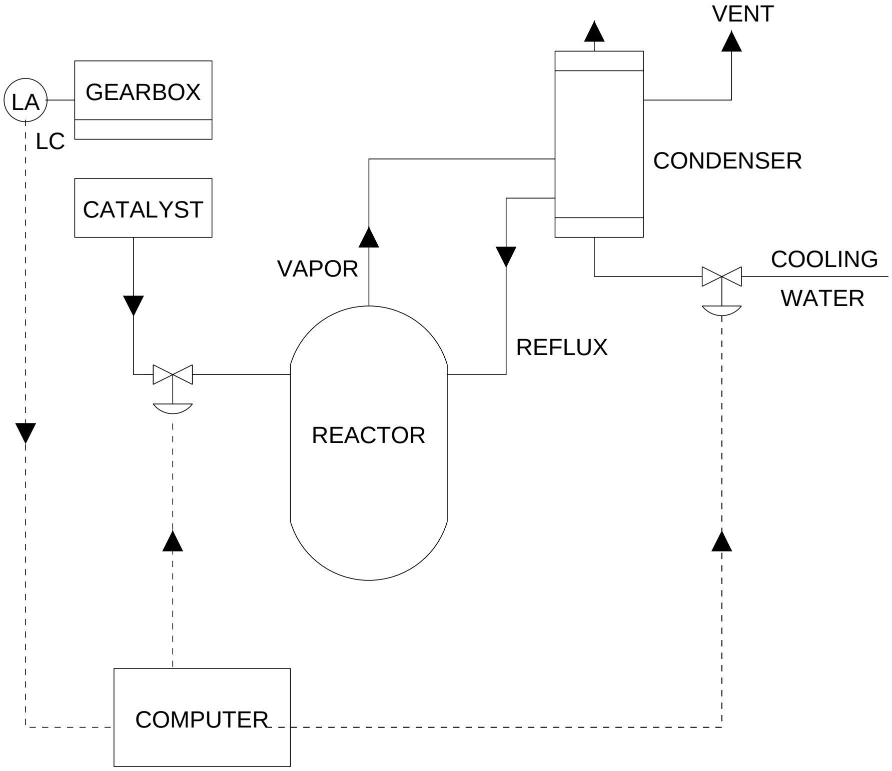
Software System Safety

Nancy G. Leveson

MIT Aero/Astro Dept.

Accident with No Component Failures



Types of Accidents

- Component Failure Accidents

  Single or multiple component failures

  Usually assume random failure

- System Accidents

  Arise in interactions among components

    No components may have "failed"

  Caused by interactive complexity and tight coupling

  Exacerbated by the introduction of computers.

Safety $\neq$ Reliability

Accidents in high−tech systems are changing their nature, and we must change our approaches to safety accordingly.

Confusing Safety and Reliability

From an FAA report on ATC software architectures:

> "The FAA's en route automation meets the criteria for consideration as a safety−critical system. Therefore, en route automation systems must posses ultra−high reliability."

From a blue ribbon panel report on the V−22 Osprey problems:

> "Safety [software]: ...
>
> Recommendation: Improve reliability, then verify by extensive test/fix/test in challenging environments."

Does Software Fail?

Failure: Nonperformance or inability of system or component to perform its intended function for a specified time under specified environmental conditions.

A basic abnormal occurrence, e.g.,

- burned out bearing in a pump
- relay not closing properly when voltage applied

Fault: Higher−order events, e.g.,

- relay closes at wrong time due to improper functioning of an upstream component.

All failures are faults but not all faults are failures.

Reliability Engineering Approach to Safety

Reliability: The probability an item will perform its required function in the specified manner over a given time period and under specified or assumed conditions.

(Note: Most software−related accidents result from errors in specified requirements or function and deviations from assumed conditions.)

- Concerned primarily with failures and failure rate reduction
  - Parallel redundancy
  - Standby sparing
  - Safety factors and margins
  - Derating
  - Screening
  - Timed replacements

Reliability Engineering Approach to Safety (2)

- Assumes accidents are the result of component failure.

  + Techniques exist to increase component reliability
  Failure rates in hardware are quantifiable.

  — Omits important factors in accidents.
  May even decrease safety.

- Many accidents occur without any component ‘‘failure’’

  e.g. Accidents may be caused by equipment operation
  outside parameters and time limits upon which
  reliability analyses are based.

  Or may be caused by interactions of components
  all operating according to specification

  Highly reliable components are not necessarily safe.

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 (con’t.)

- 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.

A Possible Solution

- Enforce discipline and control complexity
  - Limits have changed from structural integrity and physical constraints of materials to intellectual limits
- Improve communication among engineers
- Build safety in by enforcing constraints on behavior

Example (batch reactor)

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

The Problem to be Solved

- The primary safety problem in computer−based systems is the lack of appropriate constraints on design.

- The job of the system safety engineer is to identify the design constraints necessary to maintain safety and to ensure the system and software design enforces them.

An Overview of The Approach

> *Engineers should recognize that reducing risk is not an impossible task, even under financial and time constraints. All it takes in many cases is a different perspective on the design problem.*
>
> Mike Martin and Roland Schinzinger
> *Ethics in Engineering*

System Safety

- A planned, disciplined, and systematic approach to preventing or reducing accidents throughout the life cycle of a system.

- ‘‘Organized common sense ’’ (Mueller, 1968)

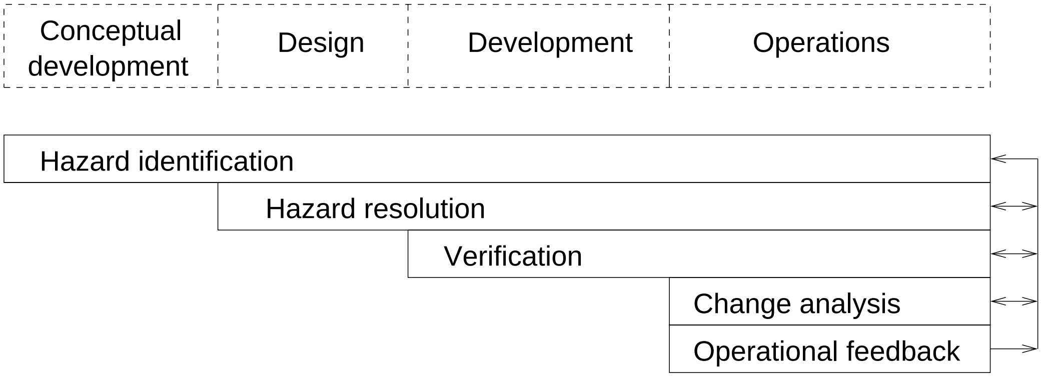
- Primary concern is the management of hazards:

    Hazard
        identification
        evaluation
        elimination
        control
    through
        analysis
        design
        management

- MIL−STD−882

System Safety (2)

- Hazard analysis and control is a continuous, iterative process throughout system development and use.



Conceptual development
Design
Development
Operations

Hazard identification
Hazard resolution
Verification
Change analysis
Operational feedback

- Hazard resolution precedence:
  1. Eliminate the hazard
  2. Prevent or minimize the occurrence of the hazard
  3. Control the hazard if it occurs.
  4. Minimize damage.
- Management

Process Steps

1. Perform a Preliminary Hazard Analysis

   Produces hazard list

2. Perform a System Hazard Analysis (not just Failure Analysis)

   Identifies potential causes of hazards

3. Identify appropriate design constraints on system, software, and humans.

4. Design at system level to eliminate or control hazards.

5. Trace unresolved hazards and system hazard controls to software requirements.

Specifying Safety Constraints

- Most software requirements only specify nominal behavior
    - Need to specify off−nominal behavior
    - Need to specify what software must NOT do
- What must not do is not inverse of what must do
- Derive from system hazard analysis

Process Steps (2)

6. Software requirements review and analysis
    - Completeness
    - Simulation and animation
    - Software hazard analysis
    - Robustness (environment) analysis
    - Mode confusion and other human error analyses
    - Human factors analyses (usability, workload, etc.)

Process Steps (3)

7. Implementation with safety in mind

   Defensive programming

   Assertions and run−time checking

   Separation of critical functions

   Elimination of unnecessary functions

   Exception−handling etc.

8. Off−nominal and safety testing

Process Steps (4)

9. Operational Analysis and Auditing

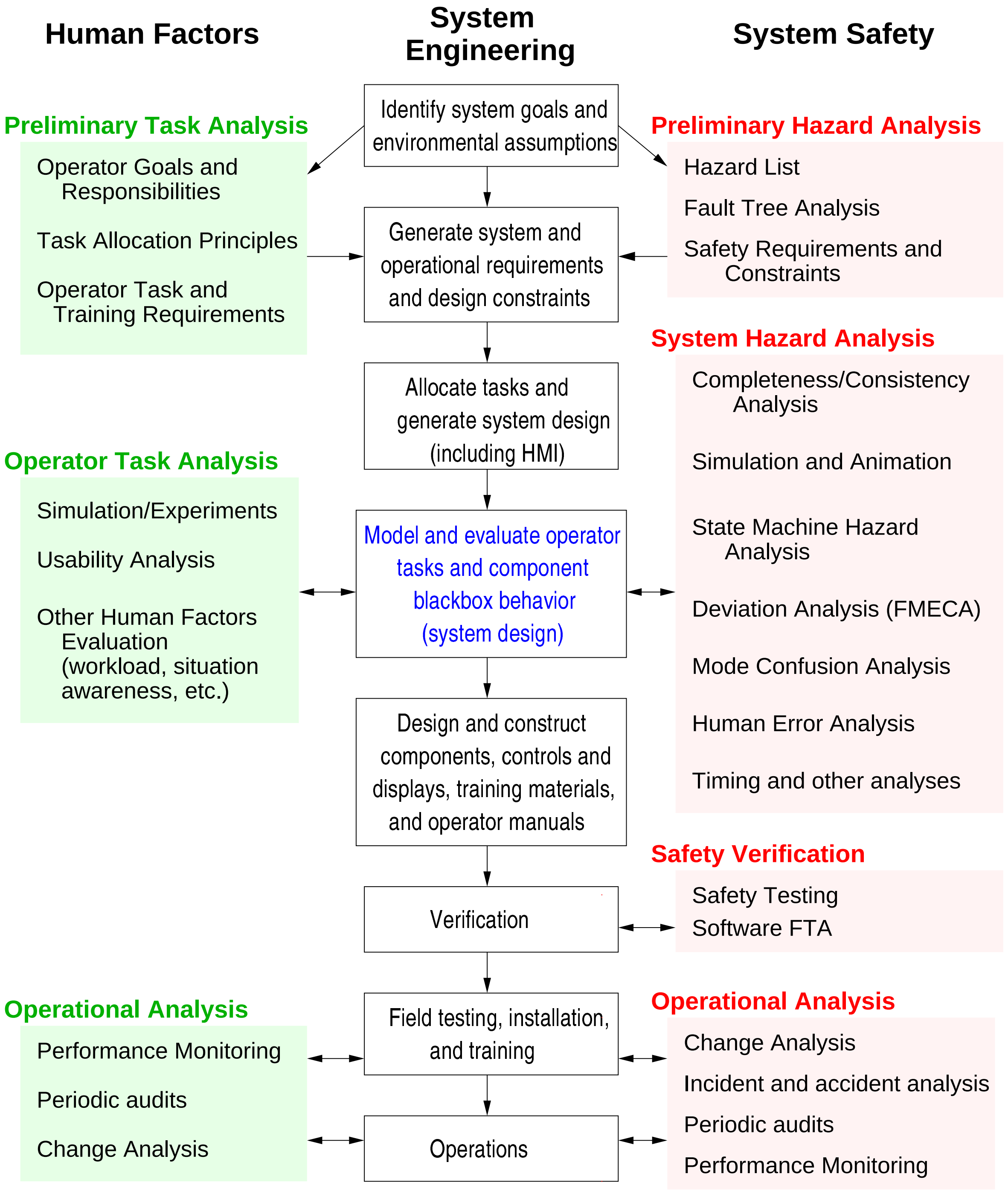
   Change analysis

   Incident and accident analysis

   Performance monitoring

   Periodic audits

A Human−Centered, Safety−Driven Design Process



Human Factors

Preliminary Task Analysis

- Operator Goals and Responsibilities
- Task Allocation Principles
- Operator Task and Training Requirements

Operator Task Analysis

- Simulation/Experiments
- Usability Analysis
- Other Human Factors Evaluation (workload, situation awareness, etc.)

Operational Analysis

- Performance Monitoring
- Periodic audits
- Change Analysis

System Engineering

- Identify system goals and environmental assumptions
- Generate system and operational requirements and design constraints
- Allocate tasks and generate system design (including HMI)
- Model and evaluate operator tasks and component blackbox behavior (system design)
- Design and construct components, controls and displays, training materials, and operator manuals
- Verification
- Field testing, installation, and training
- Operations

System Safety

Preliminary Hazard Analysis

- Hazard List
- Fault Tree Analysis
- Safety Requirements and Constraints

System Hazard Analysis

- Completeness/Consistency Analysis
- Simulation and Animation
- State Machine Hazard Analysis
- Deviation Analysis (FMECA)
- Mode Confusion Analysis
- Human Error Analysis
- Timing and other analyses

Safety Verification

- Safety Testing
- Software FTA

Operational Analysis

- Change Analysis
- Incident and accident analysis
- Periodic audits
- Performance Monitoring

Preliminary Hazard Analysis

1. Identify system hazards
2. Translate system hazards into high−level system safety design constraints.
3. Assess hazards if required to do so.
4. Establish the hazard log.

System Hazards for Automated Train Doors

- Train starts with door open.
- Door opens while train is in motion.
- Door opens while improperly aligned with station platform.
- Door closes while someone is in doorway
- Door that closes on an obstruction does not reopen or reopened door does not reclose.
- Doors cannot be opened for emergency evacuation.

System Hazards for Air Traffic Control

- Controlled aircraft violate minimum separation standards (NMAC).
- Airborne controlled aircraft enters an unsafe atmospheric region.
- Controlled airborne aircraft enters restricted airspace without authorization.
- Controlled airborne aircraft gets too close to a fixed obstable other than a safe point of touchdown on assigned runway (CFIT)
- Controlled airborne aircraft and an intruder in controlled airspace violate minimum separation.
- Controlled aircraft operates outside its performance envelope.
- Aircraft on ground comes too close to moving objects or collides with stationary objects or leaves the paved area.
- Aircraft enters a runway for which it does not have clearance.
- Controlled aircraft executes an extreme maneuver within its performance envelope.
- Loss of aircraft control.

Exercise: Identify the system hazards for this cruise−control system

The cruise control system operates only when the engine is running. When the driver turns the system on, the speed at which the car is traveling at that instant is maintained. The system monitors the car’s speed by sensing the rate at which the wheels are turning, and it maintains desired speed by controlling the throttle position. After the system has been turned on, the driver may tell it to start increasing speed, wait a period of time, and then tell it to stop increasing speed. Throughout the time period, the system will increase the speed at a fixed rate, and then will maintain the final speed reached.

The driver may turn off the system at any time. The system will turn off if it senses that the accelerator has been depressed far enough to override the throttle control. If the system is on and senses that the brake has been depressed, it will cease maintaining speed but will not turn off. The driver may tell the system to resume speed, whereupon it will return to the speed it was maintaining before braking and resume maintenance of that speed.

Hazards must be translated into design constraints.

HAZARD	DESIGN CRITERION
Train starts with door open.	Train must not be capable of moving with any door open.
Door opens while train is in motion.	Doors must remain closed while train is in motion.
Door opens while improperly aligned with station platform.	Door must be capable of opening only after train is stopped and properly aligned with platform unless emergency exists (see below).
Door closes while someone is in doorway.	Door areas must be clear before door closing begins.
Door that closes on an obstruction does not reopen or reopened door does not reclose.	An obstructed door must reopen to permit removal of obstruction and then automatically reclose.
Doors cannot be opened for emergency evacuation.	Means must be provided to open doors anywhere when the train is stopped for emergency evacuation.

Example PHA for ATC Approach Control

HAZARDS	REQUIREMENTS/CONSTRAINTS
1. A pair of controlled aircraft violate minimum separation standards.	1a. ATC shall provide advisories that maintain safe separation between aircraft. 1b. ATC shall provide conflict alerts.
2. A controlled aircraft enters an unsafe atmospheric region. (icing conditions, windshear areas, thunderstorm cells)	2a. ATC must not issue advisories that direct aircraft into areas with unsafe atmospheric conditions. 2b. ATC shall provide weather advisories and alerts to flight crews. 2c. ATC shall warn aircraft that enter an unsafe atmospheric region.

Example PHA for ATC Approach Control (2)

HAZARDS	REQUIREMENTS/CONSTRAINTS
3. A controlled aircraft enters restricted airspace without authorization.	3a. ATC must not issue advisories that direct an aircraft into restricted airspace unless avoiding a greater hazard. 3b. ATC shall provide timely warnings to aircraft to prevent their incursion into restricted airspace.
4. A controlled aircraft gets too close to a fixed obstacle or terrain other than a safe point of touchdown on assigned runway.	4. ATC shall provide advisories that maintain safe separation between aircraft and terrain or physical obstacles.
5. A controlled aircraft and an intruder in controlled airspace violate minimum separation standards.	5. ATC shall provide alerts and advisories to avoid intruders if at all possible.

HAZARDS	REQUIREMENTS/CONSTRAINTS
6. Loss of controlled flight or loss of airframe integrity.	6a. ATC must not issue advisories outside the safe performance envelope of the aircraft. 6b. ATC advisories must not distract or disrupt the crew from maintaining safety of flight. 6c. ATC must not issue advisories that the pilot or aircraft cannot fly or that degrade the continued safe flight of the aircraft. 6d. ATC must not provide advisories that cause an aircraft to fall below the standard glidepath or intersect it at the wrong place.

Classic Hazard Level Matrix

SEVERITY

LIKELIHOOD

	I Catastrophic	II Critical	III Marginal	IV Negligible
A Frequent	I−A	II−A	III−A	IV−A
B Moderate	I−B	II−B	III−B	IV−B
C Occasional	I−C	II−C	III−C	IV−C
D Remote	I−D	II−D	III−D	IV−D
E Unlikely	I−E	II−E	III−E	IV−E
F Impossible	I−F	II−F	III−F	IV−F

Another Example Hazard Level Matrix

	A Frequent	B Probable	C Occasional	D Remote	E Improbable	F Impossible
Catastrophic I	Design action required to eliminate or control hazard 1	Design action required to eliminate or control hazard 2	Design action required to eliminate or control hazard 3	Hazard must be controlled or hazard probability reduced 4	9	12
Critical II	Design action required to eliminate or control hazard 3	Design action required to eliminate or control hazard 4	Hazard must be controlled or hazard probability reduced 6	Hazard control desirable if cost effective 7	Assume will not occur 12	Impossible occurrence 12
Marginal III	Design action required to eliminate or control hazard 5	Hazard must be controlled or hazard probability reduced 6	Hazard control desirable if cost effective 8	Normally not cost effective 10	12	12
Negligible IV	10	11	Negligible hazard 12	12	12	12

Hazard Causal Analysis

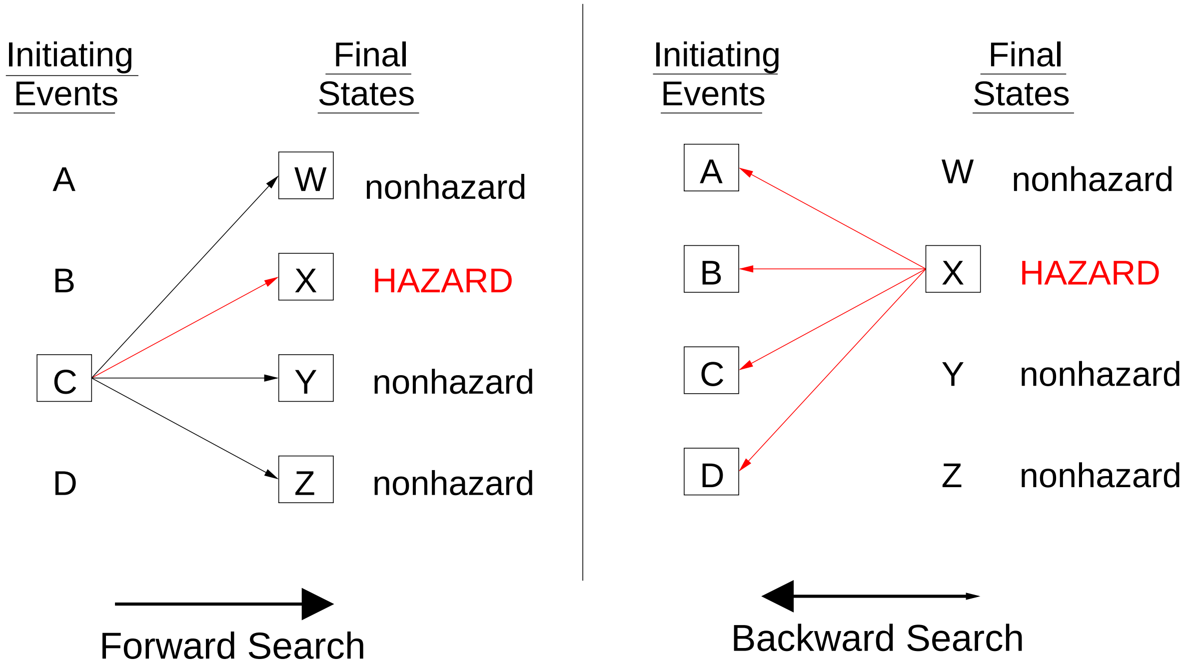
- Used to refine the high−level safety constraints into more detailed constraints.

- Requires some type of model (even if only in head of analyst)

- Almost always involves some type of search through the system design (model) for states or conditions that could lead to system hazards.

  Top−down
  Bottom−up
  Forward
  Backward

Forward vs. Backward Search



Initiating Events

Final States

A
B
C
D

W nonhazard
X HAZARD
Y nonhazard
Z nonhazard

Forward Search

Initiating Events

Final States

A
B
C
D

W nonhazard
X HAZARD
Y nonhazard
Z nonhazard

Backward Search

FTA and Software

- Appropriate for qualitative analyses, not quantitative ones
- System fault trees helpful in identifying potentially hazardous software behavior.

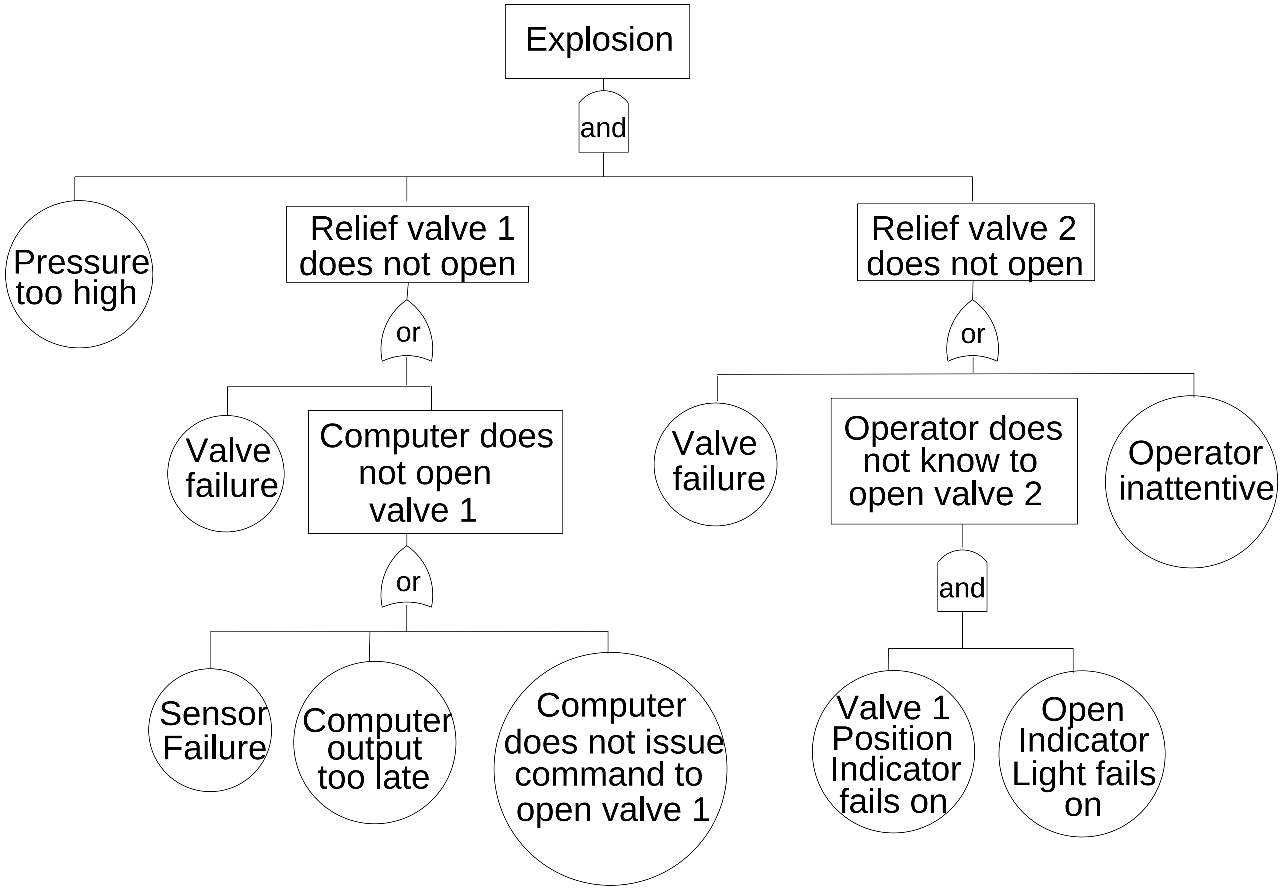
  Can use to refine system design constraints.

- FTA can be used to verify code.

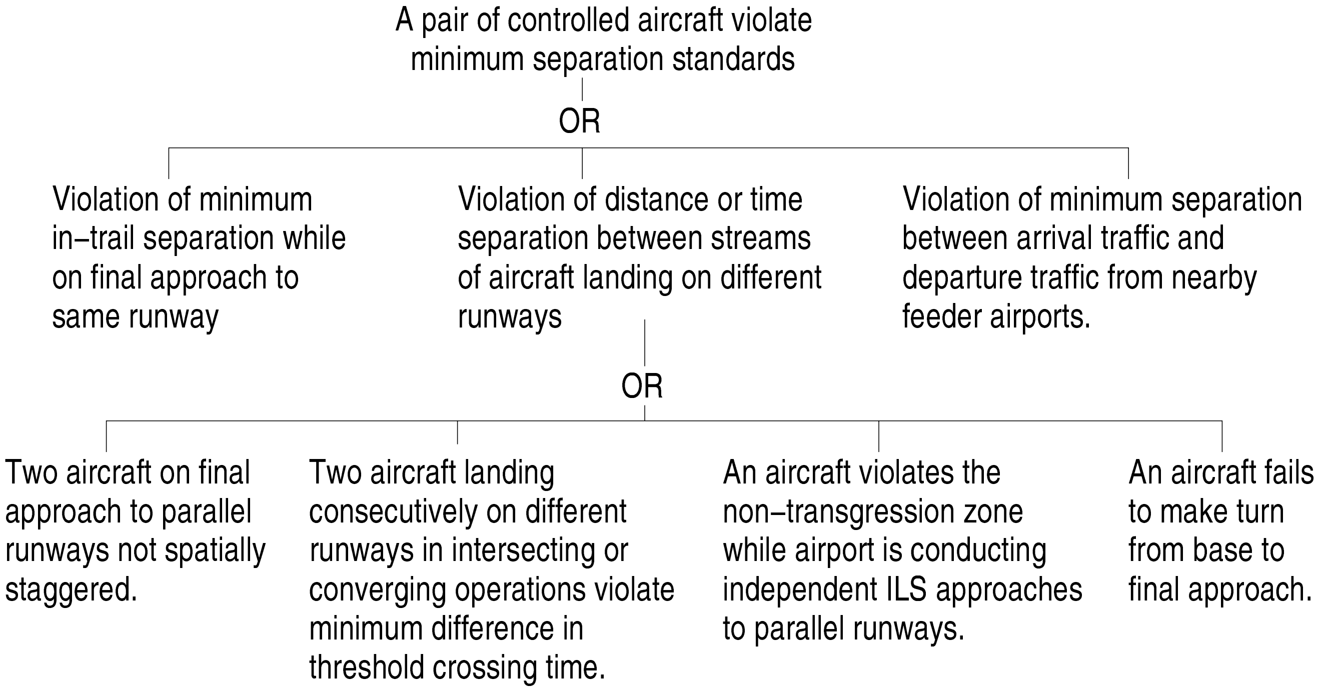
  Identifies any paths from inputs to hazardous outputs or provides some assurance they don't exist.

  Not looking for failures but incorrect paths (functions)

Fault Tree Example

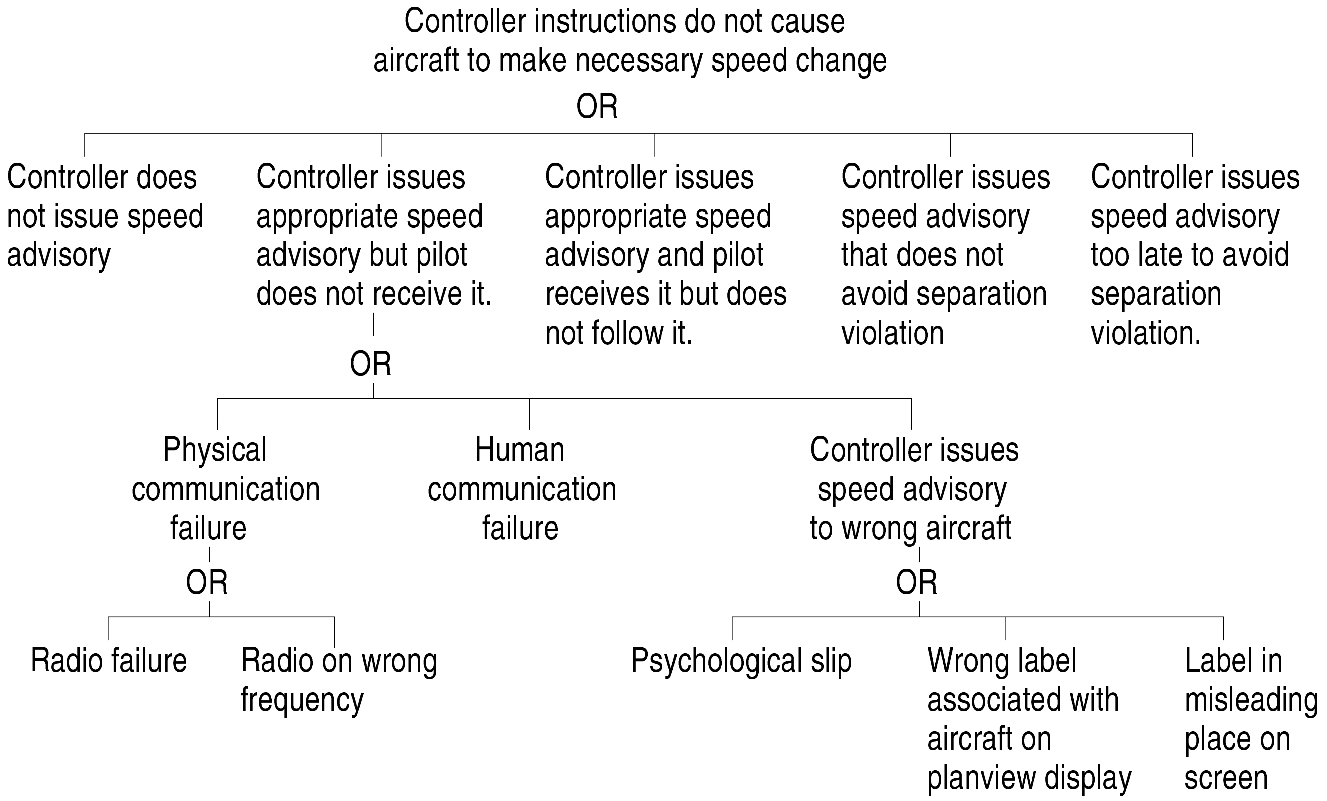


Example Fault Tree for ATC Arrival Traffic



- A pair of controlled aircraft violate minimum separation standards
  - OR
    - Violation of minimum in–trail separation while on final approach to same runway
    - Violation of distance or time separation between streams of aircraft landing on different runways
      - OR
        - Two aircraft on final approach to parallel runways not spatially staggered.
        - Two aircraft landing consecutively on different runways in intersecting or converging operations violate minimum difference in threshold crossing time.
        - An aircraft violates the non–transgression zone while airport is conducting independent ILS approaches to parallel runways.
        - An aircraft fails to make turn from base to final approach.
    - Violation of minimum separation between arrival traffic and departure traffic from nearby feeder airports.

Example Fault Tree for ATC Arrival Traffic (2)



- Controller instructions do not cause aircraft to make necessary speed change
  - OR
    - Controller does not issue speed advisory
    - Controller issues appropriate speed advisory but pilot does not receive it.
      - OR
        - Physical communication failure
          - OR
            - Radio failure
            - Radio on wrong frequency
        - Human communication failure
        - Controller issues speed advisory to wrong aircraft
          - OR
            - Psychological slip
            - Wrong label associated with aircraft on planview display
            - Label in misleading place on screen
    - Controller issues appropriate speed advisory and pilot receives it but does not follow it.
    - Controller issues speed advisory that does not avoid separation violation
    - Controller issues speed advisory too late to avoid separation violation.

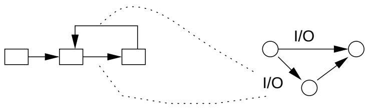
Requirements Completeness

- Most software−related accidents involve software requirements deficiencies.
- Accidents often result from unhandled and unspecified cases.
- We have defined a set of criteria to determine whether a requirements specification is complete.
- Derived from accidents and basic engineering principles.
- Validated (at JPL) and used on industrial projects.

Completeness: Requirements are sufficient to distinguish the desired behavior of the software from that of any other undesired program that might be designed.

Requirements Completeness Criteria (2)

- How were criteria derived?
  - Mapped the parts of a control loop to a state machine



  - Defined completeness for each part of state machine

    States, inputs, outputs, transitions
    Mathematical completeness

  - Added basic engineering principles (e.g., feedback)
  - Added what have learned from accidents

Requirements Completeness Criteria (3)

About 60 criteria in all including human−computer interaction.

(won’t go through them all — they are in the book)

Startup, shutdown
Mode transitions
Inputs and outputs
Value and timing
Load and capacity
Environment capacity
Failure states and transitions
Human−computer interface
Robustness
Data age
Latency
Feedback
Reversibility
Preemption
Path Robustness

Most integrated into SpecTRM−RL language design or simple tools can check them.

Requirements Analysis

- Model Execution, Animation, and Visualization
- Completeness
- State Machine Hazard Analysis (backwards reachability)
- Software Deviation Analysis
- Human Error Analysis
- Test Coverage Analysis and Test Case Generation

Automatic code generation?

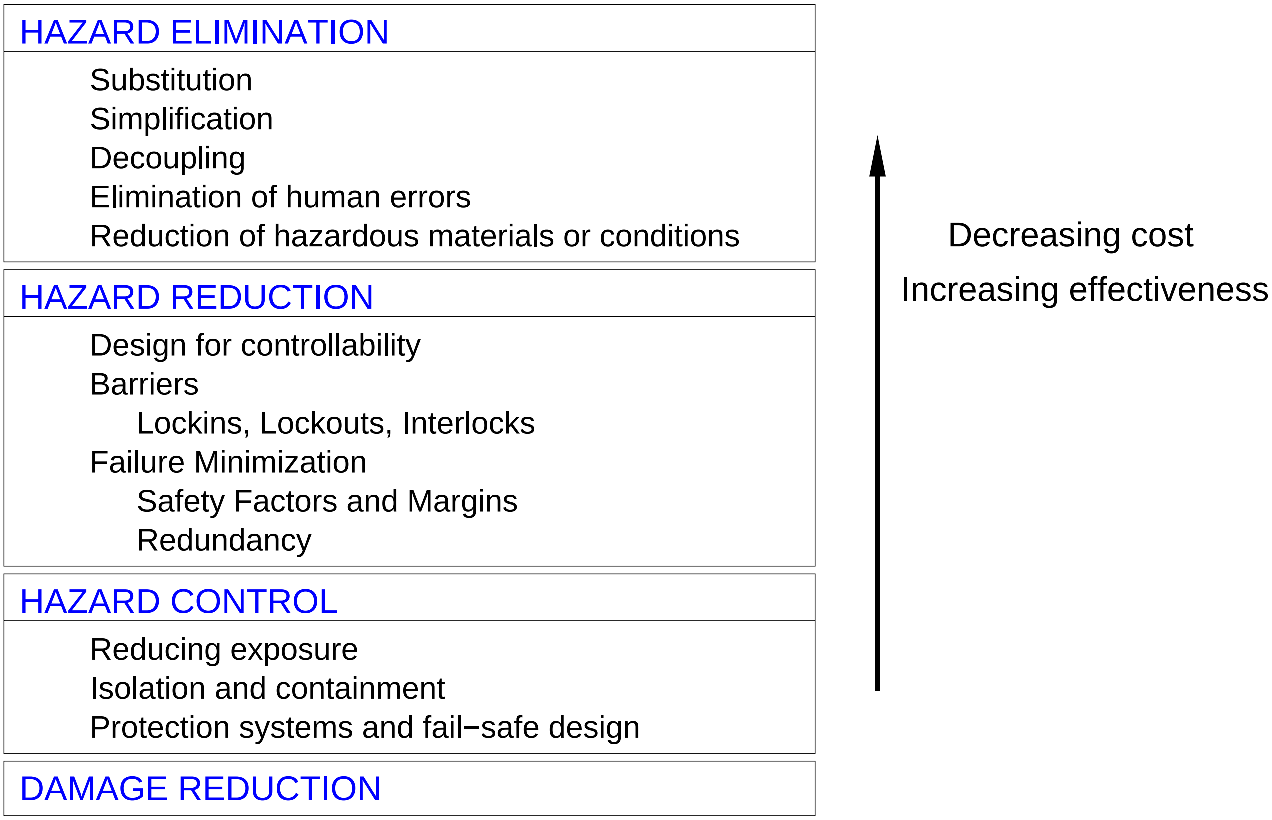
Model Execution and Animation

- SpecTRM−RL models are executable.
- Model execution is animated
- Results of execution could be input into a graphical visualization
- Inputs can come from another model or simulator and output can go into another model or simulator.

Design for Safety

- Software design must enforce safety constraints
- Should be able to trace from requirements to code (vice versa)
- Design should incorporate basic safety design principles

Safe Design Precedence



HAZARD ELIMINATION
- Substitution
- Simplification
- Decoupling
- Elimination of human errors
- Reduction of hazardous materials or conditions

HAZARD REDUCTION
- Design for controllability
- Barriers
  - Lockins, Lockouts, Interlocks
- Failure Minimization
  - Safety Factors and Margins
  - Redundancy

HAZARD CONTROL
- Reducing exposure
- Isolation and containment
- Protection systems and fail−safe design

DAMAGE REDUCTION

Decreasing cost

Increasing effectiveness

STAMP

(Systems Theory Accident Modeling and Processes)

STAMP is a new theoretical underpinning for developing more effective hazard analysis techniques for complex systems.

Accident models provide the basis for

- Investigating and analyzing accidents
- Preventing accidents
  - Hazard analysis
  - Design for safety
- Assessing risk (determining whether systems are suitable for use)
- Performance modeling and defining safety metrics

Chain−of−Events Models

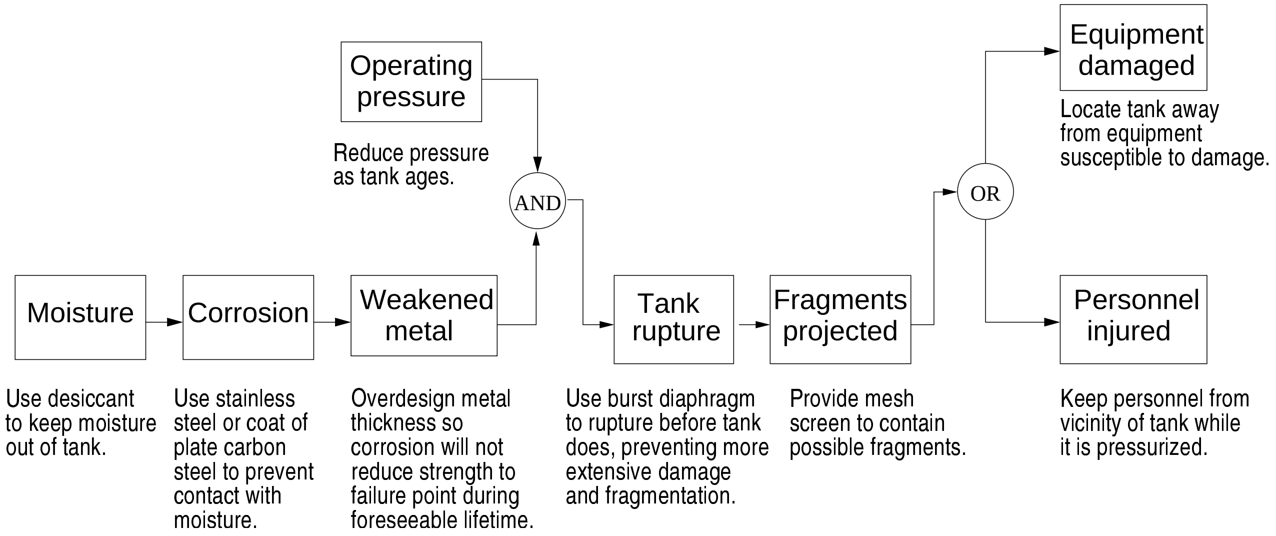
- Explain accidents in terms of multiple events, sequenced as a forward chain over time.
- Events almost always involve component failure, human error, or energy−related event
- Form the basis of most safety−engineering and reliability engineering analysis:

  e.g., Fault Tree Analysis, Probabilistic Risk Assessment, FMEA, Event Trees

  and design:

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

Chain−of−Events Example



Operating pressure
Reduce pressure as tank ages.

AND

Moisture
Use desiccant to keep moisture out of tank.

Corrosion
Use stainless steel or coat of plate carbon steel to prevent contact with moisture.

Weakened metal
Overdesign metal thickness so corrosion will not reduce strength to failure point during foreseeable lifetime.

Tank rupture
Use burst diaphragm to rupture before tank does, preventing more extensive damage and fragmentation.

Fragments projected
Provide mesh screen to contain possible fragments.

OR

Equipment damaged
Locate tank away from equipment susceptible to damage.

Personnel injured
Keep personnel from vicinity of tank while it is pressurized.

Chain−of−Events Example: Bhopal

E1: Worker washes pipes without inserting slip blind

E2: Water leaks into MIT tank

E3: Explosion occurs

E4: Relief valve opens

E5: MIC vented into air

E6: Wind carries MIC into populated area around plant

Limitations of Event Chain Models:

- Social and organizational factors in accidents

> *Underlying every technology is at least one basic science, although the technology may be well developed long before the science emerges. Overlying every technical or civil system is a social system that provides purpose, goals, and decision criteria.*
>
> *Ralph Miles Jr.*

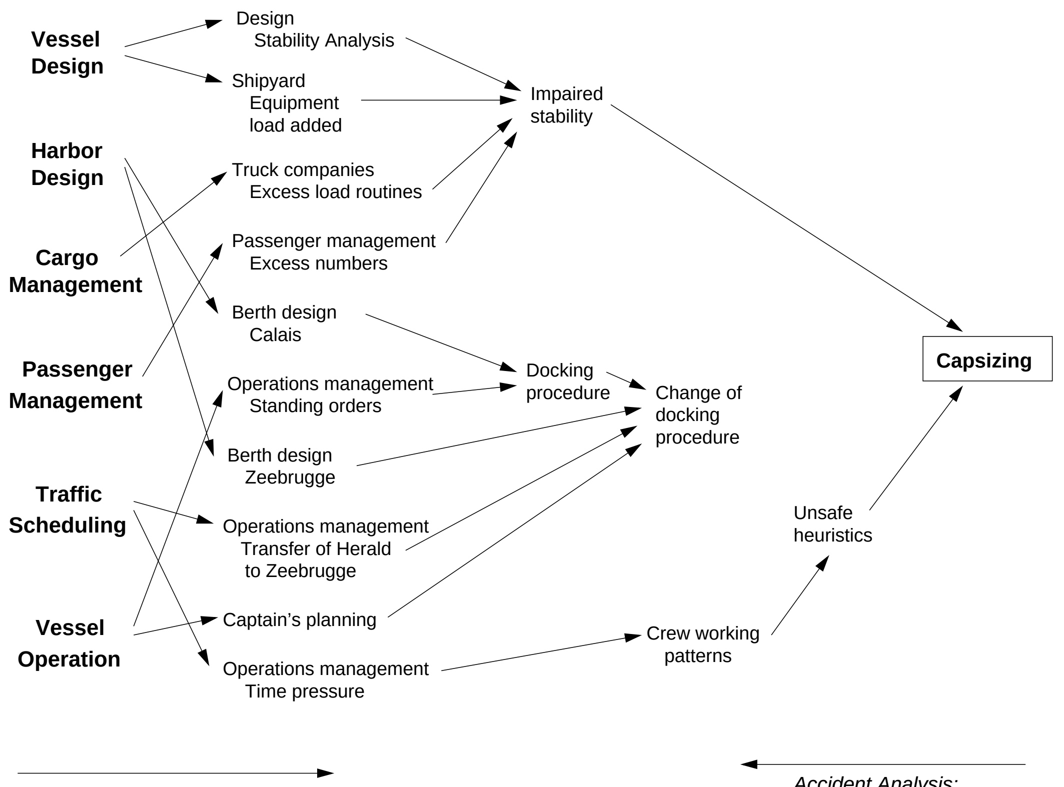
Models need to include the social system as well as the technology and its underlying science.

- System accidents

- Software error

Limitations of Event Chain Models (2)

- Human error
  - Deviation from normative procedure vs. established practice
  - Cannot effectively model human behavior by decomposing it into individual decisions and actions and studying it in isolation from the
    - physical and social context
    - value system in which it takes place
    - dynamic work process
- Adaptation
  - Major accidents involve systematic migration of organizational behavior under pressure toward cost effectiveness in an aggressive, competitive environment.



Vessel Design

Harbor Design

Cargo Management

Passenger Management

Traffic Scheduling

Vessel Operation

Design
Stability Analysis

Shipyard
Equipment
load added

Truck companies
Excess load routines

Passenger management
Excess numbers

Berth design
Calais

Operations management
Standing orders

Berth design
Zeebrugge

Operations management
Transfer of Herald
to Zeebrugge

Captain's planning

Operations management
Time pressure

Impaired stability

Docking procedure

Change of docking procedure

Crew working patterns

Unsafe heuristics

Capsizing

Operational Decision Making:
Decision makers from separate departments in operational context very likely will not see the forest for the trees.

Accident Analysis:
Combinatorial structure of possible accidents can easily be identified.

Ways to Cope with Complexity

- Analytic Reduction (Descartes)
  - Divide system into distinct parts for analysis purposes.
  - Examine the parts separately.

- Three important assumptions:
  1. The division into parts will not distort the phenomenon being studied.
  2. Components are the same when examined singly as when playing their part in the whole.
  3. Principles governing the assembling of the components into the whole are themselves straightforward.

Ways to Cope with Complexity (con’t.)

- Statistics
  - Treat as a structureless mass with interchangeable parts.
  - Use Law of Large Numbers to describe behavior in terms of averages.

- Assumes components sufficiently regular and random in their behavior that they can be studied statistically.

What about software?

- Too complex for complete analysis:
  - Separation into non−interacting subsystems distorts the results.
  - The most important properties are emergent.
- Too organized for statistics
  - Too much underlying structure that distorts the statistics.

Systems Theory

- Developed for biology (Bertalanffly) and cybernetics (Norbert Weiner)
  - For systems too complex for complete analysis
    - Separation into non−interacting subsystems distorts results
    - Most important properties are emergent.

  and too organized for statistical analysis

- Concentrates on analysis and design of whole as distinct from parts (basis of system engineering)
  - Some properties can only be treated adequately in their entirety, taking into account all social and technical aspects.
  - These properties derive from relationships between the parts of systems −− how they interact and fit together.

Systems Theory (2)

- Two pairs of ideas:
  1. Emergence and hierarchy
     - Levels of organization, each more complex than one below.
     - Levels characterized by emergent properties
       - Irreducible
       - Represent constraints upon the degree of freedom of components a lower level.
     - Safety is an emergent system property
       - It is NOT a component property.
       - It can only be analyzed in the context of the whole.

Systems Theory (3)

2. Communication and control
   - Hierarchies characterized by control processes working at the interfaces between levels.
   - A control action imposes constraints upon the activity at one level of a hierarchy.
   - Open systems are viewed as interrelated components kept in a state of dynamic equilibrium by feedback loops of information and control.
   - Control in open systems implies need for communication

A Systems Theory Model of Accidents

- Safety is an emergent system property.
  - Accidents arise from interactions among
    People
    Societal and organizational structures
    Engineering activities
    Physical system components

    that violate the constraints on safe component behavior and interactions.
  - Not simply chains of events or linear causality, but more complex types of causal connections.
- Need to include the entire socio−technical system

STAMP (Systems−Theoretic Accident Model and Processes)

- Based on systems and control theory
- Systems not treated as a static design
  - A socio−technical system is a dynamic process continually adapting to achieve its ends and to react to changes in itself and its environment
  - Preventing accidents requires designing a control structure to enforce constraints on system behavior and adaptation.

STAMP (2)

- Views accidents as a control problem

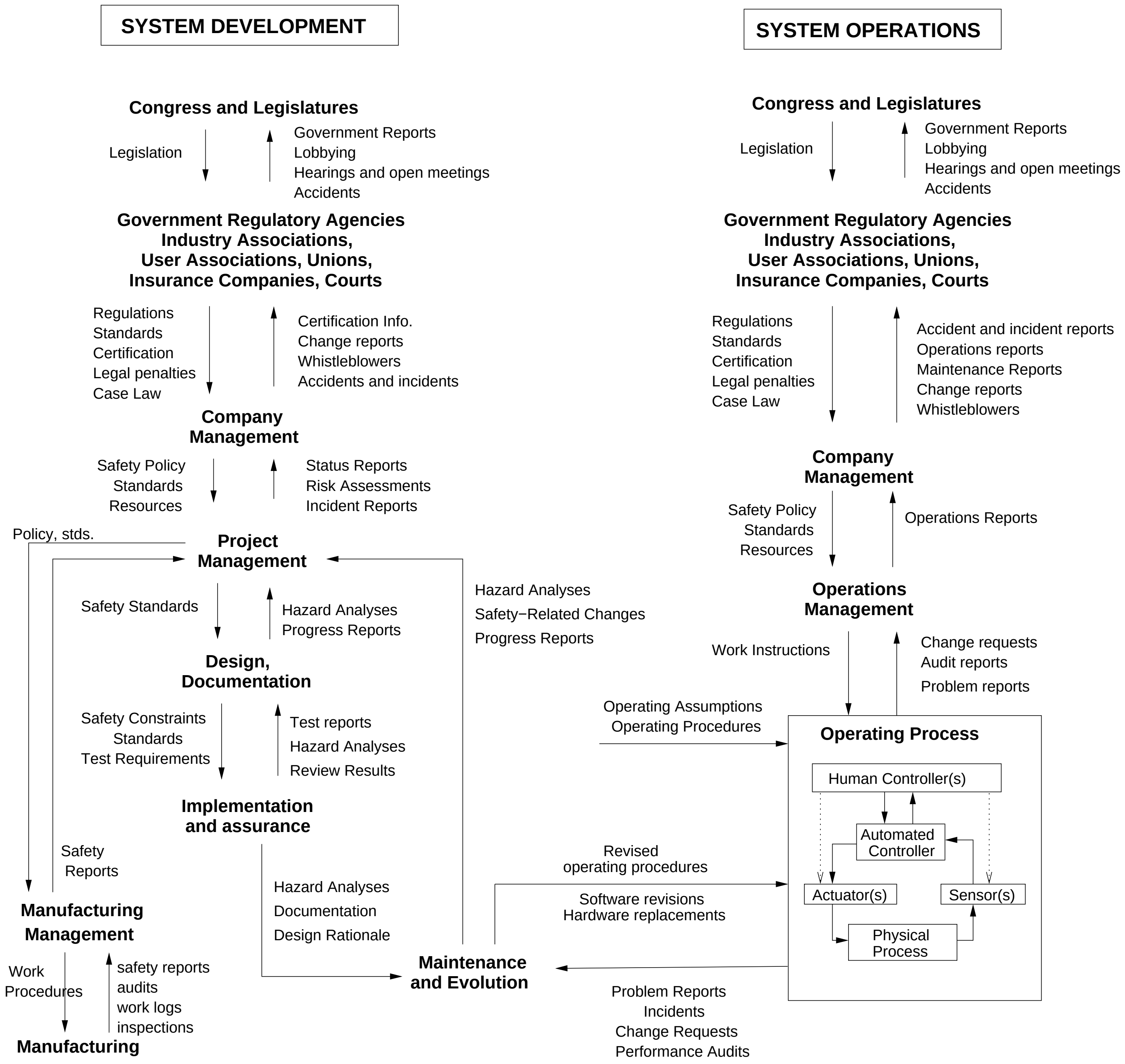
  e.g., O−ring did not control propellant gas release by sealing gap in field joint

  Software did not adequately control descent speed of Mars Polar Lander.

- Events are the result of the inadequate control

  Result from lack of enforcement of safety constraints

- To understand accidents, need to examine control structure itself to determine why inadequate to maintain safety constraints and why events occurred.


SYSTEM DEVELOPMENT
Congress and Legislatures
Legislation
Government Reports
Lobbying
Hearings and open meetings
Accidents
Government Regulatory Agencies
Industry Associations,
User Associations, Unions,
Insurance Companies, Courts
Regulations
Standards
Certification
Legal penalties
Case Law
Certification Info.
Change reports
Whistleblowers
Accidents and incidents
Company
Management
Safety Policy
Standards
Resources
Status Reports
Risk Assessments
Incident Reports
Policy, stds.
Project
Management
Safety Standards
Hazard Analyses
Progress Reports
Hazard Analyses
Safety−Related Changes
Progress Reports
Design,
Documentation
Safety Constraints
Standards
Test Requirements
Test reports
Hazard Analyses
Review Results
Implementation
and assurance
Safety
Reports
Hazard Analyses
Documentation
Design Rationale
Manufacturing
Management
Work
Procedures
safety reports
audits
work logs
inspections
Manufacturing
Maintenance
and Evolution
SYSTEM OPERATIONS
Congress and Legislatures
Legislation
Government Reports
Lobbying
Hearings and open meetings
Accidents
Government Regulatory Agencies
Industry Associations,
User Associations, Unions,
Insurance Companies, Courts
Regulations
Standards
Certification
Legal penalties
Case Law
Accident and incident reports
Operations reports
Maintenance Reports
Change reports
Whistleblowers
Company
Management
Safety Policy
Standards
Resources
Operations Reports
Operations
Management
Work Instructions
Change requests
Audit reports
Problem reports
Operating Assumptions
Operating Procedures
Operating Process
Human Controller(s)
Automated
Controller
Actuator(s)
Sensor(s)
Physical
Process
Revised
operating procedures
Software revisions
Hardware replacements
Problem Reports
Incidents
Change Requests
Performance Audits

Note:

- Does not imply need for a "controller"

  Component failures may be controlled through design
    e.g., redundancy, interlocks, fail−safe design

  or through process
    manufacturing processes and procedures
    maintenance procedures

- But does imply the need to enforce the safety constraints in some way.

- New model includes what do now and more

Accidents occur when:

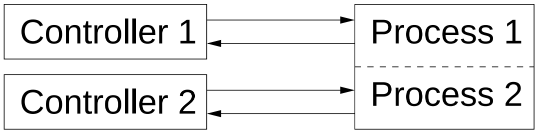
- Design does not enforce safety constraints

  unhandled disturbances, failures, dysfunctional interactions

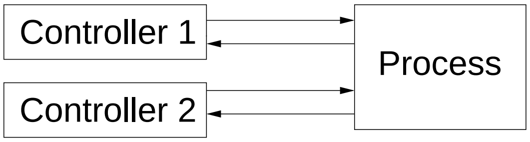
- Inadequate control actions

- Control structure degrades over time, asynchronous evolution

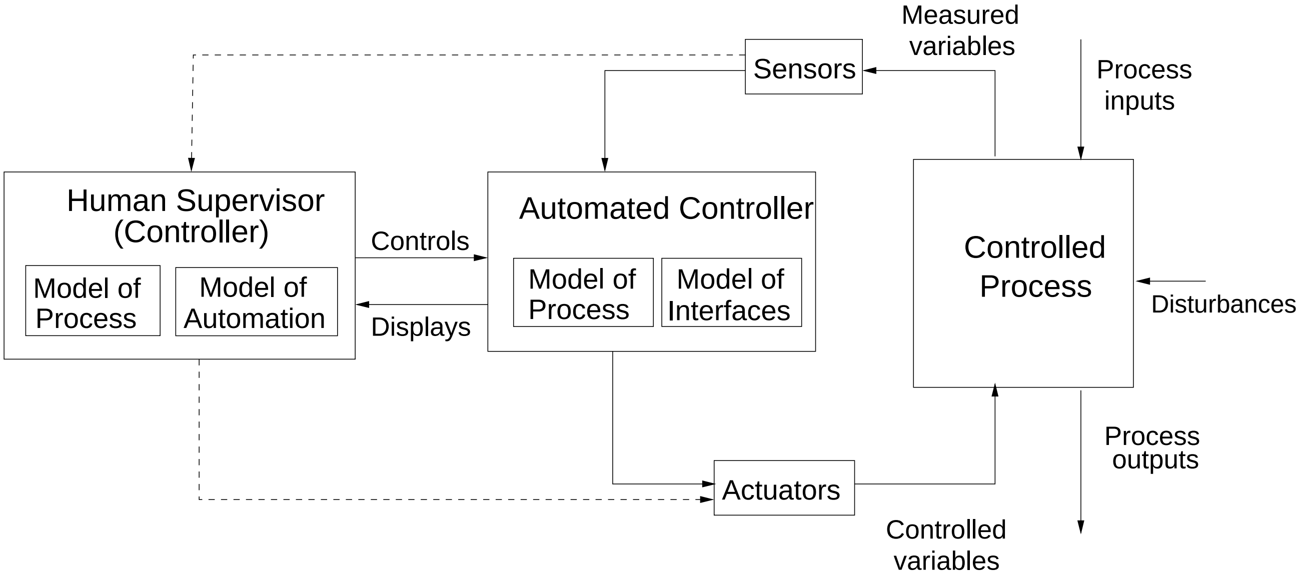
- Control actions inadequately coordinated among multiple controllers.
  - Boundary areas



  - Overlap areas (side effects of decisions and control actions)



Process Models



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 Model

- Accidents occur when the models do not match the process and incorrect control commands are given (or correct ones not given)
- How do they become inconsistent?
  - Wrong from beginning

    e.g. uncontrolled disturbances
    unhandled process states
    inadvertently commanding system into a hazardous state
    unhandled or incorrectly handled system component failures

    [Note these are related to what we called system accidents]
  - Missing or incorrect feedback and not updated correctly
  - Time lags not accounted for
- Explains most software−related accidents

Safety and Human Mental Models

- Explains developer errors
  - May have incorrect model of
    - required system or software behavior
    - development process
    - physical laws
    - etc.
- Also explains most human/computer interaction problems
  - Pilots and others are not understanding the automation

What did it just do?
Why did it do that?
What will it do next?
How did it get us into this state?
How do I get it to do what I want?

Why won't it let us do that?
What caused the failure?
What can we do so it does not happen again?

  - Or don't get feedback to update mental models or disbelieve it

---

Validating and Using the Model

- Can it explain (model) accidents that have already occurred?
- Is it useful?
  - In accident and mishap investigation
  - In preventing accidents

    Hazard analysis
    Designing for safety
- Is it better for these purposes than the chain−of−events model?

Using STAMP in Accident and Mishap Investigation and Root Cause Analysis

---

Modeling Accidents Using STAMP

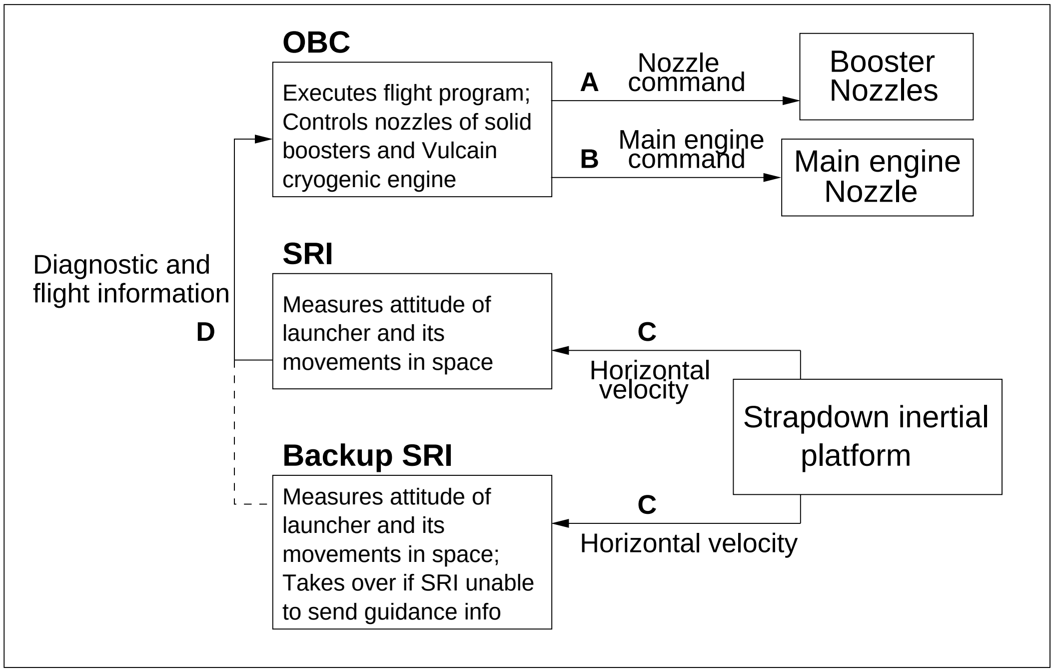
Three types of models are needed:

1. Static safety control structure
    - Safety requirements and constraints
    - Flawed control actions
    - Context (social, political, etc.)
    - Mental model flaws
    - Coordination flaws

2. Dynamic structure
    - Shows how the safety control structure changed over time

3. Behavioral dynamics
    - Dynamic processes behind the changes, i.e., why the system changes

ARIANE 5 LAUNCHER



Ariane 5: A rapid change in attitude and high aerodynamic loads stemming from a high angle of attack create aerodynamic forces that cause the launcher to disintegrate at 39 seconds after command for main engine ignition (H0).

Nozzles: Full nozzle deflections of solid boosters and main engine lead to angle of attack of more than 20 degrees.

Self−Destruct System: Triggered (as designed) by boosters separating from main stage at altitude of 4 km and 1 km from launch pad.

OBC (On−Board Computer)

OBC Safety Constraint Violated: Commands from the OBC to the nozzles must not result in the launcher operating outside its safe envelope.

Unsafe Behavior: Control command sent to booster nozzles and later to main engine nozzle to make a large correction for an attitude deviation that had not occurred.

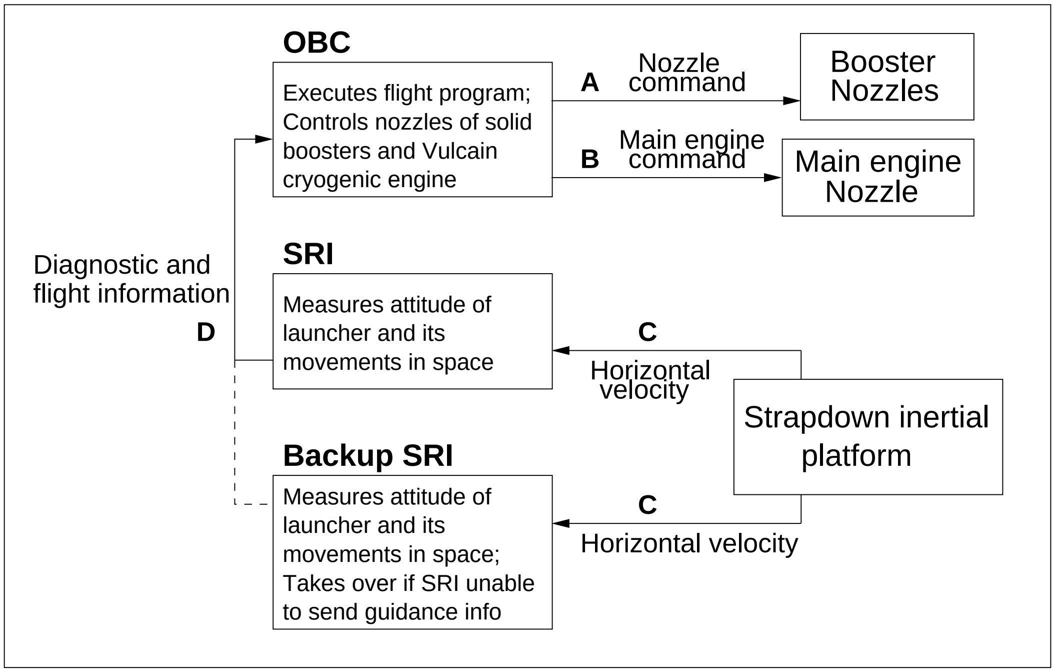
Process Model: Model of the current launch attitude is incorrect, i.e., it contains an attitude deviation that had not occurred. Results in incorrect commands being sent to nozzles.

Feedback: Diagnostic information received from SRI

Interface Model: Incomplete or incorrect (not enough information in accident report to determine which) − does not include the diagnostic information from the SRI that is available on the databus.

Control Algorithm Flaw: Interprets diagnostic information from SRI as flight data and uses it for flight control calculations. With both SRI and backup SRI shut down and therefore no possibility of getting correct guidance and attitude information, loss was inevitable.

ARIANE 5 LAUNCHER



SRI (Inertial Reference System):

SRI Safety Constraint Violated: The SRI must continue to send guidance information as long as it can get the necessary information from the strapdown inertial platform.

Unsafe Behavior: At 36.75 seconds after H0, SRI detects an internal error and turns itself off (as it was designed to do) after putting diagnostic information on the bus (D).

Control Algorithm: Calculates the Horizontal Bias (an internal alignment variable used as an indicator of alignment precision over time) using the horizontal velocity input from the strapdown inertial platform (C). Conversion from a 64−bit floating point value to a 16−bit signed integer leads to an unhandled overflow exception while calculating the horizontal bias. Algorithm reused from Ariane 4 where horizontal bias variable does not get large enough to cause an overflow.

Process Model: Does not match Ariane 5 (based on Ariane 4 trajectory data); Assumes smaller horizontal velocity values than possible on Ariane 5.

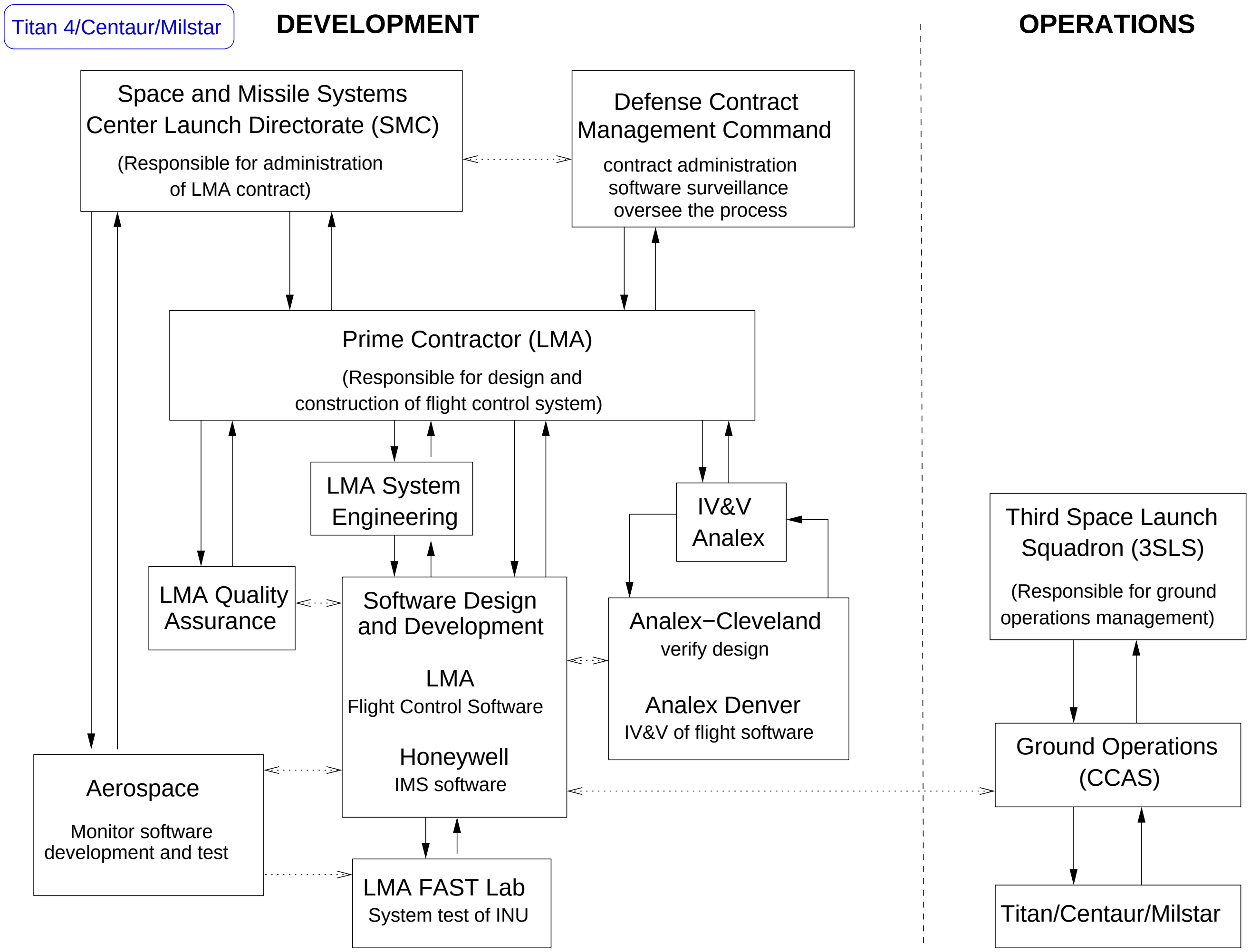
Backup SRI (Inertial Reference System):

SRI Safety Constraint Violated: The backup SRI must continue to send guidance information as long as it can get the necessary information from the strapdown inertial platform.

Unsafe Behavior: At 36.75 seconds after H0, backup SRI detects an internal error and turns itself off (as it was designed to do).

Control Algorithm: Calculates the Horizontal Bias (an internal alignment variable used as an indicator of alignment precision over time) using the horizontal velocity input from the strapdown inertial platform (C). Conversion from a 64−bit floating point value to a 16−bit signed integer leads to an unhandled overflow exception while calculating the horizontal bias. Algorithm reused from Ariane 4 where horizontal bias variable does not get large enough to cause an overflow. Because the algorithm was the same in both SRI computers, the overflow results in the same behavior, i.e., shutting itself off.

Process Model: Does not match Ariane 5 (based on Ariane 4 trajectory data); Assumes smaller horizontal velocity values than possible on Ariane 5.



Titan 4/Centaur/Milstar

DEVELOPMENT

OPERATIONS

Space and Missile Systems Center Launch Directorate (SMC)
(Responsible for administration of LMA contract)

Defense Contract Management Command
contract administration
software surveillance
oversee the process

Prime Contractor (LMA)
(Responsible for design and construction of flight control system)

LMA System Engineering

LMA Quality Assurance

Software Design and Development
LMA
Flight Control Software
Honeywell
IMS software

IV&V Analex

Analex−Cleveland
verify design
Analex Denver
IV&V of flight software

Aerospace
Monitor software development and test

LMA FAST Lab
System test of INU

Third Space Launch Squadron (3SLS)
(Responsible for ground operations management)

Ground Operations (CCAS)

Titan/Centaur/Milstar

Analex IV&V

Safety Constraint:

- IV&V must be performed on the as−flown system
- All safety−crtiical data and software must be included

Control Flaws:

- Designed an IV&V process that did not include load tape
- Used default values for testing software implementation
- Validated design constant but not actual constant

Mental Model Flaws:

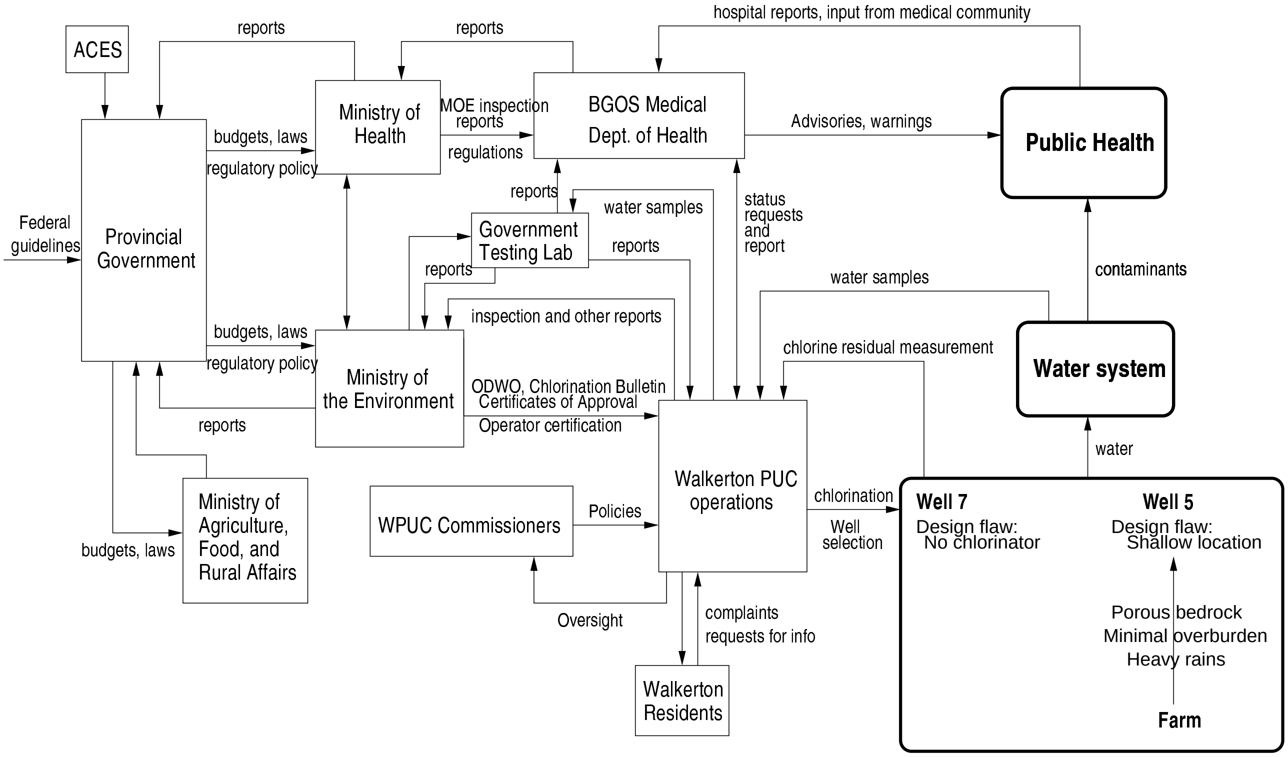
- Misunderstanding about what could be tested
- Misunderstainding of load tape creation process

System Hazard: Public is exposed to e. coli or other health–related contaminants through drinking water.

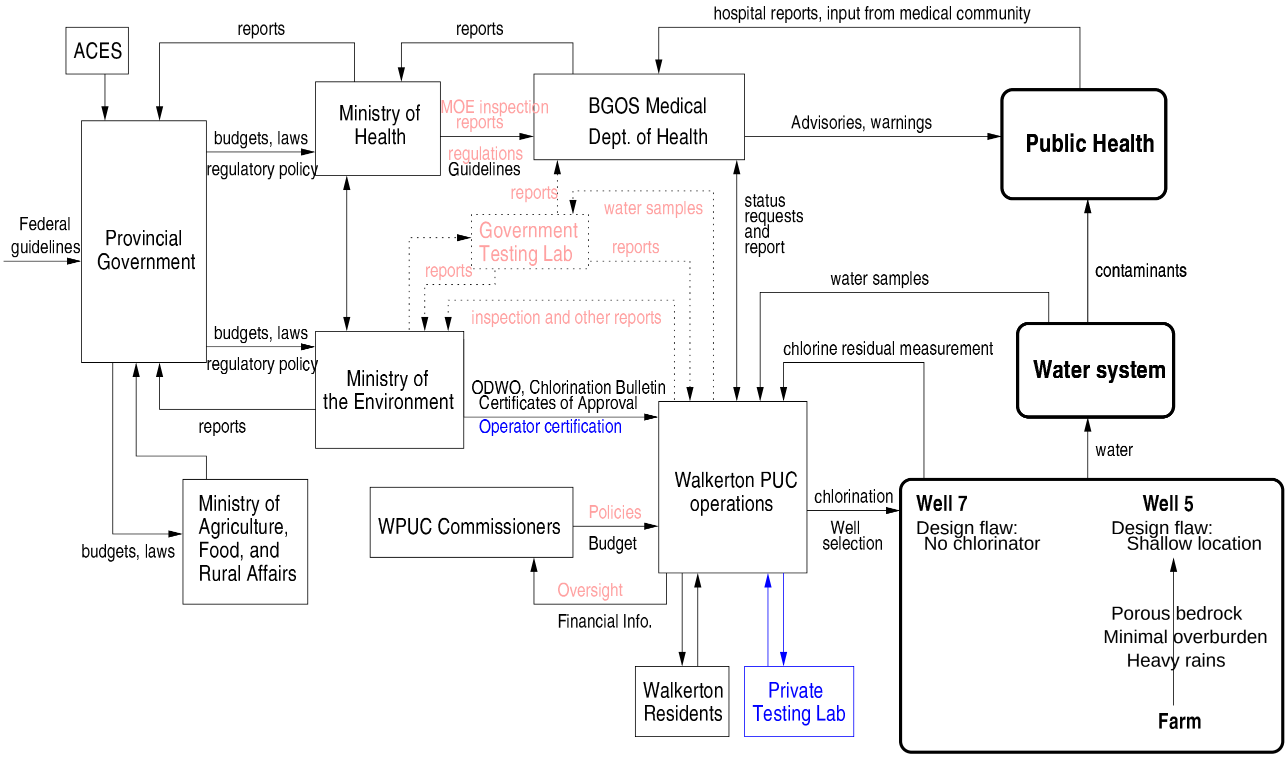
System Safety Constraints: The safety control structure must prevent exposure of the public to contaminated water.

(1) Water quality must not be compromised.

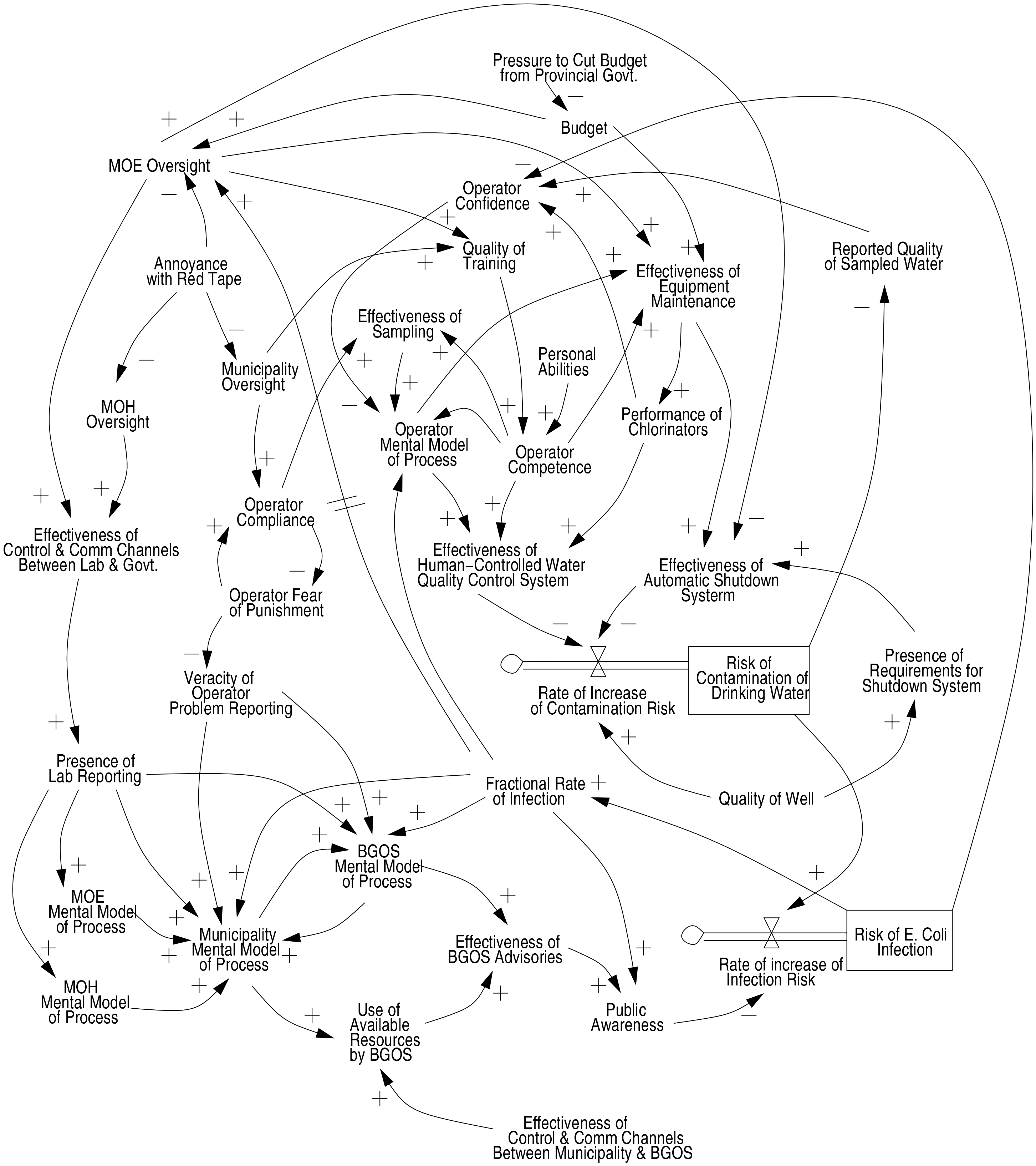
(2) Public health measures must reduce risk of exposure if water quality is compromised (e.g., notification and procedures to follow)



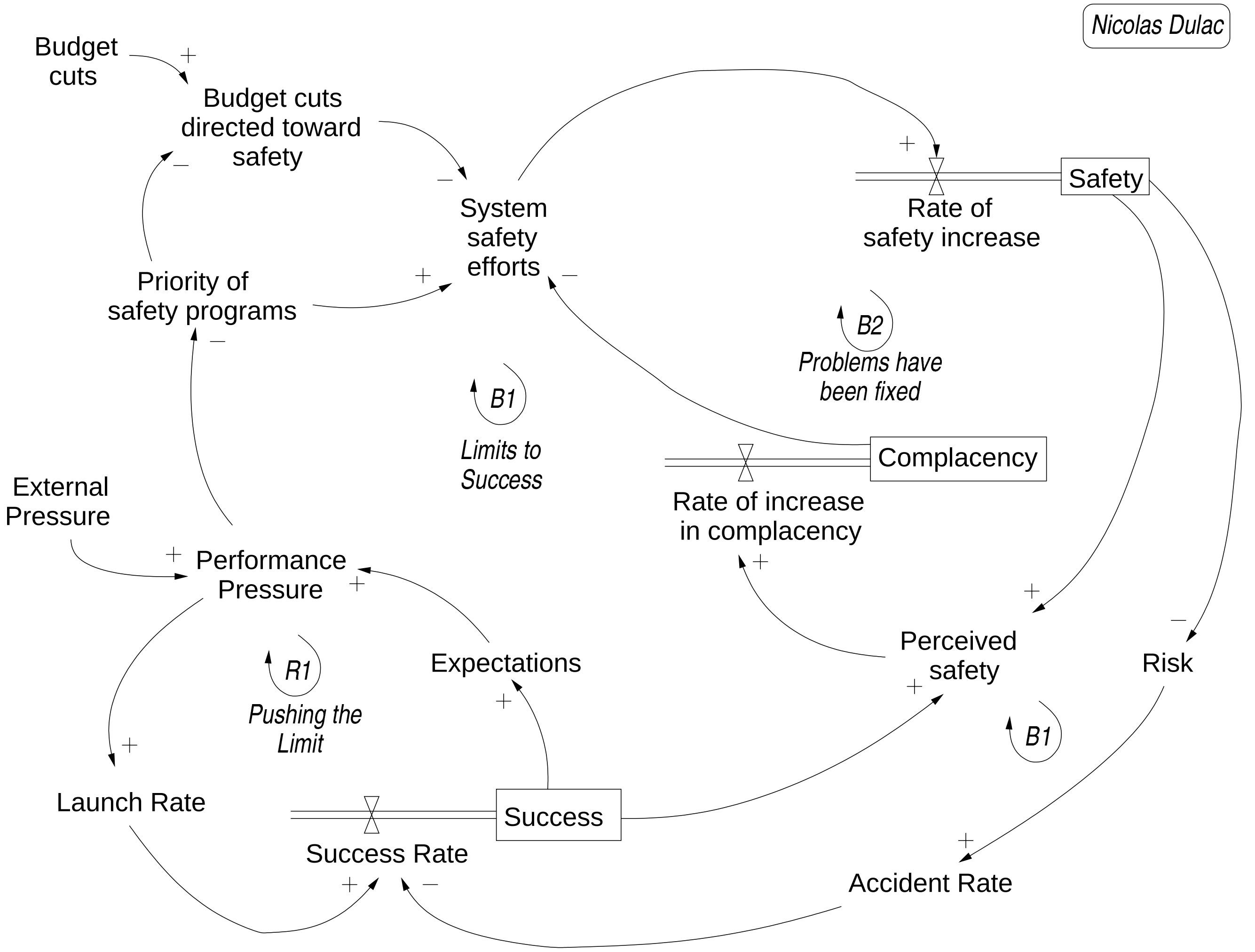
Dynamic Structure



Modeling Behavioral Dynamics



A (Partial) System Dynamics Model of the Columbia Accident



Steps in a STAMP analysis:

1. Identify
    - System hazards
    - System safety constraints and requirements
    - Control structure in place to enforce constraints
2. Model dynamic aspects of accident:
    - Changes to static safety control structure over time
    - Dynamic processes in effect that led to changes
3. Create the overall explanation for the accident
    - Inadequate control actions and decisions
    - Context in which decisions made
    - Mental model flaws
    - Control flaws (e.g., missing feedback loops)
    - Coordination flaws

STAMP vs. Traditional Accident Models

- Examines interrelationships rather than linear cause−effect chains
- Looks at the processes behind the events
- Includes entire socio−economic system
- Includes behavioral dynamics (changes over time)
    - Want to not just react to accidents and impose controls for a while, but understand why controls drift toward ineffectiveness over time and
        - Change those factors if possible
        - Detect the drift before accidents occur

Using STAMP to Prevent Accidents

Hazard Analysis

Safety Metrics and Performance Auditing

Risk Assessment

STAMP−Based Hazard Analysis (STPA)

- Provides information about how safety constraints could be violated.
    - Used to eliminate, reduce, and control hazards in system design, development, manufacturing, and operations
- Assists in designing safety into system from the beginning
    - Not just after−the−fact analysis
- Includes software, operators, system accidents, management, regulatory authorities
- Can use a concrete model of control (SpecTRM−RL) that is executable and analyzable

STPA − Step1: Identify hazards and translate into high−level requirements and constraints on behavior

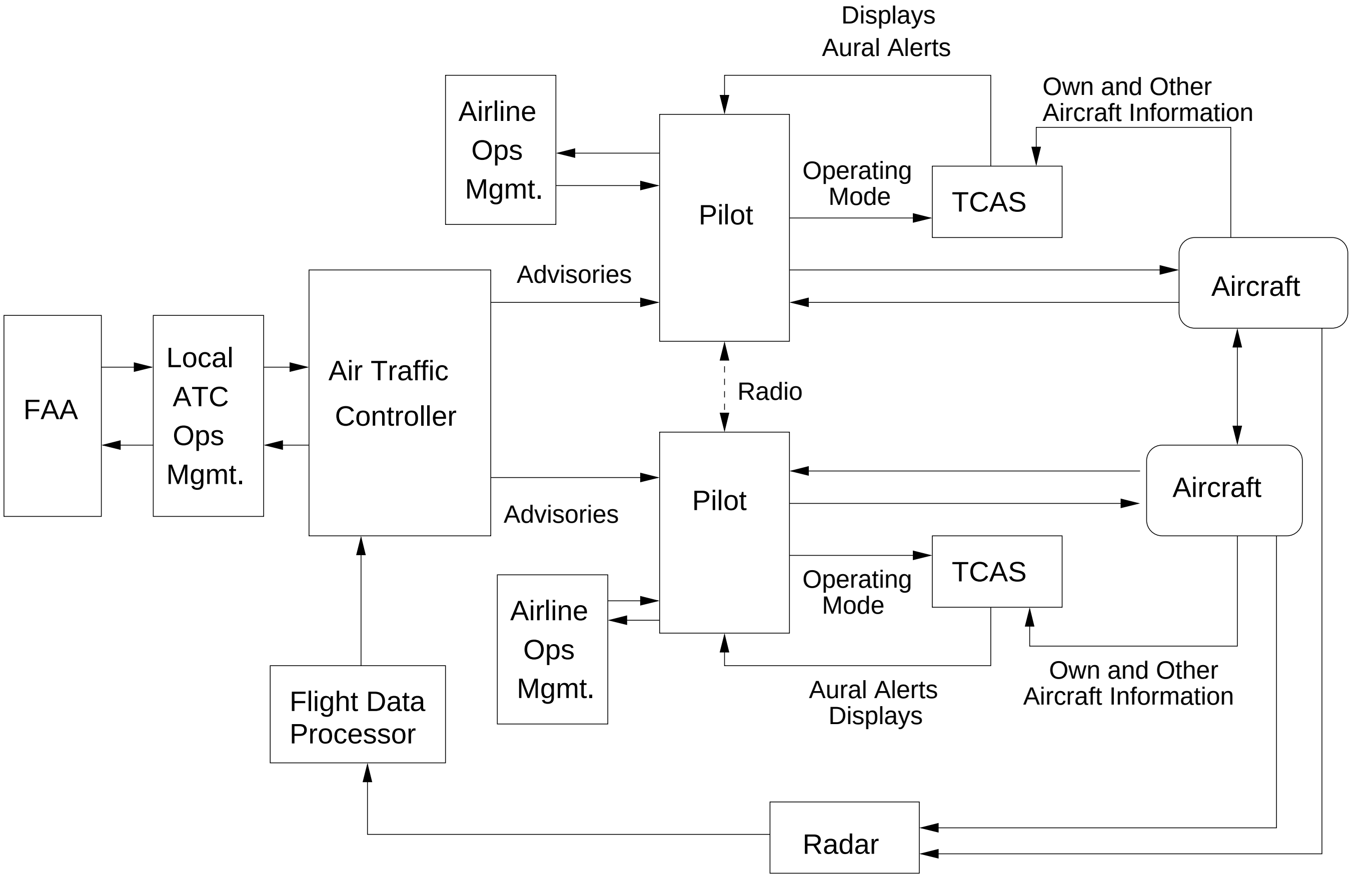
TCAS Hazards

1. A near mid−air collision (NMAC)

   (a pair of controlled aircraft violate minimum separation standards)

2. A controlled maneuver into the ground
3. Loss of control of aircraft
4. Interference with other safety−related aircraft systems
5. Interference with ground−based ATC system
6. Interference with ATC safety−related advisory

STPA − Step 2: Define basic control structure



STPA − Step 3: Identify potential inadequate control actions that could lead to hazardous process state

In general:

1. A required control action is not provided
2. An incorrect or unsafe control action is provided.
3. A potentially correct or inadequate control action is provided too late (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.

STPA − Step 4: Determine how potentially hazardous control actions could occur.

Eliminate from design or control or mitigate in design or operations

In general:

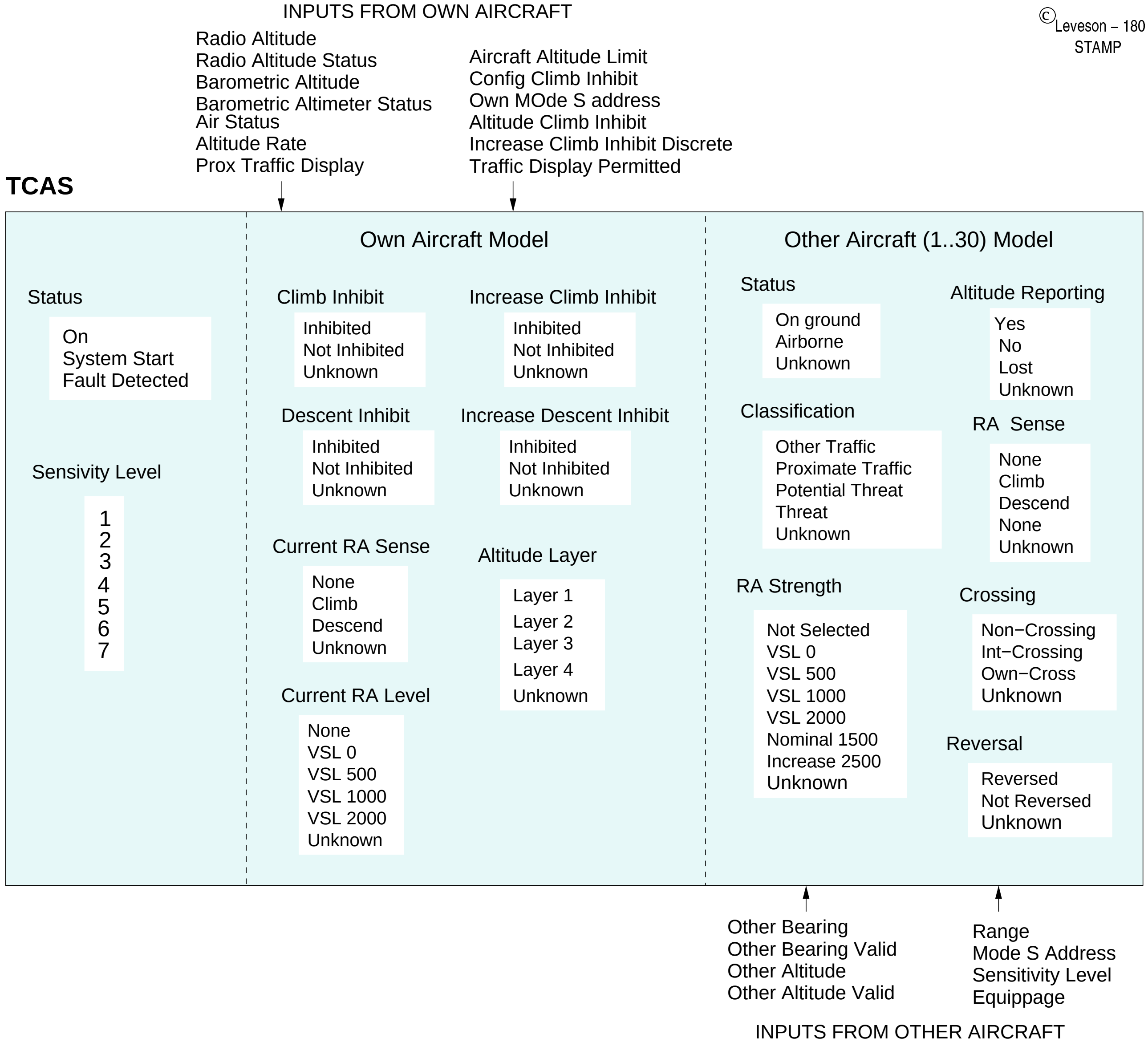
- Can use a concrete model in SpecTRM−RL
  - Assists with communication and completeness of analysis
  - Provides a continuous simulation and analysis environment to evaluate impact of faults and effectiveness of mitigation features.

Step 4a: Augment control structure with process models for each control component

Step 4b: For each of inadequate control actions, examine parts of control loop to see if could cause it.

- Guided by set of generic control loop flaws
- Where human or organization involved must evaluate:
  - Context in which decisions made
  - Behavior−shaping mechanisms (influences)

Step 4c: Consider how designed controls could degrade over time



INPUTS FROM OWN AIRCRAFT

Radio Altitude
Radio Altitude Status
Barometric Altitude
Barometric Altimeter Status
Air Status
Altitude Rate
Prox Traffic Display

Aircraft Altitude Limit
Config Climb Inhibit
Own MOde S address
Altitude Climb Inhibit
Increase Climb Inhibit Discrete
Traffic Display Permitted

TCAS

Status
On
System Start
Fault Detected

Sensivity Level
1
2
3
4
5
6
7

Own Aircraft Model

Climb Inhibit
Inhibited
Not Inhibited
Unknown

Increase Climb Inhibit
Inhibited
Not Inhibited
Unknown

Descent Inhibit
Inhibited
Not Inhibited
Unknown

Increase Descent Inhibit
Inhibited
Not Inhibited
Unknown

Current RA Sense
None
Climb
Descend
Unknown

Altitude Layer
Layer 1
Layer 2
Layer 3
Layer 4
Unknown

Current RA Level
None
VSL 0
VSL 500
VSL 1000
VSL 2000
Unknown

Other Aircraft (1..30) Model

Status
On ground
Airborne
Unknown

Altitude Reporting
Yes
No
Lost
Unknown

Classification
Other Traffic
Proximate Traffic
Potential Threat
Threat
Unknown

RA Sense
None
Climb
Descend
None
Unknown

RA Strength
Not Selected
VSL 0
VSL 500
VSL 1000
VSL 2000
Nominal 1500
Increase 2500
Unknown

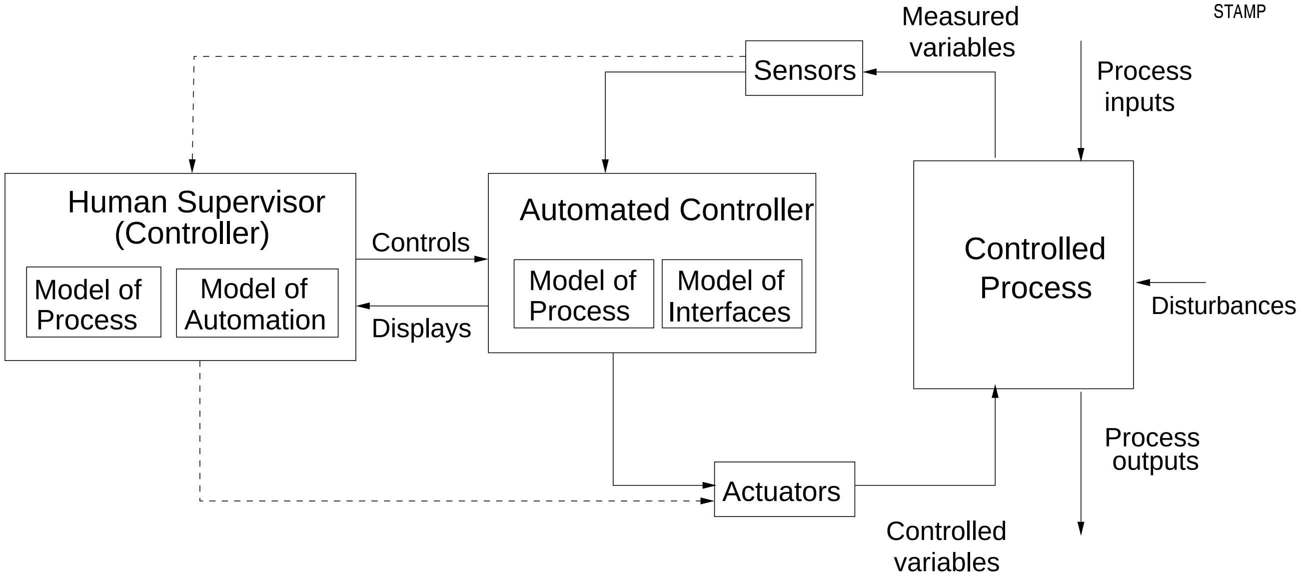
Crossing
Non−Crossing
Int−Crossing
Own−Cross
Unknown

Reversal
Reversed
Not Reversed
Unknown

Other Bearing
Other Bearing Valid
Other Altitude
Other Altitude Valid

Range
Mode S Address
Sensitivity Level
Equippage

INPUTS FROM OTHER AIRCRAFT



STPA − Step 4b: Examine control loop for potential to cause inadequate control actions

- **Inadequate Control Actions (enforcement of constraints)**
  - Design of control algorithm (process) does not enforce constraints
  - Process models inconsistent, incomplete, or incorrect (lack of linkup)
    - Flaw(s) in creation or updating process
    - Inadequate or missing feedback
      - Not provided in system design
      - Communication flaw
      - Inadequate sensor operation (incorrect or no information provided)
    - Time lags and measurement inaccuracies not accounted for
  - Inadequate coordination among controllers and decision−makers (boundary and overlap areas)
- **Inadequate Execution of Control Action**
  - Communication flaw
  - Inadequate "actuator" operation
  - Time lag

STPA − Step4c: Consider how designed controls could degrade over time.

- E.g., specified procedures ==> effective procedures
- Use system dynamics models?

- Use information to design protection against changes:

e.g. operational procedures
controls over changes and maintenance activities
auditing procedures and performance metrics
management feedback channels to detect unsafe changes

Comparisons with Traditional HA Techniques

- Top−down (vs. bottom−up like FMECA)
- Considers more than just component failures and failure events
- Guidance in doing analysis (vs. FTA)
- Handles dysfunctional interactions, software, management, etc.
- Concrete model (not just in head)
  - Not physical structure (HAZOP) but control (functional) structure
  - General model of inadequate control
    - HAZOP guidewords based on model of accidents being caused by deviations in system variables
    - Includes HAZOP model but more general
- Compared with TCAS II Fault Tree (MITRE)

  STPA results more comprehensive