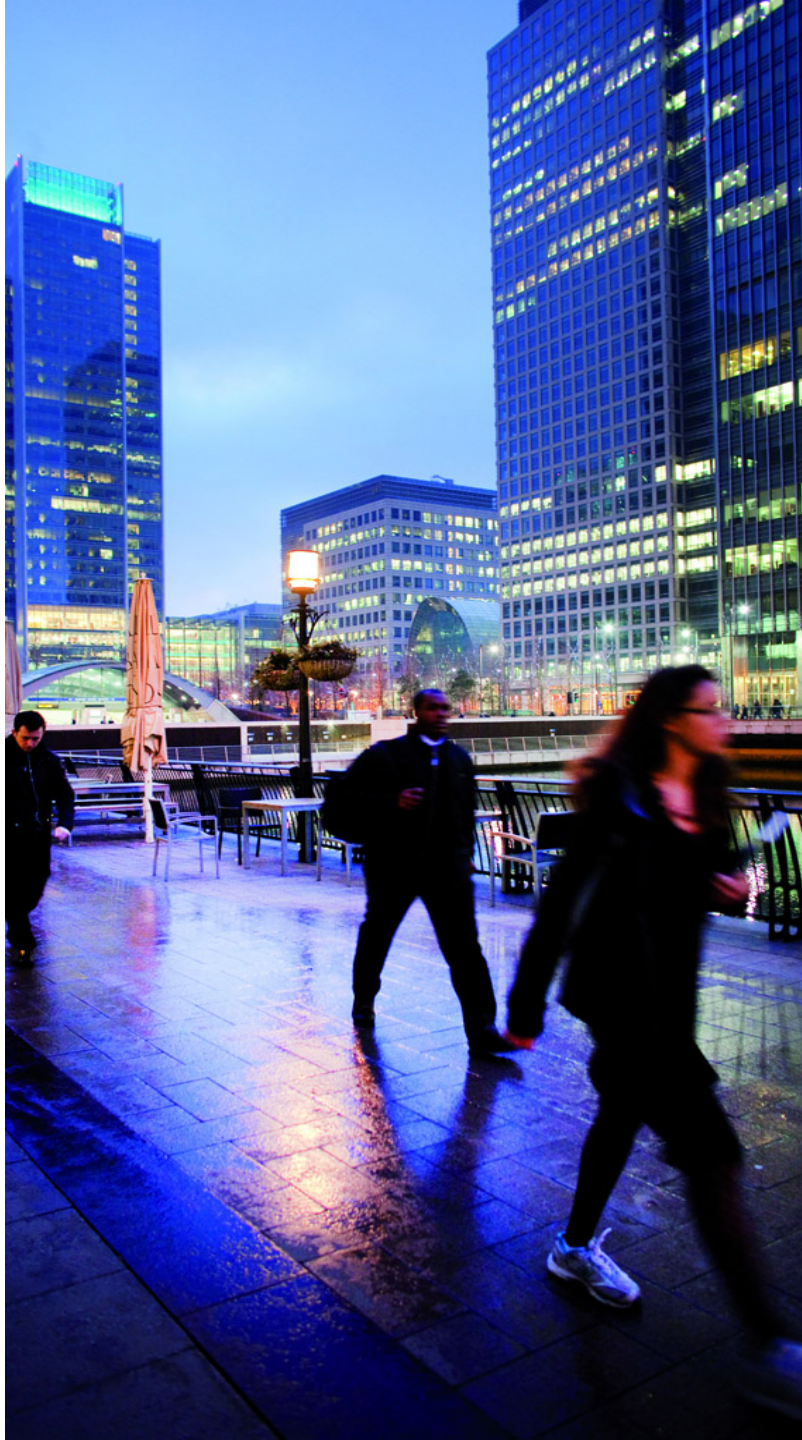
Defects in Requirements Specification: The Landing Gear System (LGS)

Cyber-Physical and Socio-Technical Systems are Different from Software-Based Systems

Thuy NGUYEN
thuy.apt[at]orange.fr

GDR GPL - IE
June 5-9, 2023
Rennes, France

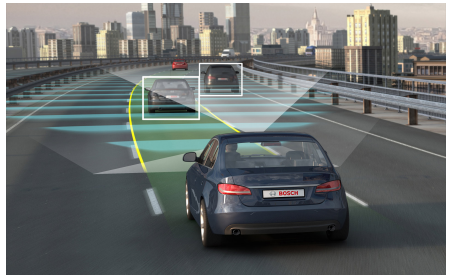


Who Am I? * *Vannes 2022*

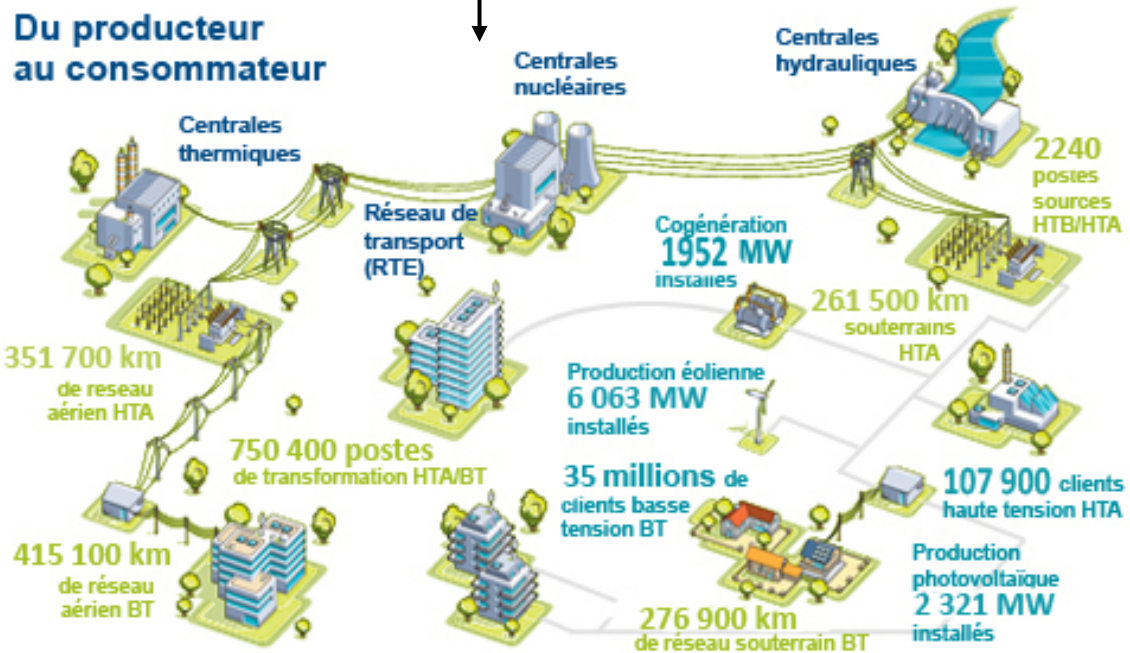
- **1975 - 1994: Software engineer and architect in the general software industry**
  - Signal acquisition & processing
  - Programming languages, compilers & interpreters
  - Computer graphics, computer-aided industrial drawing, mechanical CAD-CAM
  - Real-time, distributed digital systems
  - File & database management systems
  - Software engineering
- **1994 - 2021: Research engineer at EDF for Instrumentation & Control (I&C) systems important to power plant safety**
  - Since 1994: formal verification (complete I&C system software, and I&C system architectures)
  - Since 1999: FPGAs (Field Programmable Gate Arrays) for safety applications
  - Since 2007: simulation assisted engineering for cyber-physical systems, socio-technical systems and systems of systems
  - Since 2016: NUWARD I&C architect
    - The small modular reactor (SMR) co-developed by EDF, CEA, Technicatome and Naval Group
- **Since June 2021: Retired**
  - But still active, with the IAEA, IMT Atlantique and IRIT

Cyber - Physical Systems (CPS), Socio - Technical Systems (STS) *

- **Computation & networking**
- **Physical processes, physical proximity, physical connections, ...**
- **Human and organizational aspects**



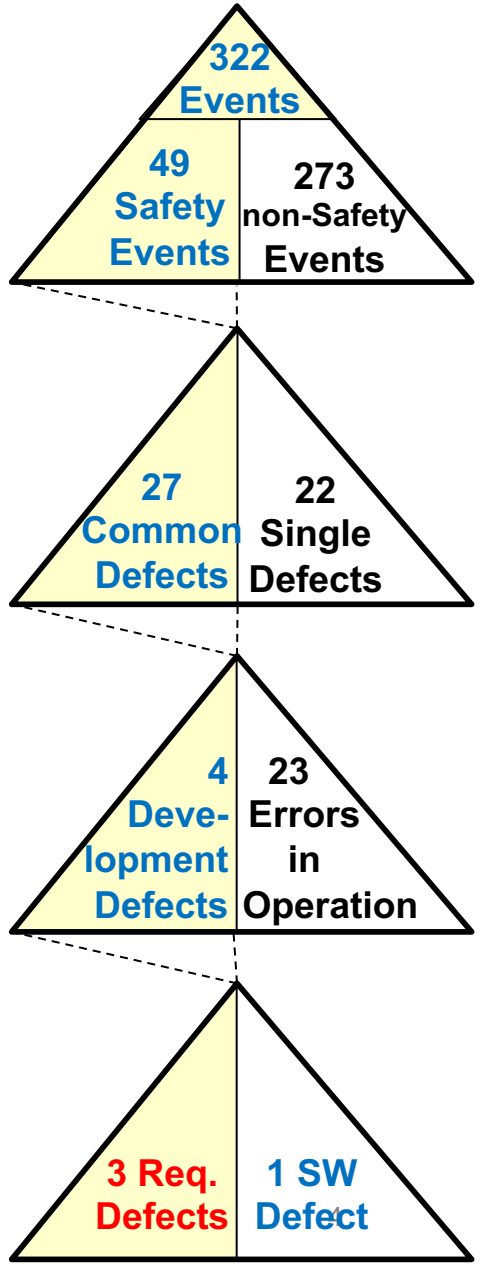
Systems of Systems (SoS)



Cyber and software aspects need to be addressed in the framework of human and physical aspects

Experience with Nuclear Power Plants Digital Safety Systems

- **2007: Study of actual and potential common-cause failures of digital safety systems in US nuclear power plants (1987-2007)**
  - EPRI 1016731 *Operating Experience Insights on CCF in Digital I&C Systems* (December 2008)



Non-software errors or defects played a much greater role than software defects, and need to be better taken into account

Design-related events were mainly due to issues at the (plant system + human) - (digital system + software) boundary, where communication between different engineering disciplines is often difficult and inadequate

Experience with Nuclear Power Plants Digital Safety Systems

- **EPRI 3002005385 *Severe Nuclear Accidents: Lessons Learned for I&C and HF* (December 2015)**

  Instrumentation & Control (I&C) or Human Factors (HF) issues contributed to every severe accident identified in this study, but failures of I&C components seldom contributed to severe accidents. Instead, the main contributors were found to be the following:

  - The sensitivity, range, or response time of a measurement indicator or the display was inadequate
  - A needed instrumentation or display function was not included in the design
  - Inadequate display characteristics were provided, e.g., range or location
  - I&C support systems failed
  - The form or location of the measurements did not give the intended information
  - Instruments were incorrectly calibrated

  Such issues typically result from incomplete or incorrect I&C system requirements. Complete, correct, and clear design requirements are essential to reducing the potential for accidents and improving the operators' ability to respond to accidents if they occur.

- **OECD/NEA *COMPSIS (COMPuter-based Systems Important to Safety) Final Report* (July 2012)**

  Weaknesses in requirements are one of the most significant contributors to systems and software failing to meet the intended goals. A better analysis is needed to understand the software's interfaces with the rest of the system and discrepancies between the documented requirements for a correct functioning system.

- **The civil aviation industry, with much larger experience in operation, has similar conclusions**

Defects in Specified Requirements *

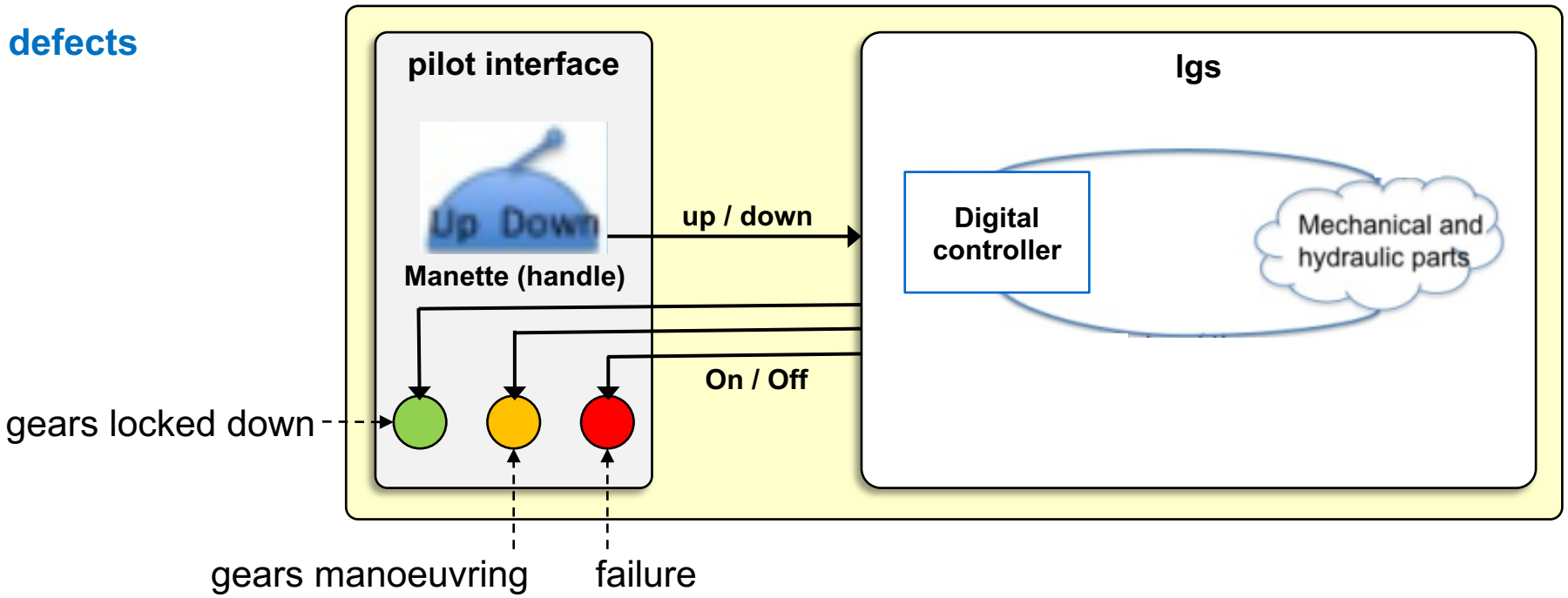
- **Inadequacy**
  - Where, in some situations, what is specified is woefully inappropriate and could lead to unacceptable consequences
  - Or where what is necessary in some situations is not specified (silence), which could also lead to unacceptable consequences
- **Ambiguity**
  - Where different people concerned could understand what is specified differently or when some necessary aspects of a requirements (e.g., margins) are left to interpretation, which could also lead to unacceptable consequences
  - Syntactic ambiguity, lexical ambiguity, value ambiguity, ...
- **Apathy**
  - Where what is specified makes no difference between what is genuinely needed and what is barely tolerated in exceptional situations
- **Over-ambition**
  - Where what is specified might be interesting but is not essential and could lead to excessive complexity, higher costs, longer delays and greater risks of errors (in design, construction, operation and / or maintenance), with possibly unacceptable consequences
- **Over-specification**
  - Where what is specified is not the problem but a technical solution, not necessarily the best and simplest, and worse, not necessarily fully solving the real problem
- **Intangibility**
  - Where what is specified is based on immaterial, abstract concepts, with no concrete, verifiable acceptance criteria (wishful thinking)
- **Infeasibility**
  - Where what is specified is not achievable
  - E.g., but not only when satisfaction of some requirements necessarily implies violation of others (contradiction)

Current Requirements Engineering Methods are Unsatisfactory for CPS-STS

- **Most were developed for software engineering, and do not take sufficient account of differences between software and CPS-STS**
- **Many are mere requirements management methods, addressing form but not substance**
  - Addressing only partially a severe defect in requirements: ambiguity
  - Not addressing the most severe one: inadequacy
  - Not addressing the others either
- **Requirements for large CPS-STS are as complex as a large piece of software**
  - No one trusts software that has never been tested or formally verified
  - That complexity is compounded with the number of teams, engineering disciplines, organizations and stakeholders
    - Which more often than not have difficulties understanding one another, with different cultures, term definitions, methods, tools, and sometimes languages
  - Defects tend to be detected very late in system development, or worse, during operation, with sometimes catastrophic and wholly unacceptable consequences
- **But very little attention is devoted to the verification of the substance of requirements**

The Landing Gear System (LGS) - Introduction

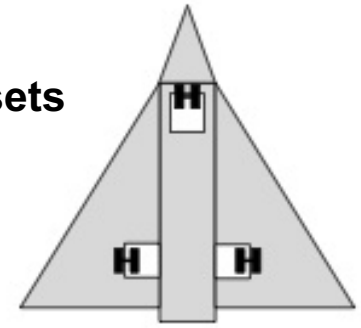
- **Benchmark for techniques and tools dedicated to the verification of behavioural properties of systems and software**
  - From Frédéric Boniol and Virginie Wiels (ONERA, Toulouse)
- **Some of the LGS requirements, and more detailed requirements of its control software, in natural language**
- **Challenge: translate them into formal requirements**
- **But there are some defects**



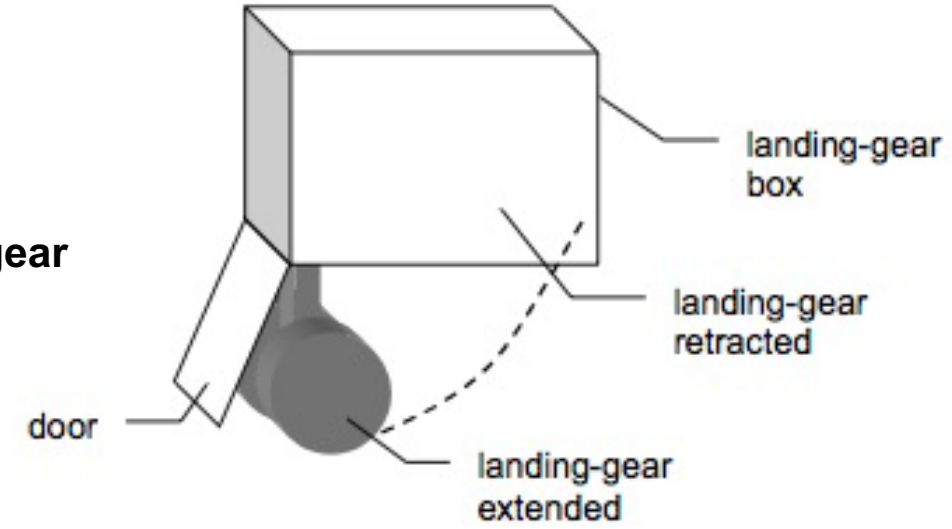
LGS Structure and Manoeuvring

- **Mission: manoeuvring 3 landing sets**
  - Front
  - Right
  - Left

  } identical



- **Each set is composed of a door and a retractable landing-gear**



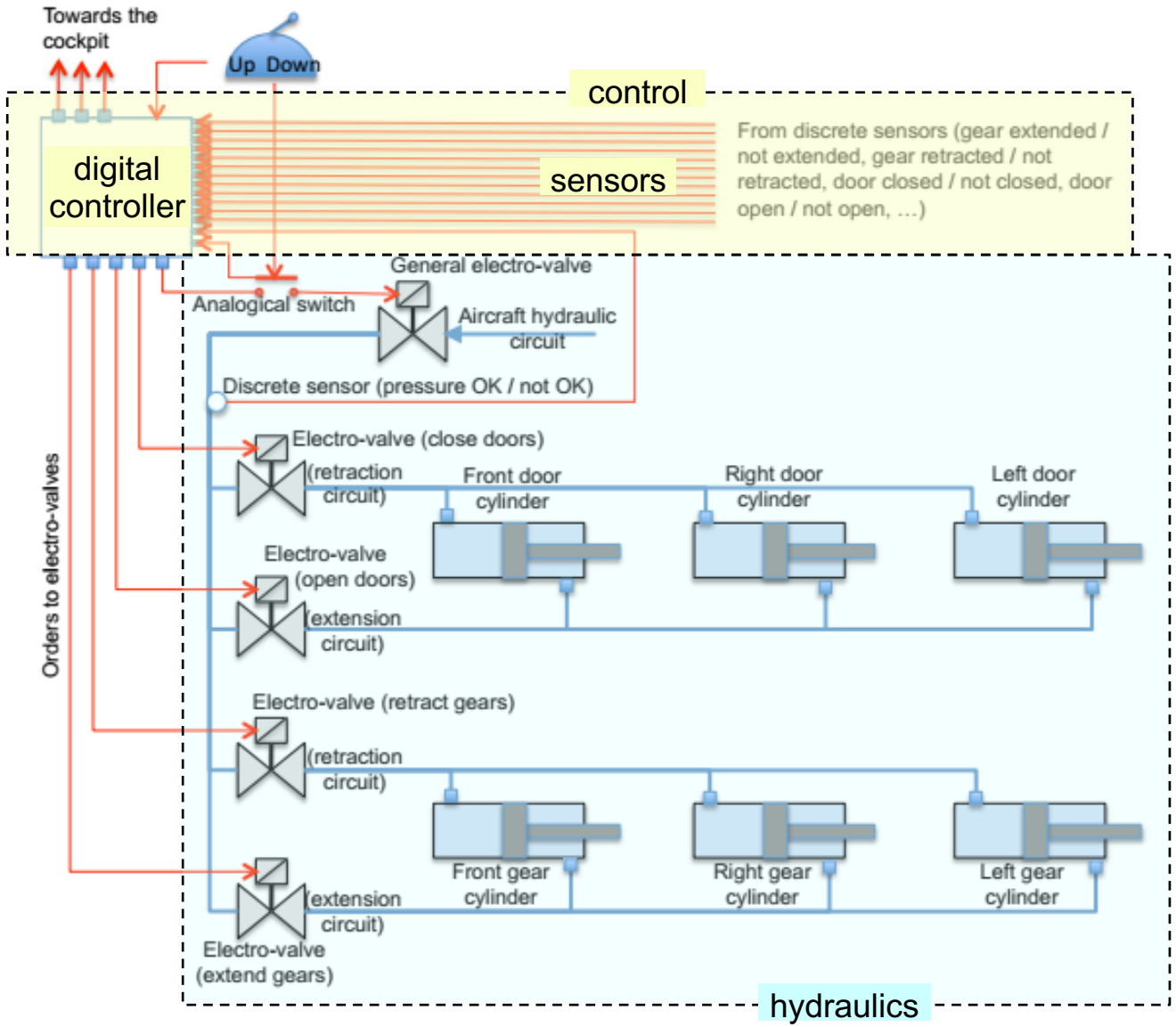
- **Landing sequence**
  - Open the door
  - Extend the landing-gear
  - Close the door
- **Retraction sequence**
  - Open the door
  - Retract the landing-gear
  - Close the door
- **Digital control in normal mode, analogical control in emergency mode**

Not in the case study

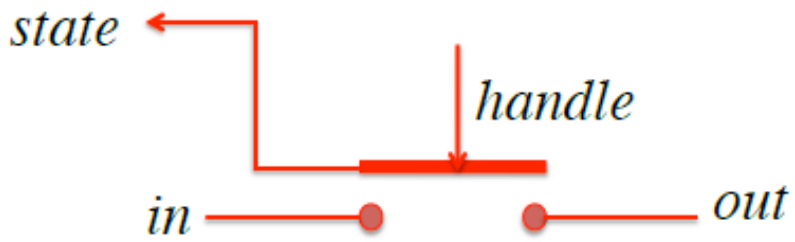
Additional mission: health monitoring
(part of the case study)

- **Challenges**
  - *model and program the software part controlling the landing and the retraction sequence*
  - *prove safety requirements taking into account the physical behavior of hydraulic devices*

LGS Hydraulics and Control

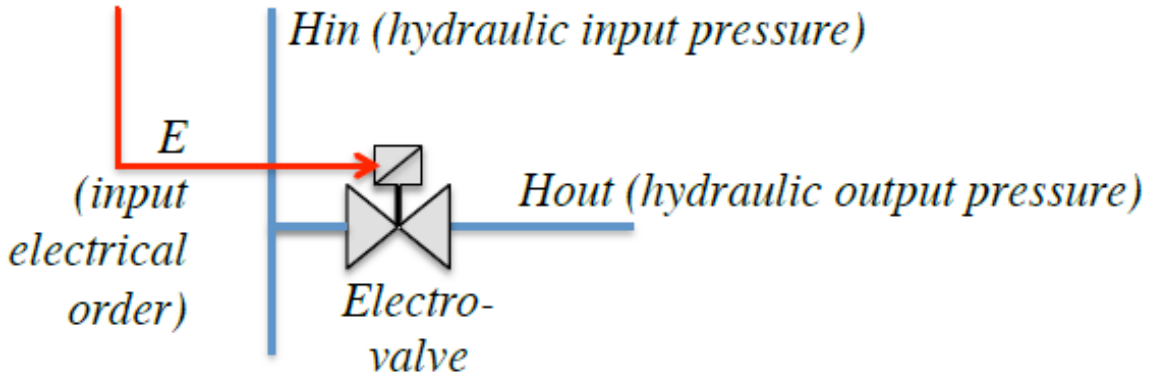


LGS Analogical Switch



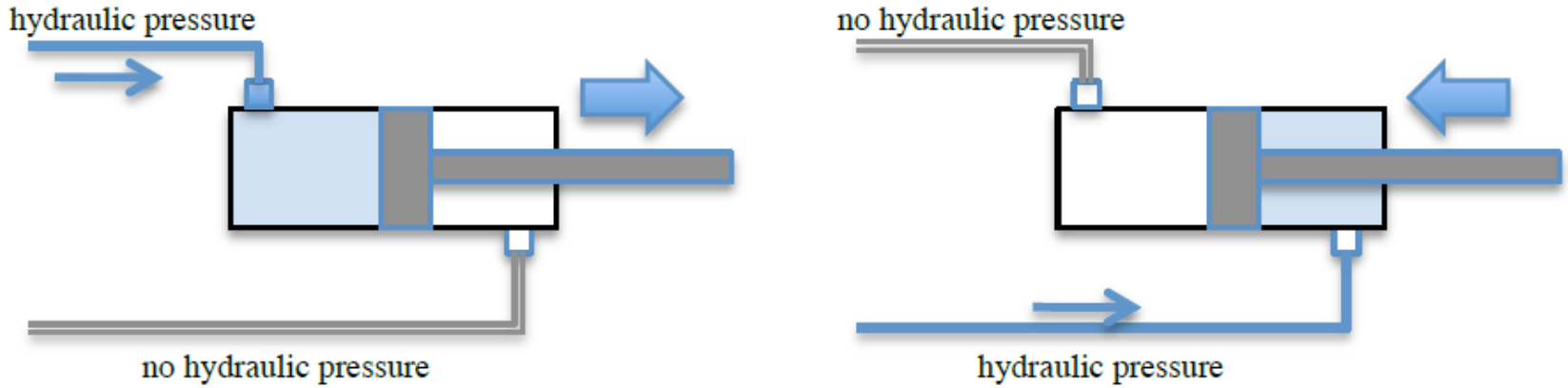
- **It protects the LGS from spurious actuation by the digital controller**
- **It is *closed* for 20 seconds when the handle is moved. After that, it *opens* automatically**
- **Transitions take time**
  - *open* → *closed*: 0.8 s
  - *closed* → *open* 1.2 s
- **It can fail and remain *open* or *closed***

LGS Electro-Valves



- **Electrical order *E* must remain *true* to maintain the *closed* position**
- ***open* → *closed*: H_{out} grows up linearly from 0 to H_{in} in 1 s**
- ***closed* → *open*: H_{out} goes down linearly from H_{in} to 0 in 3.6 s**
- **An electro-valve can fail and remain *open* or *closed***

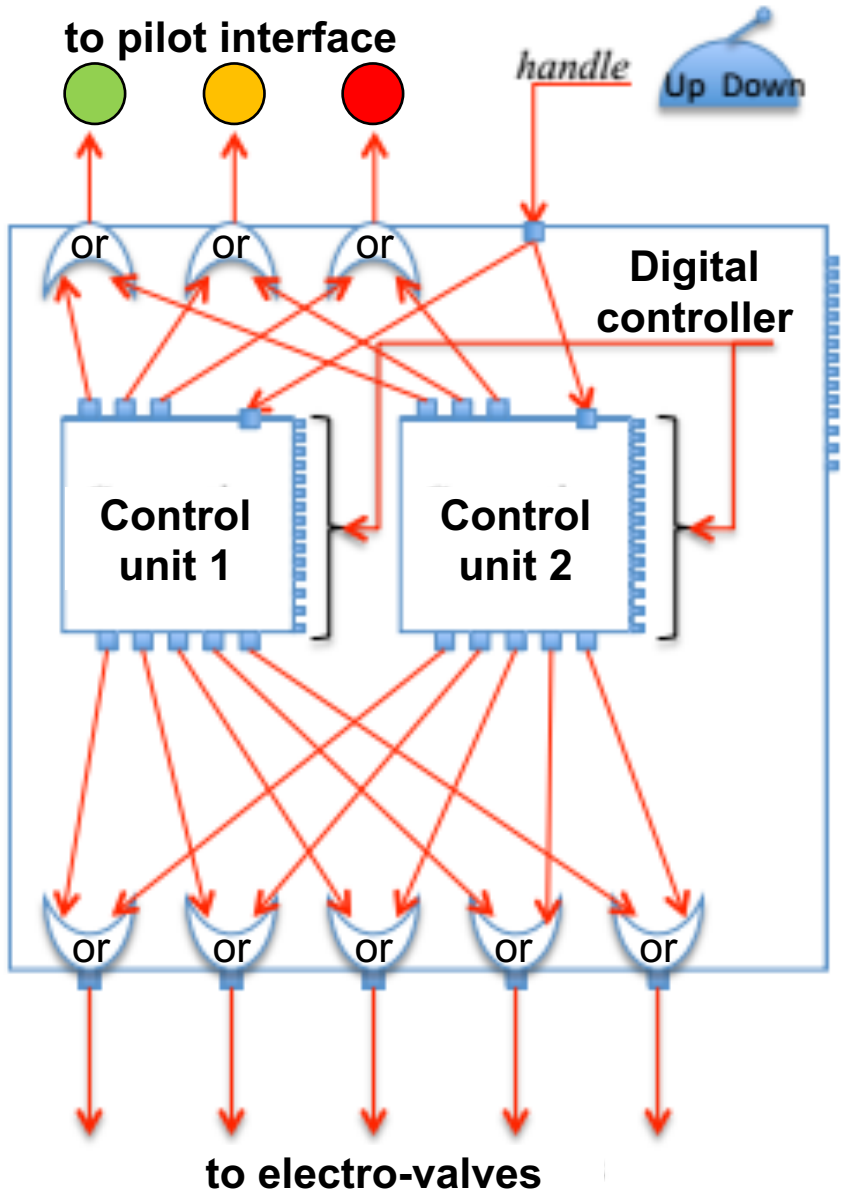
LGS Cylinders



- **At full stroke, a gear cylinder is locked in *high* or *down* position**
- **At full stroke, a door cylinder is locked only in *closed* position**
  - It is kept *open* by maintaining pressure in the extension circuit
- **The duration values given here are mean values: true durations can vary by up to 20%**
- **Cylinders can fail and remain in a fixed position**

duration (in seconds) of ...	front gear	front door	right gear	right door	left gear	left door
unlock in down position	0.8	-	0.8	-	0.8	-
from down to high position	1.6	1.2	2	1.6	2	1.6
lock in high position	0.4	0.3	0.4	0.3	0.4	0.3
unlock in high position	0.8	0.4	0.8	0.4	0.8	0.4
from high to down position	1.2	1.2	1.6	1.5	1.6	1.5
lock in down position	0.4	-	0.4	-	0.4	-

LGS Redundant Control Units



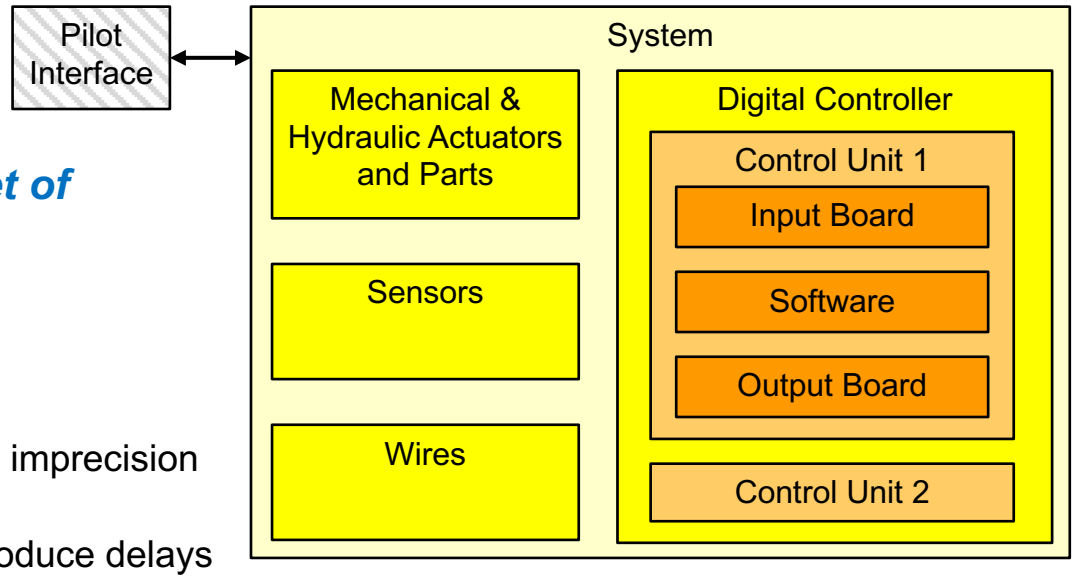
discrete sensors

- F/R/L gear *locked* / *not locked* in extended position
- F/L/R gear *locked* / *not locked* in retracted position
- F/L/R gear shock absorber *on* / *not on* ground
- F/L/R door in *open* / *not locked* in open position
- F/L/R door *locked* / *not locked* in closed position
- The hydraulic circuit (after the general electro-valve) is *pressurized* / *not pressurized*
- The analogical switch is *closed* / *open*
- Each sensor is triplicated

System Behaviour is Different from Software Behaviour

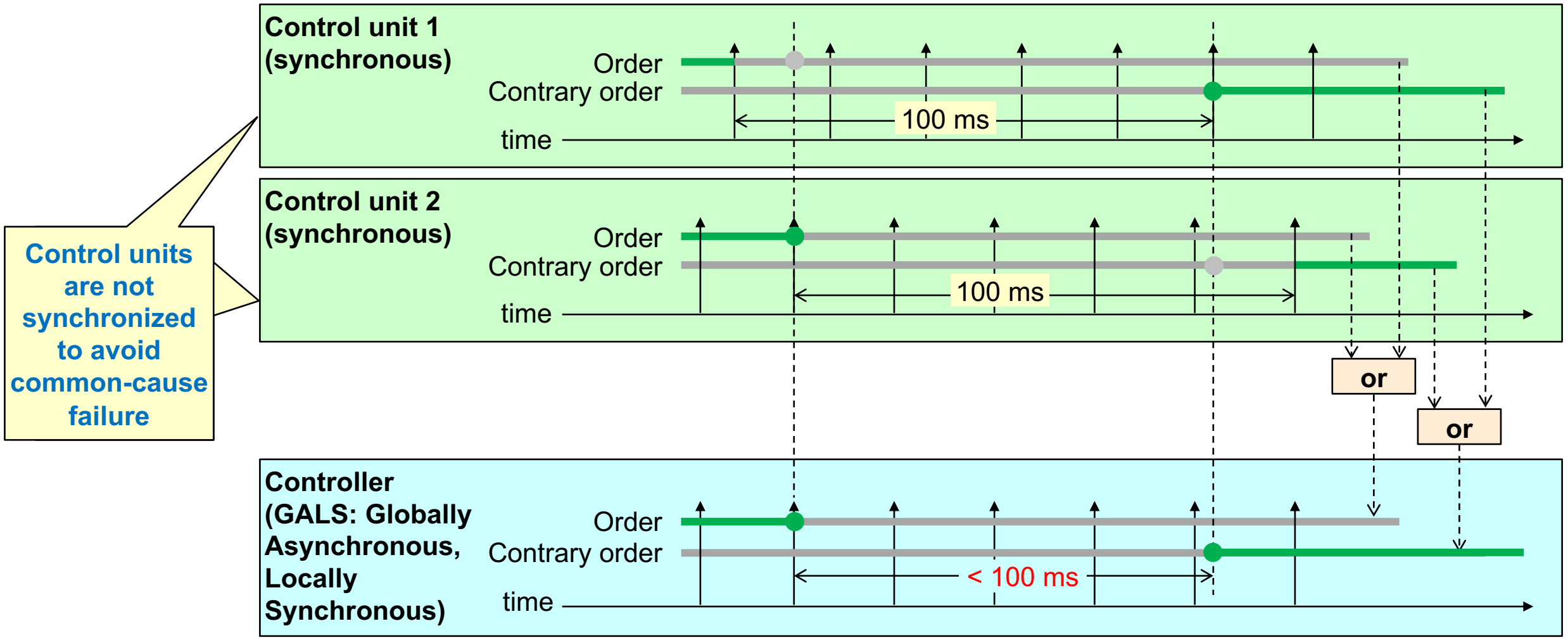
Each requirement must be attached to an object or a class (i.e., to each instance of the class)

- ***Section 4 specifies the expected behavior of the system, i.e. the behavior to be implemented by the control software***
- ***Section 5 presents the requirements of the system, that is the set of properties to be satified by the computing units of the system***
- **But**
  - System behaviour is different from controller behaviour
    - The controller receives its inputs from sensors that introduce delays, noise, imprecision and failure, through wires may break or suffer short-circuits
    - The controller sends its outputs to actuators (valves and cylinders) that introduce delays and failures, through wires that may also fail
  - Controller behaviour is different from control unit behaviour
    - With multiple synchronous but independent control units, controller inputs are sampled at different times by each unit
    - Each controller output is a combination of, and is different from, its counterparts from each control unit
  - Control unit behaviour is different from software behaviour
    - The software of a control unit gets its inputs from input boards and possibly data links internal to the control unit, which may cause additional delays and failures, and in the case of analog inputs to the control unit, additional noise, imprecision and drift
    - The control unit transforms software outputs into output signals with output boards and possibly internal data links, which may cause additional delays and failures, and in the case of analog outputs signals, additional noise, imprecision and drift



Controller Behaviour is NOT Control Unit Behaviour - Inadequacy

- ***Two contrary orders must be separated by at least 100 ms***



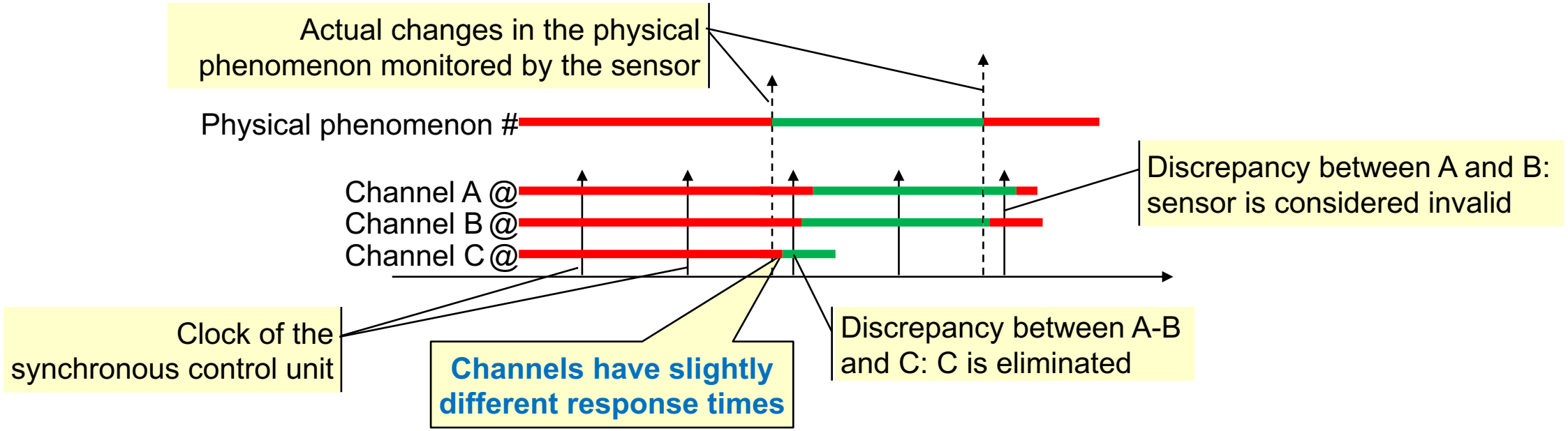
CPS-STS Have Fundamental Differences with Software

- **They operate in continuous time**
  - Even if some digital sub-systems operate in discrete time
- **They have continuous states ...**
  - Modelling only discrete states is insufficient
- **... often based on physical quantities**
  - These must be expressed in terms of physical units
  - "*When pressure > 10 do action A*" is not acceptable
  - One also needs to take account of physical laws
- **They have many random aspects**
  - Due to noise, components physical failure, external conditions, human actions and errors, manufacturing variability, ...
- **Even seemingly passive components (e.g., wires, connectors, pipes or walls) have behaviours and effects**
  - In particular in case of failure
- **In addition to engineered interfaces, one needs to take account of undesired, non-engineered interactions**
  - Due to physical proximity or physical connections
  - In particular in failure analyses
- **One also needs to take account of the physical environment**
  - Beyond official inputs & outputs
  - E.g., ambient conditions, electro-magnetic interference and compatibility (EMI - EMC), vibrations

Random Aspects Need to be Taken Into Account - Inadequacy

Such requirements are likely to invalidate perfectly operating channels and sensors

- ***If at t one channel is different from the two others for the first time, then this channel is considered invalid and is definitely eliminated***
- ***If a channel has been eliminated previously, and if at t the two remaining channels are not equal, then the sensor is definitely considered invalid***



CPS-STS Failures Can Be Catastrophic: Dependability is Essential

- **Reliability is the probability that the system will operate without failure for a given time period**
- **Availability is the probability that the system is operational at any given instant**
- **Maintainability is the probability that necessary maintenance actions can be successfully performed within a stated delay**
- **Robustness is the ability / probability of the system to tolerate non-intentional aggressions**
- **Safety is the ability / probability of the system not to harm people or the environment**
  - Generally specified in terms of required actions or states to be maintained, but also of required absence of action
- **Fault-tolerance is the ability / probability of the system to tolerate a certain number of internal failures**
- **Resilience is the ability / probability of the system, in unforeseen or exceptional situations, to enable uses that can avoid or limit unacceptable consequences**
- **Security is the ability / probability of the system to resist, at least for a certain time, to intentional aggressions**
- **Ergonomics is the adequacy of the human-system interfaces**
  - Not only of functional, but also of physical interfaces

For safety-critical CPS-STS, all behavioural requirements are probabilistic, and their engineering includes significant probabilistic studies

Apathy

To satisfy these requirements, one could just constantly set *normal mode* to false

- ***R31: When the command line is working (normal mode), the stimulation of the gears outgoing or the retraction electro-valves can only happen when the three doors are locked open***
- ***R32: When the command line is working (normal mode), the stimulation of the doors opening or closure electro-valves can only happen when the three gears are locked down or up***
- ***R41: When the command line is working (normal mode), opening and closure doors electro-valves are not stimulated simultaneously***
- ***R42: When the command line is working (normal mode), outgoing and retraction gears electro-valves are not stimulated simultaneously***
- ***R51: When the command line is working (normal mode), it is not possible to stimulate the maneuvering electro-valve (opening, closure, outgoing or retraction) without stimulating the general electro-valve***

Inadequacy

- ***R12: When the command line is working (normal mode), if the landing gear command handle has been pushed UP and stays UP, then the gears will be locked retracted and the doors will be seen closed less than 15 seconds after the handle has been pushed***

Absolutely inadequate when the aircraft is on the ground. Extremely dangerous, possibly deadly, when the aircraft is on the ground at high speed

Conclusion

- **These defects in LGS requirements are not an exception: ALL requirements specifications I studied contained defects, sometimes serious, sometimes to a much larger extent**
  - This is confirmed by the operating experience of safety systems in various industrial sectors
- **Requirements engineering (RE) is NOT just requirements management (RM)**
- **RE is not just a phase in systems engineering (SE), but an integral part of the SE process that cannot be separated from it ➔ SRE (Systems & Requirements Engineering)**
- **If and when they write CPS-STS requirements, software engineers often do not have an adequate understanding of the overall system and / or of non-software aspects**
  - Other engineers who have that understanding are generally not trained in rigorous requirements engineering
- **CPS-STS requirements tend to be voluminous and to have complex interdependencies ➔ their verification needs extensive tool support**
  - Simulation (in early phases, with tools such as StimuLus), static analysis and when possible, formal verification
- **To that end, they need to be specified in formal languages**
  - That must also be understandable to all persons concerned
  - The main purpose of formalization is not code generation, but requirements verification

Thank you for your attention

Any questions?