safety
Cost-Effective^Certification of Software-Intensive Systems

Prof. Nancy Leveson

MIT

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

- Accident: An undesired and unplanned event that results in a loss (including death and injury, property damage, environmental pollution, etc.)
- Safety: Freedom from accidents

Why are our Efforts Often Not Cost-Effective?

- Efforts superficial, isolated, or misdirected
- Safety efforts start too late
- Focus on compliance, not building in safety
- Inappropriate techniques for systems built today
- Focus efforts only on technical components of system
- Systems assumed to be static through lifetime
- Success can lead to failure (risk perception)
- Limited learning from events

Safety Regulation Approaches

- Prescriptive
  - Product
    - Specific design features (e.g., electrical codes)
    - General design features (e.g., fail-safe, protection system)
  - Process: process to be followed in
    - Designing and implementing the system
    - Assuring safety
- Goal or Performance-Based:
  - Set a goal and developer decides how to achieve it.

Mil-STD-882: Defense System Safety

- Purpose:

  "Provide uniform requirements for developing and implementing a system safety program of sufficient comprehensiveness to identify the hazards of a system and to impose design requirements and management controls to prevent mishaps."

- Applies to entire lifecycle (including operations)
- Specifies *what* but not *how*
- Tailorable: written as a set of tasks that may or may not be required

MIL-STD-882

- Standard tells <u>what</u> to do, but not <u>how</u>.
- Developers must create a System Safety Plan, which is approved by the customer.
- Tailorable: Set of tasks that can be levied, depending on type of system being developed.

    100: Program Management and Control

    200: Design and Integration (Hazard ID and analysis)

    300: Design Evaluation

    400: Compliance and Verification

MIT-STD-882 Objectives

The safety program shall specify a systematic approach to make sure that:

- Safety, consistent with mission requirements, is designed into the system in a timely, cost-effective manner.
- Hazards associated with each system are identified, tracked, evaluated, and eliminated or controlled throughout the entire life-cycle of a system.
- Historical safety data, including lessons learned from other systems, are considered and used.
- Actions taken to eliminate or control hazards are documented and rigorously reviewed.

MIL-STD-882 Objectives (2)

- Retrofit actions required to improve safety are minimized through the timely inclusion of safety features during research, technology development for, and acquisition of a system.
- Changes in design, configuration, or mission requirements are accomplished in a manner that does not introduce new hazards nor reduce control of existing ones ("management of change").
- "Lessons learned" are documented and changes made to development and operational processes.

Argument-Based Safety Cases

- Nimrod accident investigation placed primary responsibility for accident on use of safety cases.
- Almost no evidence these are effective.
- In fact, most of the ones I have seen in the safety literature are incorrect.

Argument-Based Safety Cases

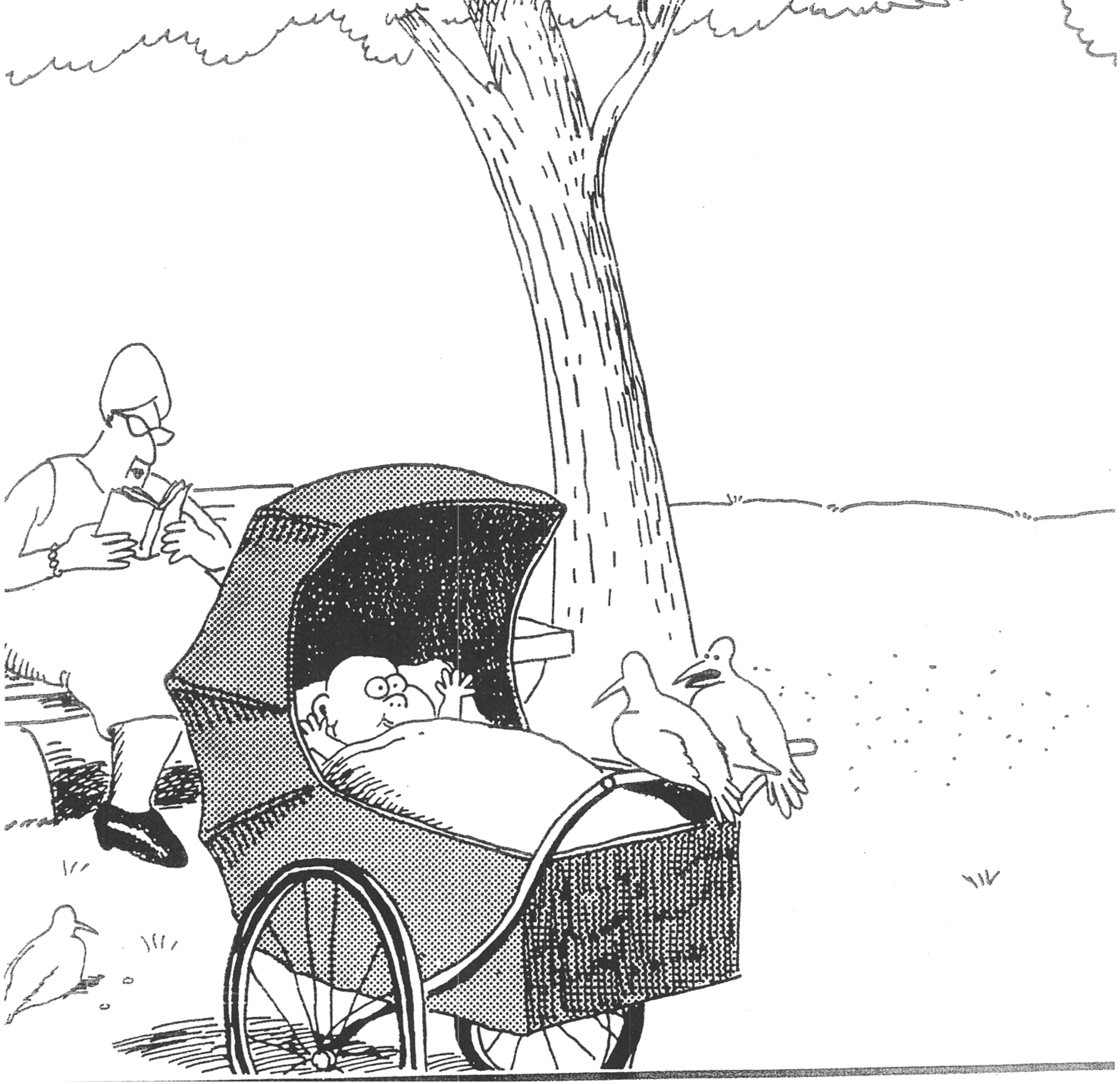
What is going on here?

– There is always a way to argue that something is safe, whether it is or not. Always possible to produce evidence that something is safe.

– *Confirmation Bias*:

  - People will focus on and interpret evidence in a way that confirms the goal they have set for themselves.
  - If the goal is to prove the system is safe, they will focus on the evidence that shows it is safe and create an argument for safety.

- The solution is <u>not</u> to use an argument for safety as the basis for certifying safety.



It’s still hungry … and I’ve been stuffing worms into it all day.

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

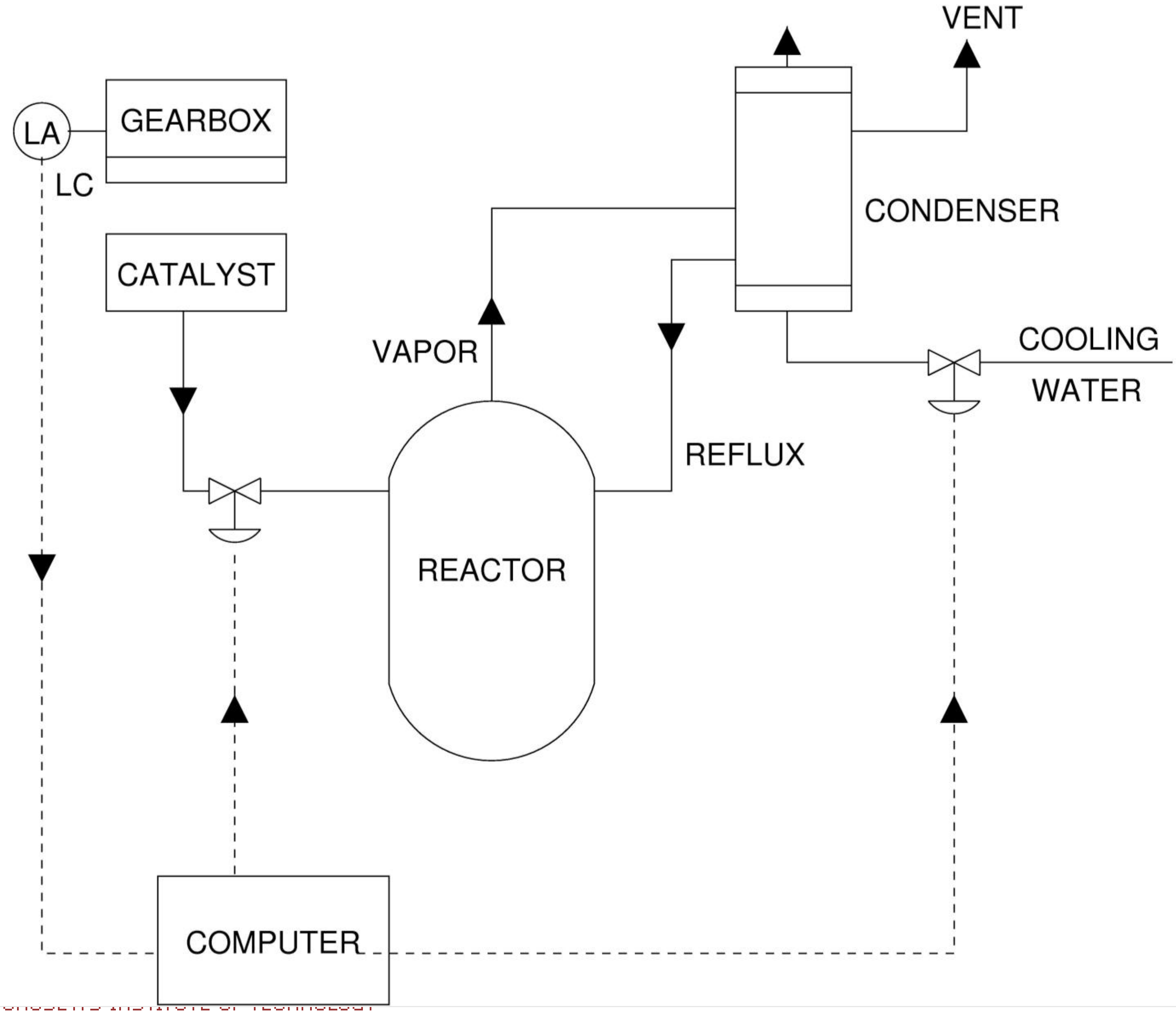
<u>Assumption</u>:

- Accidents are caused by system component failure(s)
- Safety is increased by increasing the reliability of the individual system components. If components do not fail, then accidents will not occur.

<u>Assumption</u>:

- Software can be treated just like hardware (with perhaps a few minor changes).
- Highly reliable software is safe.

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
  - Exacerbated by introduction of computers and software

Safety ≠ 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.
- But this is untrue for complex, software-intensive socio-technical systems.

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.

Engineering a Safer World (MIT Press, fall 2011)

http://sunnyday.mit.edu/safer-world

- Expanded model of accident causality based on system theory (not reliability theory)
- Treats safety as a dynamic control problem
- Handles software, hardware, human errors, management, culture …
- Overall goal is to enforce constraints on system (and software) behavior through appropriate design, analysis, operations, and management.

STAMP (2)

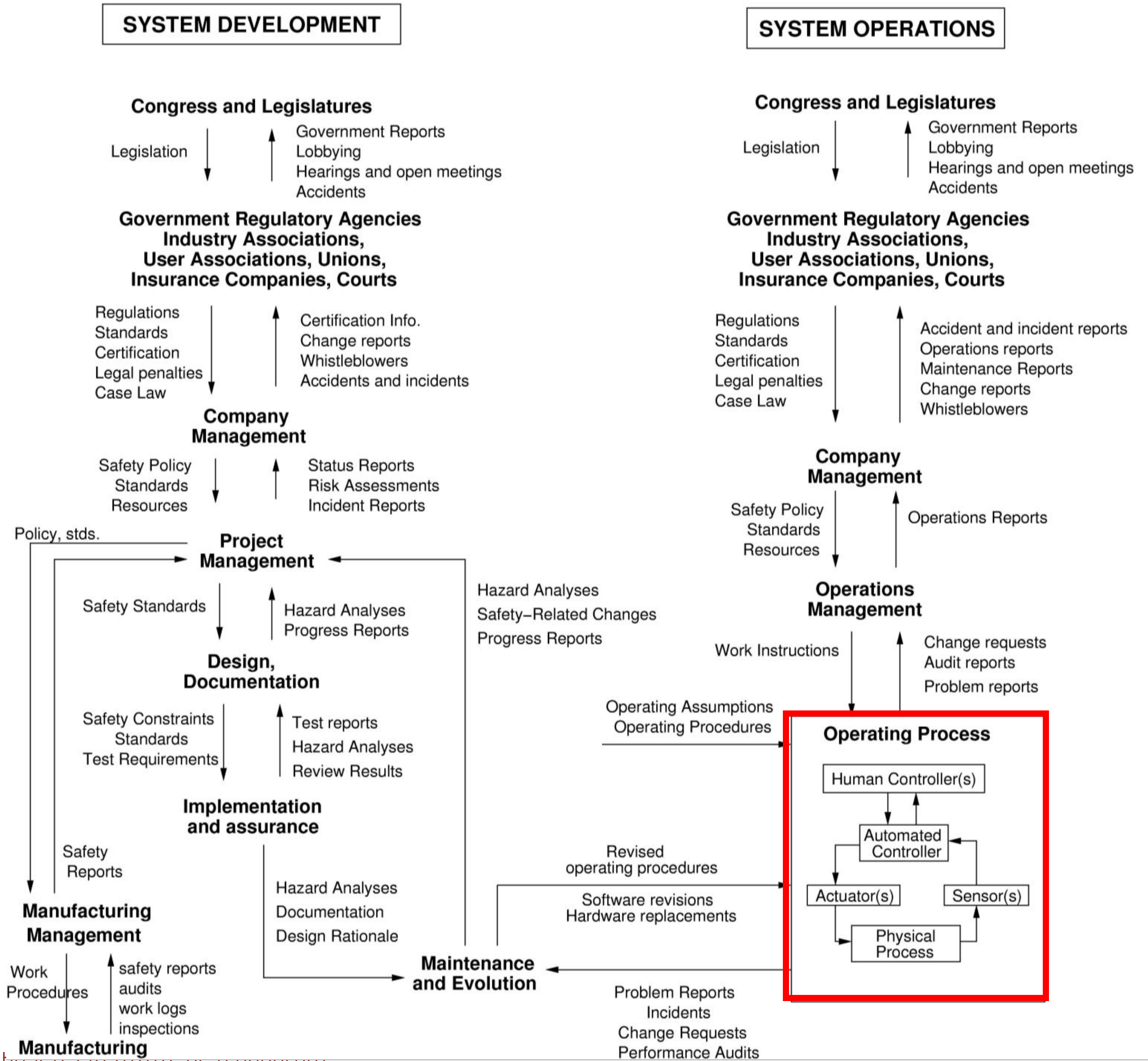
- Systems can be viewed as hierarchical control structures
  - Systems are viewed as interrelated components kept in a state of dynamic equilibrium by feedback loops of information and control
  - Controllers imposes constraints upon the activity at a lower level of the hierarchy: safety constraints
- A change in emphasis:

~~“prevent failures”~~

↓

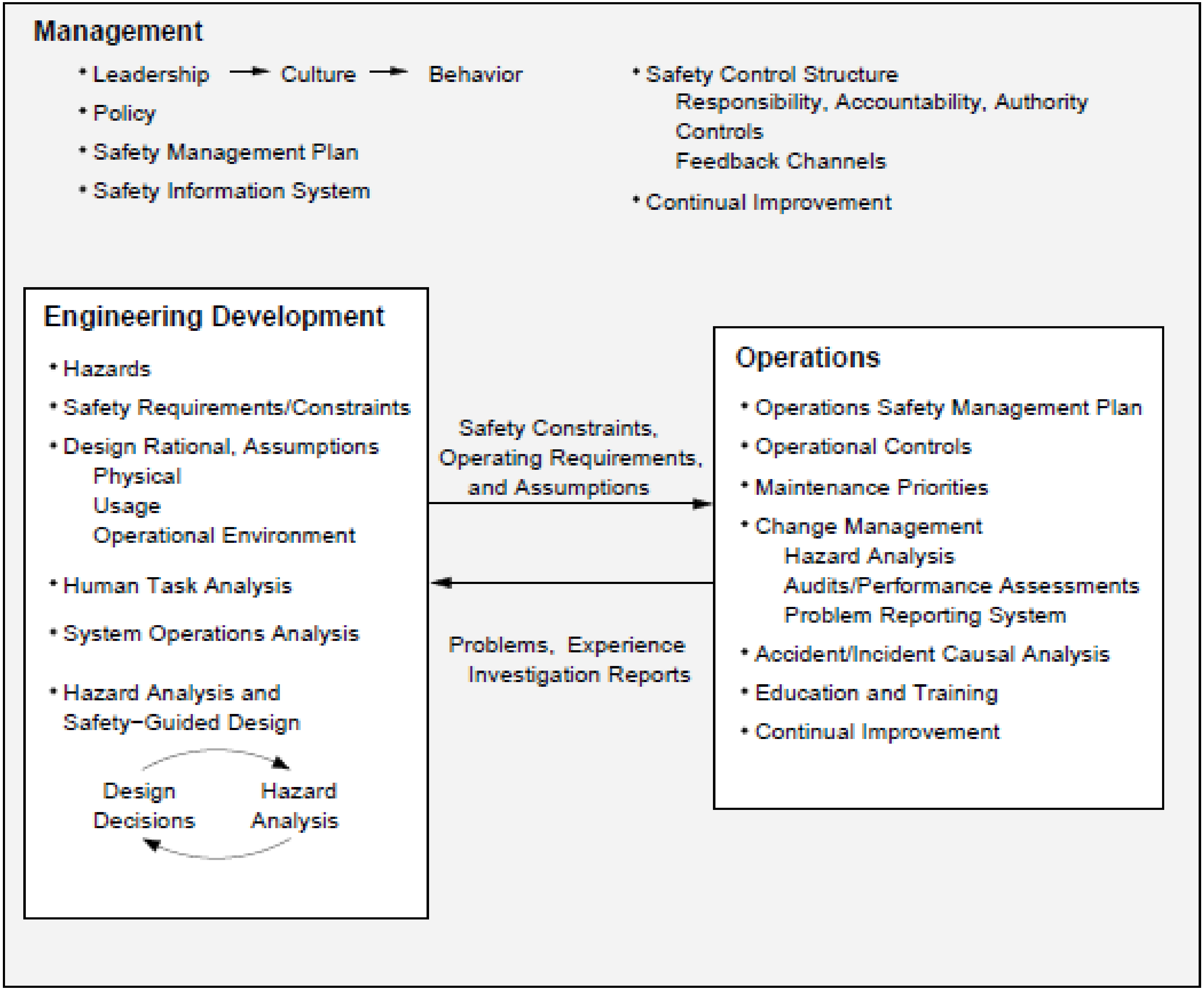
“enforce safety constraints on system behavior”

Example Safety Control Structure

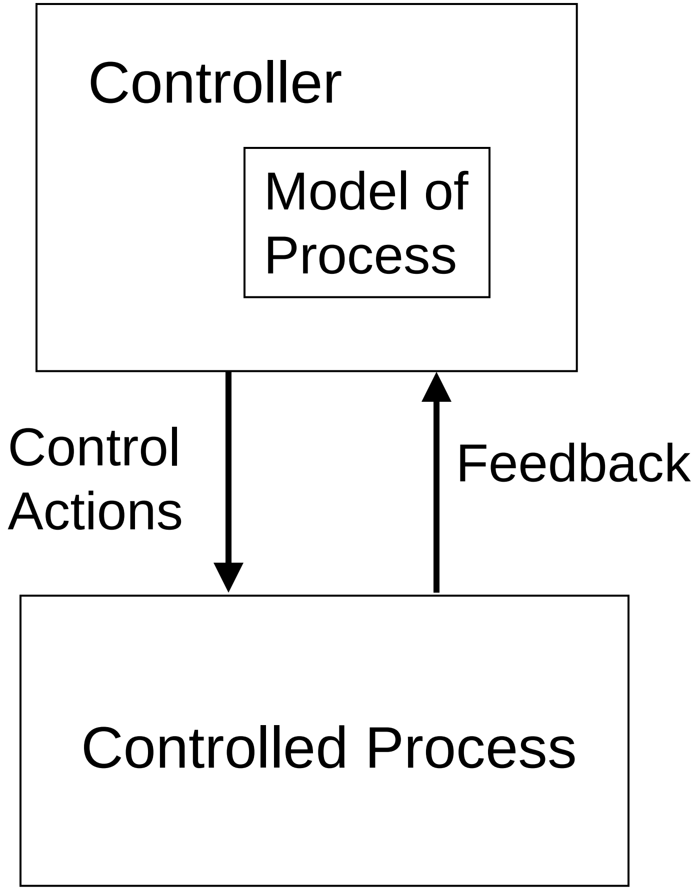


STAMP: System's Theoretic Accident Model and Processes (1)

- 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 <u>result</u> of the inadequate control
  - Result from lack of enforcement of safety constraints
  - Need to examine larger process and not just event chain

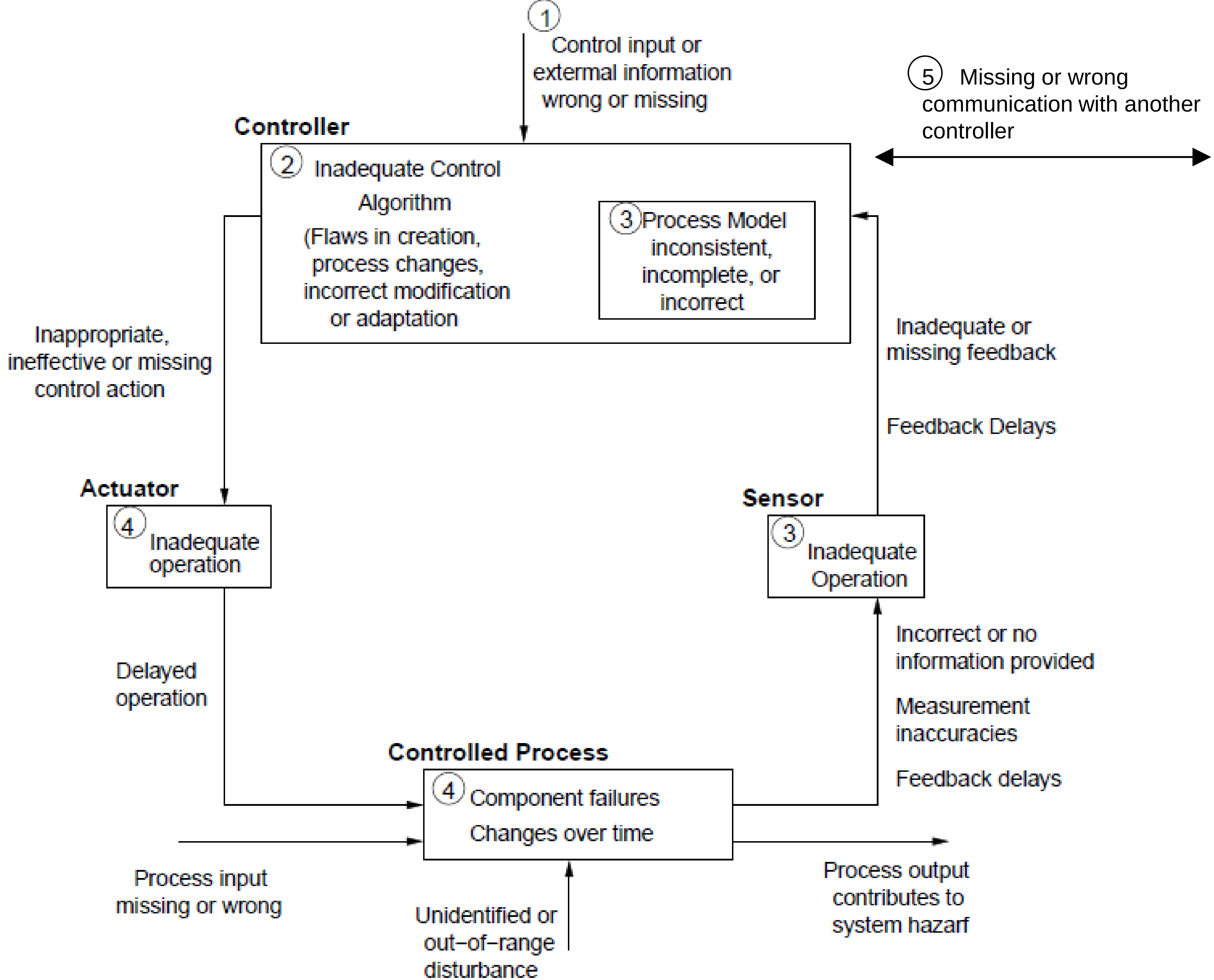

Management
Leadership → Culture → Behavior
Policy
Safety Management Plan
Safety Information System
Safety Control Structure
Responsibility, Accountability, Authority
Controls
Feedback Channels
Continual Improvement
Engineering Development
Hazards
Safety Requirements/Constraints
Design Rational, Assumptions
Physical
Usage
Operational Environment
Human Task Analysis
System Operations Analysis
Hazard Analysis and Safety−Guided Design
Design Decisions
Hazard Analysis
Safety Constraints, Operating Requirements, and Assumptions
Problems, Experience Investigation Reports
Operations
Operations Safety Management Plan
Operational Controls
Maintenance Priorities
Change Management
Hazard Analysis
Audits/Performance Assessments
Problem Reporting System
Accident/Incident Causal Analysis
Education and Training
Continual Improvement

Control processes operate between levels of control



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

Feedback channels are critical
- -- Design
- -- Operation



1
Control input or
extermal information
wrong or missing
5 Missing or 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

STPA (System Theoretic Process Analysis)

- Starts from a model of the functional control structure
  1. Identifies the behavioral safety constraints that must be enforced on the system and component behavior in order to prevent accidents (i.e., the safety requirements and constraints)
  2. Identifies potential causes of violation of the requirements (scenarios leading to unsafe system behavior).
- Has been successfully used on enormously complex systems and found many more potential problems than fault trees.
- Supports safety-guided design where intertwine design decisions with analysis to assist in decision making.

Uses for STAMP

- Create 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
- New holistic approaches to security

Does it work? Is it practical?

Technical

- Safety analysis of new missile defense system (MDA)
- Safety-driven design of new JPL outer planets explorer
- Safety analysis of the JAXA HTV (unmanned cargo spacecraft to ISS)
- Incorporating risk into early trade studies (NASA Constellation)
- Orion (Space Shuttle replacement)
- Safety of maglev trains (Japan Central Railway)
- NextGen (for NASA, just starting)
- Accident/incident analysis (aircraft, petrochemical plants, air traffic control, railway accident, …)
- Medical devices (artificial pancreas, proton therapy device)
- Automotive (adaptive cruise control)

Does it work? Is it practical?

Social and Managerial

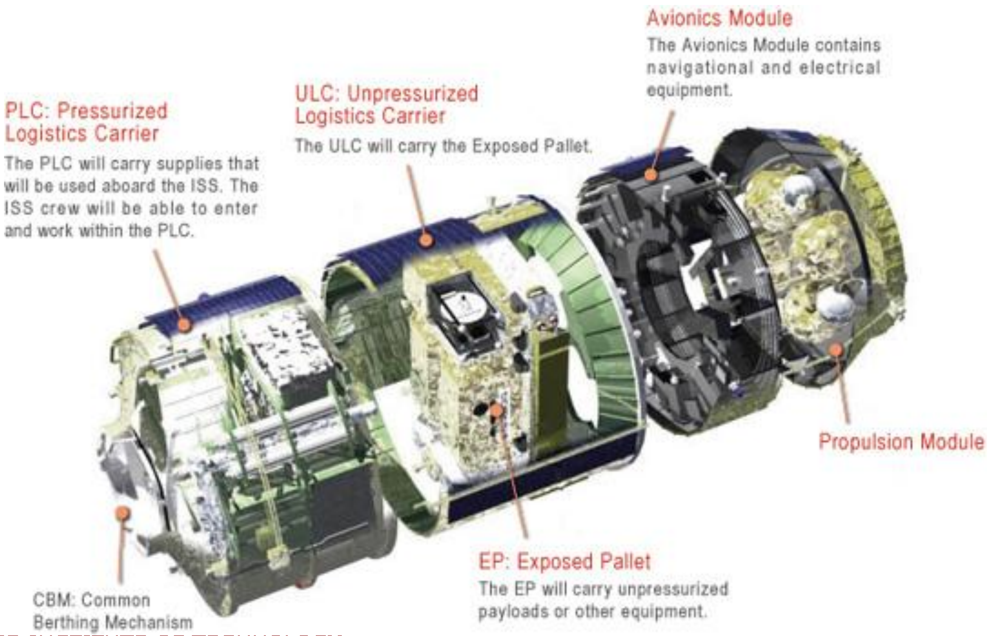
- Analysis of the management structure of the space shuttle program (post-Columbia)
- Risk management in the development of NASA’s new manned space program (Constellation)
- NASA Mission control – re-planning and changing mission control procedures safely
- Food safety
- Safety in pharmaceutical drug development
- Risk analysis of outpatient GI surgery at Beth Israel Deaconess Hospital
- Analysis and prevention of corporate fraud

Evaluation (1)

- Performed a non-advocate risk assessment for inadvertent launch on new BMDS
- Deployment and testing of BMDS 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
    - E.g., missing cases in software, obscure timing interactions
    - Could not be found by fault trees or other standard techniques
  - Complexity of component interactions led to unanticipated system behavior
  - STPA also identified component failures that could cause inadvertent launch (most analysis techniques consider only these failure events)
- Now being used proactively as changes made to system

Evaluation (2)

- Joint research project between MIT and JAXA to determine feasibility and usefulness of STPA for JAXA projects
- Comparison between STPA and FTA for HTV
  - Problems identified?
  - Resources required?



Comparison between STPA and FTA

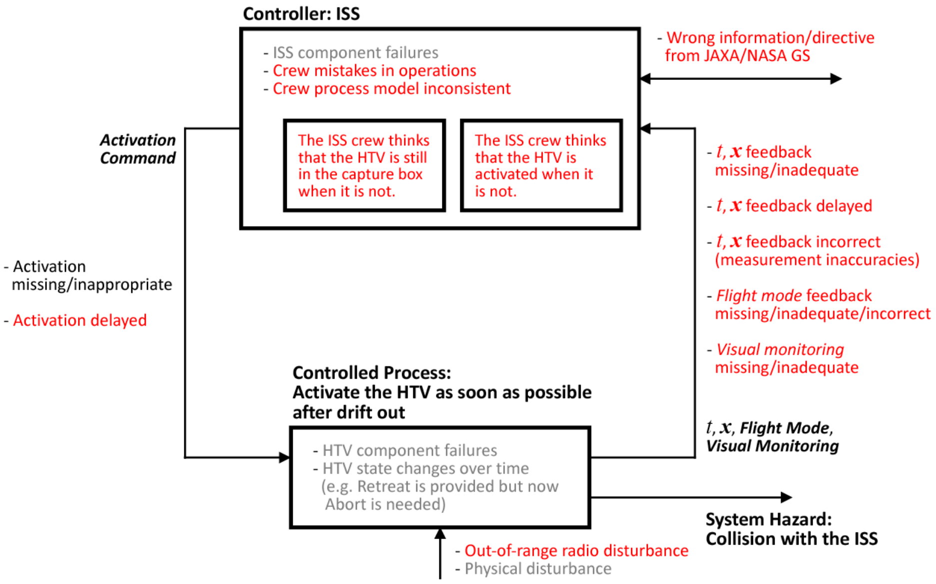
- ISS component failures
- Crew mistakes in operation
- Crew process model inconsistent

- Activation missing/inappropriate
- Activation delayed

- HTV component failures
- HTV state changes over time
- Out-of-range radio disturbance
- Physical disturbance

- t, x feedback missing/inadequate
- t, x feedback delayed
- t, x feedback incorrect
- *Flight Mode* feedback missing/inadequate
- *Flight Mode* feedback incorrect
- *Visual Monitoring* missing/inadequate

- Wrong information/directive from JAXA/NASA GS



Identified by both (STPA and FTA)
Identified by STPA only

Conclusions

- Safety is a system problem, not a software (component) problem.
  - Cannot certify software "safety" in isolation (software by itself is not safe or unsafe)
- The problem is in the requirements (externally visible behavior)
- Software that is "correct" and reliable (satisfies its specification) is not necessarily safe.
- Safety must be built in, cannot add later or certify it in

Conclusions

- Traditional safety engineering techniques are based on assumptions no longer true for the systems we are building
- Trying to add software and human error to traditional accident models and techniques is hopeless
- A new, more sophisticated causality model is needed to handle the new causes of accidents and the complexity in our modern systems
- Cannot certify software independent from a particular system (safety is a system property, not a component property)
  - Safety is an emergent system property, not a component property