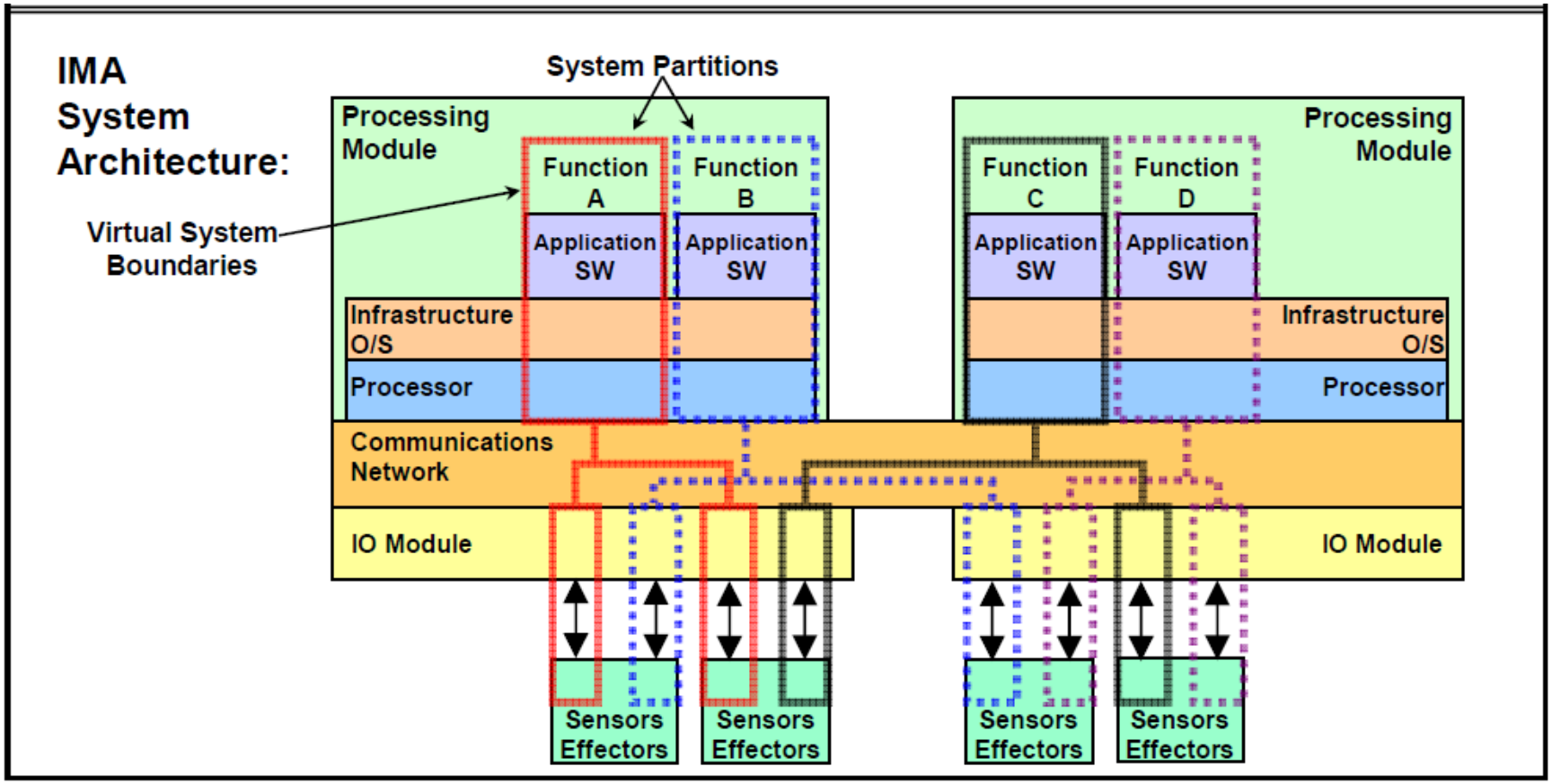
Improving Hazard Analysis and Certification of Integrated Modular Avionics

Cody Fleming

28 March 2013

Federated vs IMA Architectures



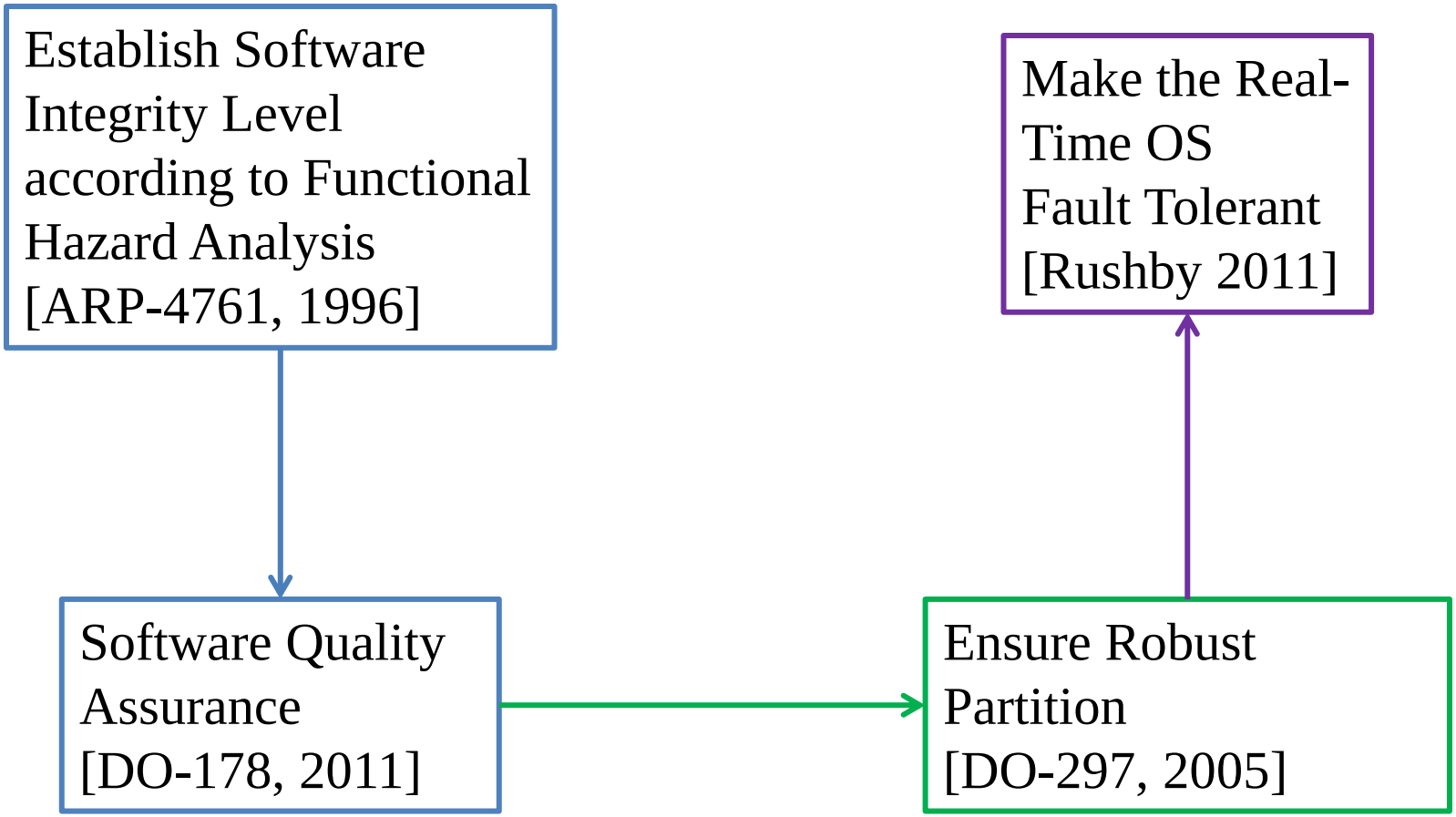
From dedicated, de-coupled systems

↓

Integrated, tightly coupled systems with real-time requirements and virtual interfaces

Background

IMA Regulatory Approach

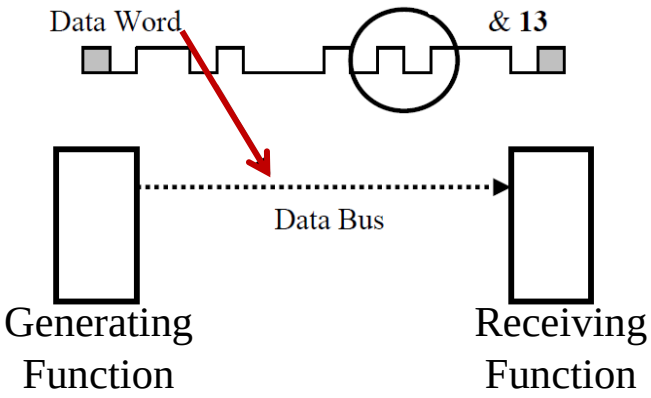


Current Approach

Partitioning & Interface Control Document (ICD)

ICD ‘defines the message structure and protocols which govern the interchange of data and communication paths’

[NASA RP–1370]

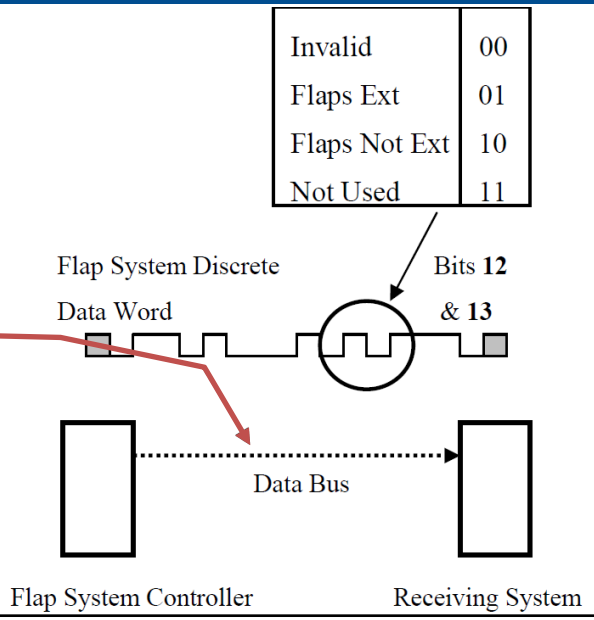


- Premise:
  - Different functions isolated from each other by robust partitioning
  - If ICD revision is not necessary for any change to function, then cross-function Change Impact Analysis is also not necessary

Problem with Approach

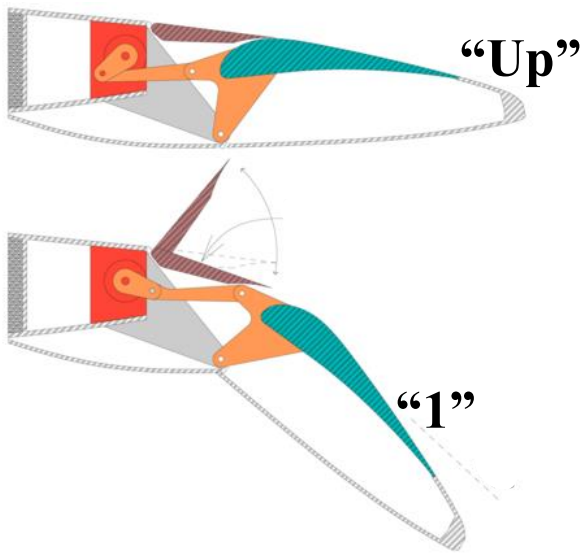
FAA illustrates their concerns with this example:

Interface Control Document (ICD) only specifies what variables go on the Data Bus, and which systems have access to them NOT HOW THEY ARE GENERATED



Examples of valid Flaps EXTENDED variable generation:

- Flap surfaces detected in the "1" or greater flap detent.
- Flap surfaces detected not in the "Up" flap detent.
- Flap Lever Handle detected in the "1" or greater flap handle detent.
- Flap Lever Handle detected not in the "Up"



http://en.wikipedia.org/wiki/Flap_(aircraft)

[Bartley 2008]

Limitations of Current Approach

1. Capturing hazardous behavior *due to component interaction*, which will become much more prevalent in an IMA regime.

2. Change Management – very little guidance & *assumes partitioning will isolate* modified functions

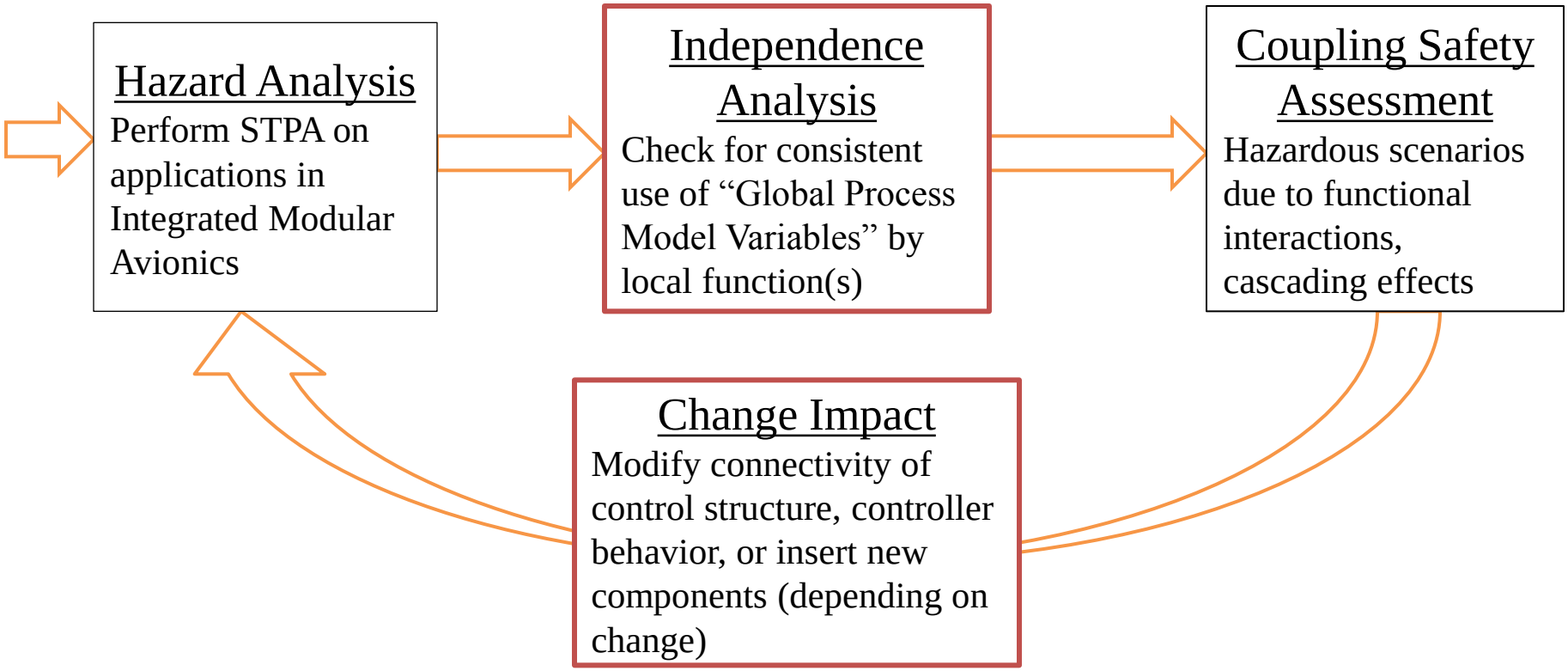
[Baker 2011] , [Bartley 2008] , [Betancourt 2012],
[Conmy & McDermid 2001] , [Graydon & Kelly 2012],
[Rushby 2011] , [Watkins 2007]

Objectives

1. Create a method to identify potentially hazardous interactions between applications in IMA and other tightly coupled avionics architectures

2. Create a method for doing Change Impact Hazard Analysis for these complex avionics systems that will be more effective than an ICD

Proposed Methodology



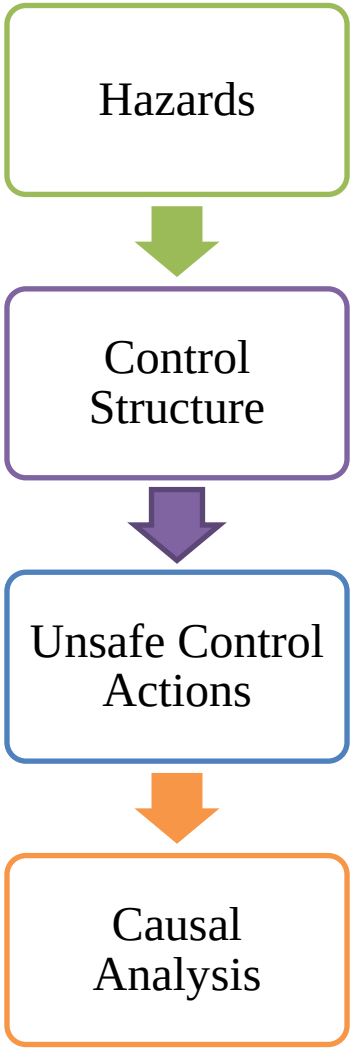
STPA : "Systems Theoretic Process Analysis"

A Note on STAMP & STPA

STAMP

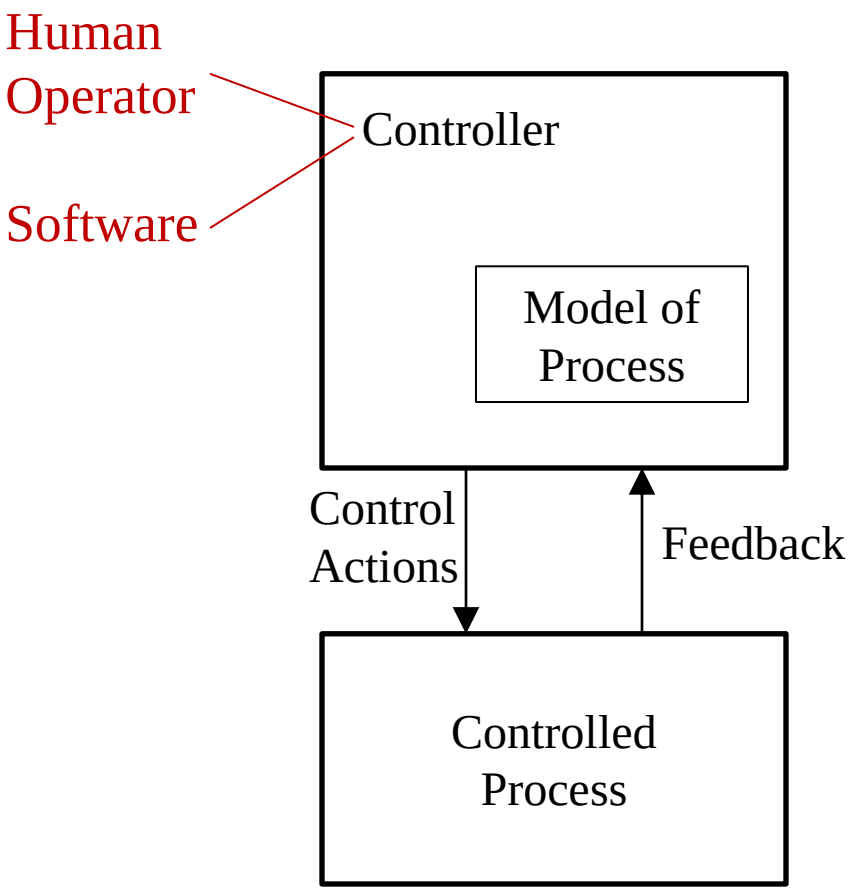
- Accidents are more than a chain of failures, they involve complex dynamic **processes**.
- Treat accidents as a **control problem**, not a failure problem
- Prevent accidents by enforcing constraints on component behavior and **interactions**
- Handles behavior that is not handled by other methods
  - Failure Modes and Effects Analysis (FMEA)
  - Fault Tree Analysis (FTA)
  - Event Tree Analysis

STPA Hazard Analysis



[Leveson 2012]

Controller Process Model



Many accidents occur when model of process is inconsistent with real state of process and controller provides inadequate control actions

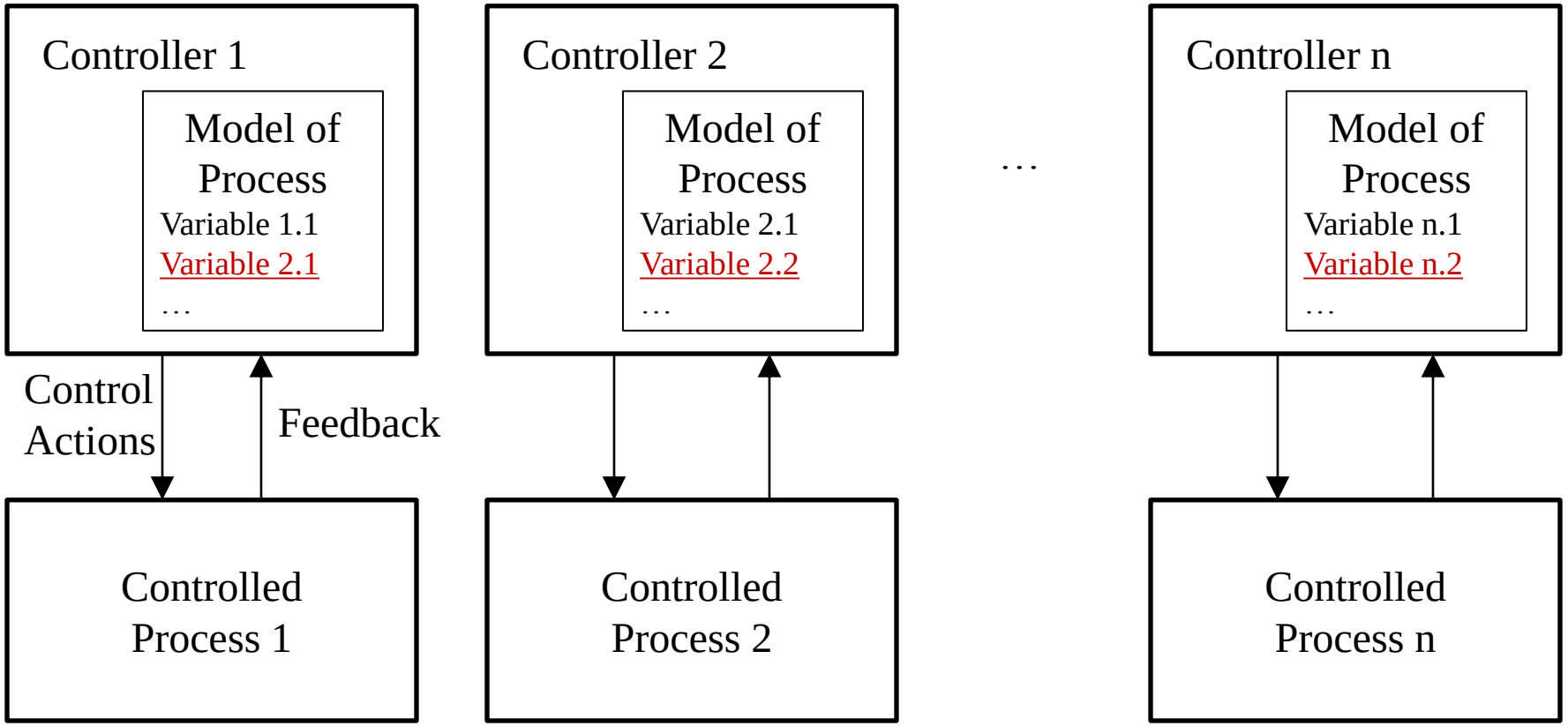
Need to have correct model to begin with

Feedback channels are critical for maintaining correct model

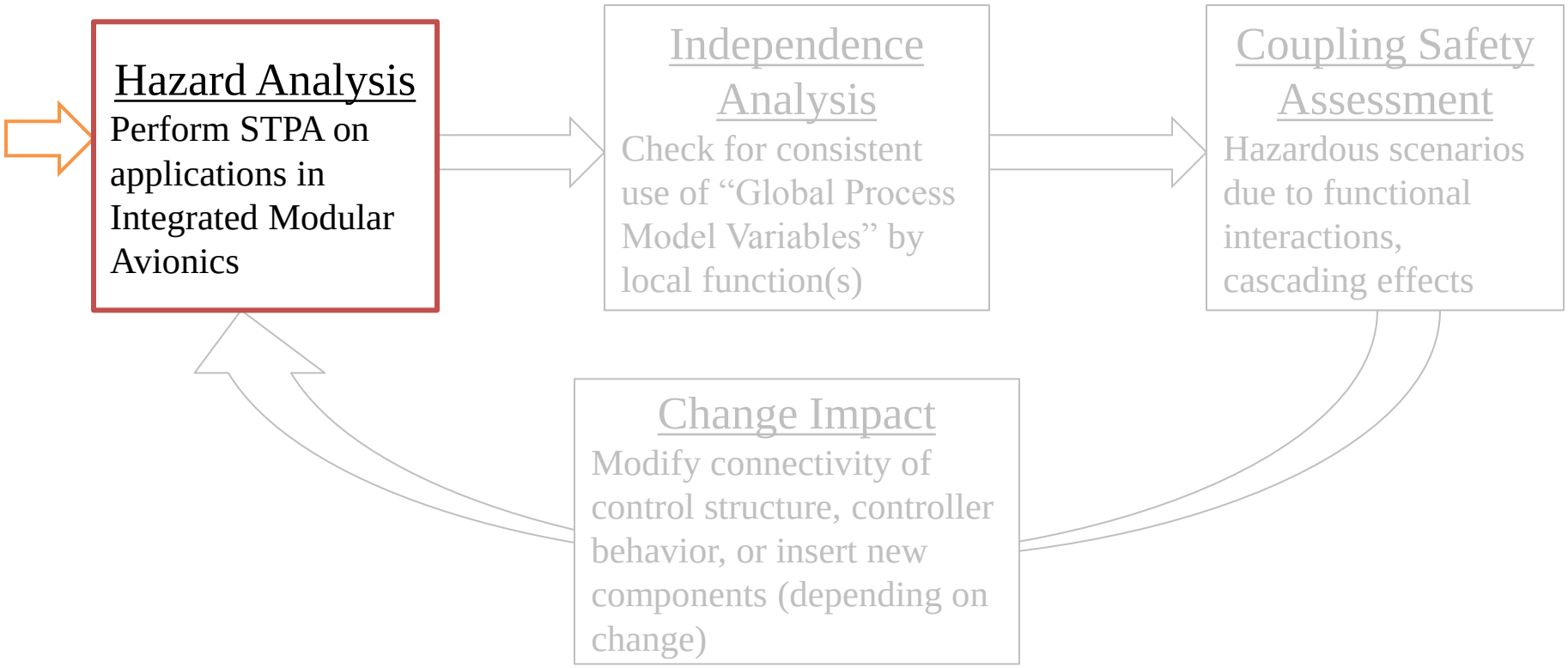
[Adapted from "1st STAMP/STPA WORKSHOP", MIT 2012]

Proposal: Global Process Model Variable

- What if different controllers (controlling different processes) need information about the same state variable?
- Inconsistent use / perception of this variable may lead to hazardous behavior



Proposed Methodology



Control Structure

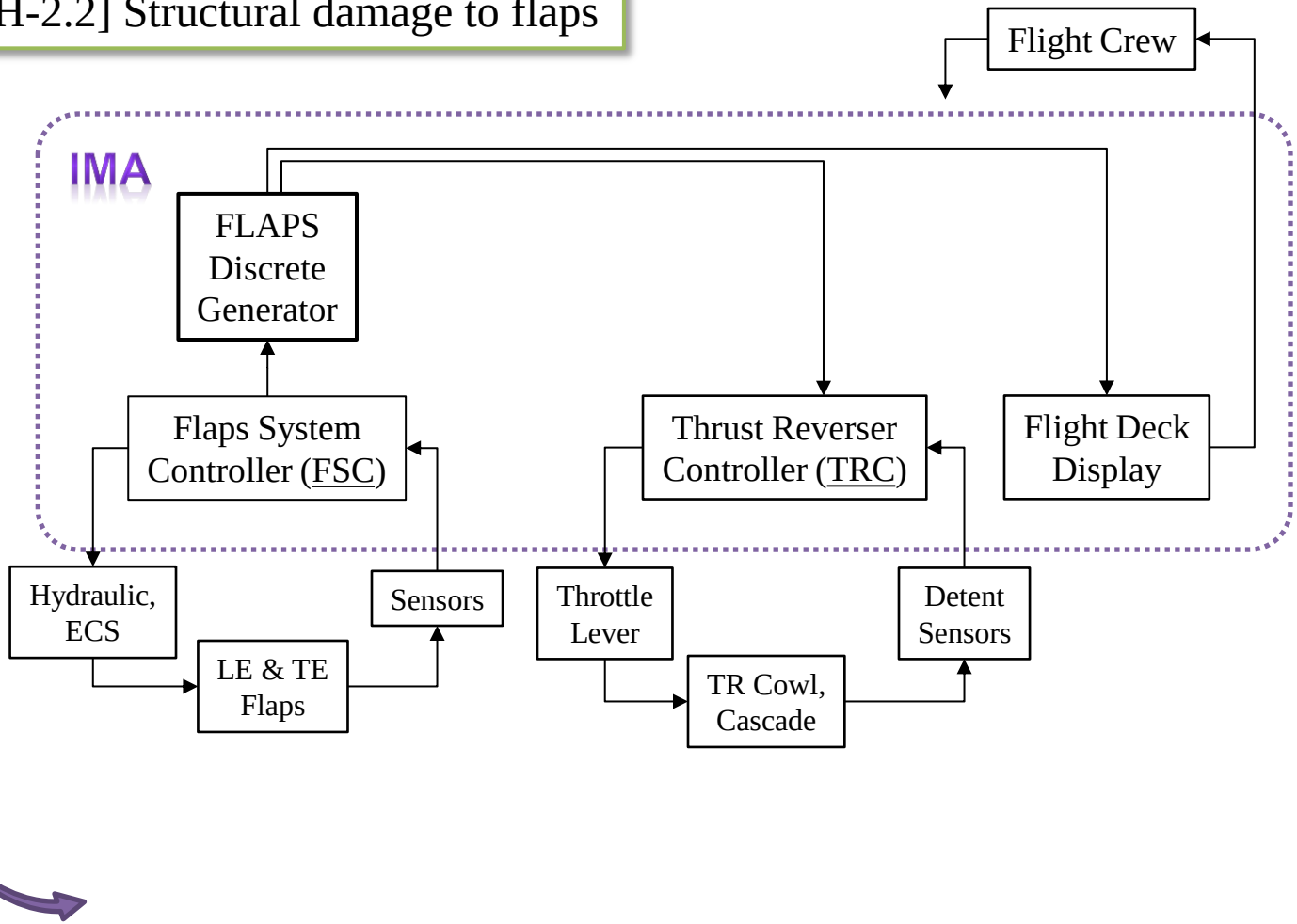
Hazards

Control Structure

Unsafe Control Actions

[H-1] Controlled Flight into Terrain
- [H-1.1] Loss of lift

[H-2] Loss of Aircraft Control
- [H-2.1] Loss of lift
- [H-2.2] Structural damage to flaps



Unsafe Control Actions

Control Structure

Unsafe Control Actions

Controller: Flight Crew	Not Provided when required for safety	Providing Causes Hazard	Too soon, too late, out of sequence	Stopped too soon, applied too long
Extend Flaps	Flaps not extended during takeoff or landing (insufficient lift during terminal ops, C_L)	LE flaps extended during thrust reversal (exhaust impingement) Flaps extended during cruise or excessive airspeed & density (flap overload)	Flaps extended too soon during approach (increased drag, loss of speed, flap overload) Flaps extended too late during approach (overspeed, missed runway)	Flaps do not achieve desired angle (e.g. stopped at incorrect discrete)

Unsafe Control Actions

Control Structure

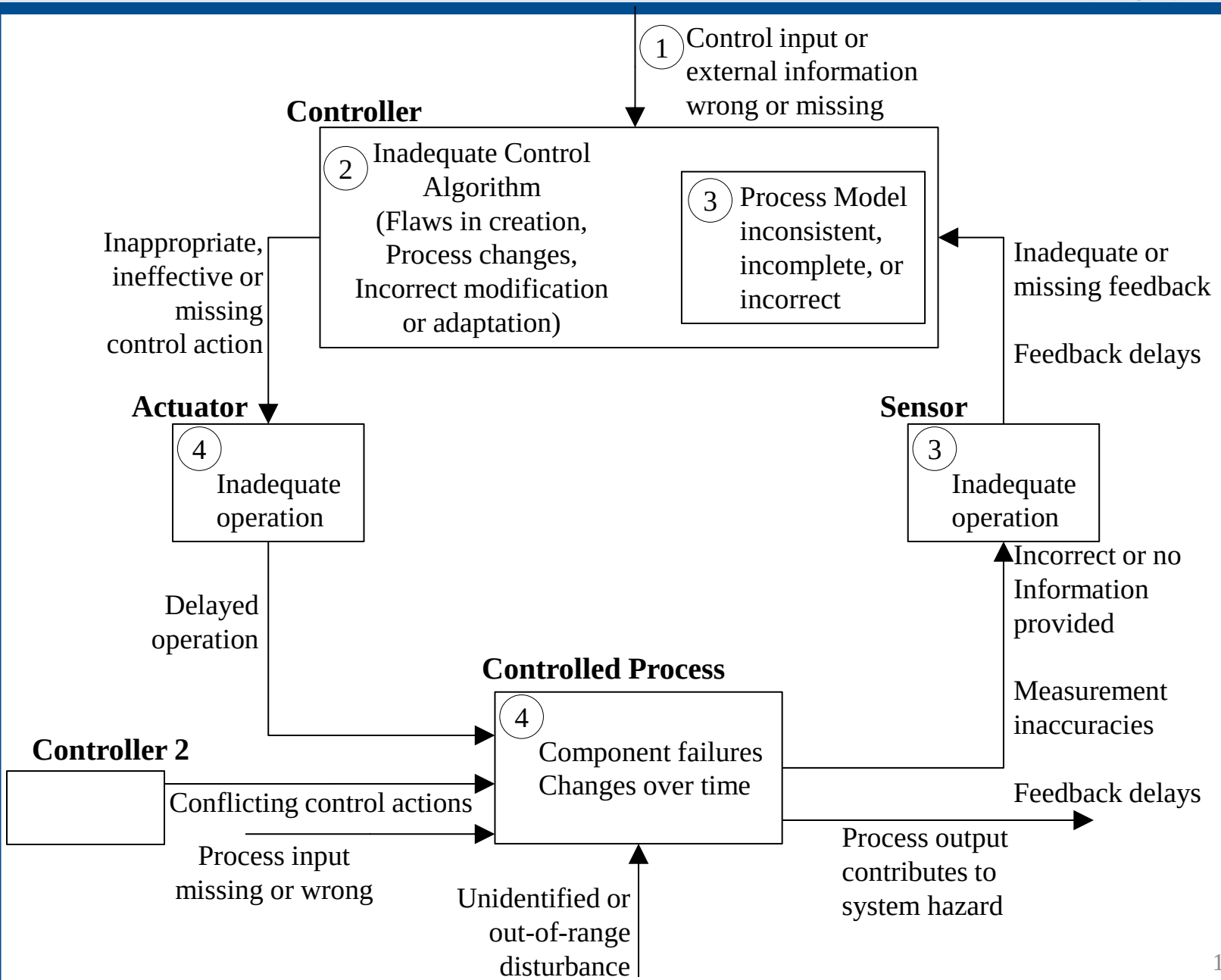
Unsafe Control Actions

Controller: Thrust Rev Ctl	Not Provided when required for safety	Providing Causes Hazard	Too soon, too late, out of sequence	Stopped too soon, applied too long
Thrust Reverse ON	No thrust reverse on short runway* (runway overshoot) Rollout takes longer than expected (conflict with other taxiing/runway operations)	Reverse thrust during flight leads to loss of *v* and therefore lift Bypass air impinges on LE flaps	Reverse thrust applied too soon before landing, resulting in loss of airspeed during approach Applied too late during rollout (Needed when C_L and high *v* limit effectiveness of friction brakes located on landing gear)	Stopped before aircraft reaches desired speed on runway

Causal Analysis – STPA Generic Loop

Unsafe Control Actions

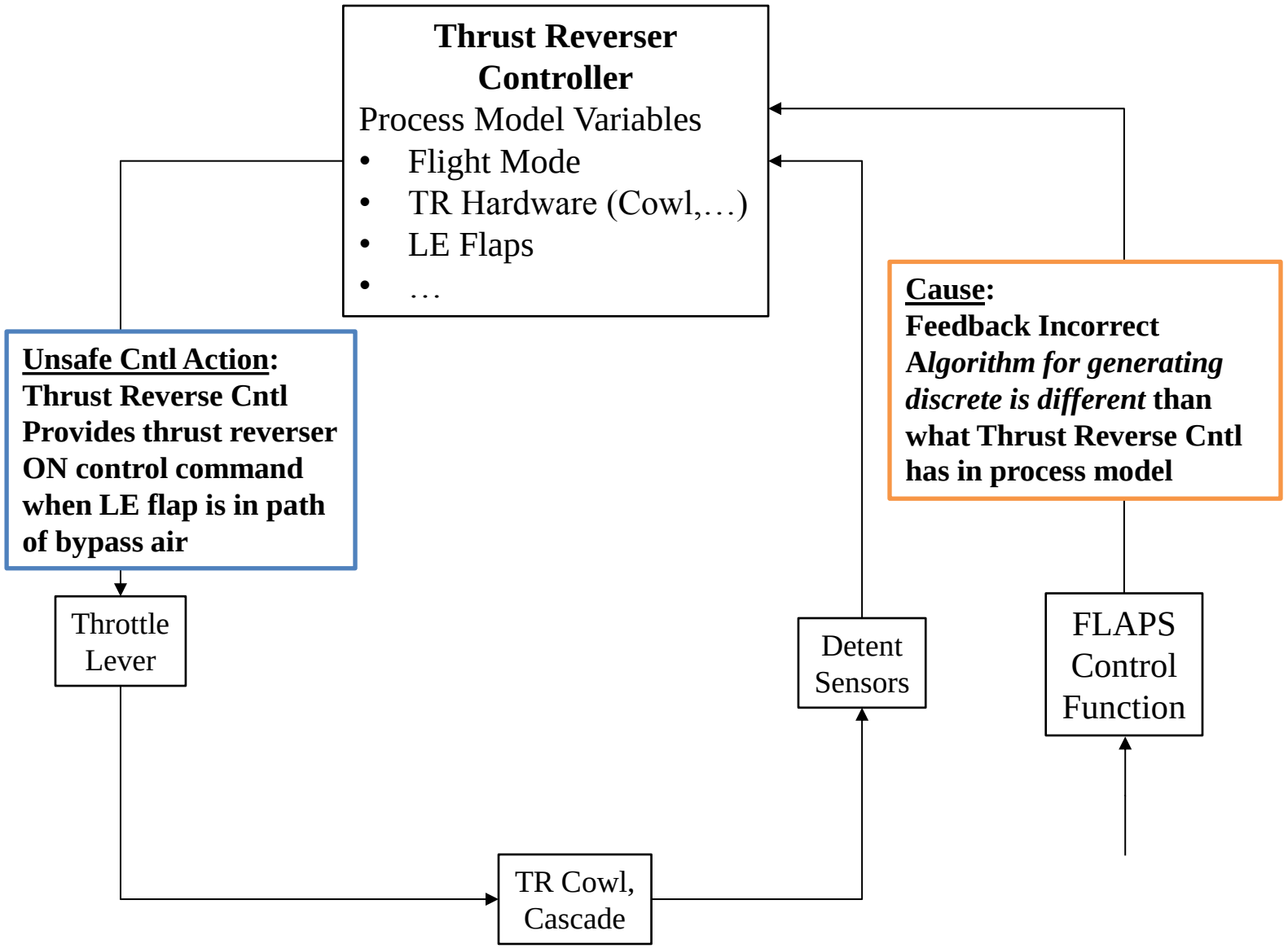
Causal Analysis



Causal Analysis – Thrust Reverse

Unsafe Control Actions

Causal Analysis



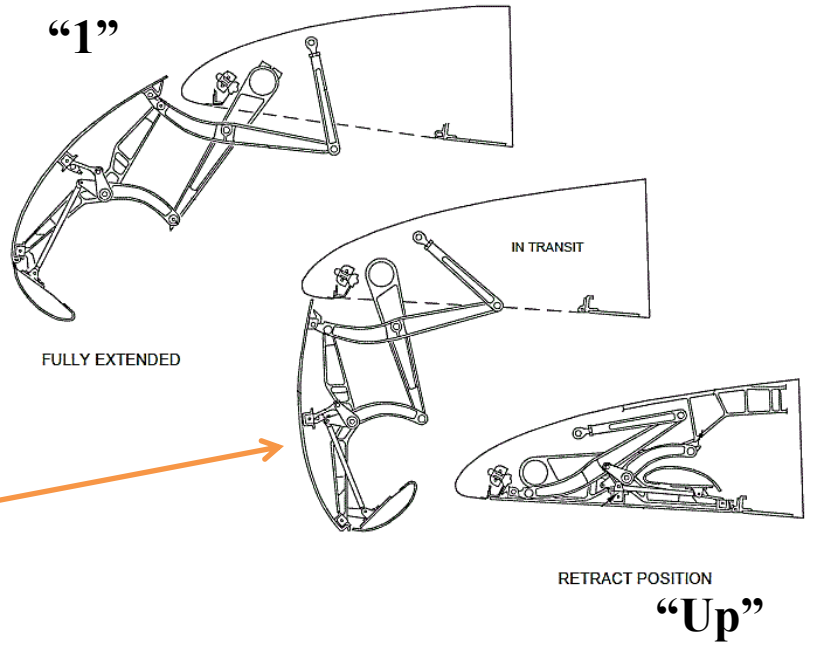
Causal Analysis – Thrust Reverse

Unsafe Control Actions

Causal Analysis

Scenario:
Flaps Control Function only sends EXTENDED message when sensor is in "1" detent

∴ Unsafe Cntl Action:
Thrust Reverse Cntl 'Provides' thrust reverser ON control law when LE flaps is between retracted and full extension

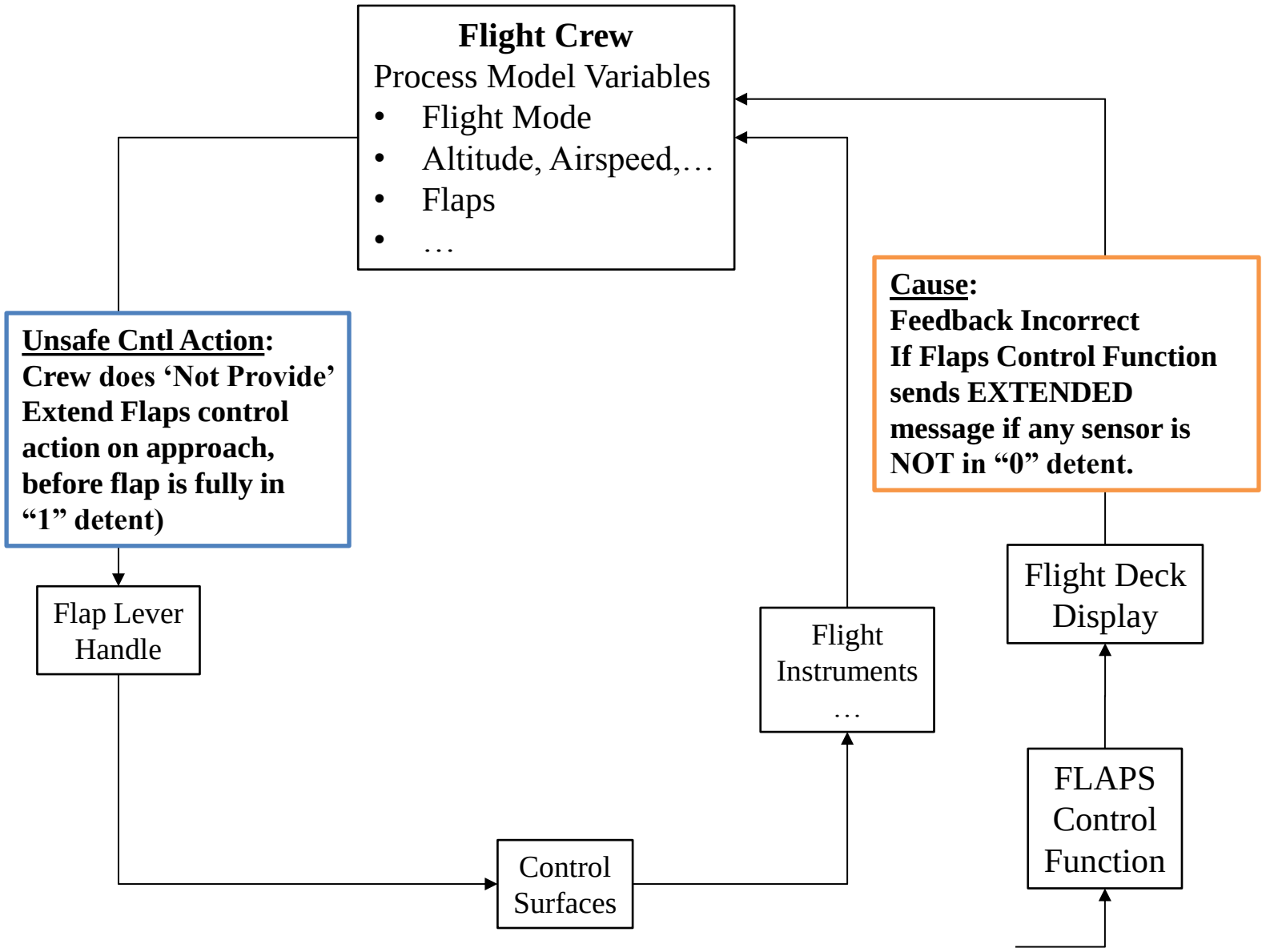


http://captainsim.org/yabb2/

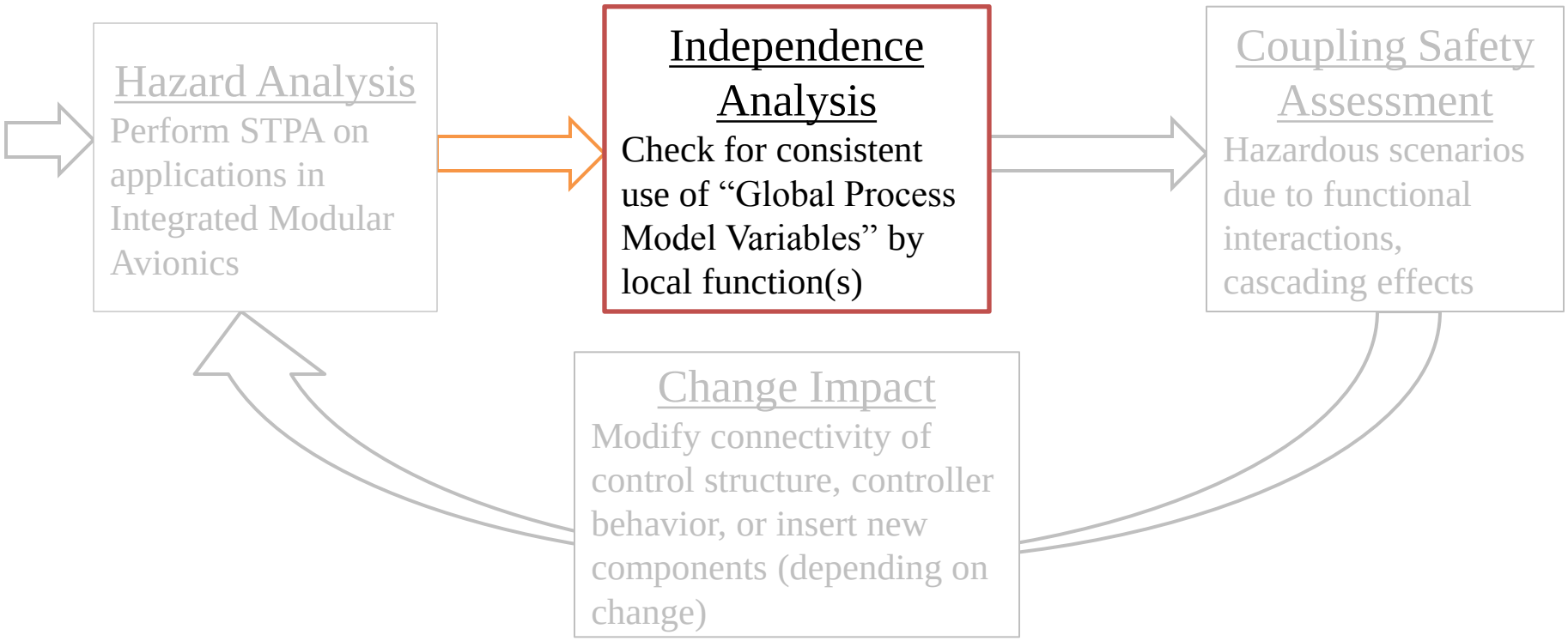
Causal Analysis – F.D. Display

Unsafe Control Actions

Causal Analysis



Proposed Methodology



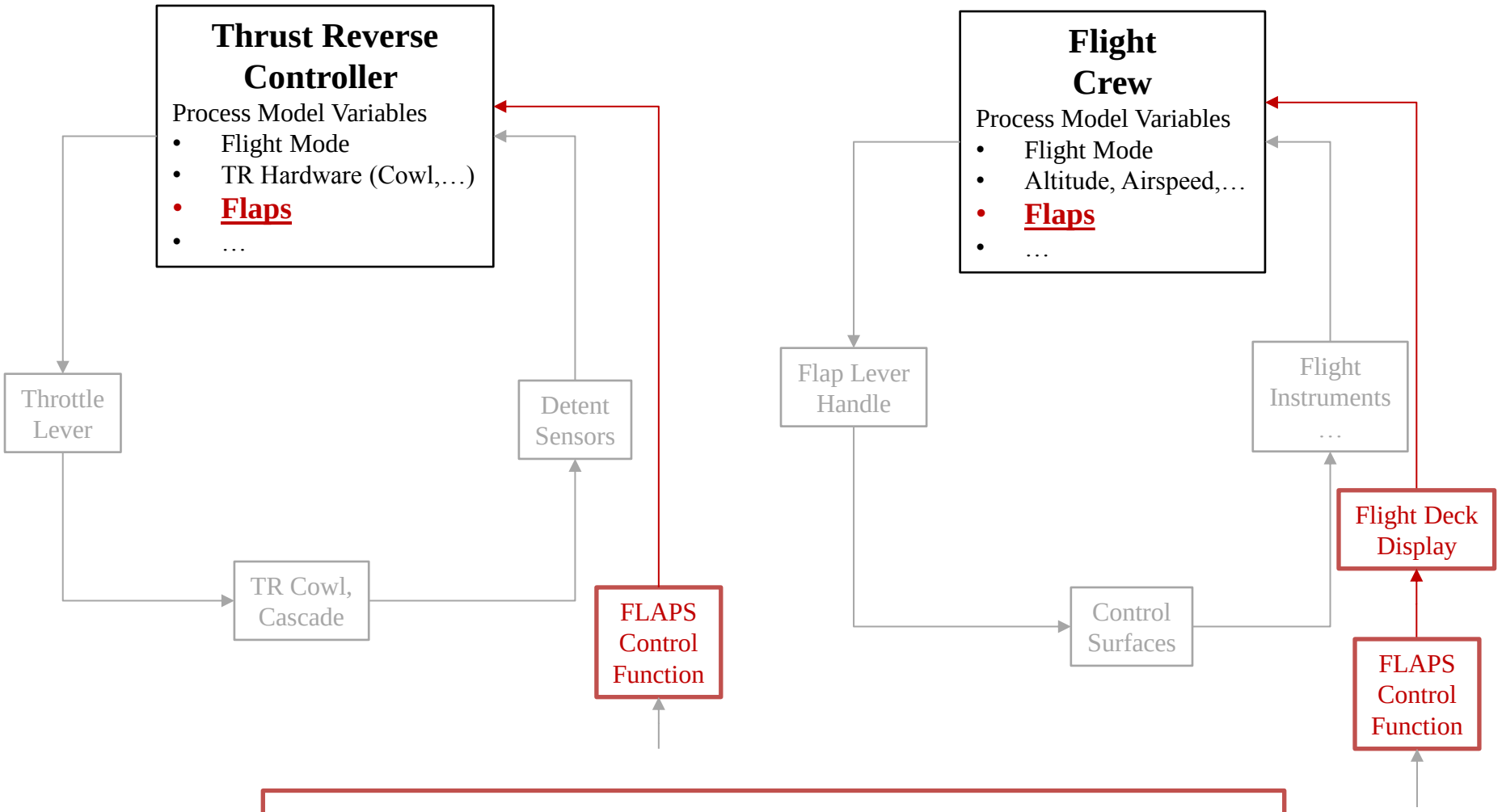
Independence Analysis

1. Identify Global Process Model Variable(s)

2. Examine each controller's use of the Global Process Model Variable

3. Analyze for potentially inconsistent use of Global Process Model Variable

Independence Analysis



How do these Controllers use the Global PM Variable to make (un)safe control actions?

Independence Analysis

Thrust Reverse Controller:

Needs "FLAPS EXTENDED" variable whenever flaps surface IS NOT in "0" detent

Assumptions:

Thrust Reverse Cntl risks impingement on flaps any time LE flaps are not stowed

Flight Deck Display:

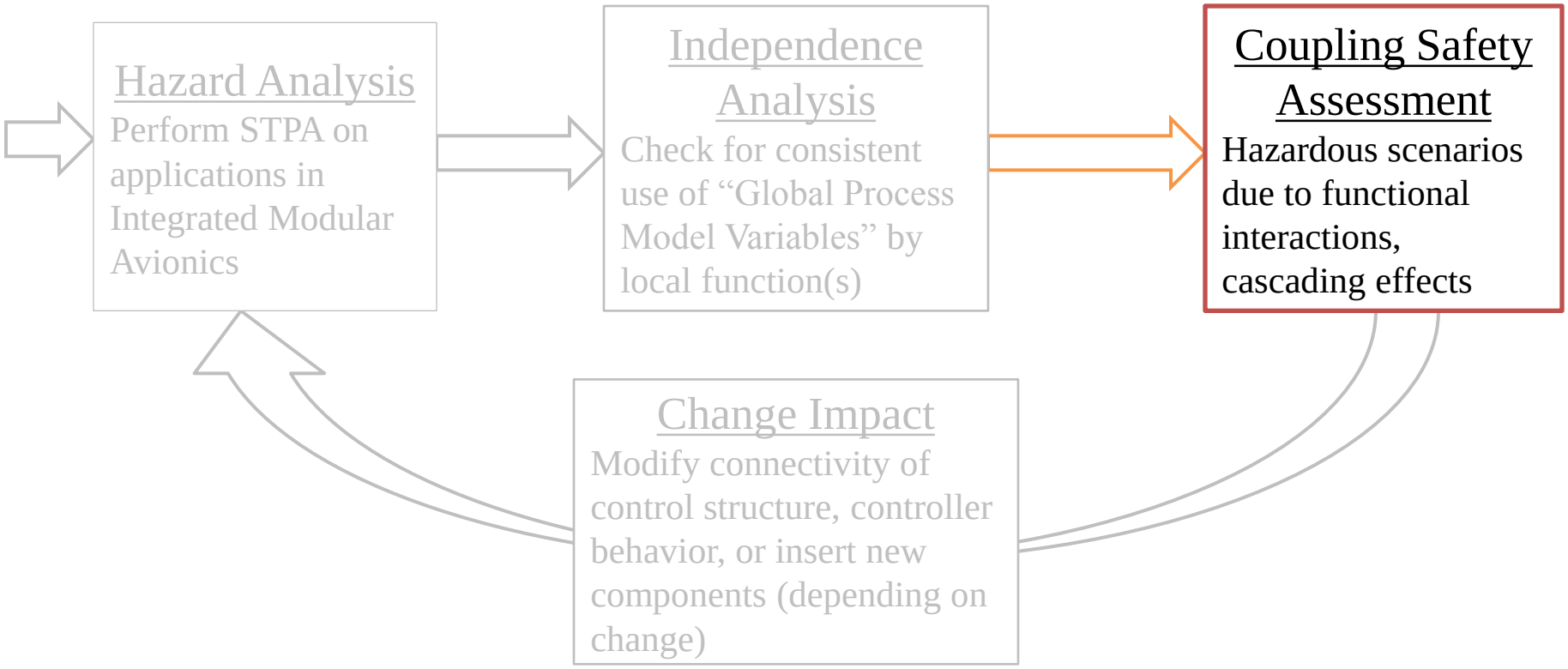
Needs "FLAPS EXTENDED" variable only when flap surface IS in "1" or greater detent

Assumptions:

Crew responsibility only complete when flap makes it fully to detent

- Thrust Reverse Cntl and Flight Deck Display *do not have direct interface*
- However, this analysis shows that *their behavior is not INDEPENDENT*

Proposed Methodology



Independence Analysis → Inconsistent uses of Global Process Model Variable

- ∴ Generate constraints on behavior w/r/t GPMV

Coupling Safety Assessment

Flaps EXTENDED generation logic should be changed, for example:

WAS:

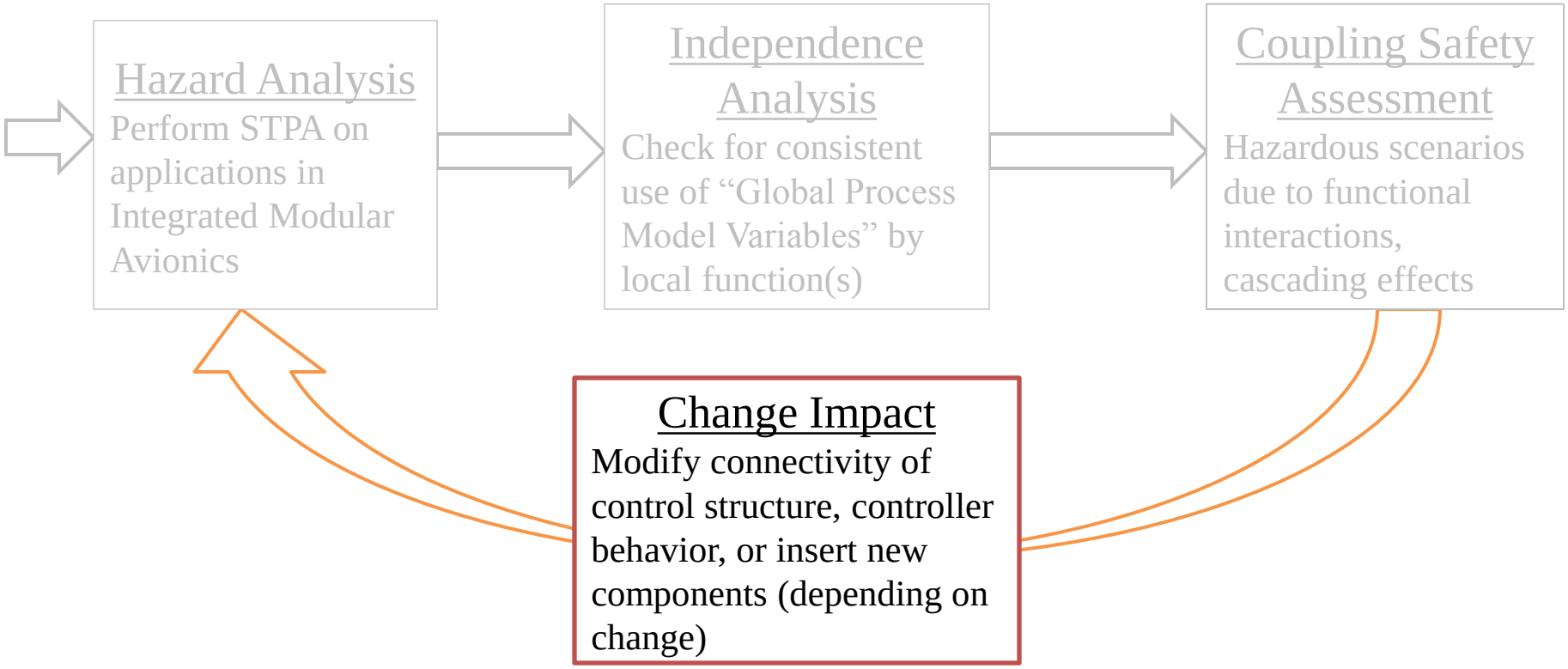
1. Flap surfaces detected in the “1” or greater flap detent, OR
2. Flap surfaces detected not in the “Up” flap detent, OR
3. …

CHANGE TO:

Flaps EXTENDED iff Flap surfaces detected in the “1” or greater flap detent

→ What does this do to the existing analysis?

Proposed Methodology



Change Impact Analysis

When changes are made, what components does it affect?
Can we reduce the amount of re-analysis?

1. Identify change:
   - Control structure
   - Component behavior
   - Information exchange between components

2. Analyze how assumptions in the previous analysis become invalid

Change Impact Analysis

Which assumptions in the analysis changed?

COMPONENT BEHAVIOR CHANGE:

Flaps EXTENDED iff Flap surfaces detected in the "1" or greater flap detent

<u>UCA</u>: TRC does 'Not Provide' thrust reverser OFF control law when LE flaps is between retracted and TBD° extension

<u>Cause</u>: Feedback Incorrect Flaps Discrete Function sends EXTENDED message if any sensor j is in "1" or greater detent.

The Thrust Reverse scenario still exists

<u>UCA</u> : Crew does 'Not Provide' Extend Flaps control action on approach, before flap is fully in "1" detent

<u>Cause</u>: Feedback Incorrect Flaps Discrete Function sends EXTENDED message if any sensor k is NOT in "0" detent.

Flight Deck Display scenario eliminated by this change, no re-analysis

Contributions

1. Created a methodology to analyze for hazardous behavior due to interaction between applications
    - Introduced the Global Process Model Variable to solve the problem
    - Method to analyze for consistency amongst controllers

2. Created Change Impact Hazard Analysis methodology
    - Analyzed how changes in behavior of one application affects another

Future Work

- Scalability

- Other types of coupling
  - e.g. when the behavior of one component directly influences another through control actions
  - Other types of data exchange

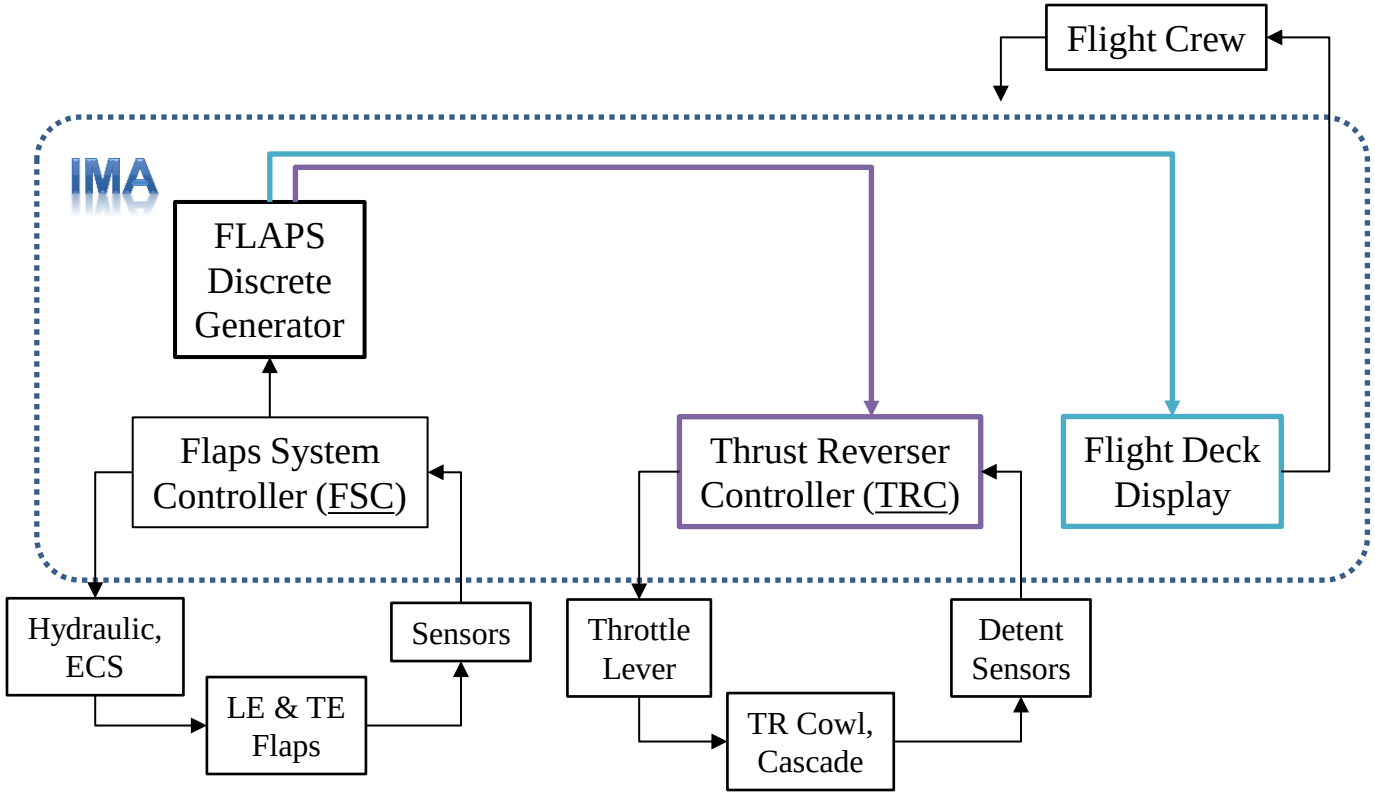
- Other types of changes
  - Connectivity (i.e. changes in control structure)
  - Timing

References

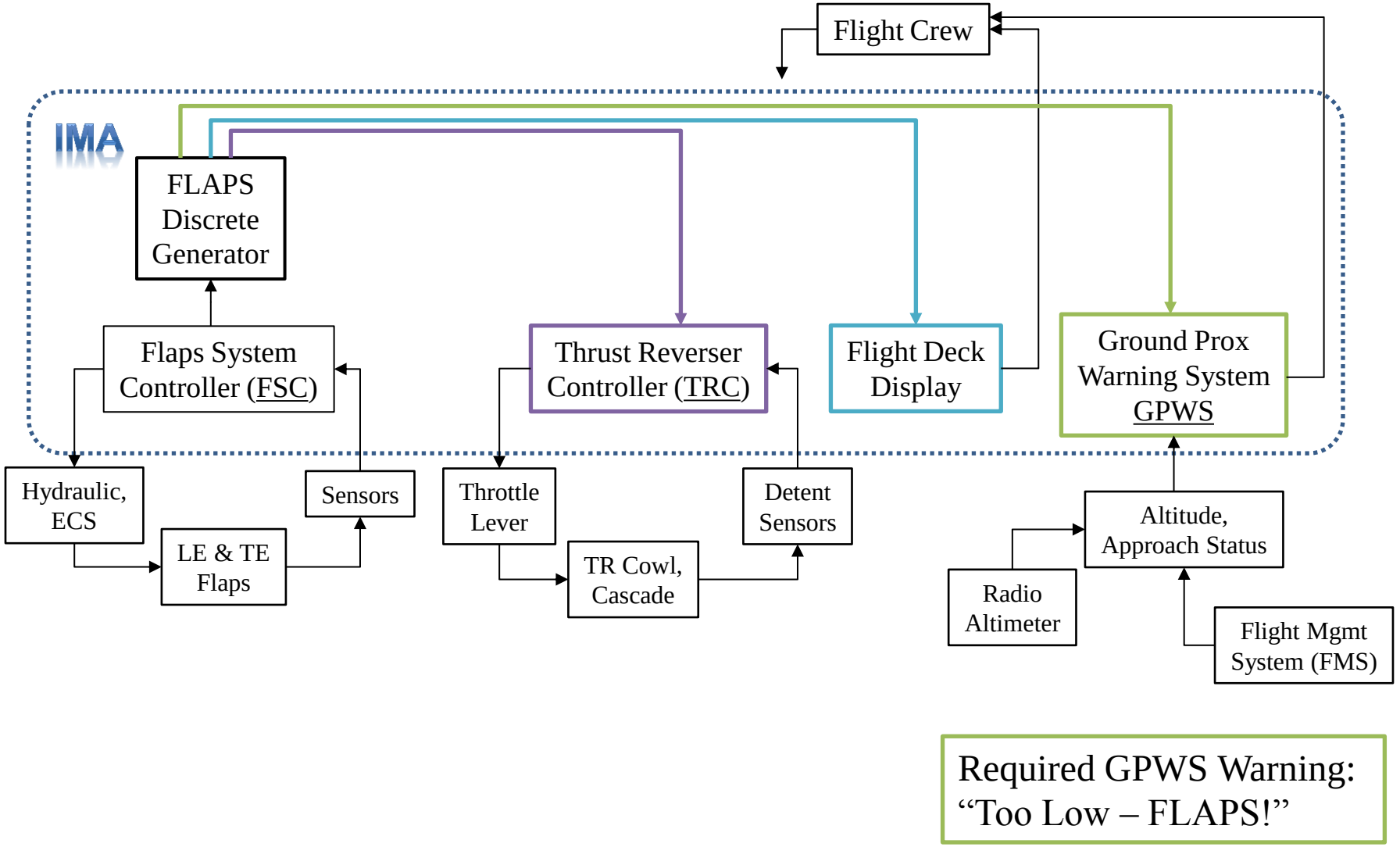
1. Baker, K. “Filling the FAA Guidance and Policy Gaps for Systems Integration and Safety Assurance” Systems”, 30th Digital Avionics Systems Conference (2011).
2. Bartley G., Lingberg B. “Certification Concerns of Integrated Modular Avionics (IMA) Systems”, 27th Digital Avionics Systems Conference (2008).
3. Betancourt, L. Birla, S. Gassino, J. Regnier, P. “Suitability of Fault Modes and Effects Analysis for Regulatory Assurance of Complex Logic in Digital Instrumentation and Control Systems” U.S. Nuclear Regulatory Commission, (2012).
4. Conmy P., McDermid J. “High level failure analysis for Integrated Modular Avionics”, 6th Australian Workshop on Safety Critical Systems and Software (2001)
5. Graydon, P., Kelly T. “Assessing Software Interference Management When Modifying Safety-Related Software”, SAFECOMP, Springer-Verlag (2012).
6. Lalli, V.R., Kastner, R.E., Hartt, H.N. Training Manual for Elements of Interface Definition and Control, NASA Reference Publication 1370 (1997).
7. Leveson, N. “Engineering a Safer World”, MIT Press (2012).
8. Prisaznuk, P. “ARINC 653 Role in Integrated Modular Avionics (IMA)”, 27th Digital Avionics Systems Conference (2008).
9. RTCA DO-178C “Software Considerations in Airborne Systems and Equipment Certification”, RTCA Incorporated, SC-205 (2011).
10. RTCA DO-297 “Integrated Modular Avionics (IMA) Development Guidance and Certification Considerations”, RTCA Incorporated, SC-200 (2011).
11. Rushby, J. “New Challenges In Certification For Aircraft Software” Proceedings of the Ninth ACM International Conference On Embedded Software (2011).
12. S–18. “Guidelines and methods for conducting the safety assessment process on civil airborne systems and equipment. Society of Automotive Engineers”, ARP4761 (1996).
13. Watkins C. “Integrated Modular Avionics: Managing the Allocation of Shared Intersystem Resources”, 25th Digital Avionics System Conference (2006).
14. Zimmerman, M. Lundqvist, K. Leveson, N. “Investigating the Readability of State-Based Formal Requirements Specification Languages”, International Conference on Software Engineering, (2002).

BACKUP

Original IMA



Technology Insertion – GPWS



Required GPWS Warning:
"Too Low – FLAPS!"

Change Analysis

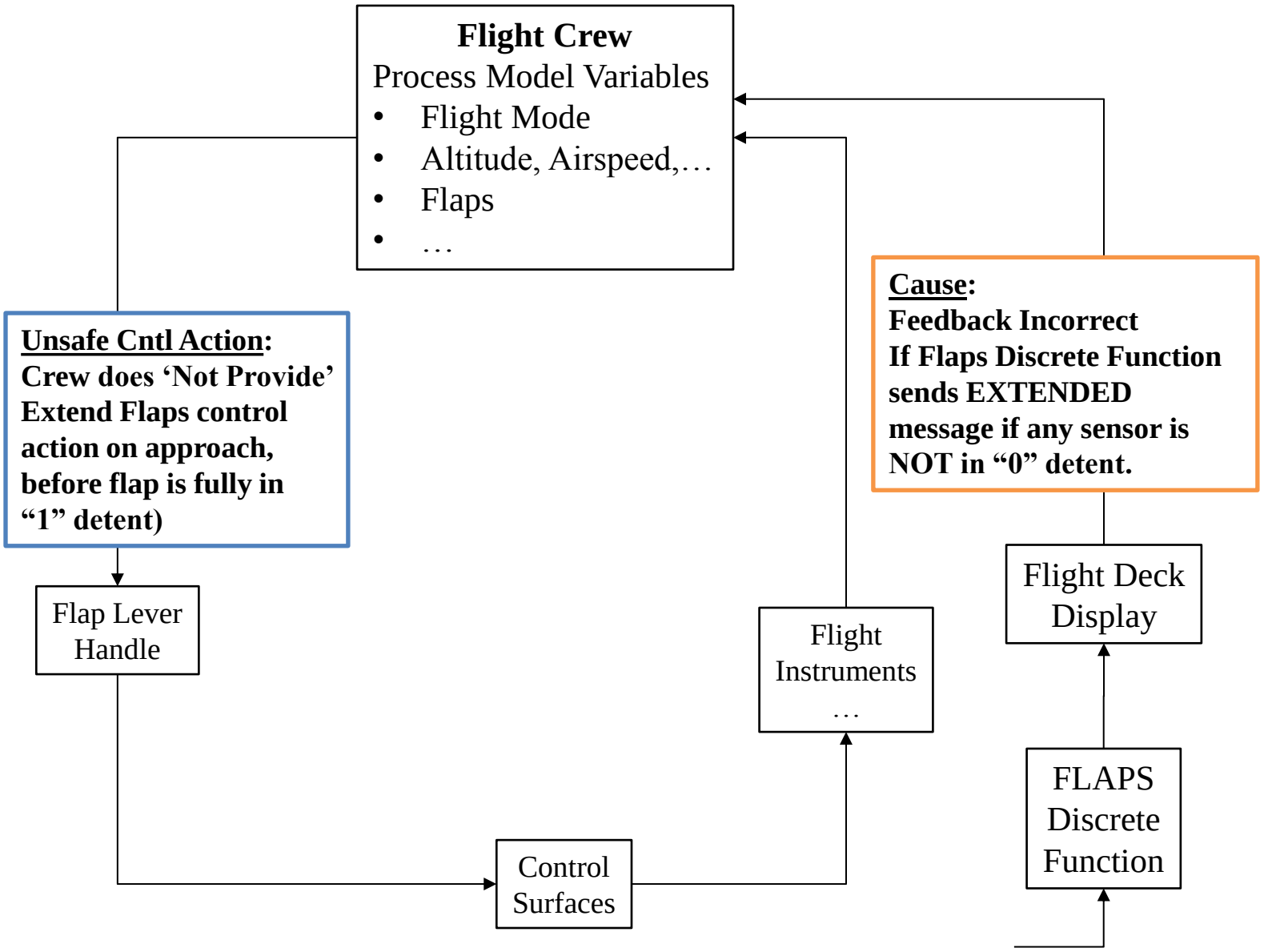
- Intuition (at least my intuition):
  - Ground Prox Warning System interfaces with Flaps Discrete Function only
  - Change impact should be minimal since *it does not exchange information* with Thrust Rev or Display functions

Change Analysis

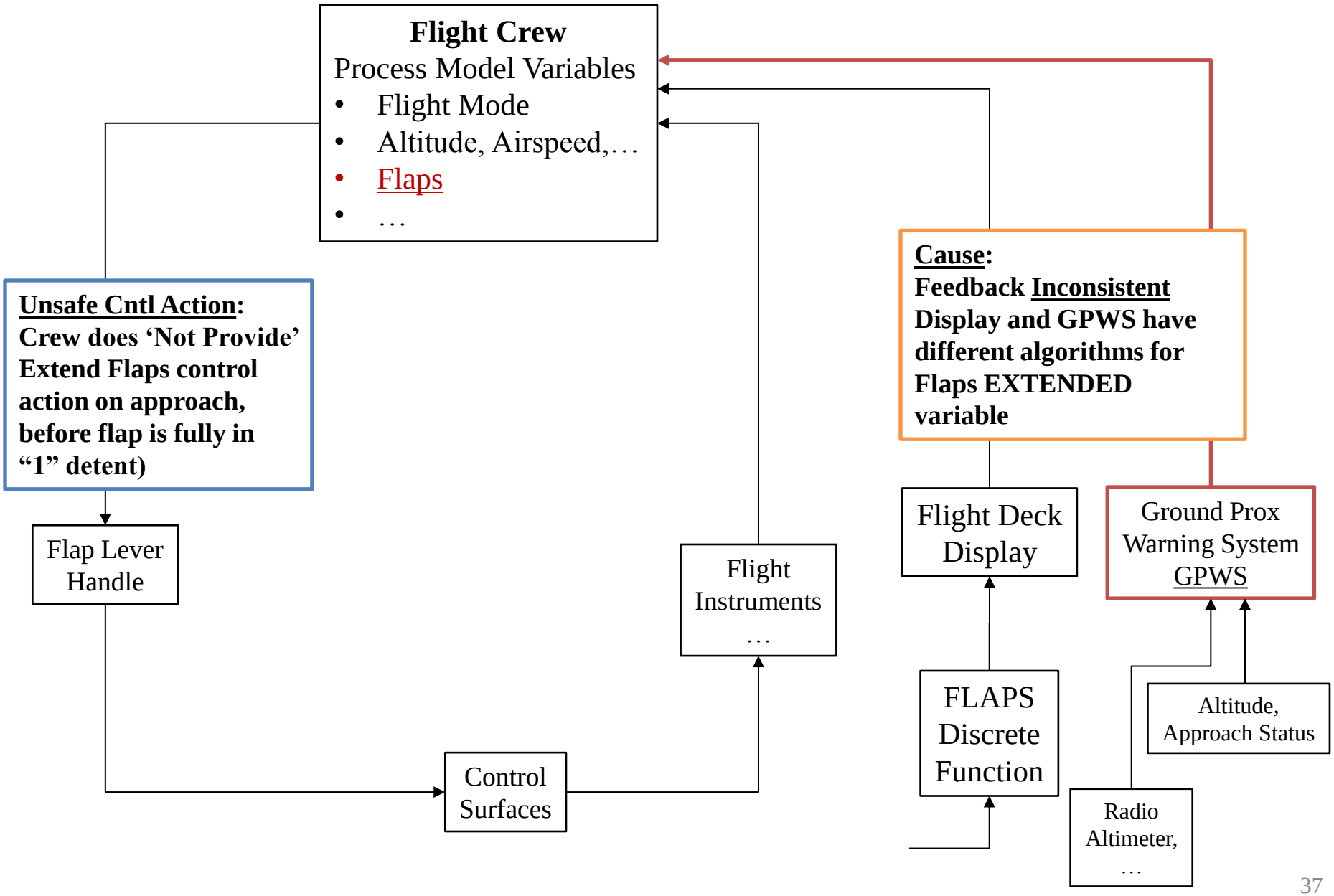
- But look at original STPA analysis:

Controller: Flight Crew	Not Provided when required for safety	Providing Causes Hazard	Too soon, too late, out of sequence	Stopped too soon, applied too long
Extend Flaps	Flaps not extended during takeoff or landing (insufficient lift during terminal ops, C_L)	LE flaps extended during thrust reversal (exhaust impingement) Flaps extended during cruise or excessive airspeed & density (flap overload)	Flaps extended too soon during approach (increased drag, loss of speed, flap overload) Flaps extended too late during approach (overspeed, missed runway)	Flaps do not achieve desired angle (e.g. stopped at incorrect discrete)

Change Analysis – WAS



Change Analysis – NOW



Implications

- Change Analysis must be top-down
  - But where is the “top”?
- System boundary is critical
  - In this case we must include the flight crew within the analysis
- Analysis demonstrates that Flight Deck Display and Ground Prox Warning System are indeed coupled

787

- …“the vast collection of components by hundred of suppliers that go into a 787 makes troubleshooting potentially more difficult. Although outsourcing has always been a part of commercial aviation, the difference now is the *complexity and co-dependence* of the electronics operating the aircraft.” [Dixon, Globe & Mail, 18 Jan 2013]

TAM - 3054

http://news.bbc.co.uk/2/hi/in_pictures/

- Both thrust levers were in CL (or "climb") position, with engine power being governed by the flight computer's autothrottle system. Two seconds prior to touchdown, an aural warning, "retard, retard," was issued by the flight's computer system, advising the pilots to "retard" the thrust lever to the recommended idle or reverse thrust lever position. This would disengage the aircraft's autothrottle system, with engine power then being governed directly by the thrust lever's position.
- At the moment of touchdown, the spoiler lever was in the "ARMED" position. According to the system logic of the A320's flight controls, in order for the spoilers to automatically deploy upon touchdown not only must the spoiler lever be in the "ARMED" position, but both thrust levers must be at or close to the "idle" position. The FDR transcript shows that immediately after the warning, the flight computer recorded the left thrust lever being retarded to the rear-most position, activating the thrust reverser on the left engine, while the right thrust lever (controlling the engine with the disabled thrust reverser) remained in the CL position. The pilots had only retarded the left engine to idle because they thought that without thrust reverser, the right engine did not need to be retarded as well. Airbus autothrust logic dictates that when one or more of the thrust levers is pulled to the idle position, the autothrust is automatically disengaged. Thus, when the pilot pulled the left engine thrust lever to idle it disconnected the autothrust system. Since the right engine thrust lever was still in the "climb" detent, the right engine accelerated to climb power while the left engine deployed its thrust reverser. The resulting asymmetric thrust condition resulted in a loss of control and a crash ensued. Moreover, the A320's spoilers did not deploy during the landing run, as the right thrust lever was above the "idle" setting required for automatic spoiler deployment

Lufthansa 2904

- Windshear → banked touch down
- Spoilers are only activated if either of these conditions are true:
  - Must be weight of over 12 tons on each main landing gear strut
  - Wheels of the plane must be turning faster than 133 km/h
- The thrust reversers are only activated if latter condition is true.
- There is no way for the pilot to override the software decision and activate either system manually.

http://www.airdisaster.com/photos/lh2904/2.shtml

B747-400 Incident (British Airways)

- All model 747 airplanes will automatically retract the Group ‘A’ LE flaps upon movement of the reverse thrust handle…to prevent thrust reverser efflux air from impinging directly onto the flap panel surfaces to improve the fatigue life of the panels and their attachments.
- During normal LE flap operation there is no separate indication on the flight deck for the position of the LE flaps. The expanded ‘FLAPS’ display appears automatically on the main EICAS for non-normal configurations
- During the takeoff roll the No. 3 ‘REV’ amber EICAS message displayed on the P2 – Pilots Center Instrument Panel. Some seconds later, a No. 2 engine ‘REV’ amber EICAS message displayed on the P2 – Center Instrument Panel.
- The ‘REV’ amber EICAS message indicated to the flight deck crew that the specific thrust reverser was out of the stowed and locked position and in transit [Note that in this case both engines #2 and #3 had one TR gearbox unlock, however the other locking gearbox and the air motor brake remained engaged and neither reverser deployed].
- The aircraft air/ground logic then signaled the Group ‘A’ LE flaps to redeploy (extend) and this occurred automatically.

Report No. CA18/3/2/0717, South African Incident Investigation

Moving Forward

- Does it scale?
  - Real project with airframe manufacturer
  - Existing hazard analysis ~2500 pages (FTA)
  - Change Management Log
  - Project engineers believe they are missing many scenarios, cannot manage existing documentation
- Retrospective
  - Does it capture past scenarios in past accidents / incidents? (TAM 3054, Lufthansa 2904, B747-400 Tambo Airport Report #CA18/3/2/0717, South African Incident Investigation

Behavioral Specification

Data: Sensor data from flap surface or flap lever handle
Result: Generate Flaps EXTENDED Variable; Control thrust reverse and generate Flight Deck display
initialization;
if {*Flap Surfaces* ∈ *"1"* ∨ *greater*} ∨
{*Flap surfaces* ∉ *"Up"* } ∨
{*Flap Lever Handle* ∈ *"1"* ∨ *greater*} ∨
{*Flap Lever Handle* ∉ *"Up"*} **then**
  Flaps Control Variable ← EXTENDED;

if *LE Flaps* ≙ *EXTENDED* **then**
  Thrust Reverse ← OFF;
else
  Thrust Reverse ← Safe;

if {*LE Flaps* ≙ *EXTENDED*} ∧ { *TE Flaps* ≙ *EXTENDED*} **then**
  Display ← "Flaps";
else
  Display ← OFF;

Algorithm 1: Original Flaps GPMV Logic

Data: Sensor data from flap surface or flap lever handle
Result: Generate Flaps EXTENDED Variable; Control thrust reverse and generate Flight Deck display
initialization;
if *Flap Surfaces* ∈ *"1"* ∨ *greater* **then**
  Flaps Control Variable ← EXTENDED (01);
else if *Flap Surfaces* ∈ *"0"* **then**
  Flaps Control Variable ← NOT EXT (10);
else if {*Flap Surfaces* ∈ *"Valid"* } ∧ {*Flap Surfaces*∉ {*"0"*∨*"1"* }} **then**
  Flaps Control Variable ← TRANSIT (11);
else
  Flaps Control Variable ← INVALID (00);

if *LE Flaps* ≙ {*EXTENDED* ∨ *TRANSIT* ∨ *INVALID*} **then**
  Thrust Reverse ← OFF;
else if *Flap Surfaces* ∈ *"NOT EXT"* **then**
  Thrust Reverse ← Safe;
else
  Thrust Reverse ← OFF;

if *Flaps Control Variable* ≙*EXTENDED* **then**
  Display ← "Flaps";
else if *Flap Control Variable* ∈ { *"NOT EXT"* ∨ *"Transit"*} **then**
  Display ← OFF;
else if *Flap Control Variable* ∈ *"INVALID"* **then**
  Display ← WARNING;

Algorithm 2: Modified Flaps GPMV Logic

Generate Requirements

Flaps Generation Function

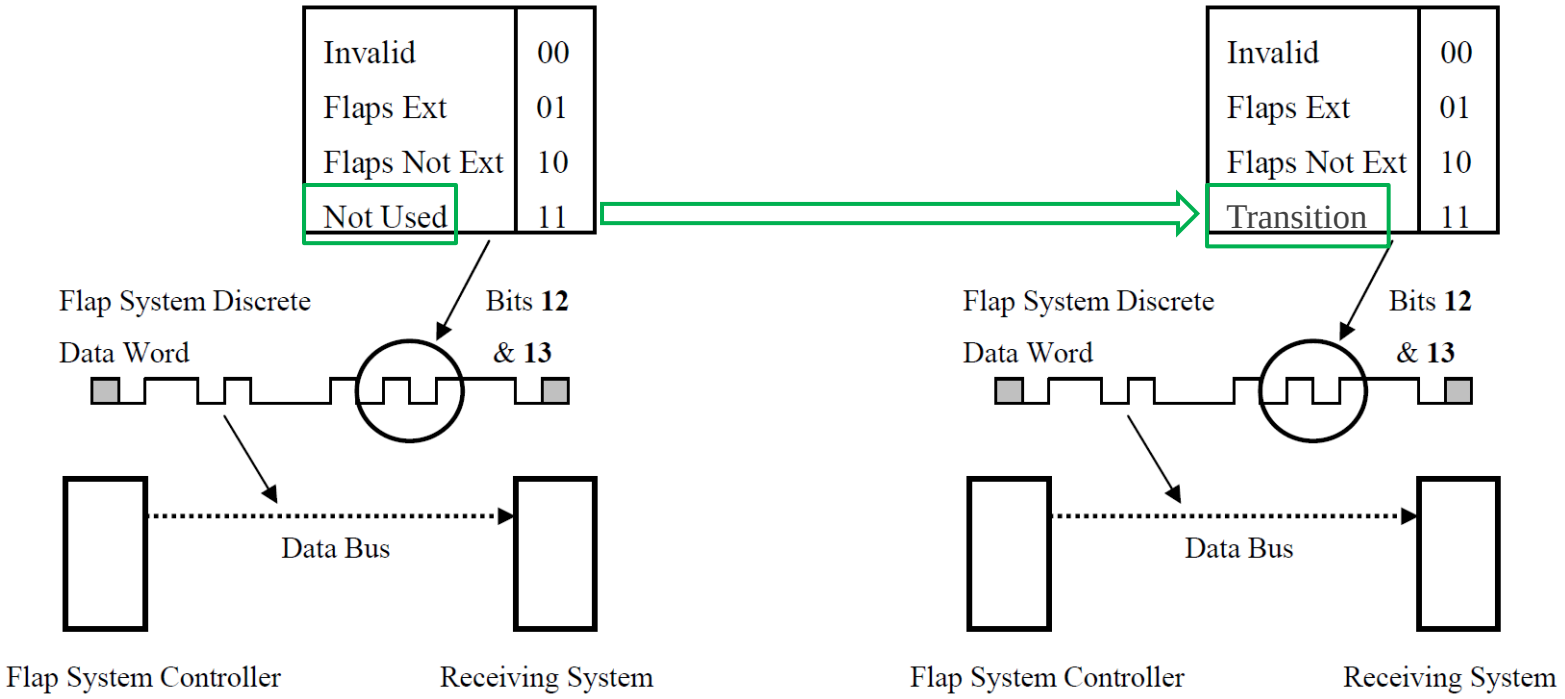
Was (e.g.):

- **If Flaps Position ≡ "1"**
  Flaps Discrete → Extended
- **ElseIf Flaps Pos ≥ "0"**
  Flaps Discrete → Not Ext
- **Else**
  Flaps Discrete → Invalid

Modified:

- **If Flaps Position ≡ "1"**
  Flaps Discrete → Extended
- **ElseIf Flaps Pos ≡ "0"**
  Flaps Discrete → Not Ext
- **ElseIf Flaps Pos "0" < P < "1"**
  Flaps Discrete → Transition
- **Else**
  Flaps Discrete → Invalid

Generate Requirements

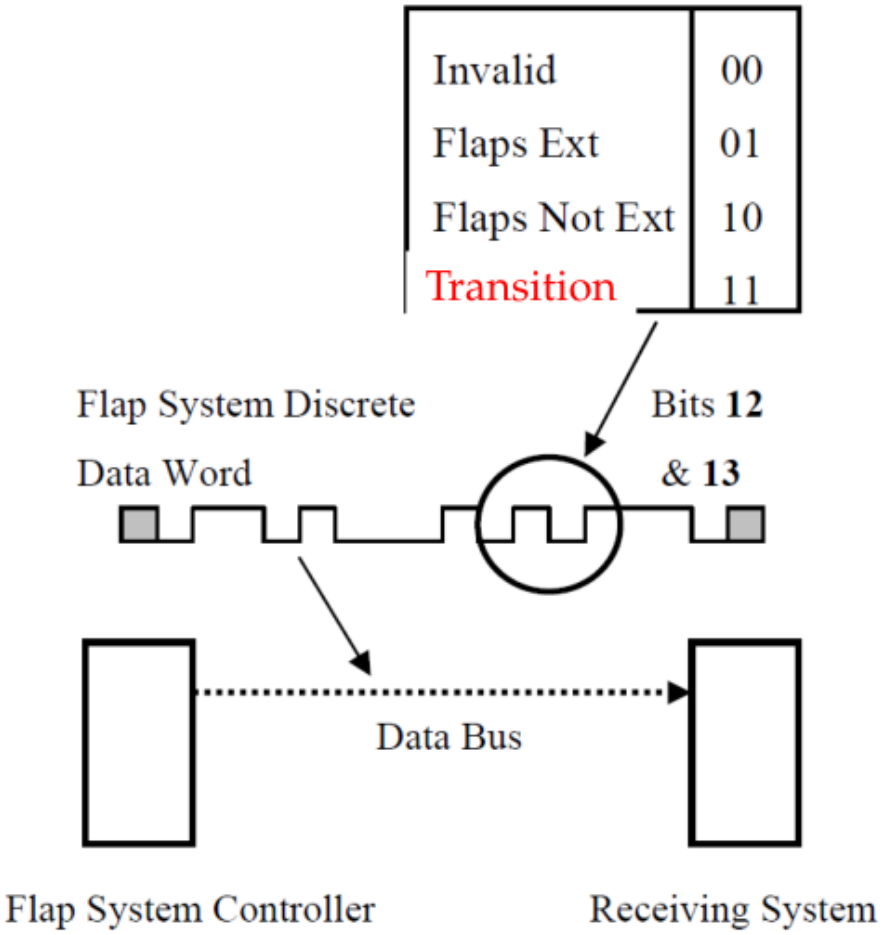


Behavioral Specification – Flaps Fnc

IMA - Flaps Discrete

FLAPS EXTENDED discrete should be modified to account for causes identified previously

- If LE ∧ TE flap surfaces ∈ {"1" ∨ greater} detent.
  → Flaps Ext (01)
- ElseIf LE ∧ TE surfaces ∈ {"0"} detent
  → Flaps Not Ext (10)
- ElseIf LE ∧ TE Flap Surfaces ∉ {0, 1} ∧ Range = Valid
  → Transition (11)
- Else*
  → Invalid (00)



"Conflicting" Causes

Without coupling of FCS and TRS via the common Flaps Discrete, we might have:

Flaps Control System

If LE $\vee$ TE Flaps Discrete = Extended (in "1" $\vee$ greater)
    Flap Control System $\rightarrow$ IDLE
ElseIf Flaps Discrete = Not Extended (Not in "1" $\vee$ greater)
    Flaps Control System $\rightarrow$ EXECUTE

Thrust Reverse System

If LE Flaps Discrete = Extended (Not in "0" Detent)
    Off Thrust Reverse $\triangleq$ Mandatory
Else Flaps Discrete = Not Extended (in "0")
    On Thrust Reverse $\triangleq$ Safe

Behavioral Specification – Thrust Rev

Use Hazard Analysis to modify the high-level algorithmic requirements of the Thrust Reverse System (using modified Flaps Discrete logic):

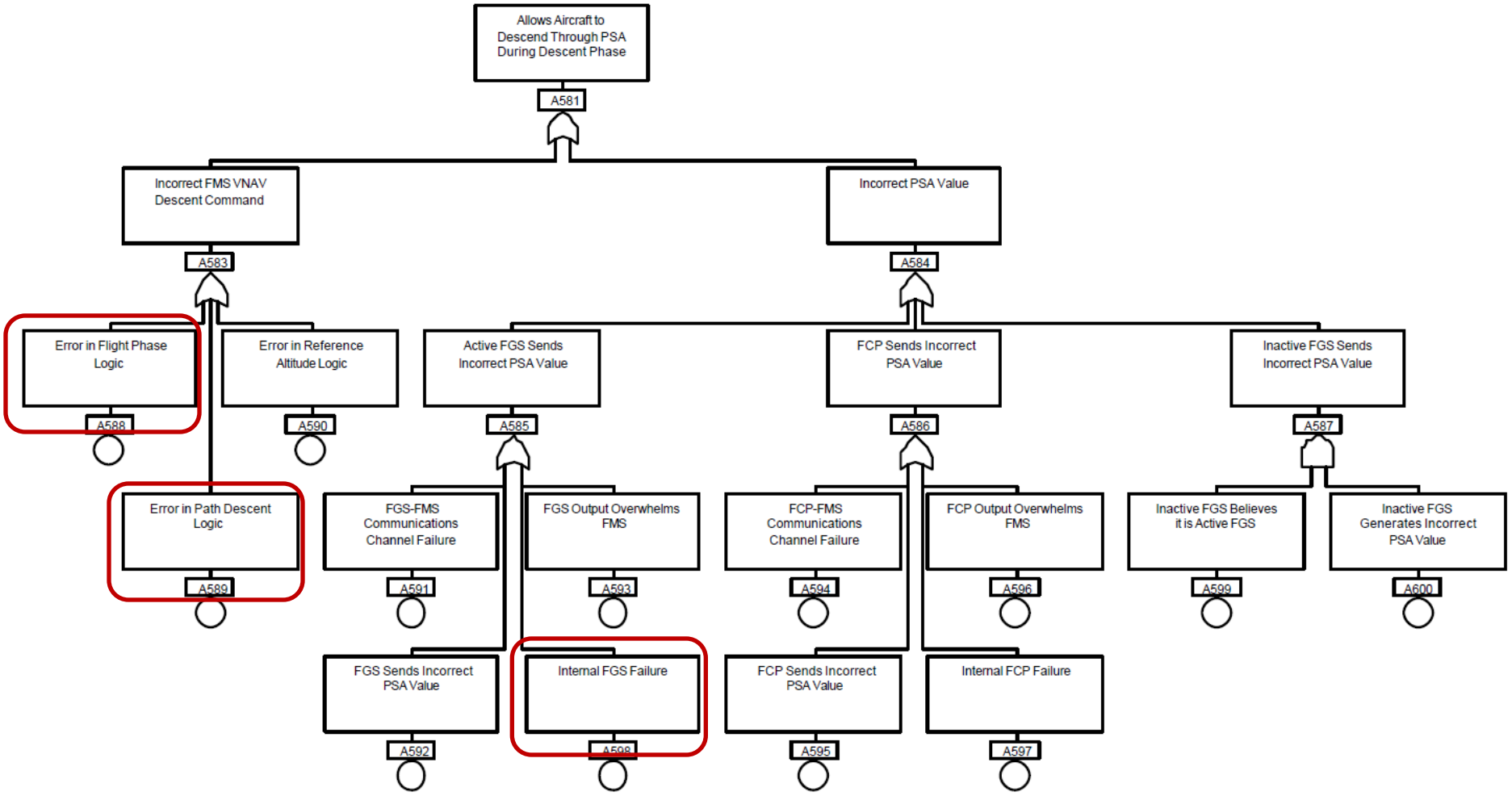
Was:

IF LE Flaps Discrete = EXTENDED (NOT in "0" Detent)
    OFF Thrust Reverse $\triangleq$ Mandatory
ELSE Flaps Discrete = NOT EXTENDED (in "0")
    ON Thrust Reverse $\triangleq$ SAFE

Modify to:

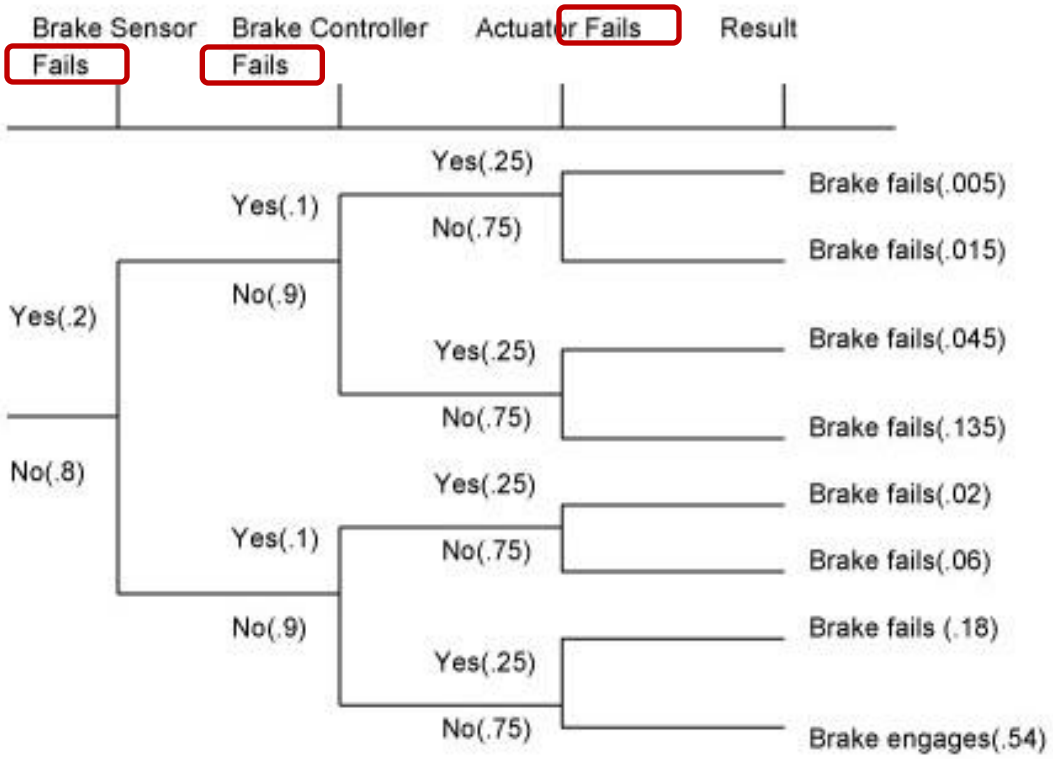
IF LE Flaps Discrete = {EXTENDED $\vee$ TRANSITION}(in "1" or greater Detent or moving in valid range)
    OFF Thrust Reverse $\triangleq$ Mandatory
ELSE Flaps Discrete = NOT EXTENDED (in "0")
    ON Thrust Reverse $\triangleq$ SAFE

Example Fault Tree



[Tribble & Miller 2003]

Example Event Tree



http://www.ece.cmu.edu/~koopman/des_s99/safety_critical/

Example FMEA

FAILURE MODE & EFFECTS ANALYSIS (FMEA)

Date: 1/1/2000
Revision: 1.3

Process Name: Left Front Seat Belt Install Process Number: SBT 445

Failure Mode	A) Severity Rate 1-10 10 = Most Severe	B) Probability of Occurance Rate 1-10 10 = Highest Probability	C) Probability of Detection Rate 1 - 10 10 = Lowest Probability	Risk Preference Number (RPN) AxBxC
1) Select Wrong Color Seat Belt	5	4	3	**60**
2) Seat Belt Bolt Not Fully Tightened	9	2	8	**144**
3) Trim Cover Clip Misaligned	2	3	4	**24**

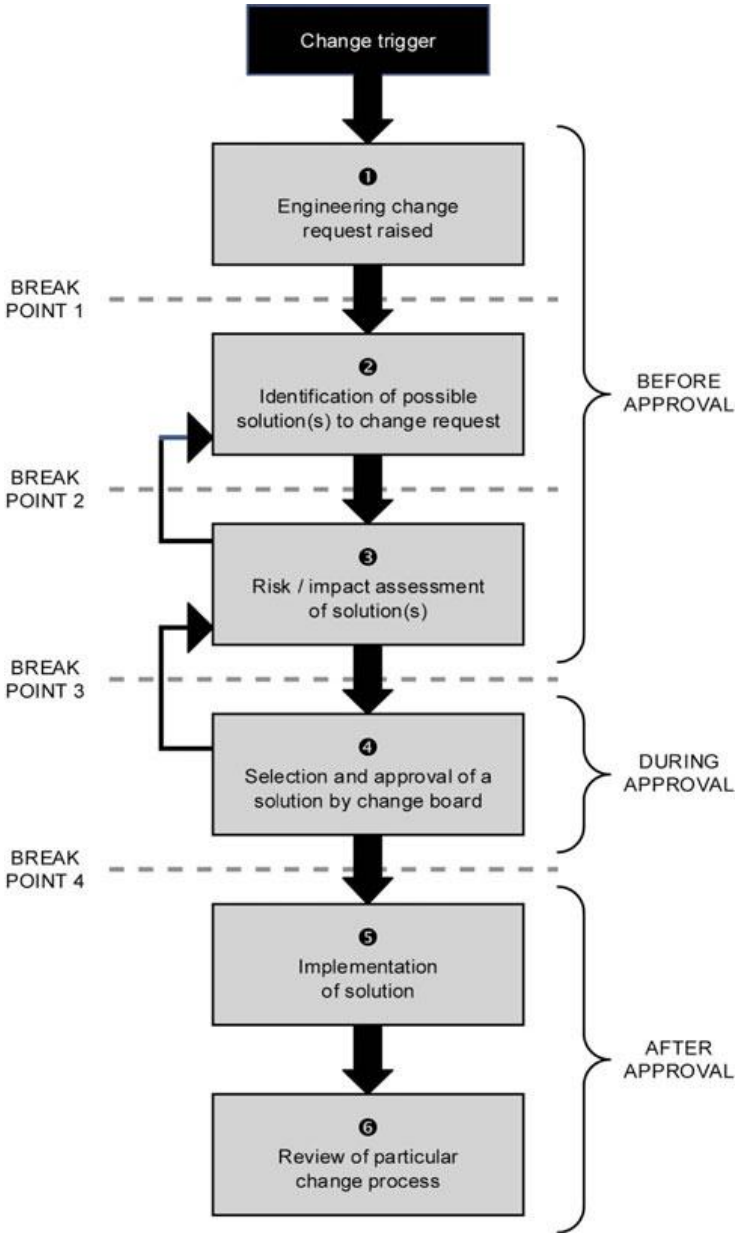
http://www.moresteam.com/toolbox/fmea.cfm

HAZOP

Parameter / Guide Word	More	Less	None	Reverse	As well as	Part of	Other than
Flow	high flow	low flow	no flow	reverse flow	deviating concentration	contamination	deviating material
Pressure	high pressure	low pressure	vacuum		delta-p		explosion
Temperature	high temperature	low temperature					
Level	high level	low level	no level		different level		
Time	too long / too late	too short / too soon	sequence step skipped	backwards	missing actions	extra actions	wrong time
Agitation	fast mixing	slow mixing	no mixing				
Reaction	fast reaction / runaway	slow reaction	no reaction				unwanted reaction
Start-up / Shut-down	too fast	too slow			actions missed		wrong recipe
Draining / Venting	too long	too short	none		deviating pressure	wrong timing	
Inertising	high pressure	low pressure	none			contamination	wrong material
Utility failure (instrument air, power)			failure				
DCS failure			failure				
Maintenance			none				
Vibrations	too low	too high	none				wrong frequency

http://en.wikipedia.org/wiki/Hazard_and_operability_study

Change Process



[Jarrett 2004]

Limitations

- Do these results contradict a central tenant of IMA?
  - That is, do these results negate the OEM ability to “plug & play”?
  - To some extent, yes
  - It was shown (briefly) that partitioning alone does not solve the safety problem – the FAA and the research community appear to agree
- So then the question becomes: can we reduce the regulatory certification burden whenever a new application is added, or an existing app is modified?
  - This research has not answered that question (it showed that iteration might be required to obtain consistency, but that “change” is within a type design)

Limitations

- This example was fairly high-level
  - Yet it asserts that there must be a top-down analysis
  - How far down do we have to go?

Future Directions

- One of the key tenets of enforcing safe behavior – Process Model consistency
  - One thing shown in this presentation is that process models can become inconsistent in the IMA/data network paradigm (if variables are not defined with enough precision)
  - A key to approaching an easily-upgradeable IMA is the idea of assuring PM consistency
    - There certainly will (should) not be as much freedom as described in [Bartley 08] and elsewhere
    - But if the OEM can assure that the update does not invalidate the assumptions embedded in the user systems' process models, then we may not have to do an entire re-analysis
- What would this look like?

Other Types of Coupling

- Addition of GPWS – coupling happens outside of IMA

Other Types of Coupling

- Control Coupling – FMS example

Unsafe Control Actions

- Four Ways Unsafe Control Can Occur
  1. A control action required for safety is not provided or is not followed
  2. An unsafe control action is provided that leads to a hazard
  3. A potentially safe control action provided too late, too early, or out of sequence
  4. A safe control action is stopped too soon or applied too long (for a continuous or non-discrete control action)

IMA RTOS UCAs

Controller: IMA - RTOS	Not Provided when required for safety	Providing Causes Hazard	Too soon, too late, out of sequence	Stopped too soon, applied too long
Generate Partition Resource Allocation	None of necessary functions required for safety can execute	Incorrect amount of memory and time provided for Flaps Discrete Function (FDF), Flaps Control System (FCS), Thrust Reverser System (TRS)	Partition started too late – functions needed to execute sooner	e.g. Only resources for FDF and FCS are generated, when all are needed for safety Partition closed too soon before functions complete Partition left open too long (next partition cannot start)
Allocate Flaps Discrete Function (FDF) to Partition x	FDF output needed by other parallel processes (inside or outside partition, inside or outside IMA)	Partition x does not contain the necessary memory and time allocation for FDF to perform	FDF generated after FCS performs its control function	

IMA RTOS UCAs

Controller: IMA - RTOS	Not Provided when required for safety	Providing Causes Hazard	Too soon, too late, out of sequence	Stopped too soon, applied too long
Allocate Flaps Control System (FCS) to Partition x	FCS control computation needed to change flaps state	Partition x does not contain the necessary memory and time allocation for FCS to perform	FCS performs flaps control function after FLAPS discrete generated FCS performed before TRS but TRS needs to go before	
Allocate Thrust Reverser System (TRS) to Partition x	TRS computation needed to maintain or change thrust reverse state	Partition x does not contain the necessary memory and time allocation for TRS to perform TRS does not need to perform and resources are wasted	TRS performs its control function after FDF generated TRS performed before FCS but FCS needs to go before	

FCS Unsafe Control Actions

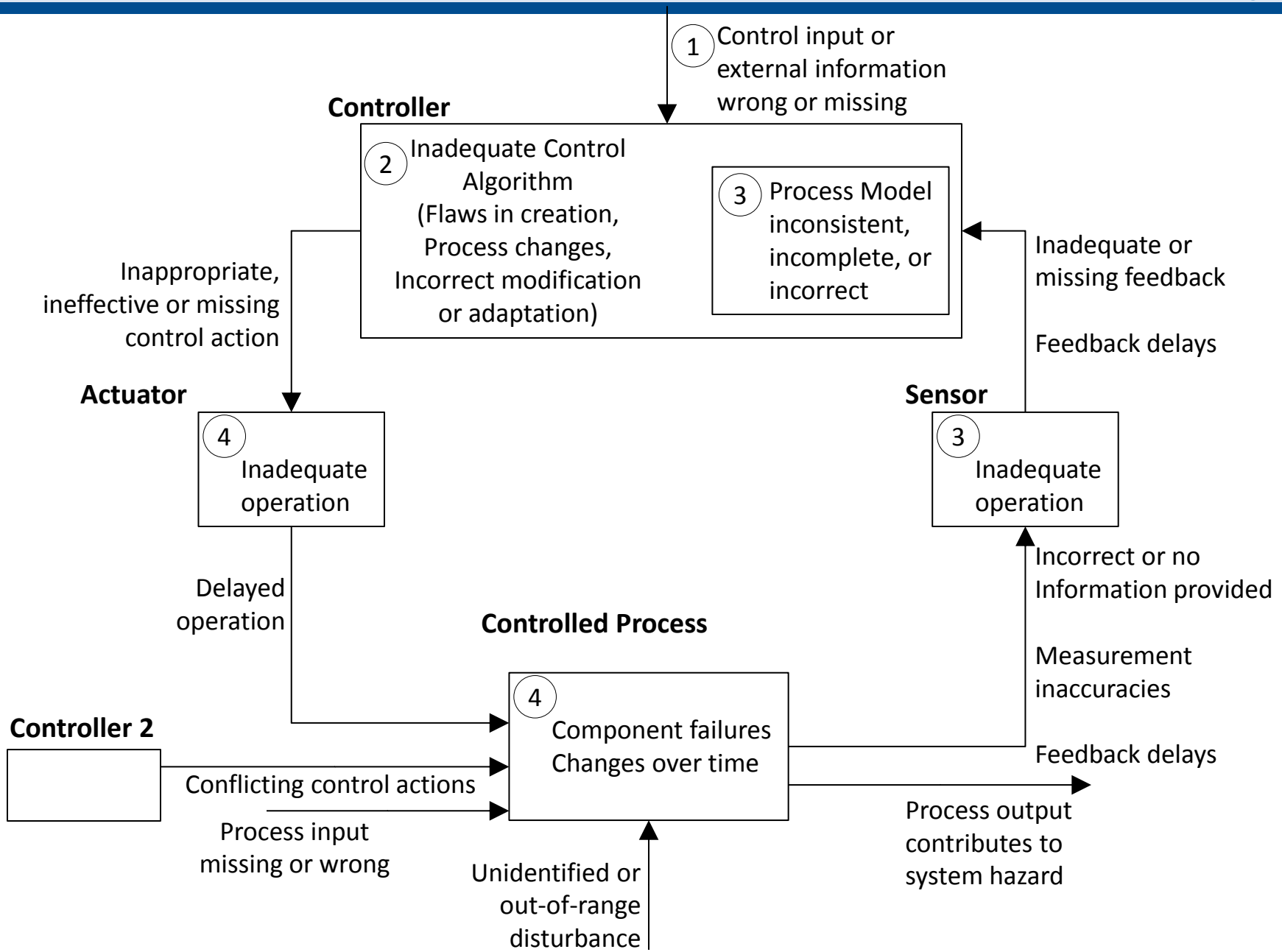
Controller: Flaps Ctlr Sys	Not Provided when required for safety	Providing Causes Hazard	Too soon, too late, out of sequence	Stopped too soon, applied too long
Extend Flaps	Flaps not extended during takeoff or landing (insufficient lift during terminal ops, C_L)	LE flaps extended during thrust reversal (exhaust impingement) Flaps extended during cruise or excessive airspeed & density (flap overload)	Flaps extended too soon during approach (increased drag, loss of speed, flap overload) Flaps extended too late during approach (overspeed, missed runway)	Flaps do not achieve desired angle (e.g. stopped at incorrect discrete)
Retract (Stow) Flaps	LE flaps not retracted during thrust reversal (exhaust impingement)	Flaps retracted during takeoff or landing (insufficient lift during terminal ops, C_L)	Retraction too late after takeoff (loss of speed, flap overload)	Not completely stowed (e.g. stopped at incorrect discrete)

TRS Unsafe Control Actions

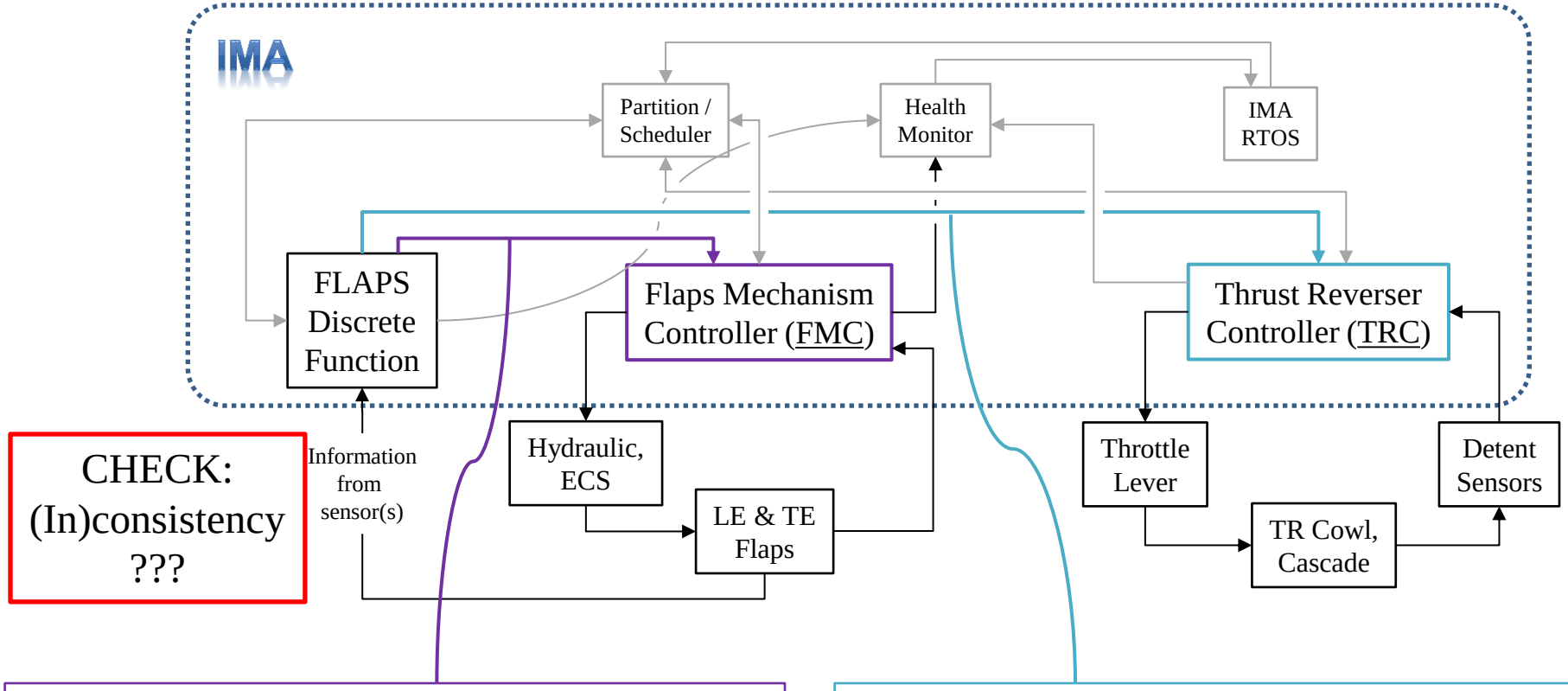
Controller: Thrust Rev Ctl	Not Provided when required for safety	Providing Causes Hazard	Too soon, too late, out of sequence	Stopped too soon, applied too long
Thrust Reverse	No thrust reverse on short runway* (runway overshoot) Rollout takes longer than expected (conflict with other taxiing/runway operations)	Reverse thrust during flight leads to loss of *v* and therefore lift Bypass air impinges on LE flaps	Reverse thrust applied too soon before landing, resulting in loss of airspeed during approach Applied too late during rollout (Needed when C_L and high *v* limit effectiveness of friction brakes located on landing gear)	Stopped before aircraft reaches desired speed on runway

* Regulations dictate that an aircraft must be able to land on a runway without the use of thrust reversers in order to be certified to land there as part of scheduled airline service

STPA Step 2 – Causal Analysis



Causal Analysis



Feedback INCORRECT:
Algorithm for generating discrete is different **than what Controller j has in process model – (TRS)**
(e.g. Flaps Discrete Function sends EXTENDED message if any sensor k is NOT in "0" detent. FMC does 'Not Provide' FLAPS extend control law on rotation, before flap is fully in "1" detent)

Feedback INCORRECT:
Algorithm for generating discrete is different **than what Controller i has in process model – (FCS)**
(e.g. Flaps Discrete Function sends EXTENDED message if any sensor j is in "1" or greater detent. TRC does 'Not Provide' thrust reverser OFF control law when LE flaps is between retracted and TBD° extension)

Proposed Approach

- Use a systems-based hazard analysis methodology, because *safety is an emergent property*
  - ICD & Robust Partitioning assumes that safety can be analyzed at the component level
- Control functions in an aircraft behave hazardously when their *process models are inconsistent with reality*
  - Inconsistent process models are due to faulty hardware…
  - …but they are also due to inadequate or late feedback, feedback from incorrect sources, etc.

Methodology

- Flag structural changes at the system level
  - Do “edges” or input/feedback links in the control structure change?
  - Are “nodes” (sensors, controllers, actuators…) introduced or deleted?
- Flag changes in blackbox behavior at the component level
  - Changes in structure (previous bullet) account for changes in Input/Output relationships
  - Changes in Blackbox behavior result in different output for a given set of inputs (need to re-word this)
- Introduction of “Global Process Model Variable”
  - Allows for…IIIIIIIIIIIIIIIIII

Change Management

- Specify structure
  - "Edges" in graph theory parlance
  - In other words, what goes into each node, and what comes out of each node?
- Specify component (node) black box behavior
  - This is based on hazard analysis that accounts for coupling between the nodes (and associated timing/missing/… causes of hazardous behavior)
- A change in either the structure or the BB behavior of a node will trigger some re-analysis
  - This looks somewhat like ΔDSM (nodes, edges, changes)
  - Difficult to capture Process Model inconsistencies with a DSM approach – how to capture timing / inconsistent feedback, etc?

Limitations of Current Approach

1. Capturing hazardous behavior *due to component interaction*, which will become much more prevalent in an IMA regime.

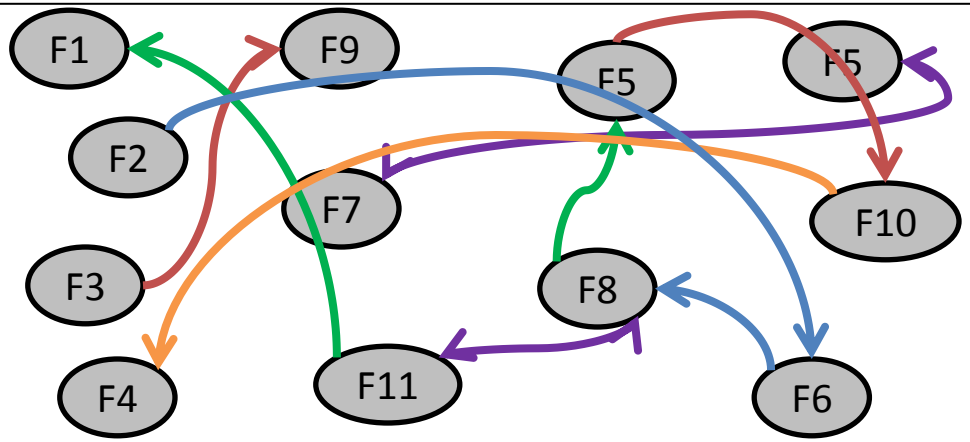
   [Baker 2011, Bartley 2008, Betancourt 2012, Conmy & McDermid 2001, Rushby 2011]

2. Change Management – very little guidance & *assumes partitioning will isolate* modified functions.

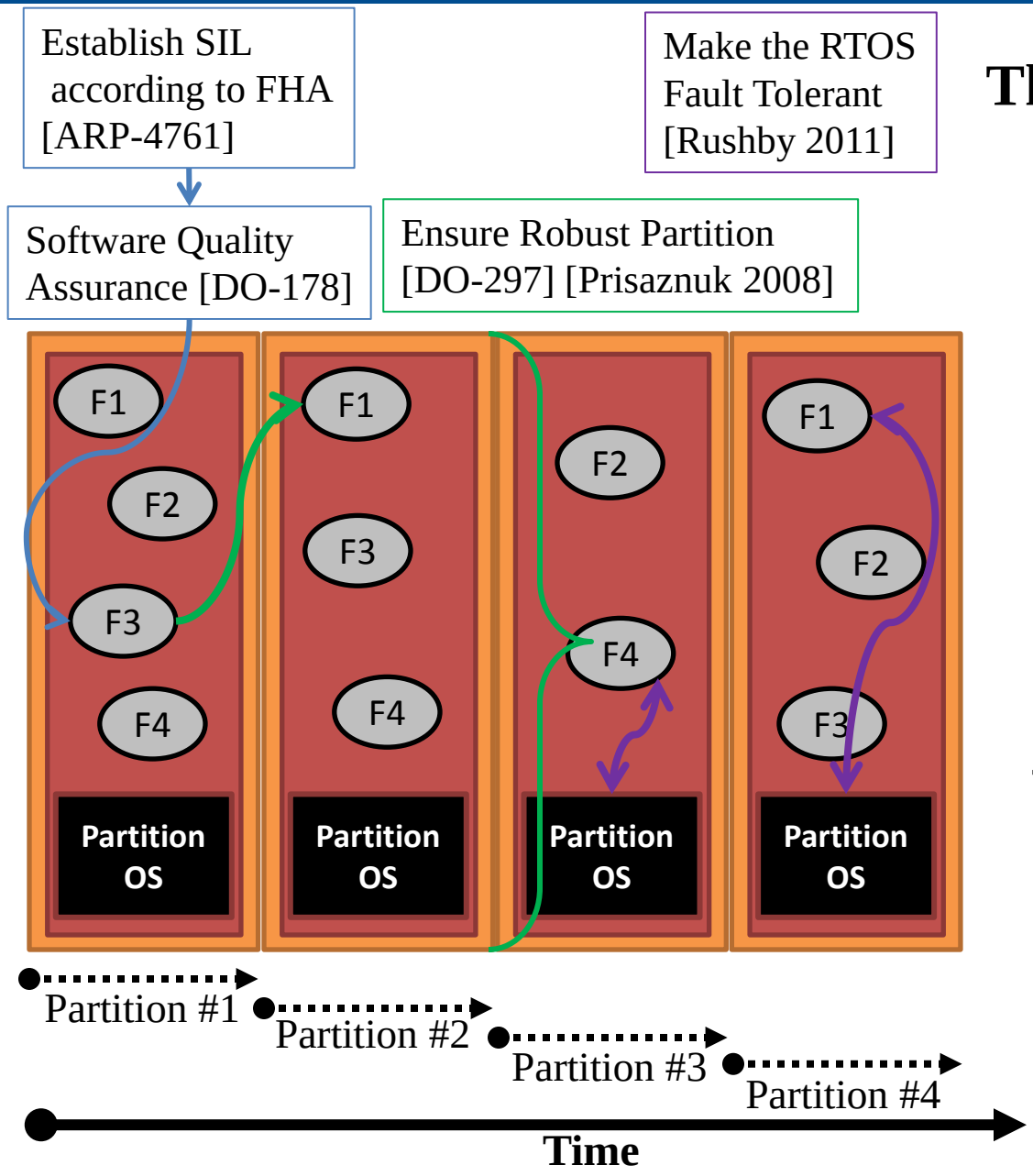
   [Bartley 2008, Graydon & Kelly 2012, Watkins 2007]

How to analyze all these interactions?

All of these functions may be developed independently, by different companies [Prisaznuk 2008]



IMA Regulatory Approach: Partitioning



This approach is limited w/r/t:

1. Capturing hazardous behavior *due to component interaction*, which will become much more prevalent in an IMA regime. [Baker 2011, Bartley 2008, Betancourt 2012, Conmy & McDermid 2001, Rushby 2011]
2. Change Management – very little guidance & *assumes partitioning will isolate* modified functions. [Bartley 2008, Graydon & Kelly 2012, Watkins 2007]

8110.49 Software Approval Guidelines

1. **Traceability analysis** identifies areas that could be affected by the software change. This includes the analysis of affected requirements, design, architecture, code, testing and analyses, as described below:
   - **(a) *Requirements and design analysis*** identifies the software requirements, software architecture, and safety-related software requirements impacted by the change. Additionally, the analysis identifies any additional features and/or functions being implemented in the system, assures that added functions are appropriately verified, and assures that the added functions do not adversely impact existing functions.
   - **(b) *Code analysis*** identifies the software components and interfaces impacted by the change.
   - **(c) *Test procedures and cases analysis*** identifies specific test procedures and cases that will need to be reexecuted to verify the changes, identifies and develops new or modified test procedures and cases (for added functionality or previously deficient testing), and assures that there are no adverse effects as a result of the changes. The absence of adverse effects may be verified by conducting regression testing at the appropriate hierarchical levels (such as aircraft flight tests, aircraft ground tests, laboratory system integration tests, simulator tests, bench tests, hardware/software integration tests, software integration tests, and module tests), as appropriate for the software level(s) of the changed software.
2. **Memory margin analysis** assures that memory allocation requirements and acceptable margins are maintained.
3. **Timing margin analysis** assures that the timing requirements, central processing unit task scheduling requirements, system resource contention characteristics, interface timing requirements, and acceptable timing margins are maintained.
4. **Data flow analysis** identifies changes to data flow and coupling between components and assures that there are no adverse impacts.
5. **Control flow analysis** identifies changes to the control flow and coupling of components and assures that there are no adverse impacts.
6. **Input/output analysis** assures that the change(s) have not adversely impacted the input and output (including bus loading, memory access, and hardware input and output device interfaces) requirements of the product.
7. **Development environment and process analyses** identify any change(s), which may adversely impact the software application or product (for example, compiler options or versions and optimization change; linker, assembler, and loader instructions or options change; or software tool change).
8. **Operational characteristics analysis** evaluates that changes (such as changes to gains, filters, limits, data validation, interrupt and exception handling, and fault mitigation) do not result in adverse effects.
9. **Certification maintenance requirements (CMR) analysis** determines whether new or changed CMRs are necessitated by the software change.
10. **Partitioning analysis** assures that the changes do not impact any protective mechanisms incorporated in the design

8110.49 Software Approval Guidelines

a) Previous hazards, identified by the system safety assessment, are changed.

b) Failure condition categories, identified by the system safety assessment, are changed.

c) Software levels are changed, particularly if the new software level is higher than the previous level.

d) Safety-related requirements, identified by the system safety assessment, are changed.

e) Safety margins are reduced.

- Motivation
- Approach
- Analysis
- Conclusions

- Motivation
- Approach
- Analysis
- Conclusions

- Motivation
- Approach
- Analysis
- Conclusions

- Motivation
- Approach
- Analysis
- Conclusions