# System Design For Safety

Gonzalo Rey

Moog Inc.

# Safe System Design Seminar

Speaker: Gonzalo J. Rey, Moog Inc.
(Visiting Professor, on invitation from Jonas Buchli, ETH ADRL)

Safe system design methodologies aim to reduce the risk of injury and fatality to tolerable or desired levels. The guidelines and regulations that exist for many domains of application share a set of common expectations. This seminar presents a set of concepts and approaches that help meet these expectations. The content focuses on "what" can and needs to be done more than in "how", which tends to be application specific. Topics are presented at the introductory level and illustrated with simple examples.

The sequence of topics is as follows:

- Safe systems and tolerable risk targets: social expectations, where they come from, what they look like, implications
- Safe system: key decisions, hazards and mitigations, engineering judgment, implications
- Architectural mitigations: redundancy, partitions, redundancy management
- Safety oriented development process: System and item requirements, validation, design, certification, life-cycle issues
- Certification oriented system development: field of use considerations, technology maturity, verification oriented requirements, life-cycles

This is an informal engagement. Slides will be available before the start of the seminar and discussion is encouraged. In order to stay within the planned time we may take deeper, longer discussions off line in smaller groups.

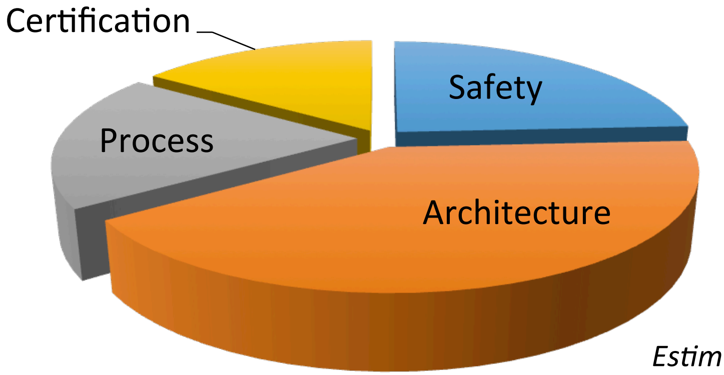
While many books and guidelines exist on this topic, typically with a target domain of application, there is no universal terminology or method for safety. Nor is there a universally accepted treatment of the topic. Typically domain-specific expectations are clear but the method to achieve them are for the most part left to the engineer. This seminar's terminology and approaches will have an "aircraft" tone since this is the speaker's domain of expertise. If this were presented by a member of the atomic power, or automotive, or mining community, this would change. The hope is that the selection of content and concepts will provide the audience with a mental picture of the type of engineering decisions that go into a "safe system" of any kind. Without claiming to be the "definitive" approach it is intended to be a reflection of a "practical" approach.

# Introduction

- Contents presented at "overview" level
  - More “what” than a “how to”
- Methods commonly used to achieve system safety
  - Without a specific certification agency target in mind
  - Concepts help find solutions that satisfy objectives such as IEC65108

# Contents

- 1. Safety *(14 slides)*
  - 1.1 Social Expectations (5)
  - 1.2 Safe System, Safety System (9)
- 2. Architectural Mitigation *(24 slides)*
  - 2.1 Redundancy (11)
  - 2.2 Redundancy Management (13)



*Estimate 7min avg/slide*
*45 x 7 / 60 = 5hr15min*

- 3. Safety Concerns in the Development Process *(11 slides)*
  - 3.1 System and Item Requirements (3)
  - 3.2 Validation, Verification and Qualification (6)
  - 3.4 Certification (2)
- 4. System Design for Certification *(9 slides)*
  - 4.1 Field of Use Considerations (2)
  - 4.2 Design Assurance Process (3)
  - 4.3 Technology Maturity (2)
  - 4.4 Safety Considerations in Design Process and Model Life-cycle (2)

# Chapter 1: Safety

# Section 1.1: Social Expectations

# Fatality Risk, Tolerable Risk

- Typically **metric**: deaths per 100,000 per year
  - Eg. 50 per 100,000 per year
- Social ***tolerance*** not equal actual statistics for injury and fatality
  - Tolerance is higher than actual rates
  - Practicality of risk reduction plays a factor
  - Type of risk-prone activity plays a factor
- Society reacts differently to mass casualties vs. individual risk
  - Tolerance of 1-2 fatalities is 10x larger than 6 or more and 3x larger than 3-5
- ***Acceptable*** *risk*: public will think of it a “low risk”
- Extensive studies of "acceptability” and “tolerance” done for policy development
  - Eg. "Reducing Risk Protecting People", Health and Safety Executive, UK, 2001

# Fatality Statistics

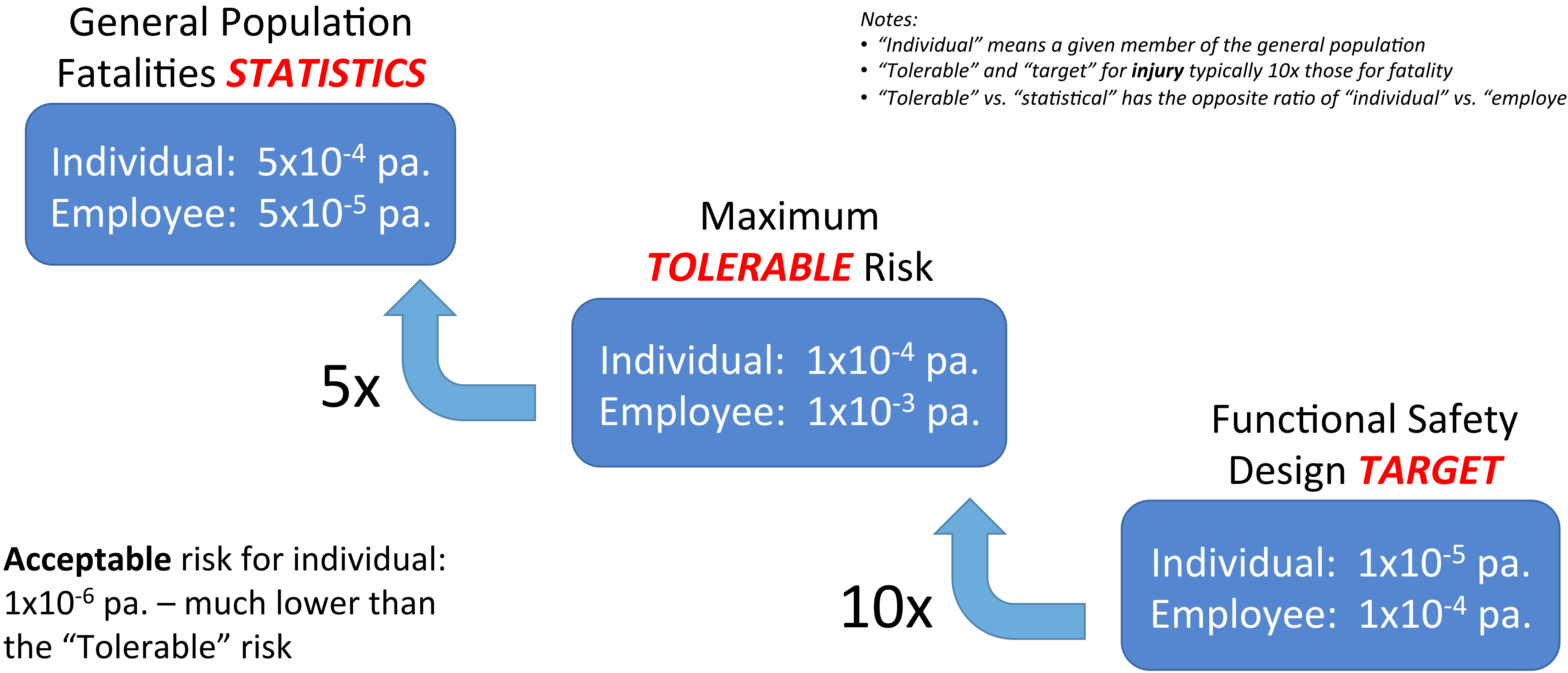
| Source | General Cause pa. per 100,000 | Accidental fatality pa. per 100,000 |
|---|---|---|
| World Health Organization | 778 | 72.1 |
| US CDC Death rates | 821 | 41.3 |
| UK Annual Abstracts of Statistics | 1000 | 20.5 |
| OSHA Worker accidental fatality rates | | Employees: 3.5, Self-employed: 9.8<br>Mining: 25, Agriculture: 28<br>Fishing: 112, Logging: 88, Pilot: 70 |

*(Note: the statistics cited above correspond to different years in the range 2011 and 2013)*

# Fatality Risk Targets for Functional Safety

*Notes:*

- *"Individual" means a given member of the general population*
- *"Tolerable" and "target" for **injury** typically 10x those for fatality*
- *"Tolerable" vs. "statistical" has the opposite ratio of "individual" vs. "employee"!*



General Population Fatalities ***STATISTICS***

Individual: $5\text{x}10^{-4}$ pa.
Employee: $5\text{x}10^{-5}$ pa.

5x

Maximum ***TOLERABLE*** Risk

Individual: $1\text{x}10^{-4}$ pa.
Employee: $1\text{x}10^{-3}$ pa.

10x

Functional Safety Design ***TARGET***

Individual: $1\text{x}10^{-5}$ pa.
Employee: $1\text{x}10^{-4}$ pa.

**Acceptable** risk for individual: $1\text{x}10^{-6}$ pa. – much lower than the "Tolerable" risk

# High and Low Demand Safety System

- *Low demand Safety System*: demand for function less than once per year
  - Eg. Car airbag
  - Failure rates are stated on a per-annum basis

- *High demand Safety System*: demand for function more than once per year
  - Eg. Car brakes
  - Failure rates are stated on a per hour of operation basis

# Example: Truck Brakes

- Exposure: Employee drives
    - 40 weeks per year,
    - 4d/week, 8 hrs/day
    - Total exposure 40x4x8 = 1,280 hrs/yr
- 50% chances that failed brakes leads to fatality (judgment call)
- Target $10^{-4}$ fatalities / year (employee)

- Target brake system failure rate:  (10-4/1,280)x2 = $1.6 \times 10^{-7}$ / hr

- Is this a realistic number to design to?
    - Numbers get small fast even for "tolerable" fatality rates!

# Section 1.2:  Safe System, Safety System

# Definitions

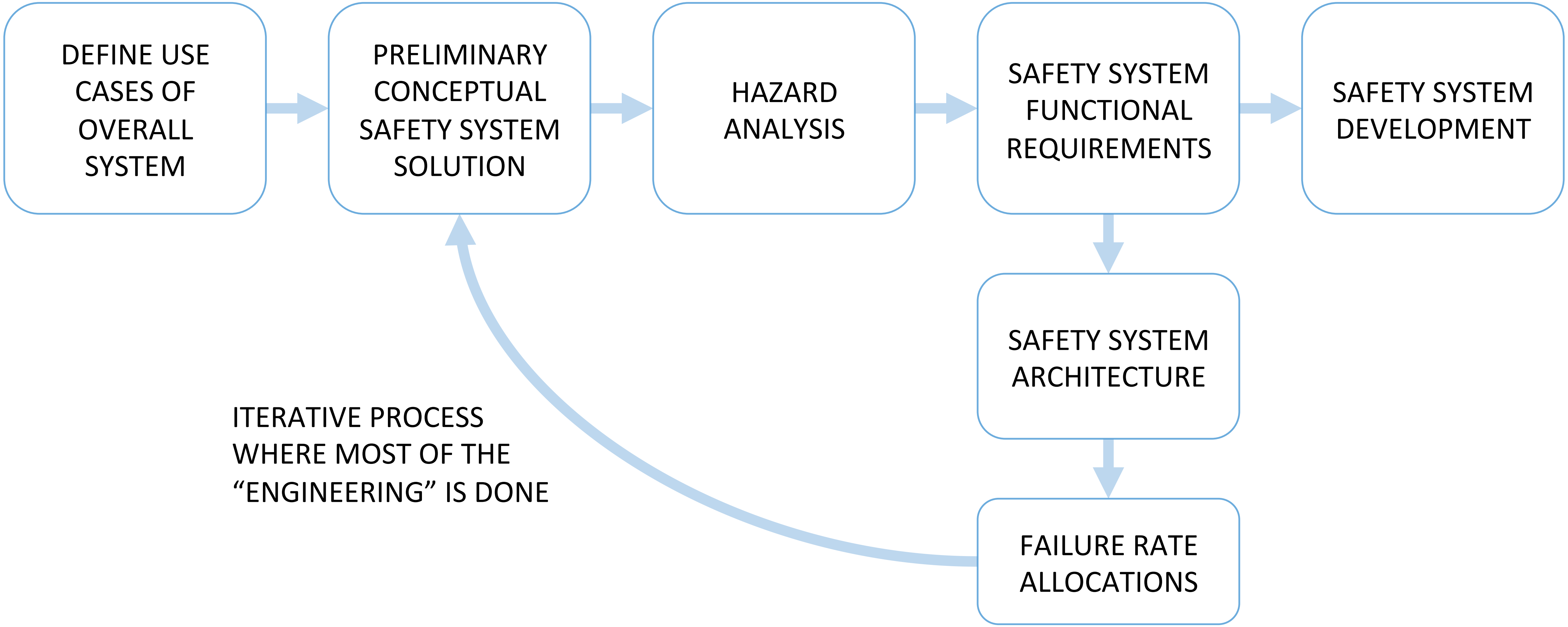
- *Hazards* or *Risks*: potential internal and external events that can impact safety
    - "Faults" or "Failures" when the events happen internally to the system
- *Development*: the life-cycle consisting of hazard analysis, system requirements, design, validation, verification, certification
- *Safe System*: a system developed to keep probabilities of injury and fatality below a specific threshold
- *Safety System*: the elements of a system that impact safety
    - Whether they "help" or "hurt"
    - Including specific technology intended to mitigate hazards

# System Partition from Safety Perspective

- Overall system partitions into Safety System and Non-Safety System
- Example:  Car
  - Safety systems:  Seatbelts, airbag, brakes, etc.
  - Non-safety system:  Entertainment system, security system

- Important to define partition early in the development process in order to avoid inadvertent co-mingling
  - Example: stereo used to make safety announcements to driver inadvertently becomes part of the safety system and must be designed accordingly
- Key decision: Is the operator part of the safety system?
  - Will the operator require training and licensing?

# Typical Safety System Development Steps



DEFINE USE CASES OF OVERALL SYSTEM

PRELIMINARY CONCEPTUAL SAFETY SYSTEM SOLUTION

HAZARD ANALYSIS

SAFETY SYSTEM FUNCTIONAL REQUIREMENTS

SAFETY SYSTEM DEVELOPMENT

SAFETY SYSTEM ARCHITECTURE

FAILURE RATE ALLOCATIONS

ITERATIVE PROCESS WHERE MOST OF THE “ENGINEERING” IS DONE

# Hazard Analysis

- *Hazard Analysis* consists of using engineering judgment to list all possible risks of injury and fatality
  - Based on a safety system solution and architecture concept
- *External hazards*:
  - Due to the operating environment
  - Due to equipment not part of the safety system
  - Due to the operator if not part of the safety system
- *Internal hazards*: Failures of the safety system (hardware, software, design flaws)
- *Combinations*: hazards occur simultaneously
- Each hazard carries a probability of occurrence and a consequence
- Sometimes this is called Failure Modes and Effects Analysis

# Safety System

- Safety system as add-on: a mechanism is added to accommodate a given hazard

- Holistic safety system: the elements of a system subject to defined hazards are designed to be safe as a whole

- Key difference:
  - Formal development process and certification for Safety Systems
  - Regardless of historical evidence, accepted lowest probability of failure of a non-safety system is $10^{-3}$ per hr

# Hazards and Mitigations

## Example

Multiple fatalities due to falling elevator

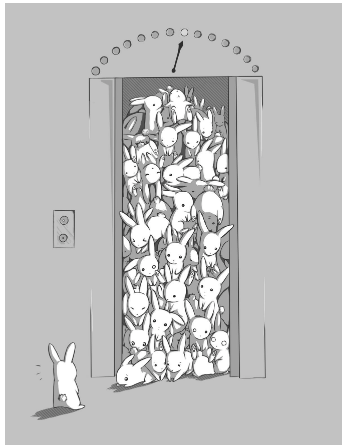
- Hazard: (there are more, but for simplicity)
  a. Cable ruptures

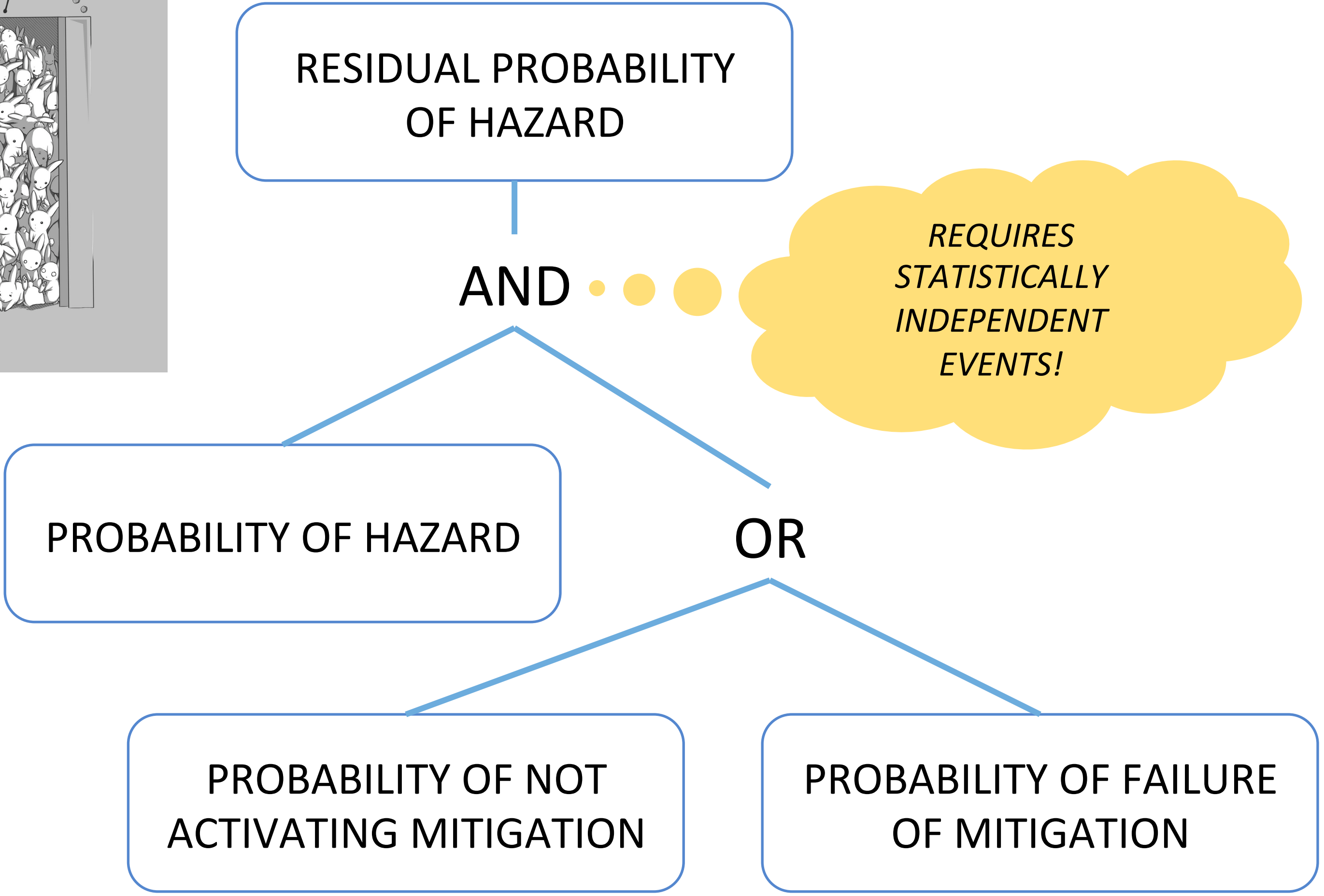
- Mitigation: emergency brakes deployed when over-speed detected. Mitigation hazards:
  b. Over-speed not detected
  c. Brakes don't deploy despite over-speed detected
  d. Brakes don't work despite being deployed

- Residual probability of hazard:
  - p(a)x[(p(b) +p(c)) + p(d)]

- Target:
  - Assume elevator carries people (4hrs/day) x (6 days/wk) x (52 wk/yr) =1,248 hr/yr
  - Target $10^{-5}$ / 1,248 = $8x10^{-9}$ / hr

- Note: If cable not part of safety system then its failure rate must be assumed > $10^{-3}$ /hr, leaving the safety system a failure rate p(b) +p(c)) + p(d) = $8x10^{-6}$ / hr
  - This is very tough to achieve so cable is part of safety system raising its integrity eg. to <$10^{-4}$ /hr





RESIDUAL PROBABILITY OF HAZARD

AND

*REQUIRES STATISTICALLY INDEPENDENT EVENTS!*

PROBABILITY OF HAZARD

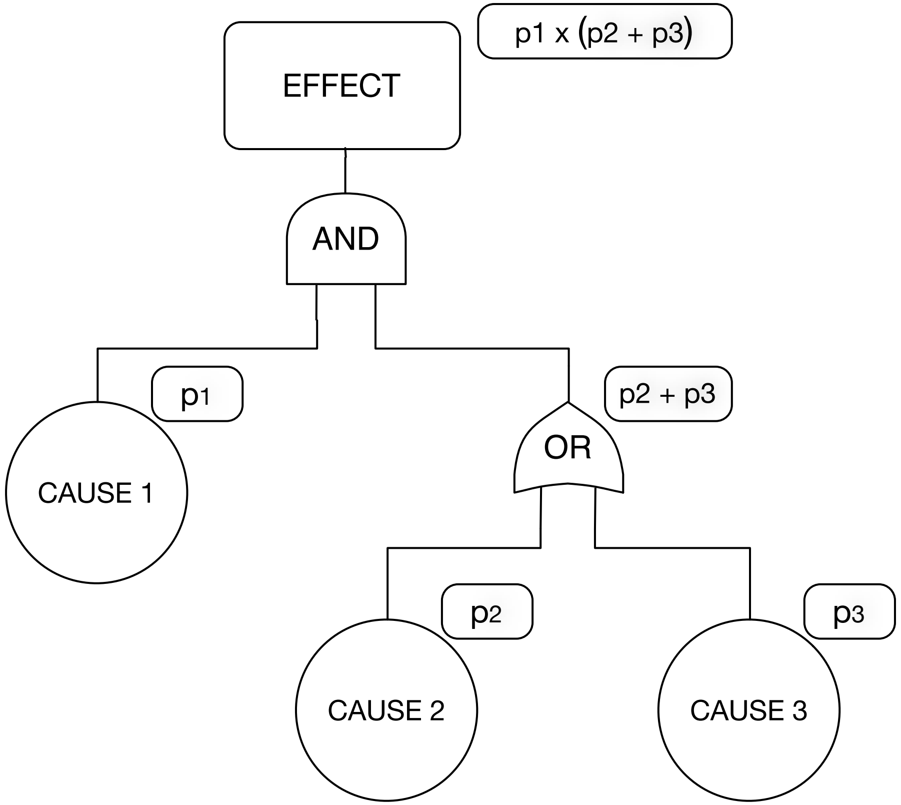
OR

PROBABILITY OF NOT ACTIVATING MITIGATION

PROBABILITY OF FAILURE OF MITIGATION

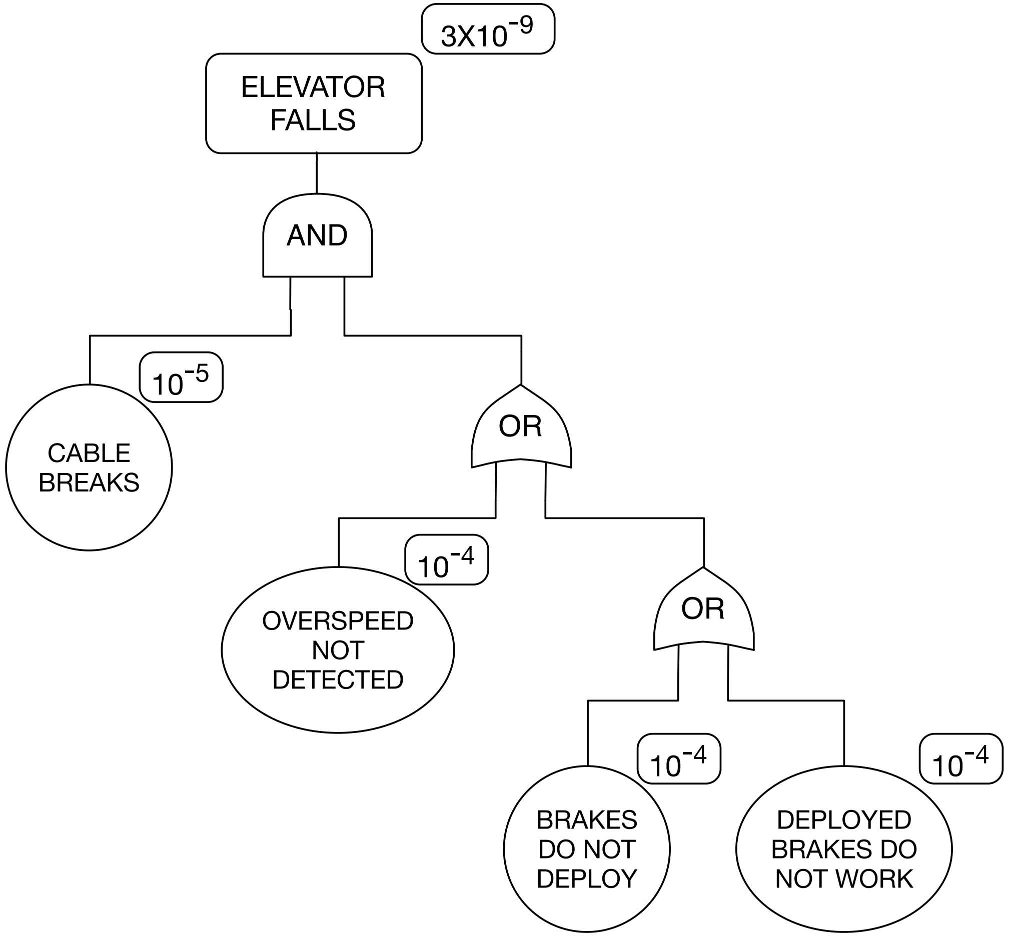
# Fault Trees

- High level event hierarchically broken down into its causes
  - If two causes are *independent* and both must occur to lead to the event, then this corresponds to a logical “AND”
  - If either cause can lead to the event, then this corresponds to a logical “OR”
- Fault trees can be used to describe failure modes and effects
  - They can either be used to demonstrate that an event has a probability of occurrence (typically lower than)
  - They can be used to justify and validate lower level requirements



# Fault Rate Allocation

- Fault rates are allocated to subsystems and components of a Safety System
- This is not “proof” of safety, only a design requirement
- A fault tree can be used to justify the allocated rates
- Example: back to elevator
  - Target: 8x10$^{-9}$ /hr
  - Assume cable is part of safety system
  - May need more than one cable to achieve 10$^{-5}$ /hr → REDUNDANCY!



# Cost Implications of Safety Systems

- Development process non-recurring costs are higher
  - Additional steps: hazard analysis, formal requirements development, flow-down, traceability, validation, verification
- Maintenance costs are higher
  - Service/inspection must be performed by a trained and certified technician
  - Must be carried out on a specific schedule and records must be kept
- Obsolescence management requires re-certification
  - System architecture can mitigate the cost by "clever" partitioning
- Redundancy may be necessary

# Chapter 2: Architectural Mitigations

# Section 2.1: Redundancy

# Redundancy

- *Redundancy*: More than one means of delivering the **safety function**
  - Commonly referred as "channels" (from electronics)
  - May or may not be exact copies of each other
- Key benefit: produces an "AND" in the fault tree
- Key drawback: complexity, size, weight and cost
- For redundancy to be effective, failures must be independent
  - Cascade failures can defeat redundancy
- Means to achieve independence:
  - Physical separation (distance or barriers)
  - Distinct technologies / solutions
  - Rigorous identification and elimination of single-point failures

# Detection and Mitigation Redundancy

- *Detection Redundancy:* Multiple means of detecting a failure
  - To detect a hazardous event with high probability it is sometimes necessary to combine multiple independent channels of data
    - Failure to detect is a failure itself
    - Failure to detect can include failure to **isolate** the faulty channel
- *Mitigation Redundancy*: Multiple means of providing the Safety Function
  - Recall that "Safety Functions" are functions of the system that affect safety
  - In addition to the primary means of delivering the safety related function there is a secondary means of insuring safety in case a fault is detected
  - The secondary means can continue to deliver the orignal function (fail-op) or not (fail-safe) – either way the hazard is mitigated

# Types of Mitigation Redundancy

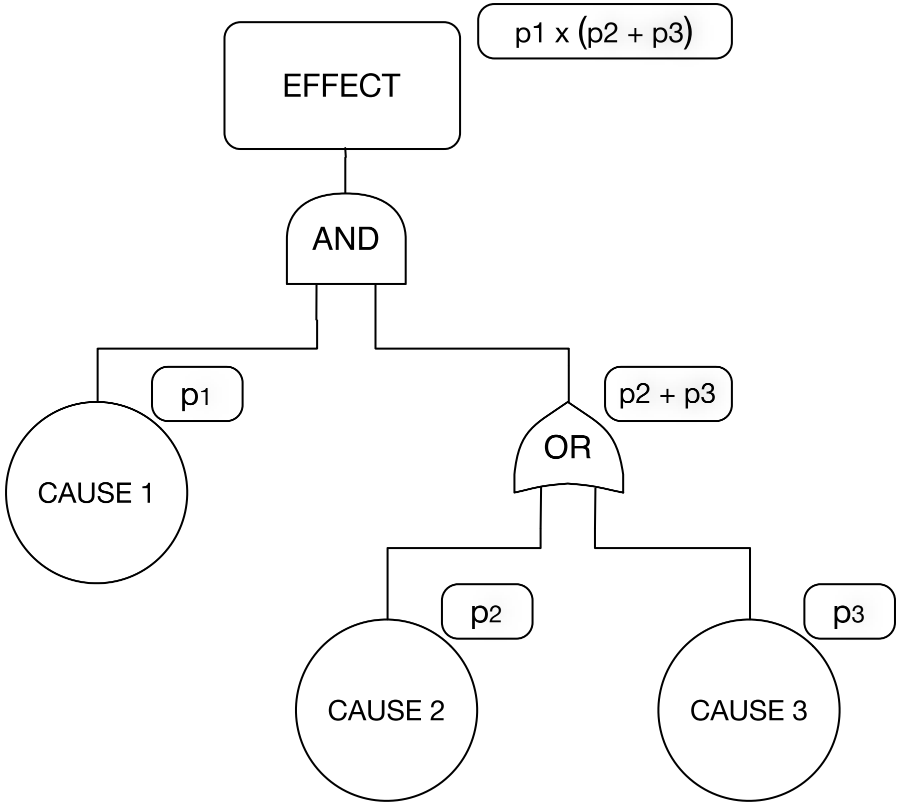
- *Fail-safe*: upon a failure the system no longer delivers the original function but instead it transitions to a safe mode
- *Fail-operational*:  upon a failure the system continues to deliver the original function
- Fail-op systems fail safe after multiple failures
- Fail-op methods:
  - Active/Standby:  Only one system operational at a time
  - Active/Active(/Active…):  Systems are active simultaneously
  - Key differences:
    - Failure transients (better in active/active)
    - Fault isolation (easier in active/standby)

# Levels of Redundancy

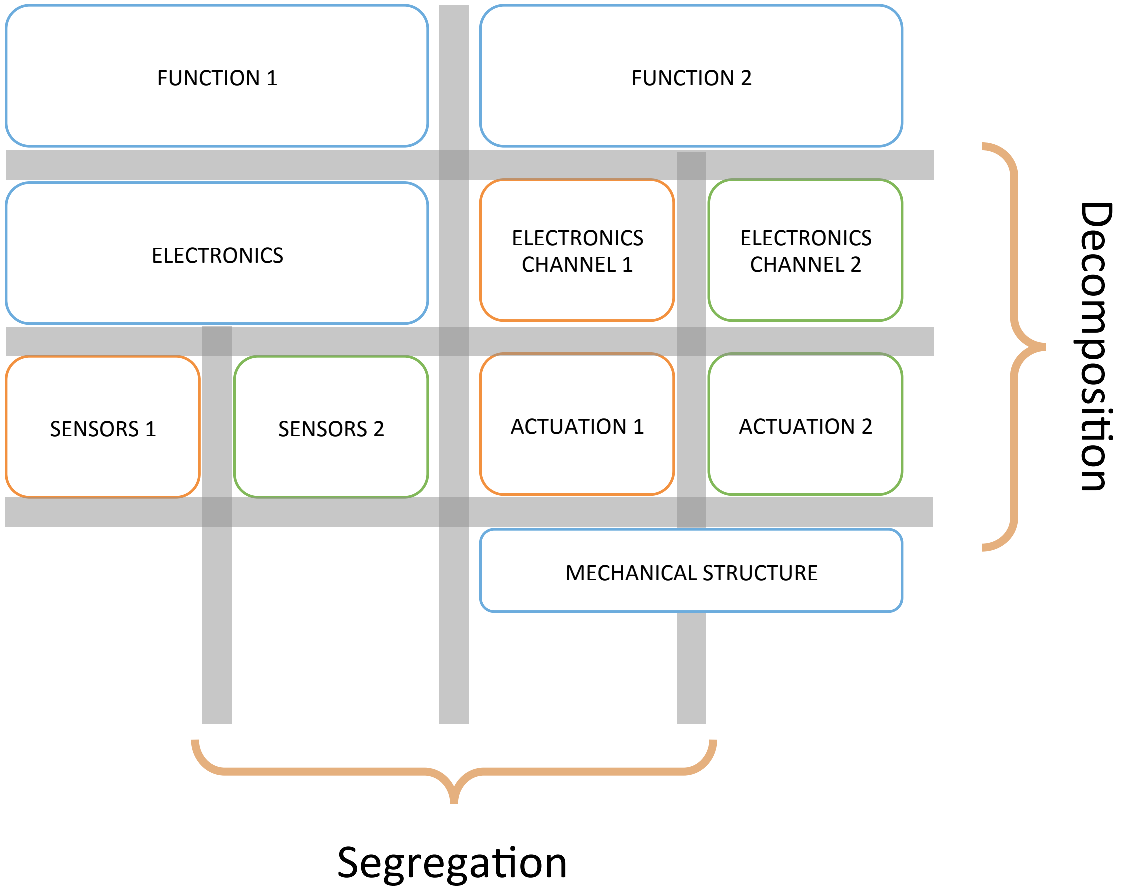
- *Level of redundancy*: Defines how many failures are accommodated before fail-safe
- Dual redundancy: two "channels"
  - Detection: Channels agree or disagree -> Fail-safe after **one** failure
    - Typical scenario: "Command/Monitor". One channel delivers a function, the other monitors
  - Mitigation: When one channel is declared failed, the other takes over
    - Typical scenario: "Active/Standby"
- Triple redundancy: three "channels"
  - Detection: Majority vote -> two channels vote the third one out, fail-safe after **second** failure
  - Mitigation: If faults can be isolated can accommodate two successive faults
- In practice "detection" is not independent from "mitigation" and "dual" and "triple" can be mixed

# Appropriate Redundancy – Example

- Assume following hazards
  - EFFECT TARGET: $2x10^{-9}$
  - CAUSE 1: Primary $p1 = 10^{-3}$
  - CAUSE 2: Fail to detect
  - CAUSE 3: Fail to mitigate
- Assume single channel failure rates
  - Detection $10^{-3}$
  - Mitigation $10^{-3}$
- Detection may be cheaper than Mitigation
  - Eg. sensors vs. actuators
- In this case we can pick
  - Triple Detection ($p2=10^{-9}$)
  - Dual Mitigation ($p3=10^{-6}$)
- Total = $1.001x10^{-9} < 2x10^{-9}$ ✔



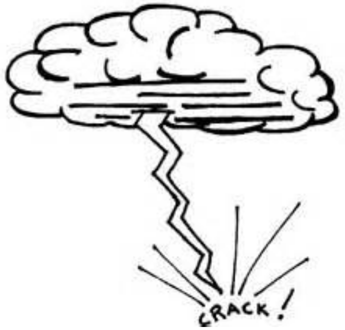
# Safety System Partitions



- Decomposition / Integration:
  - Natural layers of the solution
  - Well defined interfaces
- Segregation:
  - Designed to isolate failures
- Benefits:
  - Clarity of architecture helps avoid inadvertent common mode failures
  - Compartmentalizes certification
- Drawback:
  - Usually leads to some unwanted duplication
- This is not a “requirements” hierarchy
  - Partitions are “physical”
- Each layer of decomposition can have a different redundancy solution

# Commond Mode Failures

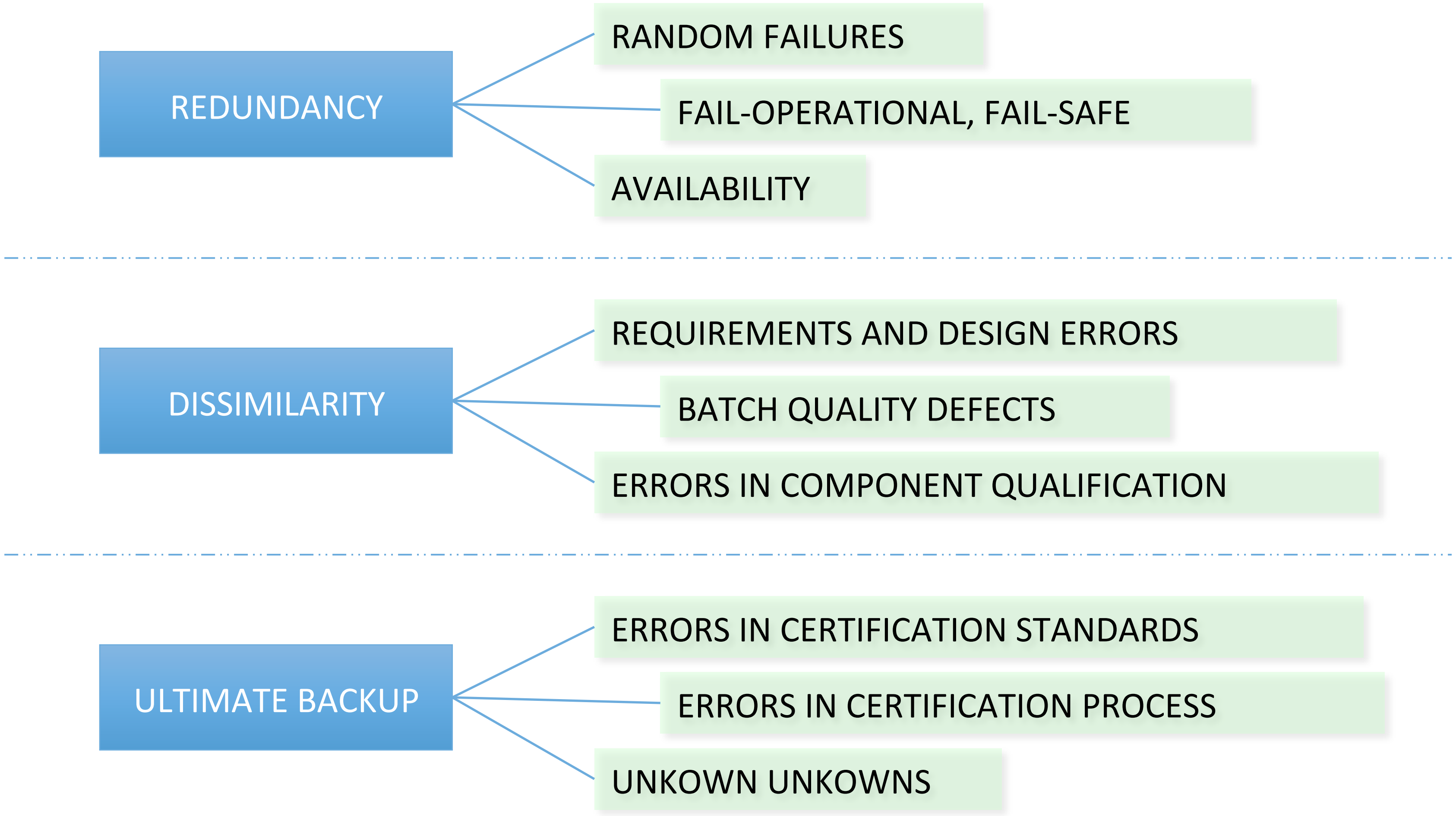
- *Common Mode Failure*: Failures that impact multiple redundant channels and/or multiple safety functions at once
  - The "AND"s in the fault tree are invalid
- Examples of sources of common mode failures
  - Environmental: lightning, fire, icing
  - Manufacturing: bad electronic components batch
  - Erroneous requirements and/or design
    - Eg. a short circuit in one channel overloads the other via the common power supply
- No "generic" approach to avoiding these, just best practices
  - Thorough review using **experienced** engineering judgment



# Mitigation of Common Mode: Dissimilarity

- *Dissimilarity*: distinct solutions for each segregated system
- Complex tools and devices (“black box” COTS solutions)
  - Different microprocessors and FPGAs – mitigate error in silicon
  - Different IDE’s – mitigate errors in compilers, linkers, etc.
- Manufacturing defects and qualification test errors
  - Either avoid using same batch of parts or use different equivalent part in each segregated system (eg. capacitors, connectors, sheer pins, bearings)
- Requirements and design errors
  - Decompose function differently in each segregated system in order to force requirements dissimilarity for subsystems/components

# Summary of Architectural Mitigations



REDUNDANCY

- RANDOM FAILURES
- FAIL-OPERATIONAL, FAIL-SAFE
- AVAILABILITY

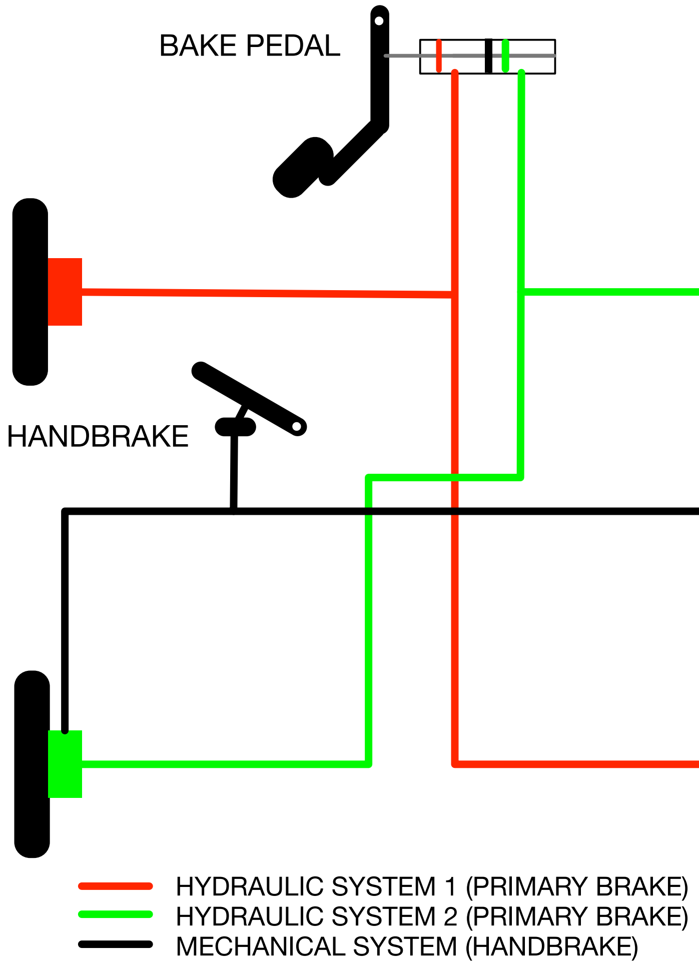
DISSIMILARITY

- REQUIREMENTS AND DESIGN ERRORS
- BATCH QUALITY DEFECTS
- ERRORS IN COMPONENT QUALIFICATION

ULTIMATE BACKUP

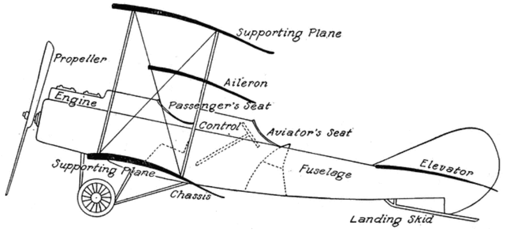
- ERRORS IN CERTIFICATION STANDARDS
- ERRORS IN CERTIFICATION PROCESS
- UNKOWN UNKOWNS

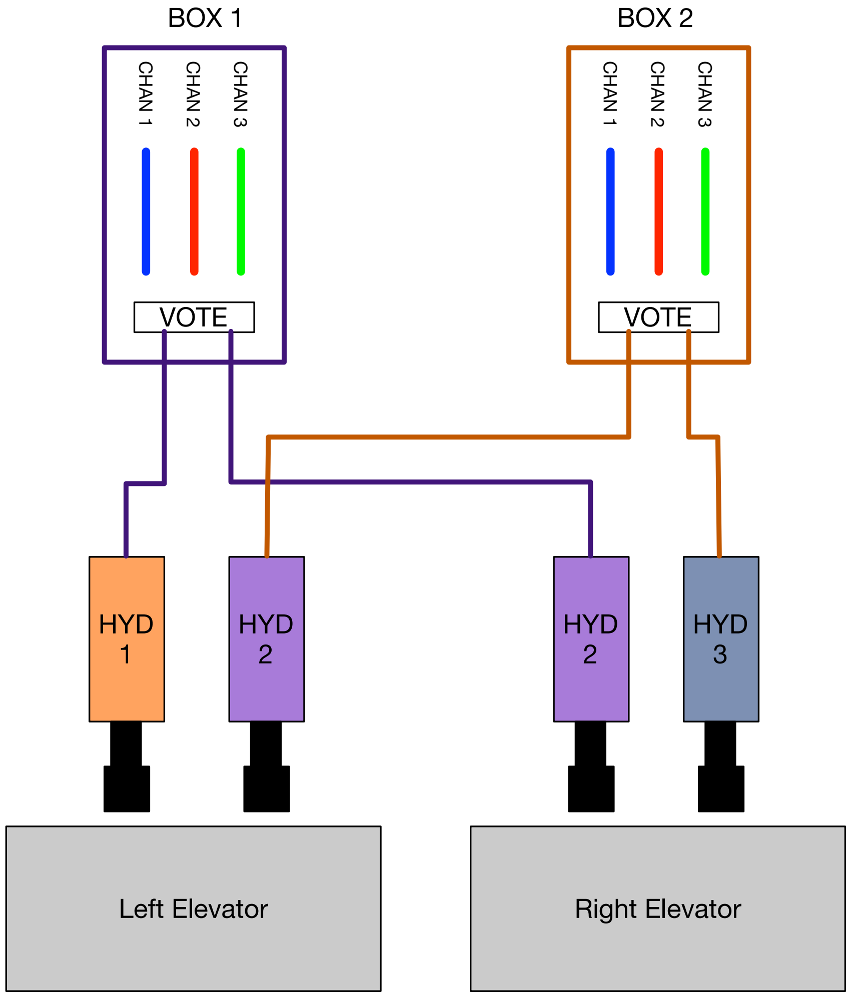
# Example:  Automobile Brake System


BAKE PEDAL
HANDBRAKE
HYDRAULIC SYSTEM 1 (PRIMARY BRAKE)
HYDRAULIC SYSTEM 2 (PRIMARY BRAKE)
MECHANICAL SYSTEM (HANDBRAKE)

- Primary dual redundant system
  - Two hydraulic circuits cross fed fr-rl and fl-rr
  - Single circuit stops car
  - Common mode exist, eg. brake pedal assembly
  - Failures are detected and indicated
  - Meets safety specifications
- Secondary dissimilar system: parking brake
  - Mechanical input to rear calipers
  - Dedicated control (lever, button, etc.)
  - Braking power less than single primary system
  - Allows steering even if back wheels lock up
  - Not part of "safety" system – ultimate backup
- Only common mode at the wheel hubs
  - Eg. all brakes overheat

# Example: Aircraft Pitch Control System





- Two boxes in different physical locations of aircraft
- Each box has three computers
  - Computer commands are voted and sent to actuators
- Actuators are powered by three segregated hydraulic systems
- Two elevator surfaces
- Each computer can control both surfaces
- Each surface powered by two different hydraulic systems
- Both boxes can continue to fully control pitch after one computer fails
  - Need four computer failures to loose pitch control
- Need three hydraulic system failures to loose pitch control
- Not covered in example: structural redundancy and electric power redundancy

# Section 2.2: Redundancy Management

# Intoduction to Redundancy Management

- Elements of redundancy management:
  - *Fault detection and isolation*: use measurements to detect a fault
    - In fail-op systems it is necessary to isolate the faulty channel
  - *Voting planes*: defines where in the system decomposition (horizontal partitions) each fault will be detected
  - *Fault accommodation*:  defines the actions taken when a fault is detected
- Redundancy management typically works in combination with maintenance and inspections

# Fault Detection

- *Monitors:* a method to detect faults based on measurements
  - If in software, monitors are "algorithms"
  - Typical monitor will process measurements and compare agains a "threshold"
- Monitors are binary statistical classification functions with false *positive and false negative (type I and II error)* probabilities
- *In-line Monitor*: use the data contained within a single "channel" to detect a fault
- *Cross-channel Monitor*: use multiple independent channels of data to detect a fault

# Voting

- When multiple versions of the same quantity or signal are available these can be "voted":
  a) Detect: Check their consistency withing a threshold – if not a fault may be declared
  b) Validate:  Decide which version or combination there of is declared "safe"
- There are many ways to define the validated value of the variable including:
  - Average value
  - Mid-value – typically used when more than one version is available
- A voting system may need to deal with the "Byzantine choice" – ie. even matched disagreement (eg. two-on-two)

# Improving Type I & II Error Probabilities

| Source or Error | Mitigation | |
|---|---|---|
| Noise in signals | Pre-filter | Inputs to monitor are smoothed |
| | Persistency | Fault not declared on first threshold crossing.  Typically this consists of low-pass filtering the output of the monitor before making a decision |
| | Count-up | Option for software monitors that count the times the threshold is crossed in a row before detection is declared.  The counter is reset if the count threshold is not crossed. |
| | Cout-up-count down | Option for software monitors that count "up" the number of samples that the threshold is crossed and "down" to zero each time the threshold is not crossed.  This is a form of "rate limited" persistence |
| System transients | Inhibit monitors during transients | Transients can be system startup or commanded transients. "Inhibit" means that the fault is not declared despite the threshold been crossed |
| Erroneous measurements | Redundancy | This is the "erroneous detection mitigation" case |

# Discussion on Fault Detection

- The typical system will have significant robustness to positive and false negative
- The "false negative" error is defined as a **requirement** for the safety system (recall the "Appropriate Redundancy" example)
- The "false positive" error (false alarm) affects availability of the system and therefore may have **commercial** impact
  - Though safety may be compromised as well depending on architecture
- As a result these are dealt with differently:
  - False negative is typically assessed against "worst case stack-up" of errors – which in practice it means that measurement errors (eg. noise) is taken at a probability 10x better than the spec for the monitor
  - False positive is typically established via simulation and development testing if not "good enough" (vague) then measures are taken – unfortunately typically too late…
  - Rigorous approach is to use Montecarlo simulation early on

# Active and Passive Monitors

- *Active monitors*: A fault declared by a active monitor triggers the safety system to execute a fault **accommodation** action
- *Passive monitors*: A fault declared by a passive monitor triggers the safety system to generate an **annunciation** for the operator or a maintenance action or both without changing the operating mode of the safety system
  - Operator annunciations are a complex topic relating to workload…
- Fault accommodation may be different in the presence of prior faults declared by passive monitors
  - The difficulty is in how these events combine
- This topic is also related to the topic of "exposure" discussed later

# Built-In Test (BIT)

- *BIT*: a test that the safety system can conduct by itself
- Typically these consist of a monitor combined with a specific maneuver in the system
  - Eg. Apply the brakes and try to move at the same time
- *Initiated BIT*:  The operator or maintenance must trigger this test
- *Power-on BIT*:  The test occurs each time the system is turned on
- *Power-down BIT*:  The test occurs each time the system is turned off
- *Event triggered BIT*:  The test is run at times established to appropriate for the test
- *Continuous BIT*:  Test continuously runs in the background

# Coverage, Hidden Faults and Exposure

- *Coverage*: The extent to which hazards can be detected by the safety system

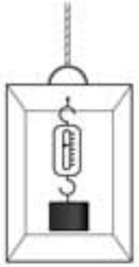
- *Hidden faults*: The hazards that are undetectable by the safety system

- *Inspection Exposure*: The length of time between actions taken to eliminate a hazard by means not available to the safety system

# Discussion on Inspection Exposure

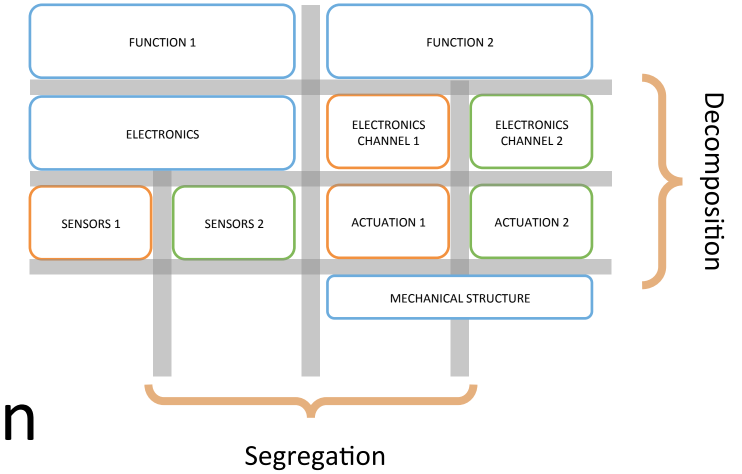
- Typically inspection exposure is applied to hazards that have a well established fault (wear out) time-line such as:
  - Corrosion, structural crack propagation, insulation resistance, abrasion, fatigue
- It requires a statistical model of how probability of failure per hour of operation depends on time between inspections
  - In practice the estimates are very conservative
- The choice to use inspection exposure as opposed to detection can be driven by
  - Feasibility of detection
  - The relative cost of regular inspections vs. the life-cycle cost of detection

# Back to the Elevator Example



- BIT: the elevator engages one brake at a time and tries to move up and down to check that all brakes are functional
- Acive Monitor: if elevator speed exceeds a threshold (up or down) declare a fault
- Fault Acomodation: deploy brakes
- Inspection Exposure: If the rope is part of the safety system, then inspection interval is designed to achieve the desired probability of rope failure

# Voting Planes



- *Voting plane*: the point in the system partition where a signal is declared "safe" by a fault detection monitor – after which the system uses a unique value of the signal
- Faults detection and signal validation can occur at anywhere in the safety system decomposition
- Selecting the voting plane has to trade off:
  - Co-mingling: if the same "validated" signal is used in two separate channels, this erodes the independence of down-stream decisions and faults
  - Divergence: if different values are used in each channel then the actions in both channels will differ, even when the signals differ in the acceptable range

# Fault Accommodation

- *Fault Accommodation*: the action taken by the safety system to mitigate the effects of a fault
- Fail-safe systems transitions to a safety backup mode
- Fail-op systems transition to a secondary means of delivering the safety function
  - For safe operation it must be know a-priori that the secondary means contains no faults
  - This requires fault isolation in an active-active system
  - In an active/standby system, the standby system fault condition is typically established during IBIT or via inspection exposure
    - In some cases the system will utilize an alternate "active" channel each time it is turned on in order to use its prior run as a means of "inspection"

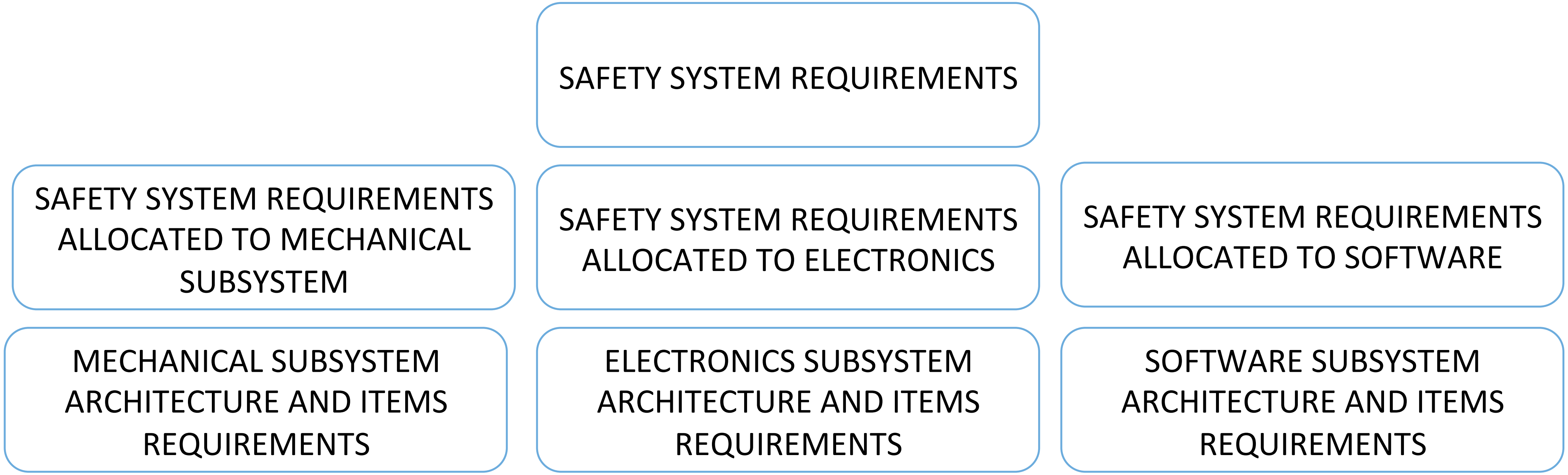
# Failure and Fault Accommodation Transients

- *Failure Transient*: is the dynamic system response that results from a failure
- *Fault Accommodation Transient*: is the dynamic response that results from the system transition necessary to mititage a failure
- Safety system requirements are typicall for both transients combined
- A system dynamic simulation study is typically used to validate the design for failure transient and fault accommodation transient
  - Operating conditions are unique and both software and hardware must be specifically designed to meet transient respose requirements

# Chapter 3:  Safety Concerns in the Development Process

# Section 3.1: System and Item Requirements

# Requirements Tree



SAFETY SYSTEM REQUIREMENTS

SAFETY SYSTEM REQUIREMENTS ALLOCATED TO MECHANICAL SUBSYSTEM

SAFETY SYSTEM REQUIREMENTS ALLOCATED TO ELECTRONICS

SAFETY SYSTEM REQUIREMENTS ALLOCATED TO SOFTWARE

MECHANICAL SUBSYSTEM ARCHITECTURE AND ITEMS REQUIREMENTS

ELECTRONICS SUBSYSTEM ARCHITECTURE AND ITEMS REQUIREMENTS

SOFTWARE SUBSYSTEM ARCHITECTURE AND ITEMS REQUIREMENTS

- Sometimes electronics and software are combined in an overall allocation and then separate item requirements are flowed down
- Verification happens at "requirements" levels (ie. Layers 1 and 3)
- Interface definitions held at level 2 defines how the subsystems "fit" together
- Validation happens "upwards" between all the levels and interfaces
- Safety system requirements are validated against "use cases"

# Good Requirements

- Typically a system fails due to incorrect requirements making requirements validation critical to safety
- Safety system requirements validation requires a lot of engineering "judgement" and past experience (this is the "art" in the process)
- Presecriptive requirements (ie. requirements that specify a design solution) lead to sub-optimal systems
  - It is a constraint that limits sub-system "experts" flexibility and optimization
- Requirements have a multiplicative impact on total design, validation and qualification/verification labor so it is best to minimize the number of requirements
- Requirements are testable
  - Design guidelines and "rules" are an alternative to "non-testable" requireents

# Sources of Error and Mitigation

| Source or Error | Mitigation |
|---|---|
| Incorrect or inconsistent requirements | Peer review, system operator review and requirements validation |
| Missing requirements | Trace matrix and peer review of trace matrix |
| Non-testable requirements | V&V team |
| Prescriptive requirements | Review by sub-system owners |
| Omissions | Adherence to industry standards and successful legacy systems |

# Section 3.2: Validation, Verification and Qualification

# Requirements Analysis

- *Requirements tracing*: a matrix that maps requirements to each other
- *Parent requirement*: The requirement that is flowed down and satisfied by allocations and allocations to item requirements
    - And item requirements to design, and design to implementation
- *Orphan*: requirements, item designs or item implementations that do not have a parent
- Requirements analysis consists veryfing that no orphans exist

# System Modeling and Simulation

- The purpose of system modeling and simulation is to two-fold
  - "Upwards" in the tree is validation
  - "Downwards" in the tree justifies allocations and enables trade-studies
- Necessary at all stages of system development and accuracy/precision improves as detailed definition of subsystems and items is completed
  - At first "requirements simulation" is used, later detailed models of the design are used
- Aspects of system that require simulation:
  - Network traffic and processors load
  - Dynamics, stability and robustness
  - Fleet-level variability issues (driven by variability in serial production, wear and environment) typically use Montecarlo simulations

# Risk Reduction and Prototype Testing

- Sources of risk
  - Complexity
  - Technology (lack of) maturity
- Risk reduction consists of identifying "all" risks and systematically producing evidence that eliminates them
- Prototype testing is an effective means of eliminating risks
  - Target specific risks using subsystem stand-alone testing
  - Focus on "principles" and test early on even before design concept is complete
  - Must account for uncertainty about eventual design solution

# Requirements Based Testing

- Each test must be tied to a specific requirement and fully establish whether it is met or not by the system
- A trace-matrix is generated by the mapping between requirements, tests and results
- Whenever an element of the system is changed, the requirement and test trace-matrices are used to generate a "regression test" program

# Verification

- Primarily test "function" (typically of system, electronics and software)
- Verification consists of generating a test report that demonstrates
    - System being tested under strick configuration control
    - All requirements have been tested and documented in a coverage analisys document
    - Each requirement has a "pass" or a "fail" determination
    - An accomplishments summary document provides an overal overview of the verification work done
- The expectations on the test equipment and tools are:
    - Fully configuration controlled
    - Fully qualified – ie. equipment and tools requirements and requirements based testing for the equimpent and tools has been successful
- Independent Quality Control is required to confirm all expectations
    - Including checks that full test can be reproduced any time in the future

# Qualification

- Typically tests performance and durability of mechanical subsystems
- Tests can include:
  - Strength: Static strength, ultimate load, proof-pressure
  - Performance: dynamic response, power utilization, speed
  - Environmental tolerance: vibration, temperature, lightning, altitude, depth
  - Durability: humidity and icing, fungus, salt-spray (corrosion), endurance (long term wear)
- Industry standards exist that specify how these tests must be conducted
  - Otherwise an agreement has to be reached with stakeholders

# Section 3.3:  Certification

# What is "Certification"?

- *Certification*:  the formal process whereby individuals with the apropriate authority formaly sign-off (i.e. testify) that to the best of their knowledge the system meets all safety needs
- It is about *people* taking responsibility for safety
  - Including the design team, the process assurance team and the certification authority representatives
- It is not as much a guarantee of safety but documentation that all known criterions, relevant industry expectations and best practices have been followed
  - People going on the record to assert that the system was developed responsibly

# The Problem with “Open Source”

- Often it is stated that "open source" systems that have accrued sufficient operating time can be "statistically" declared "safe"
- Problems:
  - What is the exact configuration of the systems that generate the statistics?
  - If one wishes to develop a follow-on version, what requirements was the legacy version designed to that makes subsystems/components reusable?
  - If I wish to certify an open source system or subsystem, who will sign on the dotted line?

# Chapter 4:  System Design for Certification

# Section 4.1: Field of Use Considerations

# Architecture for Certifiability

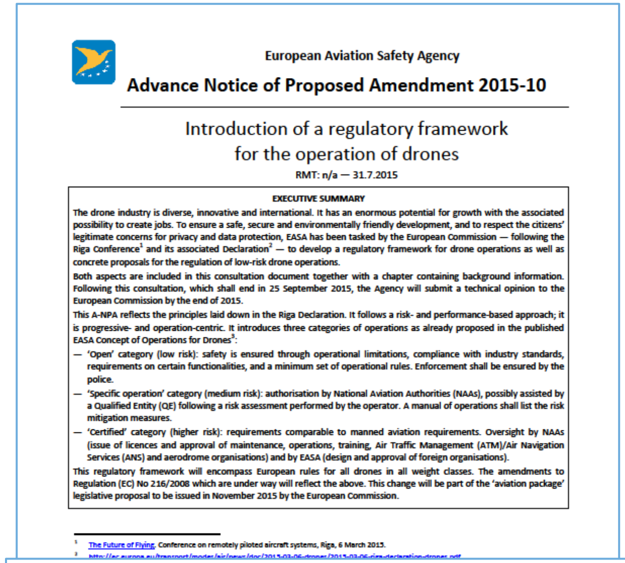
- Certification is simplified by compartmentalizing solutions to specific safety needs
  - Recall system partitions
- Problem is that each partition gives rise to additional interfaces increasing overal design complexity
- So this is a balancing act
- Key is to keep as simple as possible mapping between safety functions and system partition

# Certification Authority Management Best Practices

- Minimal indispensable and crisp assurance plan
  - More is less… no need to show off!
- Follow assurance plan rigorously
  - Team training is critical because certification authorities have a "nuanced" view of compliance
- Keep accurate records throught development from day one
- Rigurous change management, especially in requirements
- Keep certification authorities up to date
  - Typically there are defined engagements throught the project. However, it is good practice to keep them informed of key developments in-between
- Continuity of the (small) team facing the authorities help develop a fluid understanding
- Self-audit often, generate findings, close findings – nothing more reasuring to a cert authority that problems are found and resolved (no problems found = not looking hard enough!)

# Regulation Commensurate to Risk Level

- Examples of proposed standards for drones:
  - EASA A-NPA 2015-10, Advance Notice of Proposed Amendment: *Introduction of a regulatory framework for the operation of drones*
  - FAA RIN 2120–AJ60, Notice of Public Rulemaking: *Operation and Certification of Small Unmanned Aircraft Systems*
- The rigor on a sliding scale to risk of the specific operation
  - Example: Ref (1) proposes a progressive three tiered risk management approach for non-hobby commercial use of drones
    - Open (Low Risk, Operating Restrictions with Local Law Enforcement)
    - Specific (Med Risk, Authorization)
    - Certified (High Risk, Certification)



European Aviation Safety Agency

**Advance Notice of Proposed Amendment 2015-10**

Introduction of a regulatory framework
for the operation of drones

RMT: n/a — 31.7.2015

**EXECUTIVE SUMMARY**

The drone industry is diverse, innovative and international. It has an enormous potential for growth with the associated possibility to create jobs. To ensure a safe, secure and environmentally friendly development, and to respect the citizens' legitimate concerns for privacy and data protection, EASA has been tasked by the European Commission — following the Riga Conference[1] and its associated Declaration[2] — to develop a regulatory framework for drone operations as well as concrete proposals for the regulation of low-risk drone operations.

Both aspects are included in this consultation document together with a chapter containing background information. Following this consultation, which shall end in 25 September 2015, the Agency will submit a technical opinion to the European Commission by the end of 2015.

This A-NPA reflects the principles laid down in the Riga Declaration. It follows a risk- and performance-based approach; it is progressive- and operation-centric. It introduces three categories of operations as already proposed in the published EASA Concept of Operations for Drones[3]:

— 'Open' category (low risk): safety is ensured through operational limitations, compliance with industry standards, requirements on certain functionalities, and a minimum set of operational rules. Enforcement shall be ensured by the police.

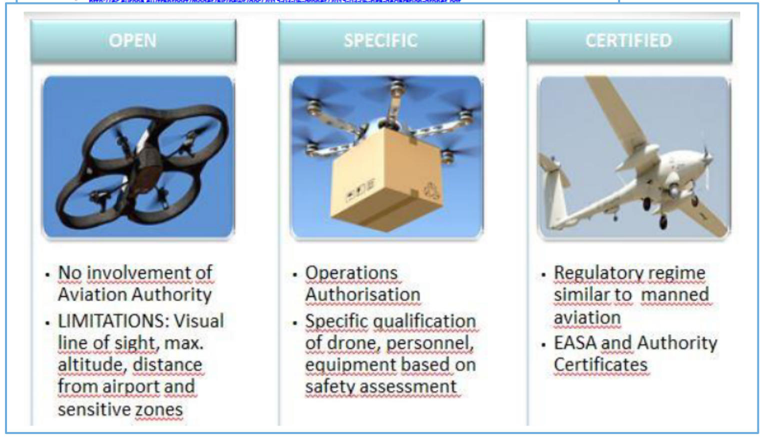
— 'Specific operation' category (medium risk): authorisation by National Aviation Authorities (NAAs), possibly assisted by a Qualified Entity (QE) following a risk assessment performed by the operator. A manual of operations shall list the risk mitigation measures.

— 'Certified' category (higher risk): requirements comparable to manned aviation requirements. Oversight by NAAs (issue of licences and approval of maintenance, operations, training, Air Traffic Management (ATM)/Air Navigation Services (ANS) and aerodrome organisations) and by EASA (design and approval of foreign organisations).

This regulatory framework will encompass European rules for all drones in all weight classes. The amendments to Regulation (EC) No 216/2008 which are under way will reflect the above. This change will be part of the 'aviation package' legislative proposal to be issued in November 2015 by the European Commission.

[1] The Future of Flying. Conference on remotely piloted aircraft systems, Riga, 6 March 2015.

[2] http://ec.europa.eu/transport/modes/air/news/doc/2015-03-06-drones/2015-03-06-riga-declaration-drones.pdf



OPEN

- No involvement of Aviation Authority
- LIMITATIONS: Visual line of sight, max. altitude, distance from airport and sensitive zones

SPECIFIC

- Operations Authorisation
- Specific qualification of drone, personnel, equipment based on safety assessment

CERTIFIED

- Regulatory regime similar to manned aviation
- EASA and Authority Certificates

# Section 4.2: Design Assurance Process

# Overview

- *Design Assurance*: process followed in order to insure that safety-related expectations are met
- Key elements of design assurance:
  - Assurance **plan** is defined before design begins (and is followed)
  - Design **standards** are tailored to the specific system and defined
  - Design team members' roles and responsibilities and **authority** delegation tree are defined
  - Project specific **training** requirements are defined and all team members have completed training
  - Design **history** is preserved and can be reviewed
  - **Configuration control** is in place and formally managed after a defined point in the project (eg. Before testing begins)
  - Validation, verification and qualification approach is documented and conducted by **independent** team
  - All **accomplishments** (eg. Verifcation) are documented
  - **Peer reviews** are conducted on all the project's work products following pre-defined checklists
  - Independent **process assurance** (quality) personnel conducts regular process compliance audits

# Design Release

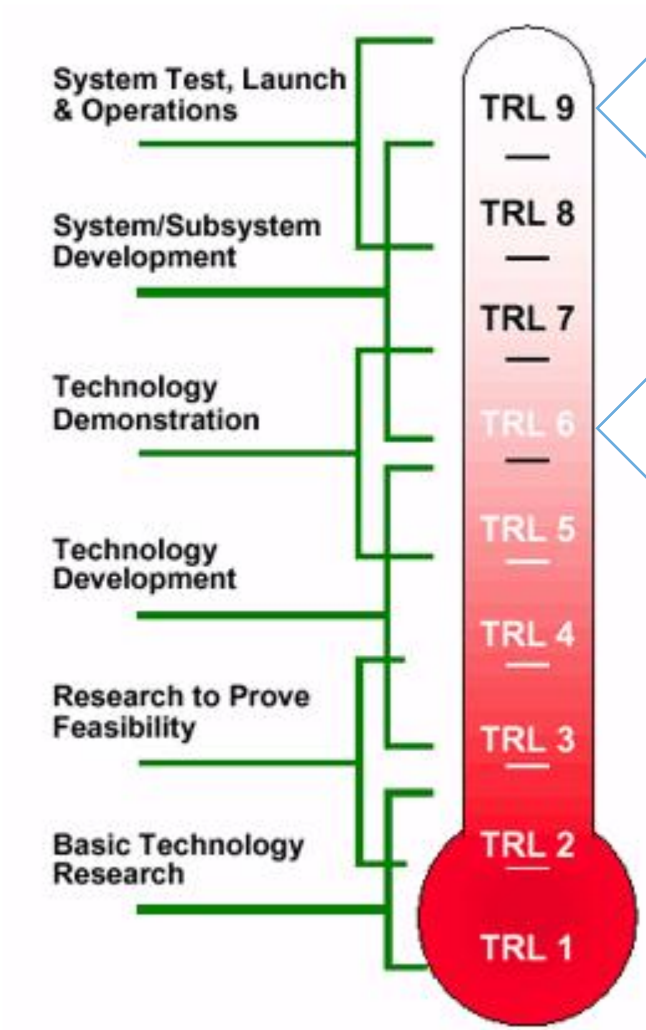
- *Design Release*: The formal process followed to document concurrence by all necessary pre-established parties that the design is satisfactory
- Typically during the early stages of the development process the project team maintains a flexible configuration control system that allows individual members to check-out and check-in contents of the configuration management database
- After design release a full **change control board** has to authorize any modifications to the configuration control database – a re-release has to occur

# Integration and (formal) Test - best practices

- Progressive system integration with development tests that confirm each addition is working properly
  - "Hardware in the loop" simulation helps operate subsystems in isolation by creating a "synthetic system" around them
- Requirements testing is best done at the lowest possible level of integration
  - While it would seem that using the "whole" system in every test saves time by avoiding the need for multiple test setups it means that whenever a test must be re-done, the whole system must be available
  - Instrumentation of "whole" system tends to be harder than for stand-alone sub-systems
- End-to-end functionality has to be fully tested
  - This has to be balanced against the previous point

# Section 4.3:  Technology Maturity

# Technology Readiness Level (NASA)


System Test, Launch & Operations
System/Subsystem Development
Technology Demonstration
Technology Development
Research to Prove Feasibility
Basic Technology Research
TRL 9
TRL 8
TRL 7
TRL 6
TRL 5
TRL 4
TRL 3
TRL 2
TRL 1

TYPICAL EXPECTATION AT ENTRY INTO SERVICE

MINIMUM EXPECTATION AT PRELIMINARY DESIGN REVIEW

- Prototype, risk reduction testing done
- Durability / endurance understood
- Design methodology known
- Qualification / Verification methodology demonstrated
- Transition to production plan in place

# Means of Assessing Maturity Level

- Operational data and similarity
  - "New" technology can be broken down into constituent elements and matched with legacy systems to "inherit" maturity
  - This works for mechanical hardware and very simple functionality
  - Requirements need to be consistent with legacy application
  - Typically requires sophisticated analysis to justify
  - Does not work for SW and FW because too complex
- Risk mitigation testing
  - Fully reproduce operating environment
  - Important for fault transients and single-string solutions

# Section 4.4: Saftey Considerations in Design and Model Life-Cycle

# Life-cycles

- *Development life-cycle*: Encompasses from concept generation to certification
- *Model life-cycle*: From first entry into service until model is retired
- *Product life-cycle*: The time the product is in service

- Life-cycles for a safety system are defined as part of the assurance plan
  - Typically segmented into phases with clear entry and exit criterion
  - Phase completion requires formal documentation of concurrance by all relevant authorities (ie. Sign-off)

# Example

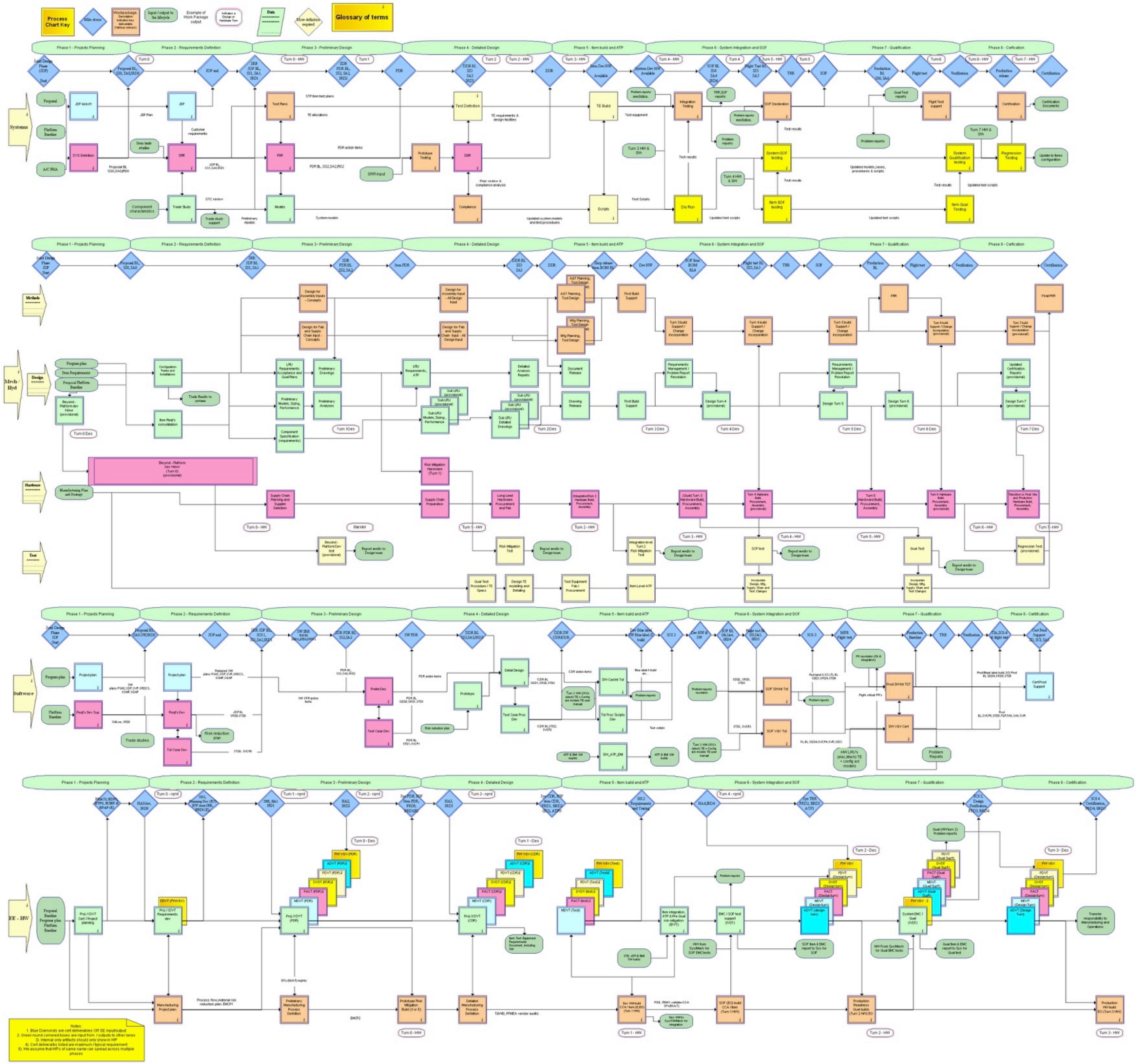
## Flight Controls Development Life-cycle

SYSTEM

MECHANICAL

SOFTWARE

ELECTRONICS


Process Chart Key
Mile stone
Workpackage. Description indicates key deliverable (Various colours)
Input / output to the lifecycle
Example of Work Package output
Indicates a Design or Hardware Turn
Data
More definition required
Glossary of terms
Phase 1 - Projects Planning
Phase 2 - Requirements Definition
Phase 3 - Preliminary Design
Phase 4 - Detailed Design
Phase 5 - Item build and ATP
Phase 6 - System Integration and SOF
Phase 7 - Qualification
Phase 8 - Certification
Joint Design Phase (JDP) Start
Proposal BL, SS0, SA0, IRD0
JDP end
SRR JDP BL, SS1, SA1, IRD1
SDR PDR BL, SS2, SA2, IRD2
PDR
DDR BL, SS3, SA3, IRD3
DDR
Item Dev HW Available
System Dev HW Available
SOF BL, SS4, SA4, IRD4
Flight Test BL, SS5, SA5
TRR
SOF
Production BL, SS6, SA6
Flight test
Verification
Production release
Certification
Systems
Mech / Hyd
Software
EE - HW

# Considerations in Life-cycle Definition

- Product Life-cycle is a key element in requirements validation
  - It defines how long a product will be in service and the inspection and maintenance intervals
- Commercial tradeoffs
  - Between incurring non-recurring costs and repeat costs during production
  - Between inspection intervals and cost of the (self-inspecting) equipment
  - Typically all these require negotiations among those who will incur the cost!
- Safety cosiderations – testing:
  - Key is up-front decision on how much unit **acceptance testing** during serial production will be relied upon as part of the safety solution
    - **Alternatives**: more monitors or BIT, more robust product, more qualification testing