart environmental awareness:
  - Obstacle detection/avoidance
  - Assisted auto-navigation
- International patent



The future is nice for engineers

- http://uk.businessinsider.com/amazing-elon-musk-believes-will-happen-future-tesla-spacex-2016-6?r=US&IR=T
- 7 mind-blowing things Elon Musk believes
- Elon Musk: a disruptive mobility pioneer:
  - TESLA: an autonomous and electric sports limo
  - Hyperloop: "trains" at Mach 1
  - Space-X: cheap transport to Mars
- Trustworthy Engineering makes it happen!

All other matters

- Questions
- eric.verhulst@altreonic.com
- tuur.benoit@siemens.com (the organiser!)
- Thanks for listening