# System Safety

Prof. Nancy G. Leveson

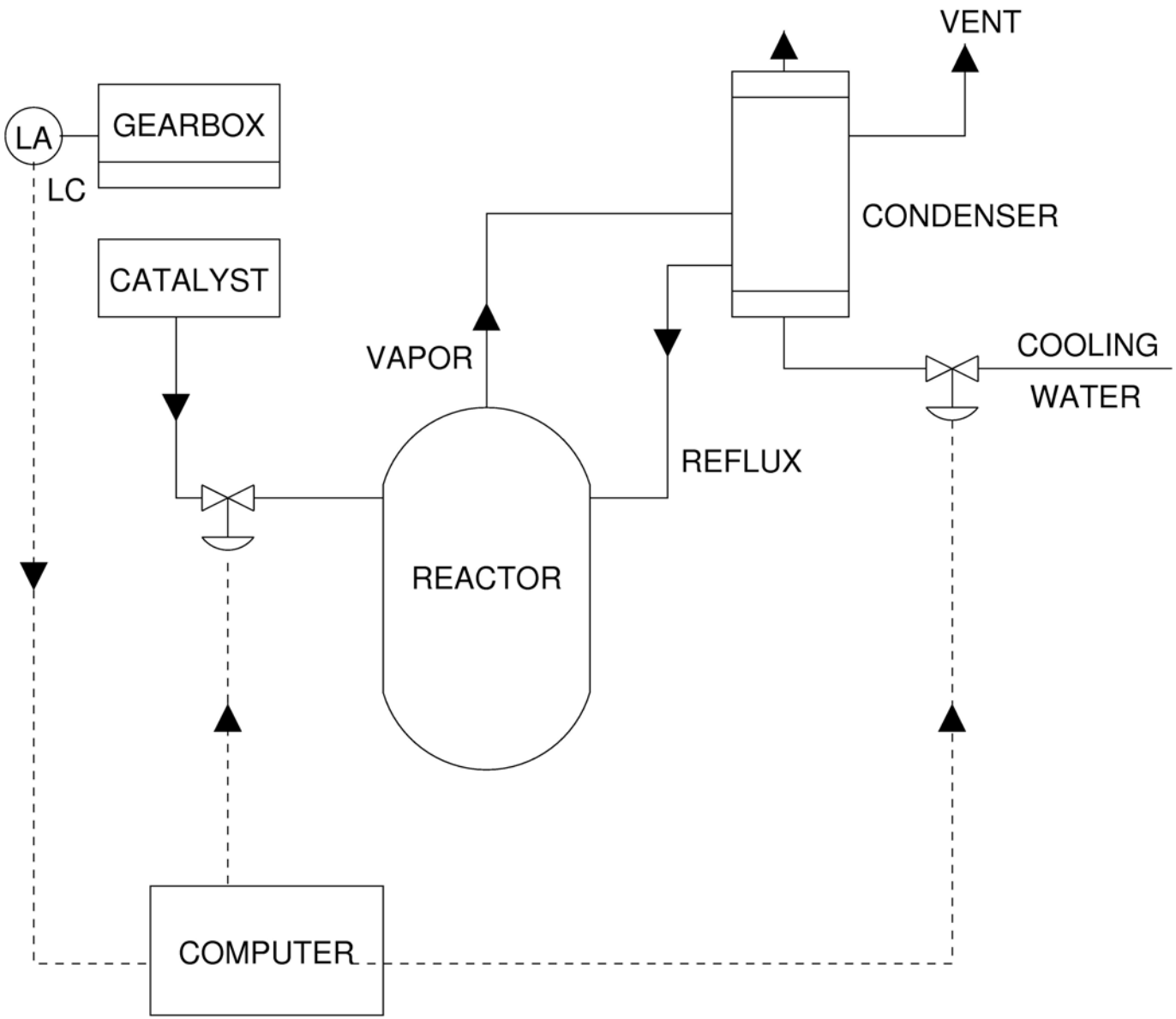
*"It's never what we don't know that stops us. It's what we do know that just ain't so."*

*Dean Kamen*

*"Without changing our patterns of thought, we will not be able to solve the problems we created with our current patterns of thought."*

*Albert Einstein*

# Accident with No Component Failures



# Types of Accidents

- Component Failure Accidents
  - Single or multiple component failures
  - Usually assume random failure
- Component Interaction Accidents
  - Arise in interactions among components
  - Related to
    - Interactive complexity and tight coupling
    - Use of computers and software

# Interactive Complexity

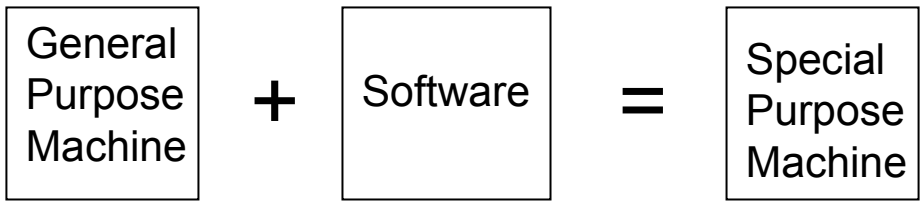
- Critical factor is intellectual manageability
  - A simple system has a small number of unknowns in its interactions (within system and with environment)
  - Interactively complex (intellectually unmanageable) when level of interactions reaches point where can no longer be thoroughly
    - Planned
    - Understood
    - Anticipated
    - Guarded against

# Safety vs. Reliability

- Safety and reliability are NOT the same
  - Sometimes increasing one can even decrease the other.
  - Making all the components highly reliable will have no impact on component interaction accidents.
- For relatively simple, electro-mechanical systems with primarily component failure accidents, reliability engineering can increase safety.
- For complex, software-intensive or human-intensive systems, we need something else.

# Software Changes System Engineering

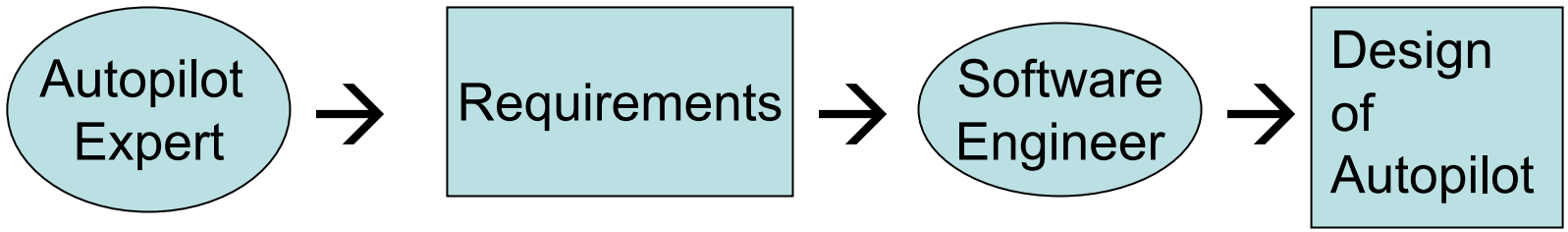
- Software is simply the design of a machine abstracted from its physical realization




- Software “failure” modes are different (do abstractions fail?)
  - Usually does exactly what you tell it to do
  - Problems occur from operation, not lack of operation
  - Usually doing exactly what software engineers wanted

# Abstraction from Physical Design

- Software engineers are doing system design




- Most operational software errors related to requirements (particularly incompleteness)

# Software-Related Accidents

- Are usually caused by flawed requirements
  - Incomplete or wrong assumptions about operation of controlled system or required operation of computer
  - Unhandled controlled-system states and environmental conditions

- Merely trying to get the software “correct” or to make it reliable will not make it safer under these conditions.

# Software-Related Accidents (2)

- Software may be highly reliable and “correct” and still be unsafe:
  - Correctly implements requirements but specified behavior unsafe from a system perspective.
  - Requirements do not specify some particular behavior required for system safety (incomplete)
  - Software has unintended (and unsafe) behavior beyond what is specified in requirements.
- While these things true for hardware, we can thoroughly test hardware and get out requirements and design errors
  - Can only test a small part of potential software behavior

# MPL Requirements Tracing Flaw

| SYSTEM REQUIREMENTS | SOFTWARE REQUIREMENTS |
| --- | --- |
| 1. The touchdown sensors shall be sampled at 100-HZ rate. | 1. The lander flight software shall cyclically check the state of each of the three touchdown sensors (one per leg) at 100-HZ during EDL. |
| 2. The sampling process shall be initiated prior to lander entry to keep processor demand constant. | 2. The lander flight software shall be able to cyclically check the touchdown event state with or without touchdown event generation enabled. |
| 3. However, the use of the touchdown sensor data shall not begin until 12 m above the surface. | ???? |

# Software Safety

Safety involves more than simply getting the software correct:

Example: Altitude Switch (Sends out signal when a specific altitude reached)

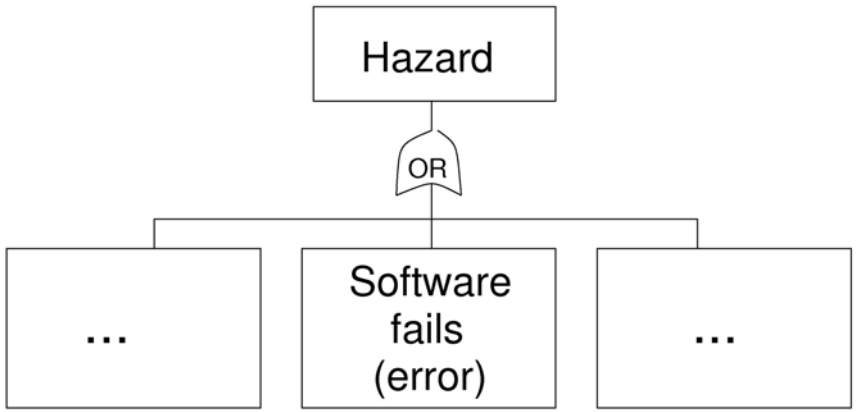
1. Signal safety-increasing ➔

   Require <u>any</u> of three altimeters to report below threshhold

2. Signal safety-decreasing ➔

   Require <u>all</u> three altimeters to report below threshhold

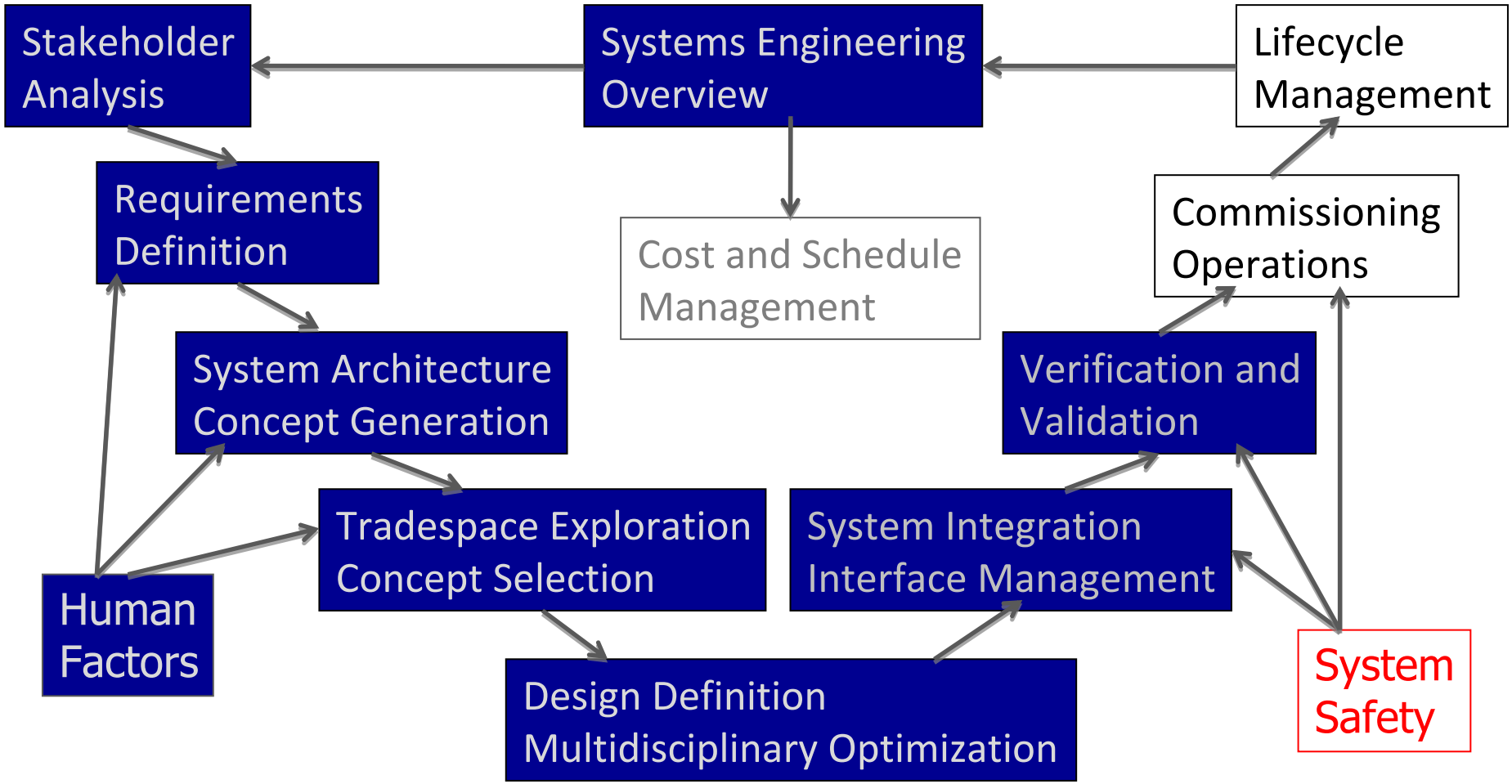
# Typical Fault Trees




| Hazard Cause | Probability | Mitigation |
|---|---|---|
| Software Error | 0 | Test software |

**All software errors do not cause hazards or any particular hazard**

# V-Model – Nov 13, 2009




Wrong!

# Traditional Accident Causation Model

# Chain-of-Events Model

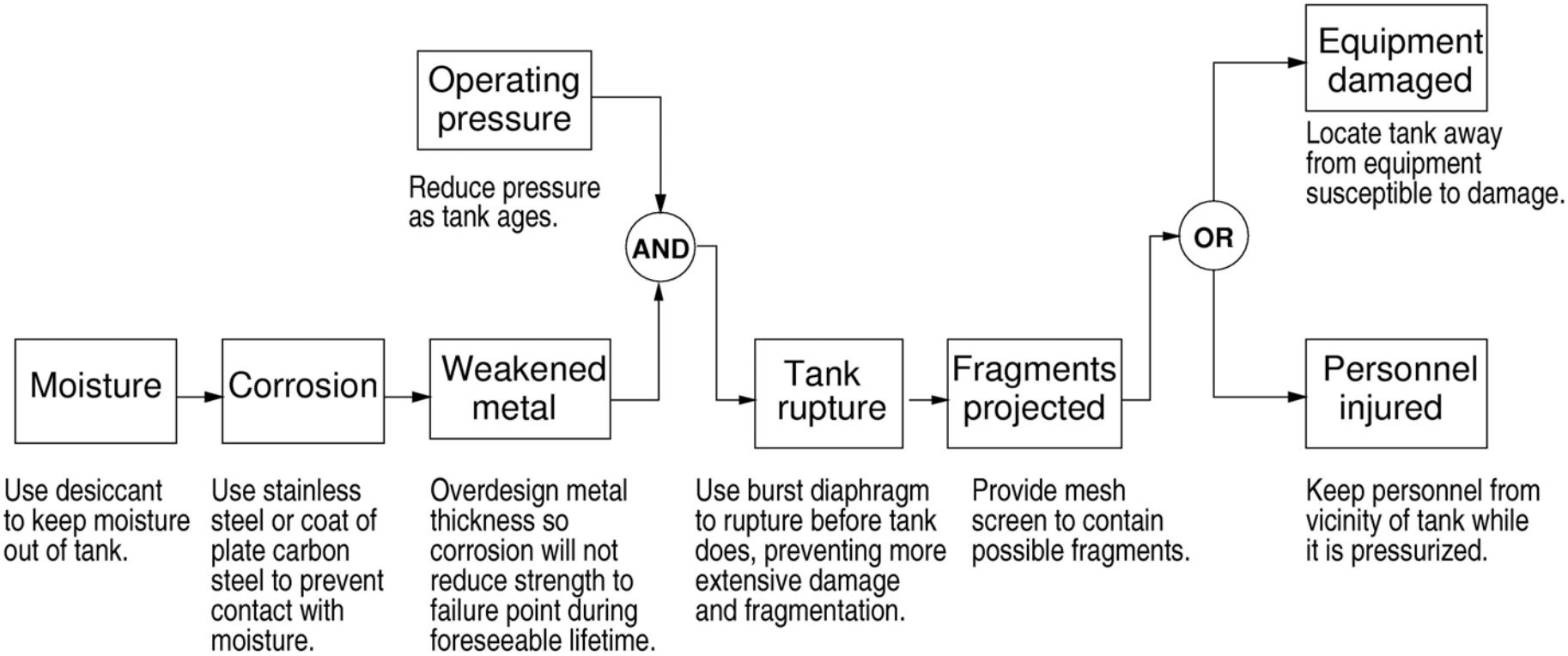
- Explains accidents in terms of multiple events, sequenced as a forward chain over time.
  - Simple, direct relationship between events in chain
  - Ignores non-linear relationships, feedback, etc.
- Events almost always involve component failure, human error, or energy-related event
- Forms the basis for most safety-engineering and reliability engineering analysis:

  e,g, FTA, PRA, FMECA, Event Trees, etc.

  and design:

  e.g., redundancy, overdesign, safety margins, ….

# Chain-of-events example



# Limitations of Chain-of-Events Model

- Social and organizational factors in accidents
- Component interaction accidents
- Software
- Adaptation
  - Systems are continually changing
  - Systems and organizations migrate toward accidents (states of high risk) under cost and productivity pressures in an aggressive, competitive environment
- Human error

# STAMP

## A new accident causation model using Systems Theory (vs. Reliability Theory)

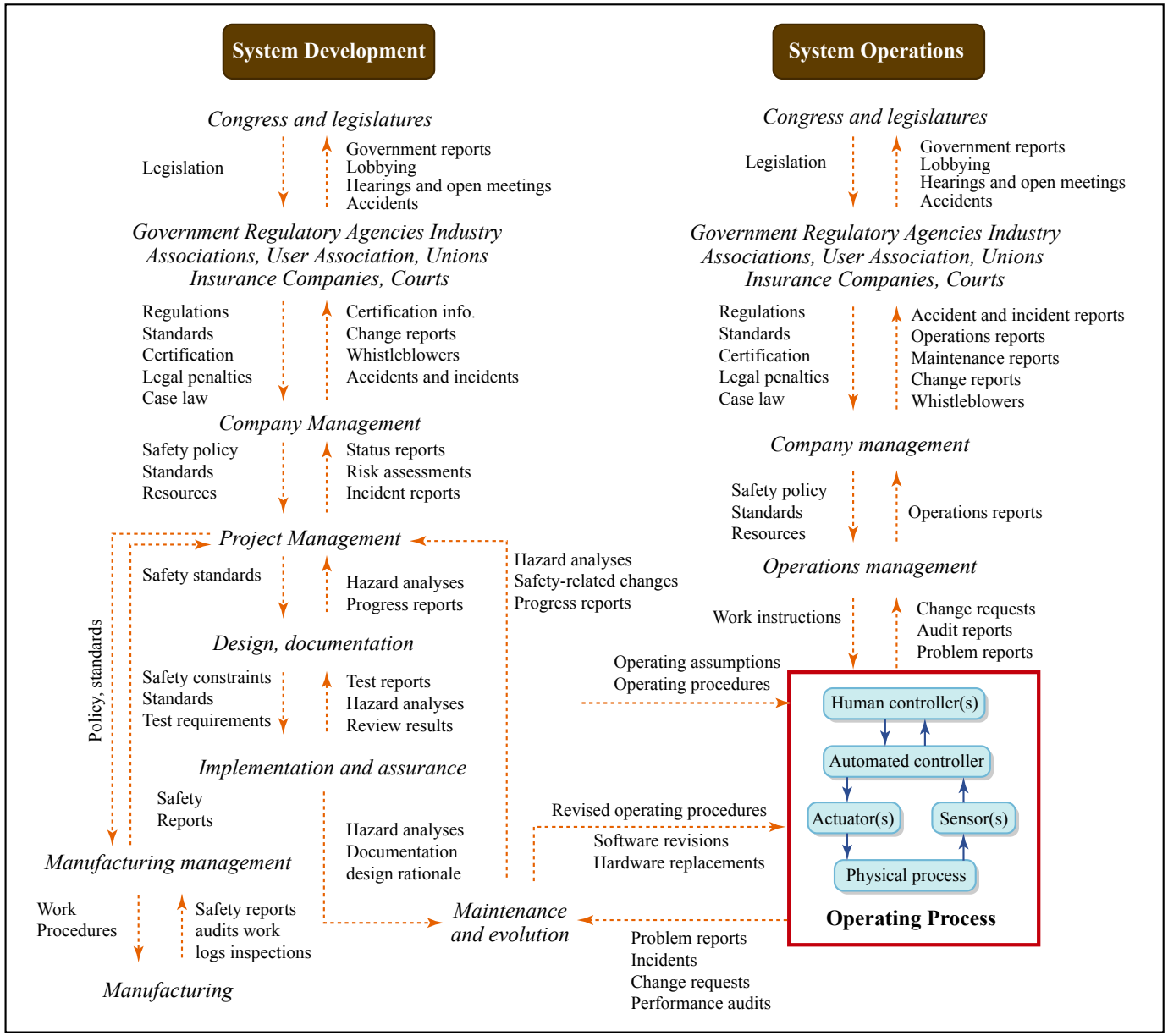
# System's Theoretic View of Safety

- Safety is an emergent system property
  - Accidents arise from interactions among system components (human, physical, social)
  - That violate the constraints on safe component behavior and interactions
- Losses are the result of complex processes, not simply chains of failure events
- Most major accidents arise from a slow migration of the entire system toward a state of high-risk
- Based on systems theory rather than reliability theory

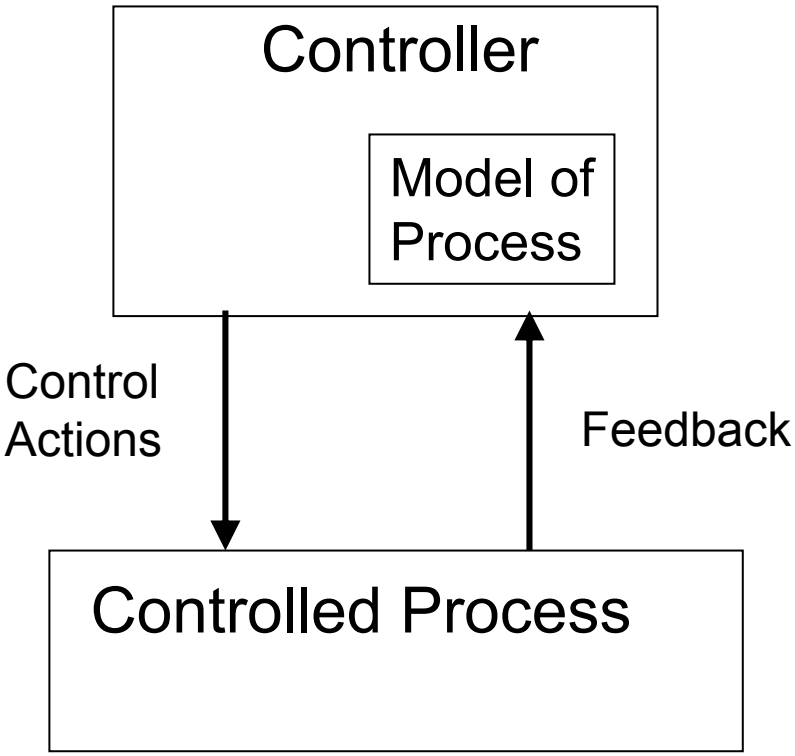
# STAMP: System's Theoretic Accident Model and Processes (1)

- Views safety as a dynamic control problem rather than a component failure problem, e.g.,
  - MPL software did not adequately control descent speed
  - O-ring did not control release of hot gases from Shuttle field joint
  - Public health system did not adequately control contamination of the milk supply with melamine
  - Financial system did not adequately control the use of financial instruments
- Events are the result of the inadequate control
  - Result from lack of enforcement of safety constraints
  - Need to examine larger process and not just event chain

# Example Control Structure



# Controlling and managing dynamic systems requires a model of the controlled process




Process models must contain:
- Required relationship among process variables
- Current state (values of process variables
- The ways the process can change state

# Relationship Between Safety and Process Models

- Accidents occur when models do not match process and
  - Incorrect control commands given
  - Correct ones not given
  - Correct commands given at wrong time (too early, too late)
  - Control stops too soon

# Relationship Between Safety and Process Models (2)

- How do they become inconsistent?
  - Wrong from beginning
  - Missing or incorrect feedback
  - Not updated correctly
  - Time lags not accounted for

  Resulting in

  Uncontrolled disturbances

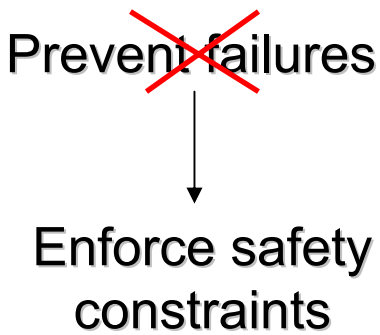
  Unhandled process states

  Inadvertently commanding system into a hazardous state

  Unhandled or incorrectly handled system component failures

# STAMP: System's Theoretic Accident Model and Processes (2)

- To understand accidents, need to examine control structure to determine
  - Why inadequate to enforce safety constraints
  - Why events occurred
- To prevent accidents, need to create a control structure that will enforce the safety constraints
- Critical paradigm change:



# Safety Constraints

Build safety in by enforcing safety constraints on behavior in system design and operations

**System Safety Constraint**:

Water must be flowing into reflux condenser whenever catalyst is added to reactor

**Software Safety Constraint**:

Software must always open water valve before catalyst valve

We have new hazard analysis and safety-driven design techniques to:

- Identify system and component safety constraints
- Perform hazard analysis in parallel with design to guide engineering design process

# The Nature of Controls (1)

Component failures and unsafe interactions may be "controlled" through design

(e.g., redundancy, interlocks, fail-safe design)

or through process

- Manufacturing processes and procedures
- Maintenance processes
- Operations

or through social controls

# The Nature of Controls (2)

- Social controls need not necessarily be governmental or regulatory
  - Controls may also be cultural, policy, individual (self-interest)

    Example: When investment banks went public, individual controls to reduce personal risk and long-term profits were eliminated and risk shifted to shareholders and others who had few and weak controls over those taking the risks.
  - Controls must be designed and implemented throughout the whole system, not just externally

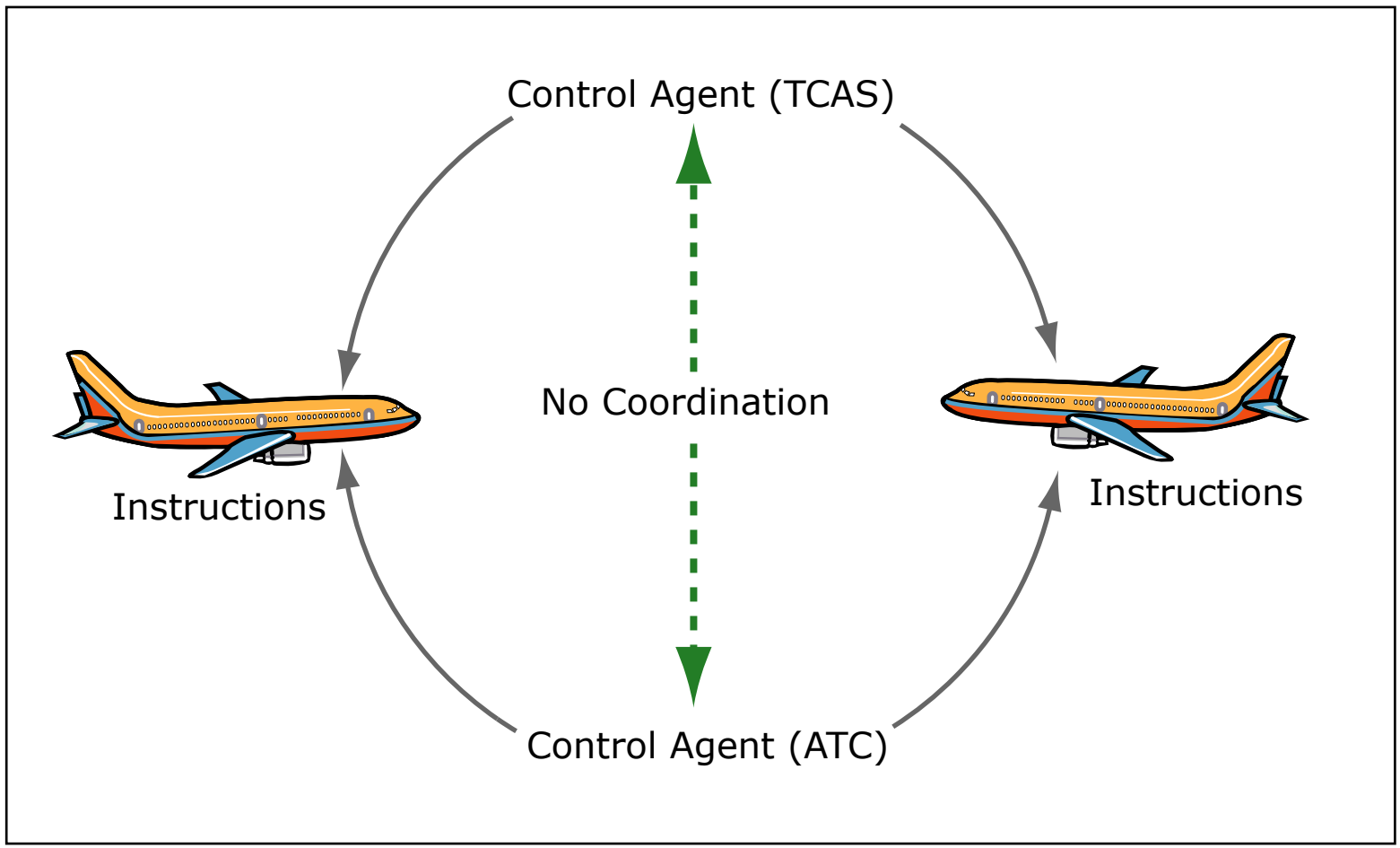
# Summary: Accident Causality

- Accidents occur when
  - Control structure or control actions do not enforce safety constraints
    - Unhandled environmental disturbances or conditions
    - Unhandled or uncontrolled component failures
    - Dysfunctional (unsafe) interactions among components
  - Control structure degrades over time (asynchronous evolution)
  - Control actions inadequately coordinated among multiple controllers

# Uncoordinated “Control Agents”

**“UNSAFE STATE”**

**BOTH TCAS and ATC provide uncoordinated & independent instructions**



# Uses for STAMP

- Basis for new, more powerful hazard analysis techniques (STPA)
- Safety-driven design (physical, operational, organizational)
- More comprehensive accident/incident investigation and root cause analysis
- Organizational and cultural risk analysis
  - Identifying physical and project risks
  - Defining safety metrics and performance audits
  - Designing and evaluating potential policy and structural improvements
  - Identifying leading indicators of increasing risk (“canary in the coal mine”)
- New holistic approaches to security

# System-Theoretic Process Analysis (STPA)

- Supports a safety-driven design process where
  - Hazard analysis influences and shapes early design decisions
  - Hazard analysis iterated and refined as design evolves
- Also supports accident analysis
- Goals (same as any hazard analysis)
  - Identification of system hazards and related safety constraints necessary to ensure acceptable risk
  - Accumulation of information about how hazards can be violated, which is used to eliminate, reduce and control hazards in system design, development, manufacturing, and operations

# Step 1: Identify hazards and translate into high-level requirements and constraints on behavior

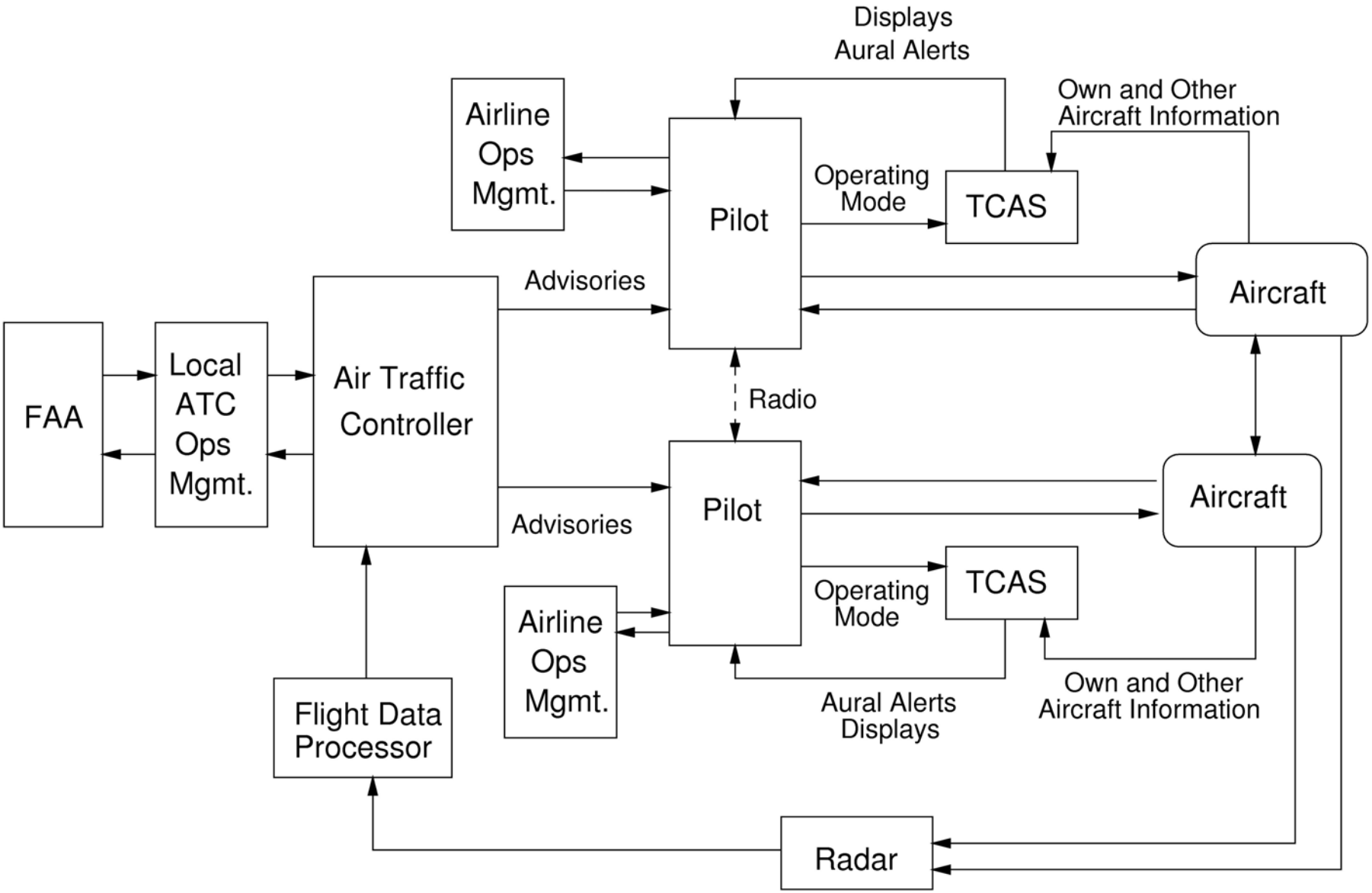
TCAS Hazards:

1. A near mid-air collision (NMAC): Two controlled aircraft violate minimum separation standards)
2. A controlled maneuver into ground
3. Loss of control of aircraft
4. Interference with other safety-related aircraft systems
5. Interference with the ground-based ATC system
6. Interference with ATC safety-related advisory

System Safety Design Constraints:

- TCAS must not cause or contribute to an NMAC
- TCAS must not cause or contribute to a controlled maneuver into the ground
- …

# Step 2: Define basic control structure



# Component Responsibilities

TCAS:

- Receive and update information about its own and other aircraft
- Analyze information received and provide pilot with
  - Information about where other aircraft in the vicinity are located
  - An escape maneuver to avoid potential NMAC threats

Pilot

- Maintain separation between own and other aircraft using visual scanning
- Monitor TCAS displays and implement TCAS escape maneuvers
- Follow ATC advisories

Air Traffic Controller

- Maintain separation between aircraft in controlled airspace by providing advisories (control action) for pilot to follow

Aircraft components (e.g., transponders, antennas)

- Execute control maneuvers
- Receive and send messages to/from aircraft
- Etc.

Airline Operations Management

- Provide procedures for using TCAS and following TCAS advisories
- Train pilots
- Audit pilot performance

Air Traffic Control Operations Management

- Provide procedures
- Train controllers,
- Audit performance of controllers
- Audit performance of overall collision avoidance system

# Step 3a: Identify potential inadequate control actions that could lead to a hazardous state.

In general:

1. A required control action is not provided or not followed
2. An incorrect or unsafe control action is provided
3. A potentially correct or inadequate control action is provided too late or too early (at the wrong time)
4. A correct control action is stopped too soon.

For the NMAC hazard:

TCAS:

1. The aircraft are on a near collision course and TCAS does not provide an RA
2. The aircraft are in close proximity and TCAS provides an RA that degrades vertical separation.
3. The aircraft are on a near collision course and TCAS provides an RA too late to avoid an NMAC
4. TCAS removes an RA too soon

Pilot:

1. The pilot does not follow the resolution advisory provided by TCAS (does not respond to the RA)
2. The pilot incorrectly executes the TCAS resolution advisory.
3. The pilot applies the RA but too late to avoid the NMAC
4. The pilot stops the RA maneuver too soon.

# Step 3b: Use identified inadequate control actions to refine system safety design constraints

- When two aircraft are on a collision course, TCAS must always provide an RA to avoid the collision
- TCAS must not provide RAs that degrades vertical separation
- …
- The pilot must always follow the RA provided by TCAS
- …

## Step 4: Determine how potentially hazardous control actions could occur (scenarios of how constraints can be violated). Eliminate from design or control in design or operations.

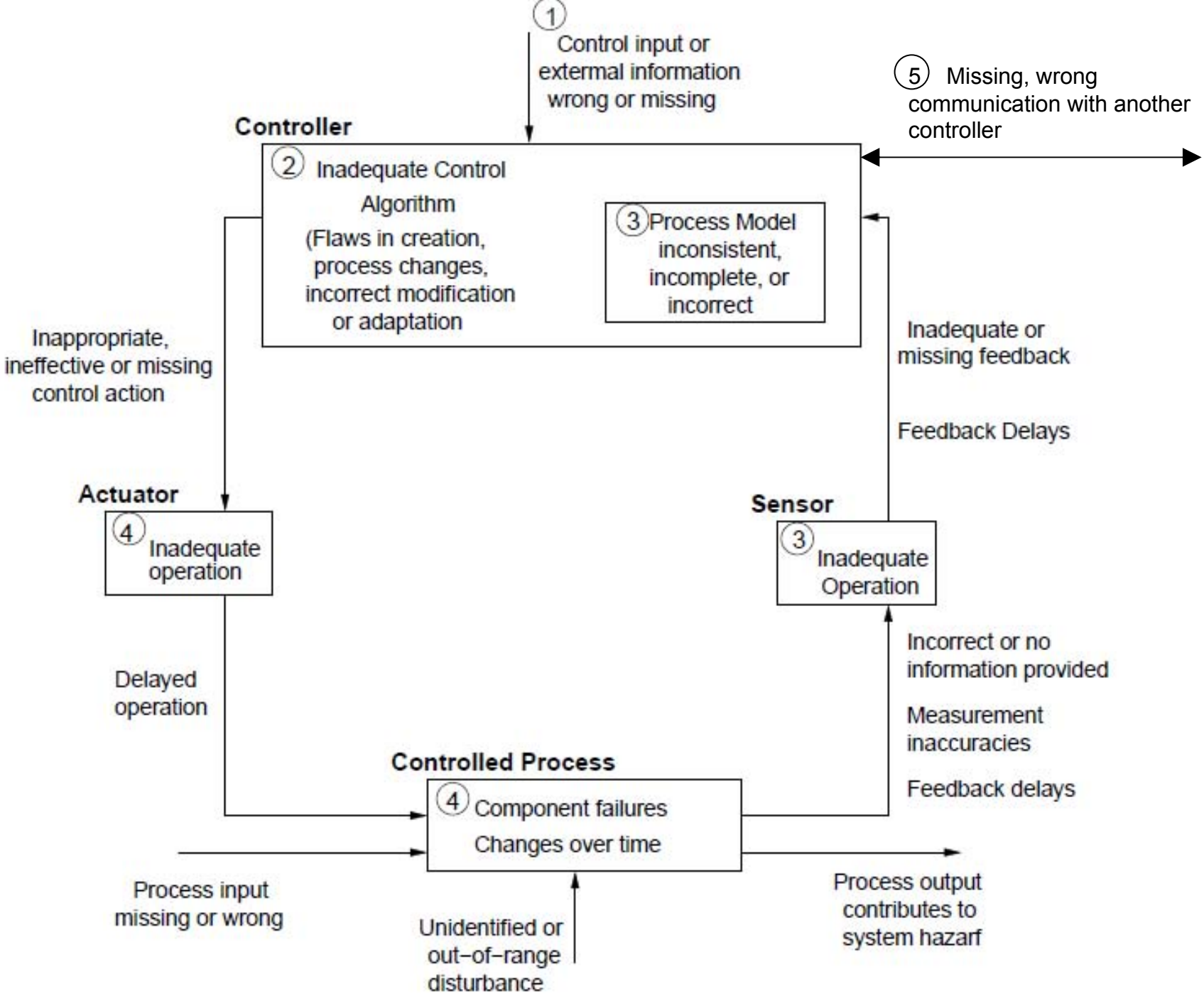
Step4a: Augment control structure with process models for each control component.

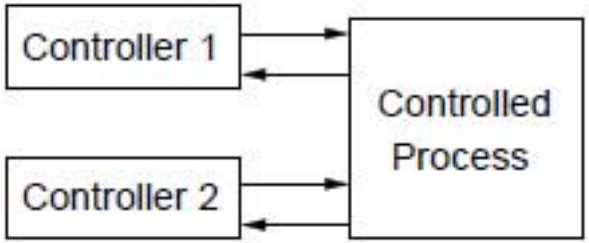
Step4b: For each of inadequate control actions, examine parts of control loop to see if could cause it.

- Guided by a set of generic control flaws

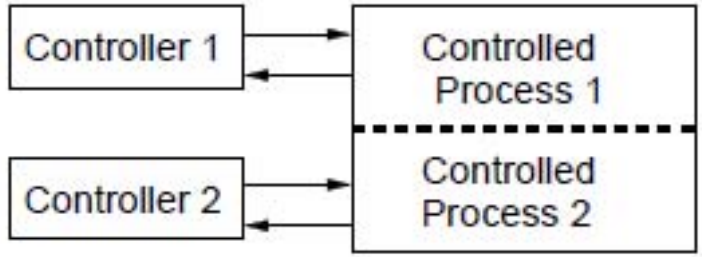
Step 4c: Design controls and mitigation measures

Step4d: Consider how designed controls could degrade over time.


1
Control input or external information wrong or missing
5
Missing, wrong communication with another controller
Controller
2
Inadequate Control Algorithm
(Flaws in creation, process changes, incorrect modification or adaptation
3
Process Model inconsistent, incomplete, or incorrect
Inappropriate, ineffective or missing control action
Inadequate or missing feedback
Feedback Delays
Actuator
4
Inadequate operation
Sensor
3
Inadequate Operation
Delayed operation
Incorrect or no information provided
Measurement inaccuracies
Feedback delays
Controlled Process
4
Component failures
Changes over time
Process input missing or wrong
Unidentified or out-of-range disturbance
Process output contributes to system hazarf




(a) Example of an overlap




(b) Example of a boundary area

Problems can occur when there is shared control or at the boundary areas of separately controlled processes.

# Ballistic Missile Defense System (BMDS) Non-Advocate Safety Assessment using STPA

- A layered defense to defeat all ranges of threats in all phases of flight (boost, mid-course, and terminal)
- Made up of many existing systems (BMDS Element)
  - Early warning radars
  - Aegis
  - Ground-Based Midcourse Defense (GMD)
  - Command and Control Battle Management and Communications (C2BMC)
  - Others
- MDA used STPA to evaluate the residual safety risk of inadvertent launch prior to deployment and test

# Safety Control Structure Diagram for FMIS

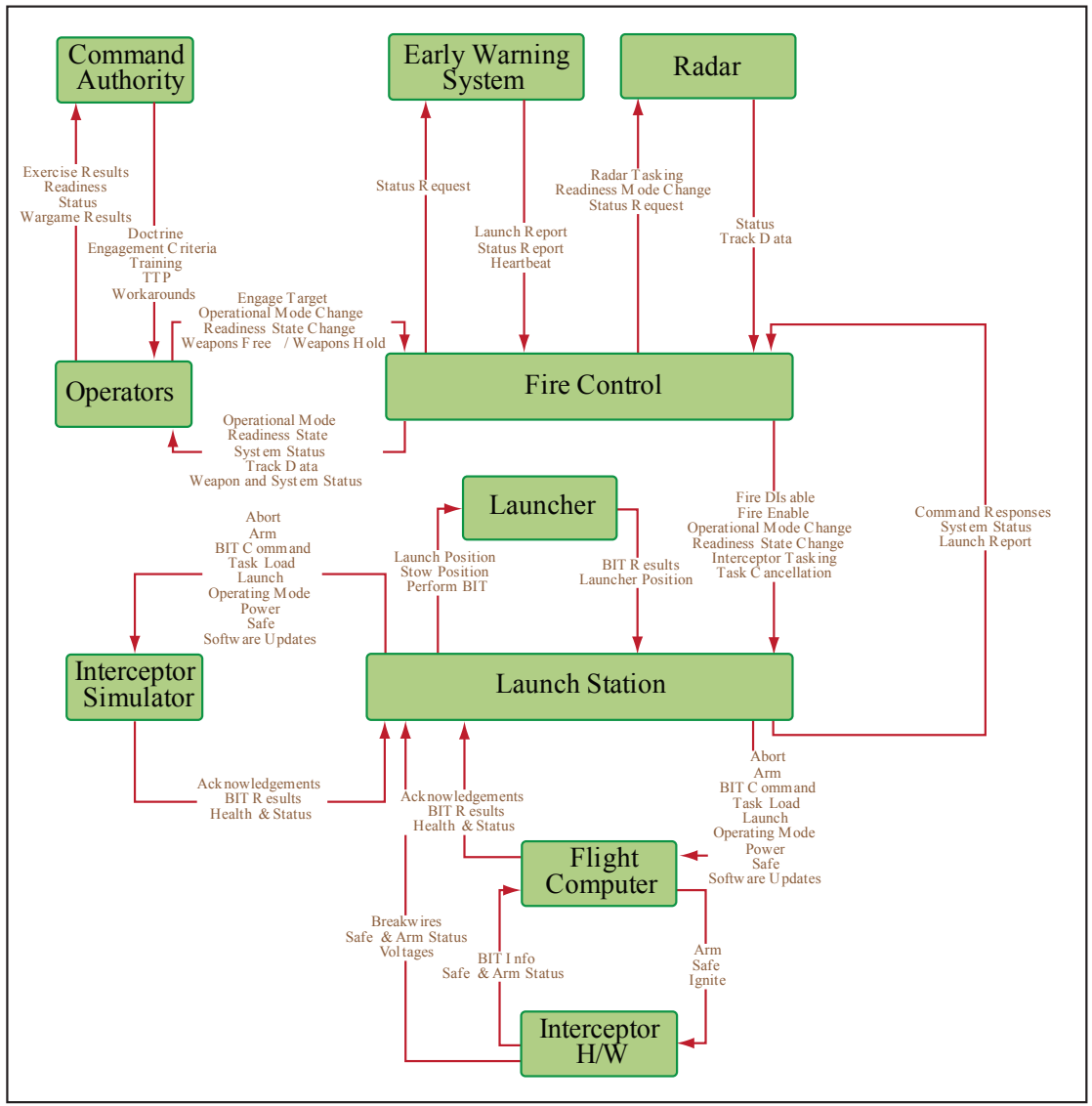



Image by MIT OpenCourseWare.

# Results

- Deployment and testing held up for 6 months because so many scenarios identified for inadvertent launch. In many of these scenarios:
  - All components were operating exactly as intended
  - Complexity of component interactions led to unanticipated system behavior
- STPA also identified component failures that could cause inadequate control (most analysis techniques consider only these failure events)
- As changes are made to the system, the differences are assessed by updating the control structure diagrams and assessment analysis templates.
- Adopted as primary safety approach for BMDS

# Current Uses (1)

- Safety analysis of new missile defense system (MDA)
- Safety-driven new JPL outer planets explorer
- Incorporating risk into early trade studies
- Analysis of the management structure of the space shuttle program (post-Columbia)
- Risk management in the development of NASA's new manned space program (Constellation)
- NASA Mission control – Re-planning and changing mission control procedures safely
- Orion (Space Shuttle replacement)

# Current Uses (2)

- Accident/incident analysis (aircraft, petrochemical plants, …)
- Analysis and prevention of corporate fraud
- Safety of maglev trains (Japan Central Railway)
- Food safety
- Safety in pharmaceutical drug development
- Risk analysis of outpatient (ambulatory) GI surgery at Beth Israel Deaconess Hospital (NIH/AHRQ)
- Children's Hospital (Boston)?
- Demonstration project for JAXA

16.842 Fundamentals of Systems Engineering

Fall 2009