A Process for STPA

STAMP Accident Model of HITOMI and Expansion to Future Safety Culture

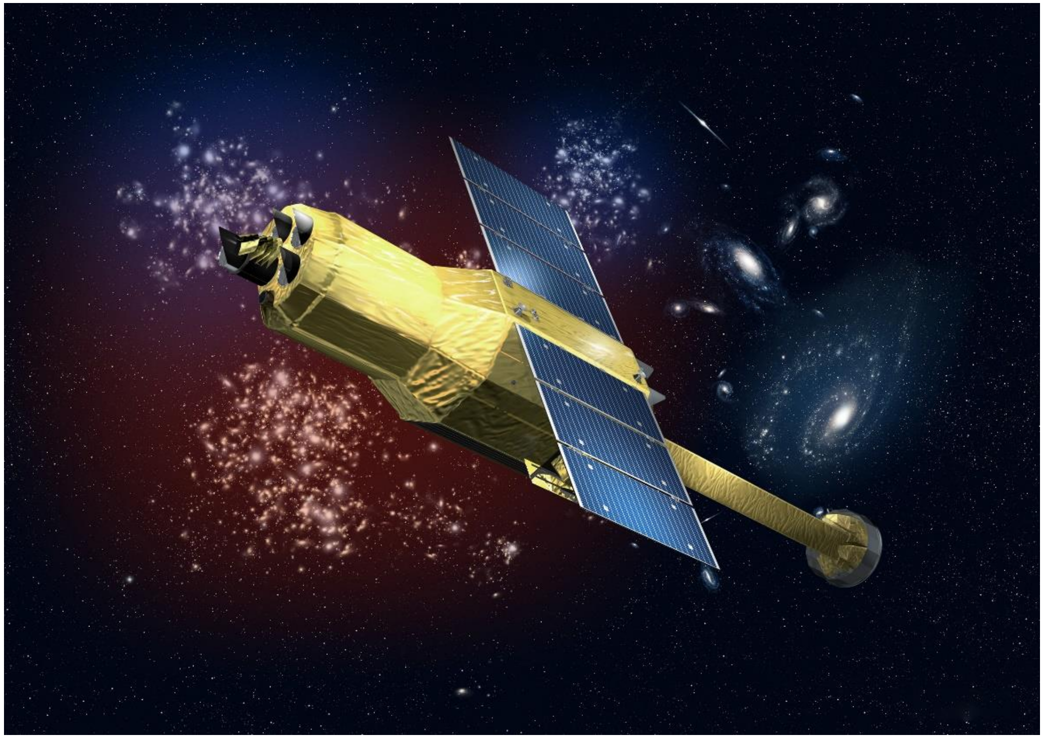
John Thomas, Nancy Leveson (MIT),

Naoki Ishimama, Masa Katahira (JAXA),

Nobuyuki Hoshino, Kazuki Kakimoto (JAMSS)

HITOMI ASTRO-H Satellite (2016)

- Unexpected behavior during a mode change
    - Process model flaw: computer suddenly believed it was spinning (it wasn't)
    - Computer commanded faster and faster rotation
    - Ripped itself apart
- Engineers had discussed this process model flaw
    - Decided not to fix
    - In normal operation, would correct itself automatically
    - BUT: other contexts and interactions easy to overlook
- Investigation result:
    - Project was lacking an "approach to examine the overall design of the spacecraft"
- JAXA statement:
    - "We were unable to let go of our usual methods"



All components operated as designed!
Not a simple component failure!

http://www.asahi.com/ajw/articles/AJ201606200006.html

STPA: Accidents and Hazards

- Accidents
  - A-1: Scientific mission is not performed (mission loss)
- System Hazards
  - H-1: ASTRO-H unable to collect scientific data
  - H-2: ASTRO-H unable to communicate scientific data

System Block Diagram

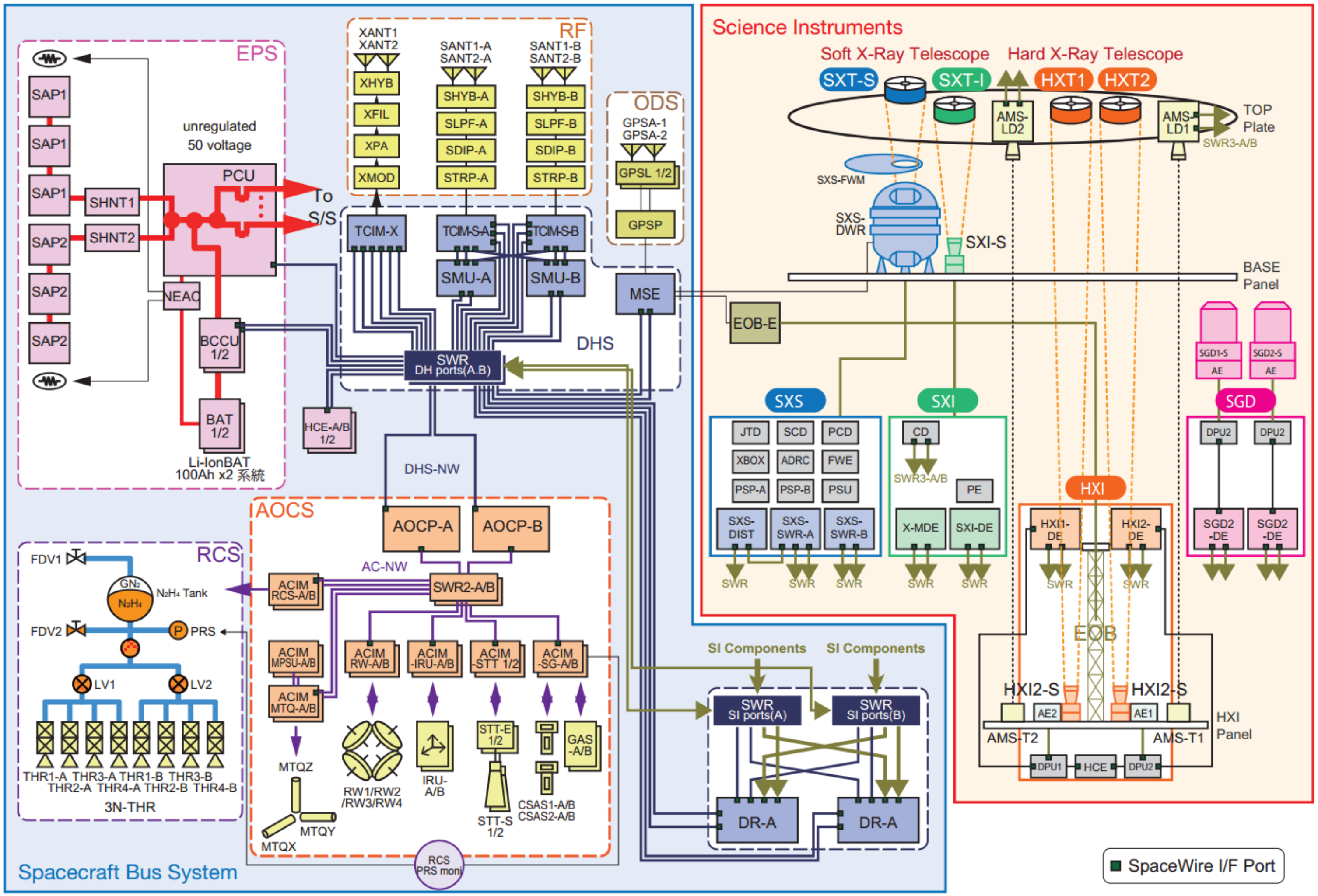
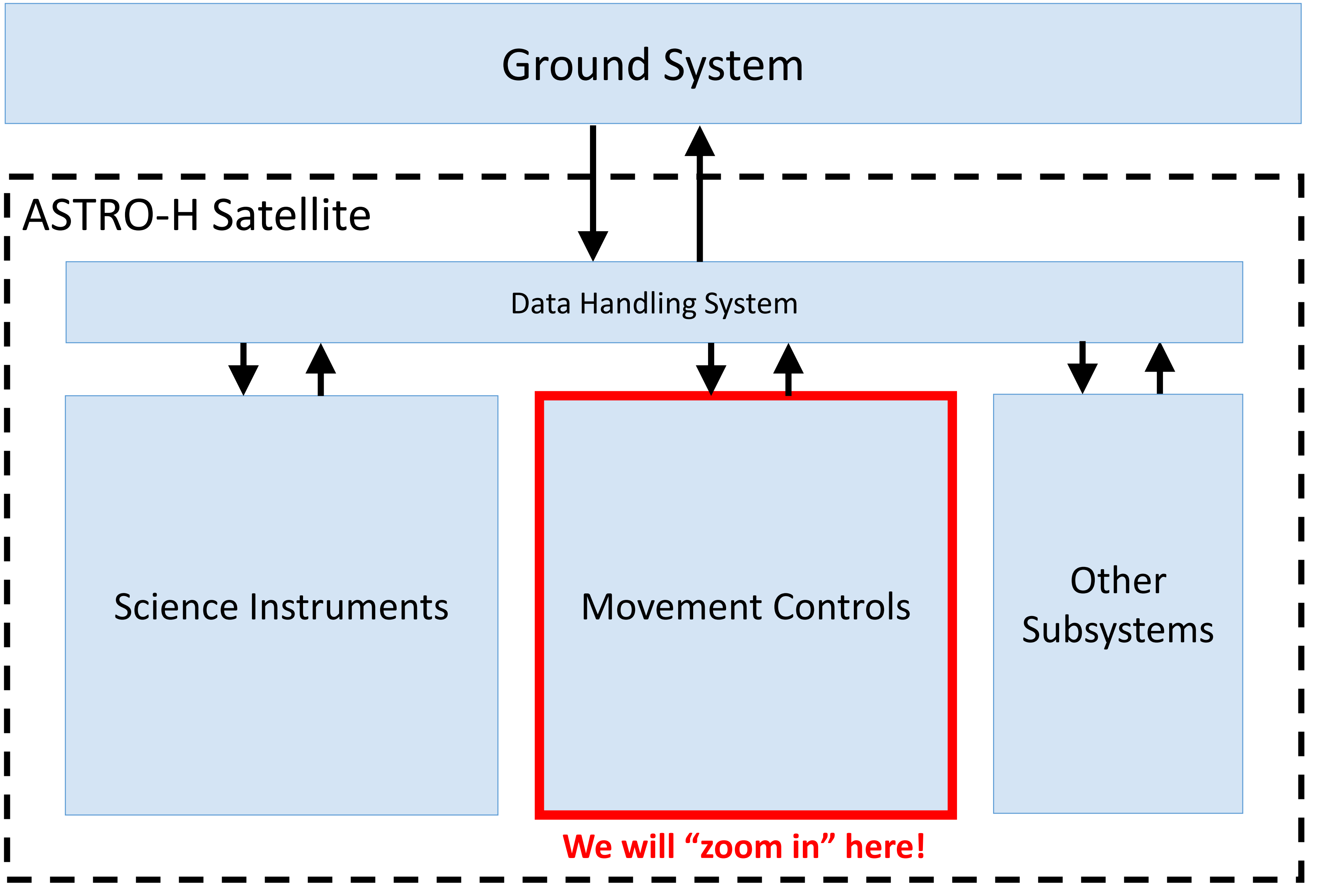


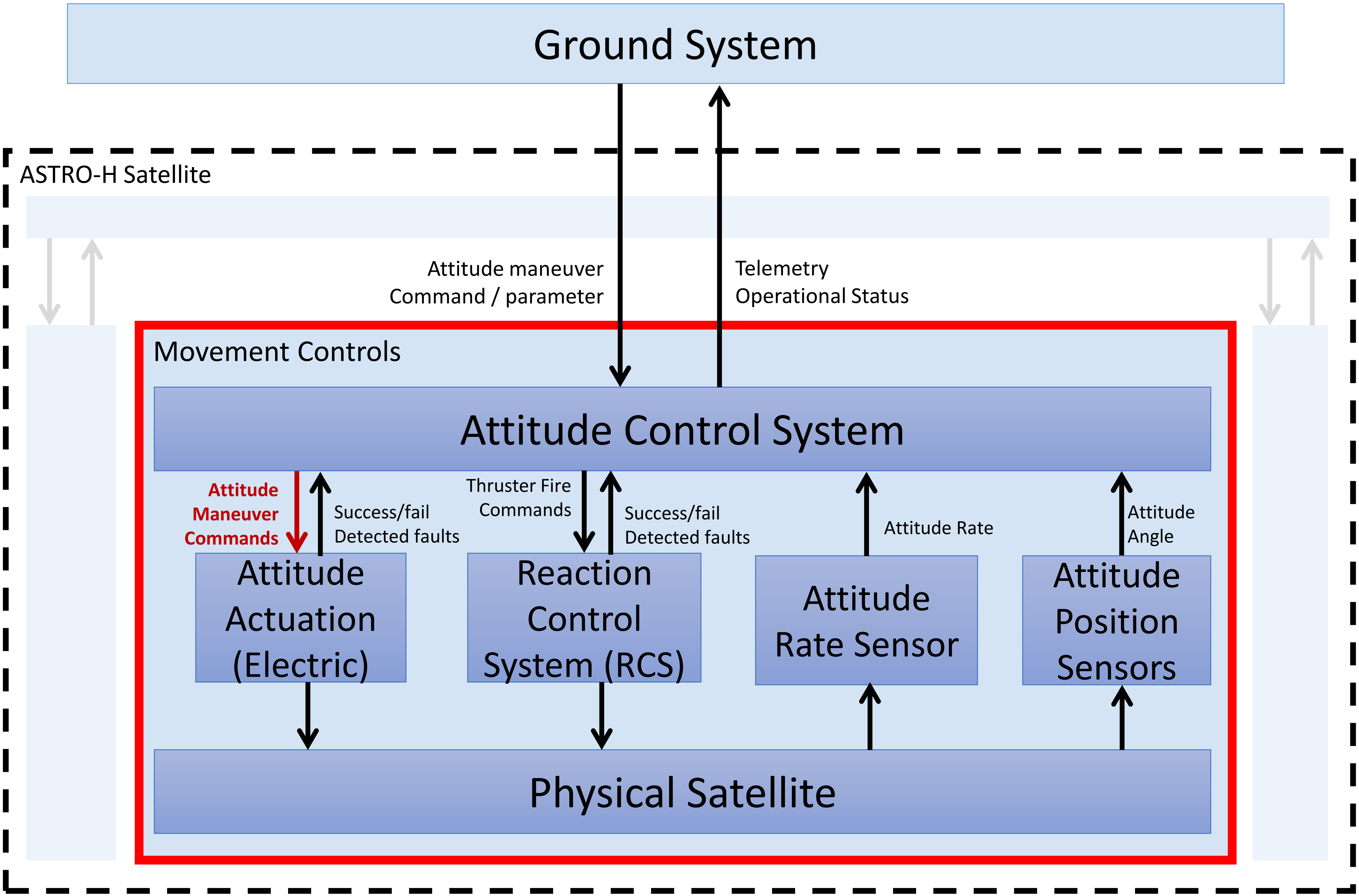
Figure 3.9: System block diagram. A is the primary and B is the redundant system.

Don't start by trying to include every detail immediately!
Start with a high-level control structure, then refine

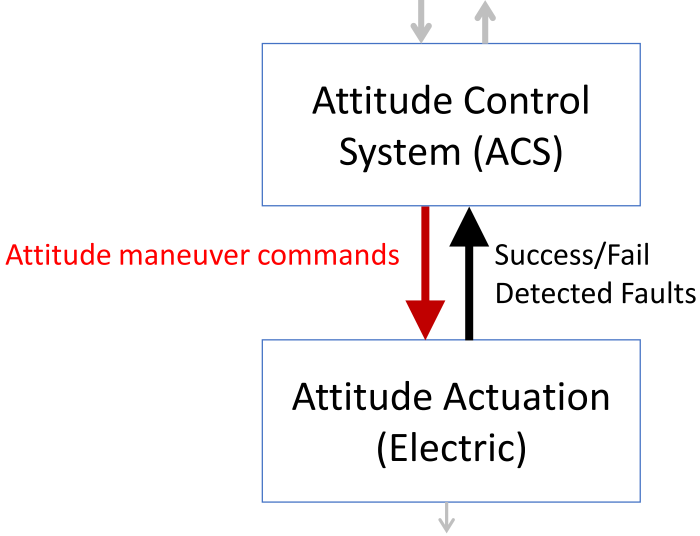
High-level control structure



Refined control structure



Identify Unsafe Control Actions



	Not Providing causes hazard	Providing causes hazard	Too early, too late, out of order	Stopped too soon, Applied too long
Attitude maneuver commands	UCA-1: ACS does not provide attitude maneuver commands when ASTRO-H is rotating [H-1,H-2]	UCA-2: **ACS provides attitude maneuver commands when maneuver direction is same as satellite rotation** [H-1,H-2] UCA-3: ACS provides attitude maneuver commands when ASTRO-H is not rotating [H-1,H-2] UCA-4: ACS provides attitude maneuver commands with insufficient strength to slow ASTRO-H quickly [H-1,H-2]	UCA-5: ACS provides attitude maneuver commands too late after satellite attitude rate is high [H-1,H-2]	UCA-6: ACS stops providing attitude maneuver commands too soon before satellite stops rotating [H-1,H-2] UCA-7: ACS continues providing attitude maneuver commands too long after satellite stopped rotating [H-1,H-2]

*All conditions can be defined in precise engineering terms. For example, “Is Rotating” means the rotational velocity is sufficient to require dumping the attitude rate

Additional Guidance for UCAs

Be sure to consider 3 types of conditions:

- Conditions in which the control action is never safe
- Conditions in which an insufficient or excessive control action is unsafe
- Conditions in which the direction of the control action is unsafe

	Not Providing causes hazard	Providing causes hazard	Too early/late, order	Stopped too soon, Applied too long
Attitude maneuver commands	UCA-1: ACS does not provide attitude maneuver commands when ASTRO-H is rotating [H-1,H-2]	UCA-2: ACS provides attitude maneuver commands when maneuver direction is same as satellite rotation [H-1,H-2] **(wrong direction)** UCA-3: ACS provides attitude maneuver commands when ASTRO-H is not rotating [H-1,H-2] **(never safe)** UCA-4: ACS provides attitude maneuver commands with insufficient strength to slow ASTRO-H quickly [H-1,H-2] **(insufficient/excessive)**	UCA-5: ACS provides attitude maneuver commands too late after satellite attitude rate is high [H-1,H-2]	UCA-6: ACS stops providing attitude maneuver commands too soon before satellite stops rotating [H-1,H-2] UCA-7: ACS continues providing attitude maneuver commands too long after satellite stopped rotating [H-1,H-2]

*All conditions can be defined in precise engineering terms. For example, "Is Rotating" means the rotational velocity is sufficient to require dumping the attitude rate

Derive Safety Constraints

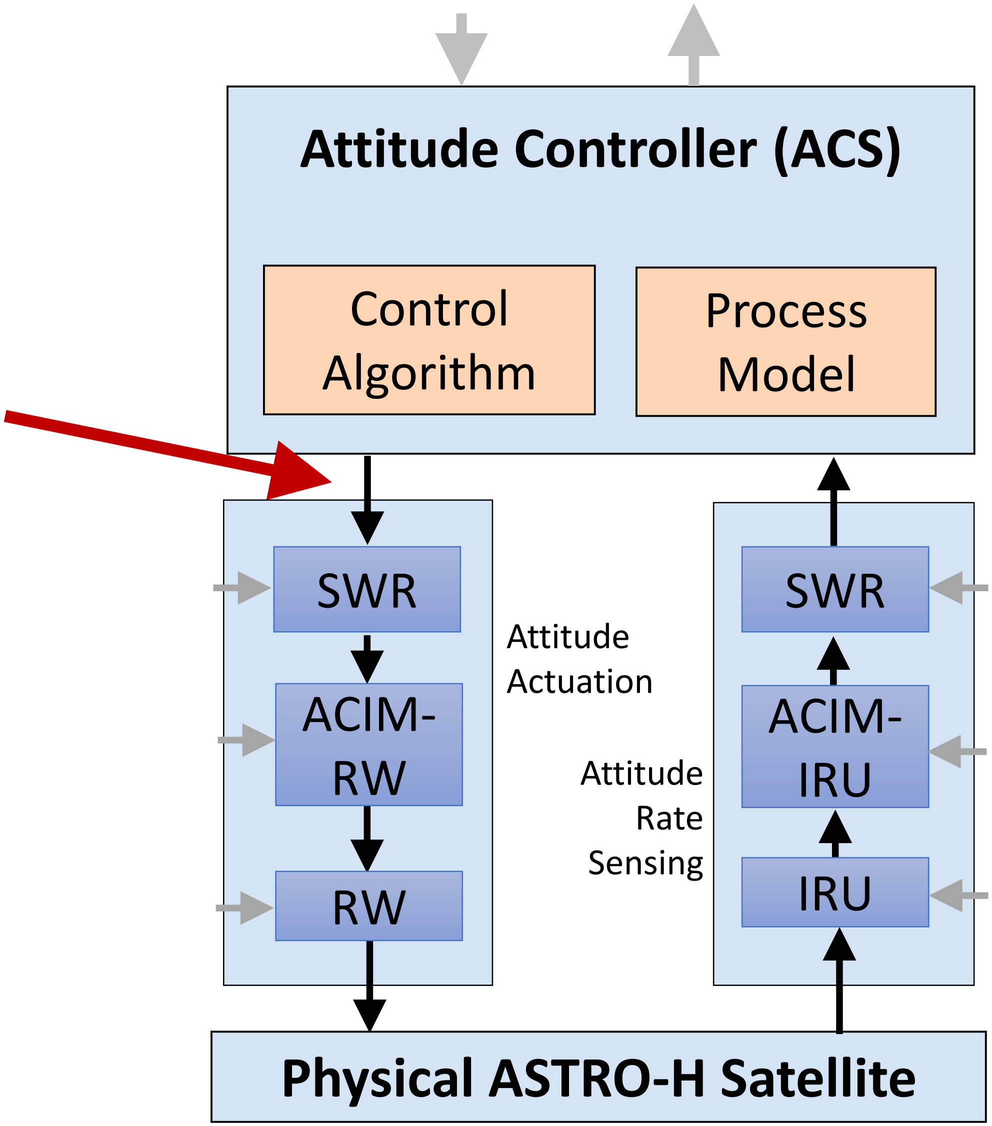
Unsafe Control Action (UCA)	Safety Constraint (SC)
UCA-1: ACS does not provide attitude maneuver commands when ASTRO-H is rotating [H-1,H-2]	SC-1: ACS must provide attitude maneuver commands when ASTRO-H is rotating [H-1,H-2]
UCA-2: ACS provides attitude maneuver commands when maneuver direction is same as satellite rotation [H-1,H-2]	SC-2: ACS must not provide attitude maneuver commands in the same direction as rotation [H-1,H-2]
UCA-3: ACS provides attitude maneuver commands when ASTRO-H is not rotating [H-1,H-2]	SC-3: ACS must not provide attitude maneuver commands when ASTRO-H is not rotating [H-1,H-2]
UCA-4: ACS provides attitude maneuver commands with insufficient strength to slow ASTRO-H quickly [H-1,H-2]	SC-4: ACS must provide attitude maneuver commands that are sufficient to slow ASTRO-H quickly [H-1,H-2]
UCA-5: ACS provides attitude maneuver commands too late after ASTRO-H has rotated too far [H-1,H-2]	SC-5: ACS must not provide attitude maneuver commands too late after ASTRO-H has rotated too far [H-1,H-2]
UCA-6: ACS provides attitude maneuver commands too early to achieve desired attitude [H-1,H-2]	SC-6: ACS must not provide attitude maneuver commands too early to achieve desired attitude [H-1,H-2]
UCA-7: ACS stops providing attitude commands too soon before attitude has stabilized [H-1,H-2]	SC-7: ACS must not stop providing attitude commands too soon before attitude has stabilized [H-1,H-2]
UCA-8: ACS continues providing attitude maneuver commands too long after attitude has stabilized [H-1,H-2]	SC-8: ACS must not continue providing attitude maneuver commands too long after attitude has stabilized [H-1,H-2]

*All conditions must be defined in precise engineering terms. For example, “Is Rotating” means the rotational velocity is sufficient to require dumping the attitude rate

Identifying Scenarios

UCA result:

UCA-2: ACS provides attitude maneuver commands in the same direction as rotation

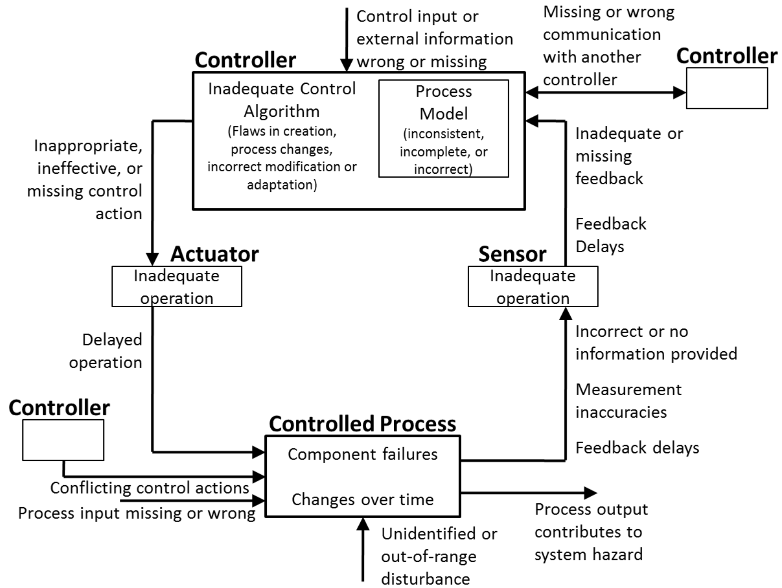


Identify scenarios
- But how?

Example of "checklist" approach

Causal factors:

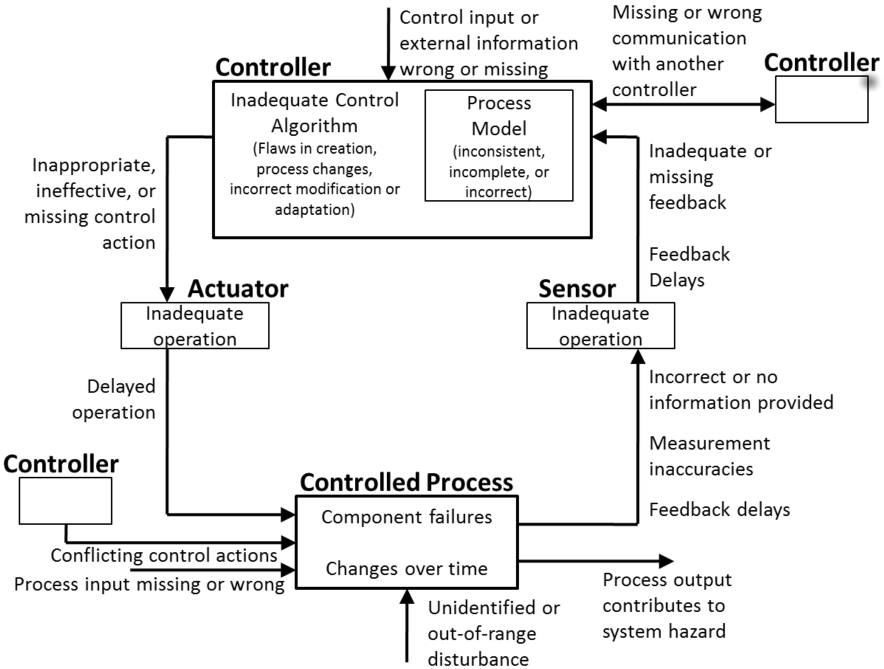
- Level sensor failure
- Level feedback not provided
- Incorrect low level feedback
- Incorrect isolation signal
- Pressure too low
- Pressure feedback delayed
- Pressure feedback missing
- Incorrect pressure feedback
- Incorrect signal of initiation
- Startup/shutdown not recognized
- Etc.



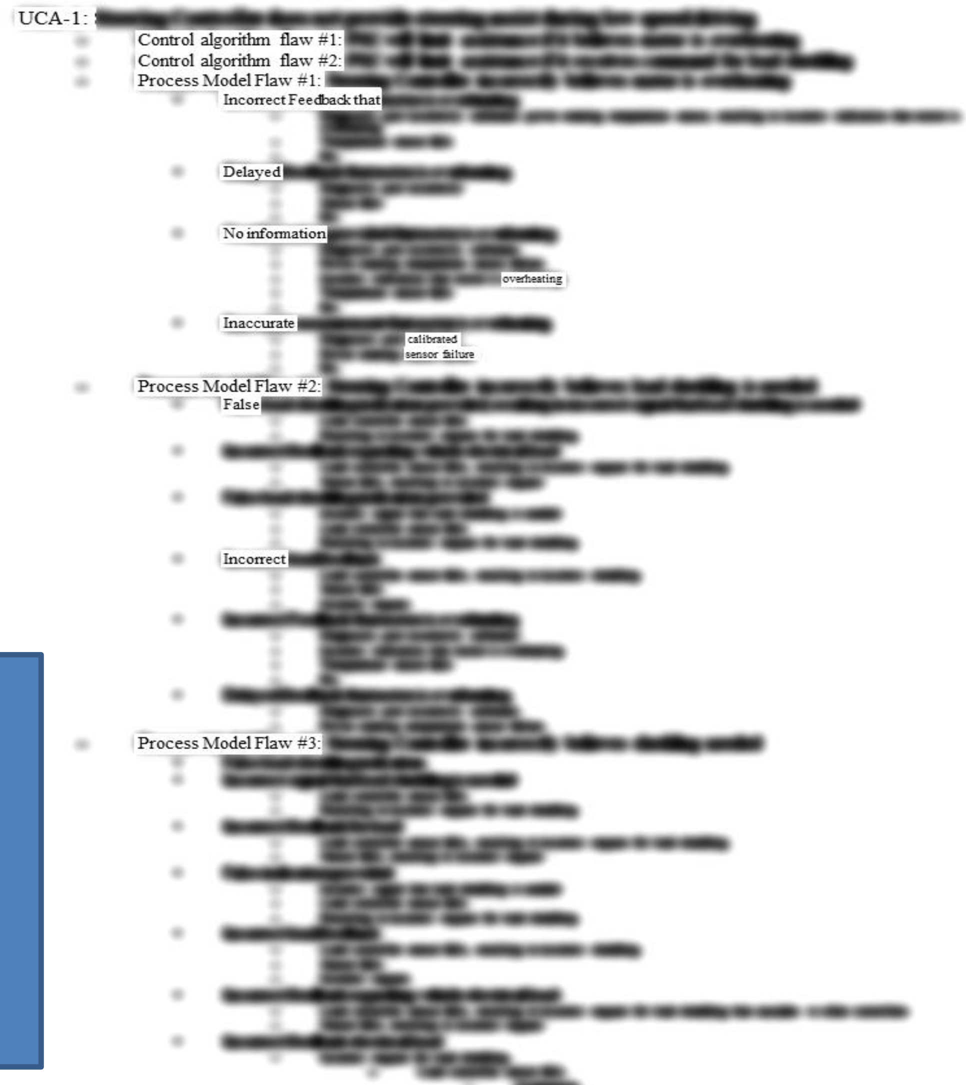
Labels above used as checklist

Bad approach!
Can provide misleading results
Focuses on single-point issues
Can miss interactions, context
May obscure complex (but critical) scenarios

Option 1: Work backward, keep asking why



Example analysis of single UCA:
(design details obscured)


UCA-1:
- Control algorithm flaw #1:
- Control algorithm flaw #2:
- Process Model Flaw #1:
  - Incorrect Feedback that
  - Delayed
  - No information
    - overheating
  - Inaccurate
    - calibrated
    - sensor failure
- Process Model Flaw #2:
  - False
  - Incorrect
- Process Model Flaw #3:

Can be done, but…
- Grows very large very quickly!
- Time, Effort
- Very detailed/specific
  - Limits how early it can be used

Option 1 using graphical tree format

Steering assist is not provided during low speed driving
(Unsafe Control Action)

Steering controller does not send steering assist command during low speeds
(Inappropriate Decision)

PSC is programmed to prevent steering assist commands
(Inadequate Control Algorithm)

PSC incorrectly believes motor is overheating
(Process Model Flaw)

Incorrect temperature feedback
(Incorrect Feedback)

Sensor re-calibrated over DLC
(Incorrect Input)

#2: Other controller sends false steering assist commands
(Conflicting Control Action)

Same issue: grows very quickly for complex systems

Option 2: Scenario Building

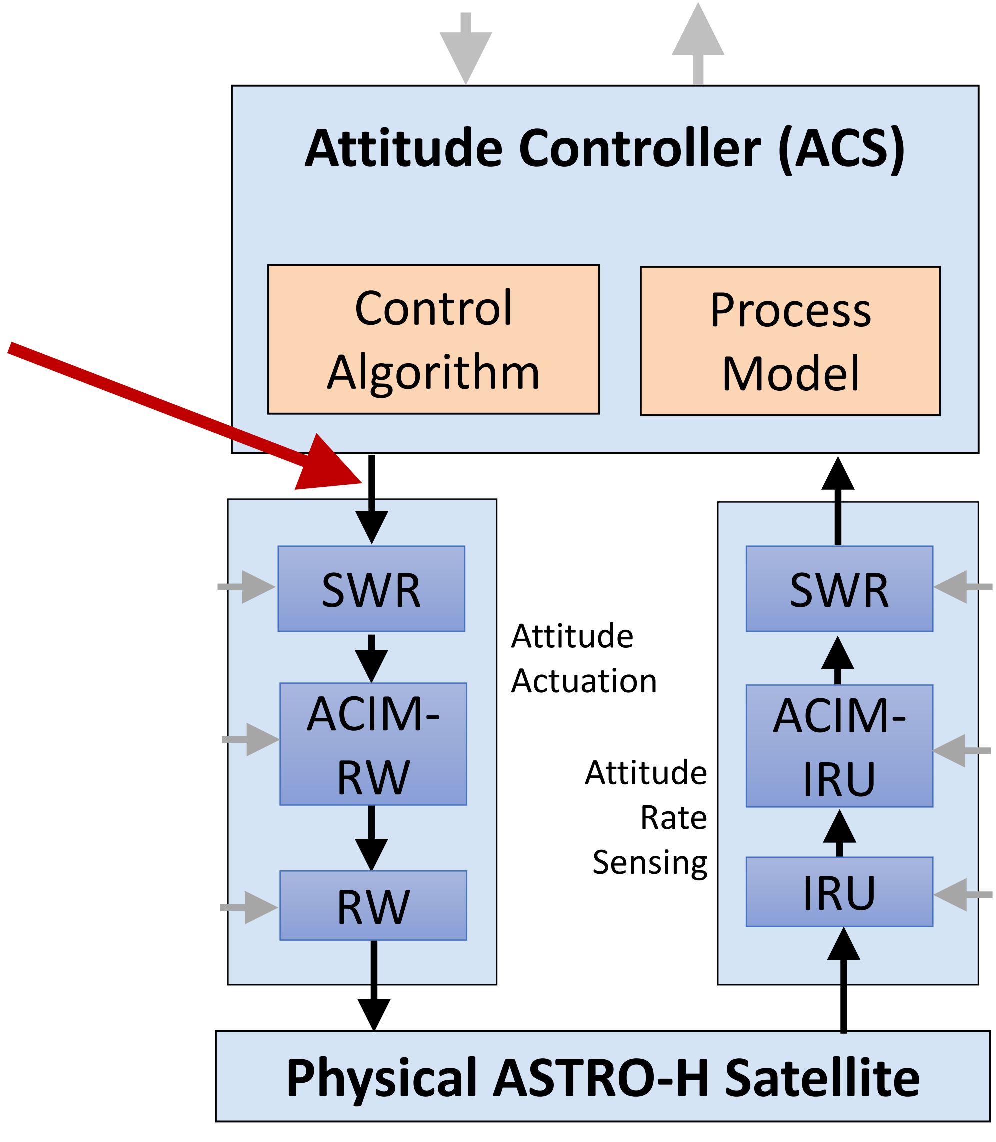
UCA result:

UCA-2: ACS provides attitude maneuver commands in the same direction as rotation

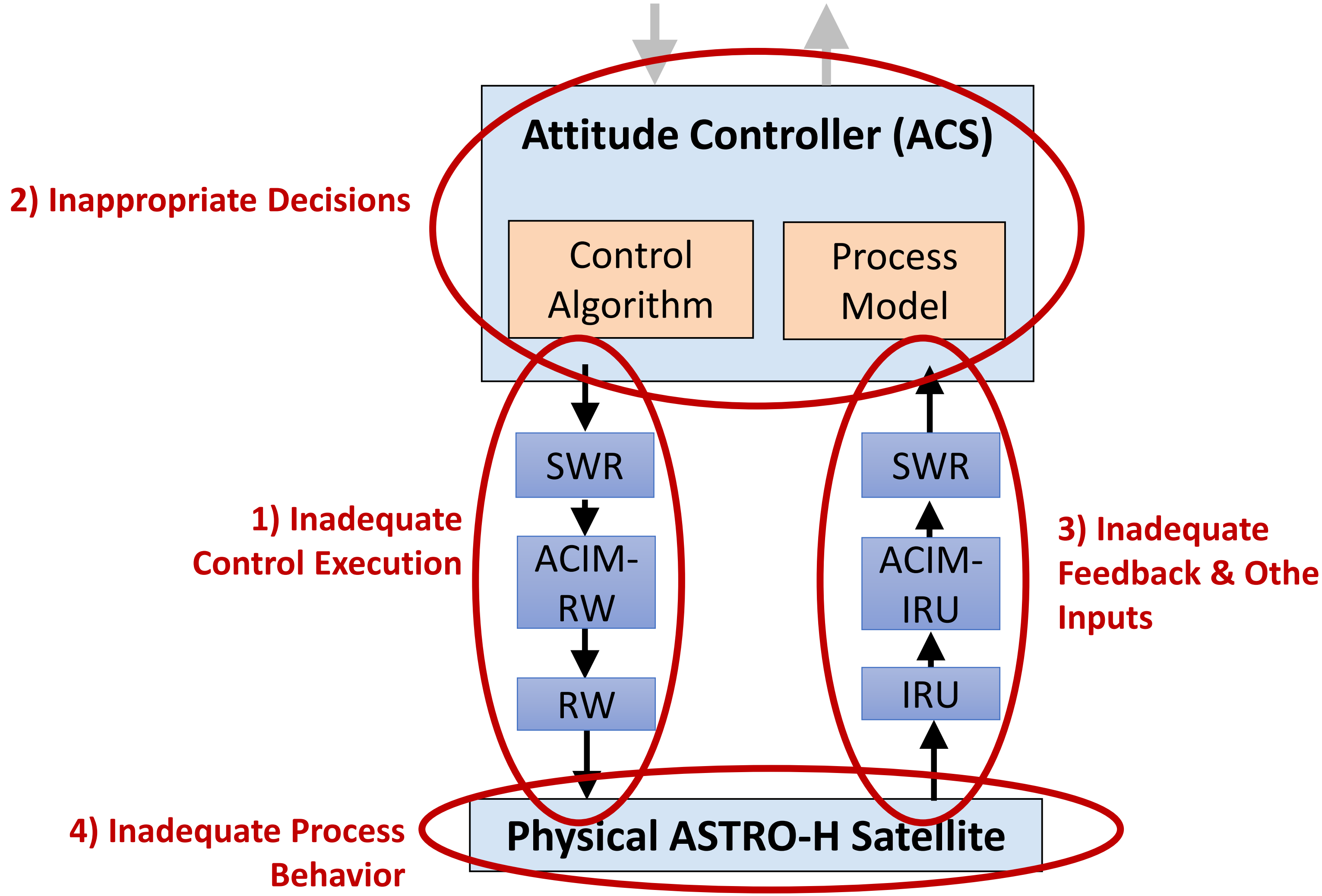
Identify scenarios

- But how?

<u>Start with high-level abstract scenarios and refine them!</u>



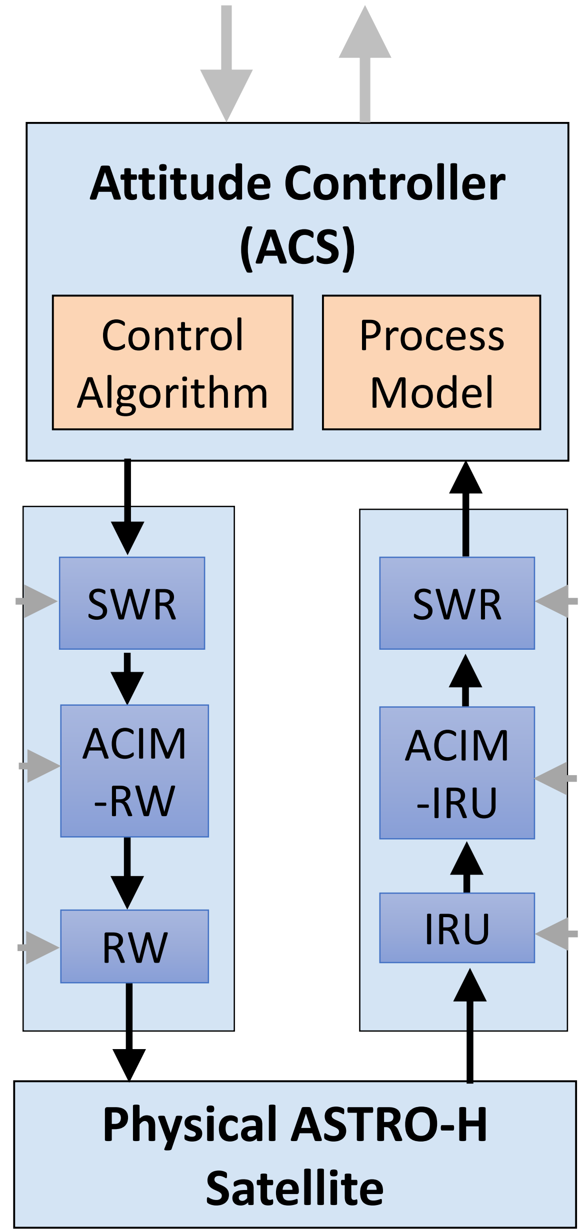
Solution: Start with high-level abstract scenarios and refine them!



We can provide specific guidance for each type of scenario

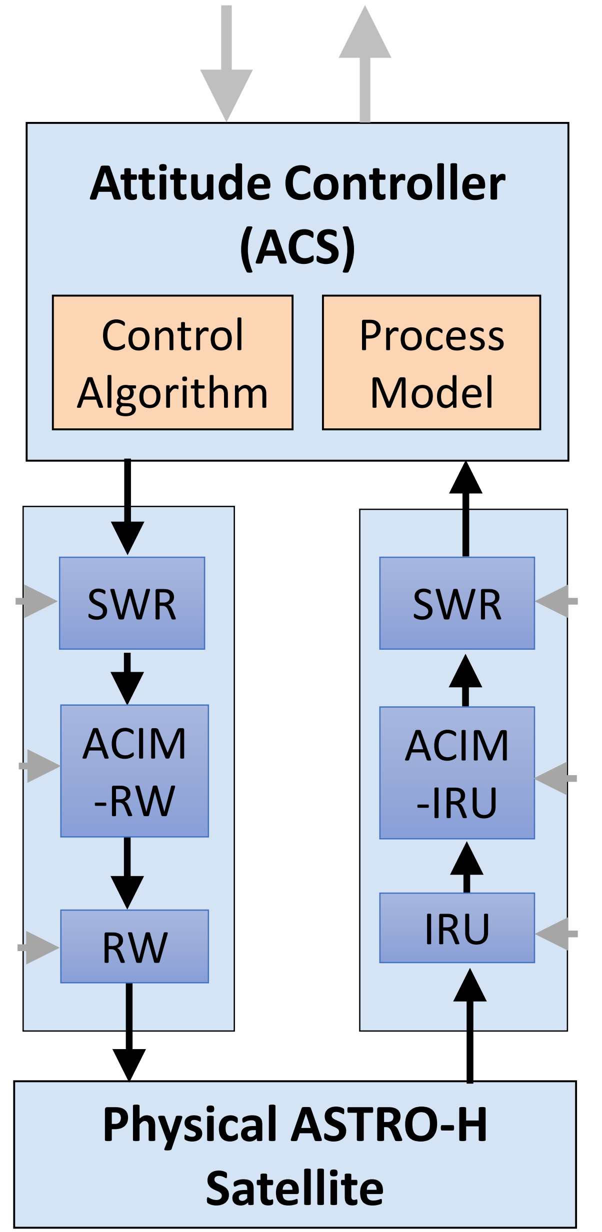
New process for Step 2:

1. Define small number of high-level scenarios
   – Start with few broad, abstract scenarios
   – Consider each scenario type
   – Easy to review, show coverage, completeness, etc.
2. Identify potential solutions
   – Requirements
   – Modify control actions
   – Modify types of feedback
   – Modify responsibilities
   – Etc.
3. Refine high-level scenarios (if solutions not found)
   – Include more design detail
   – Can be done in parallel with development



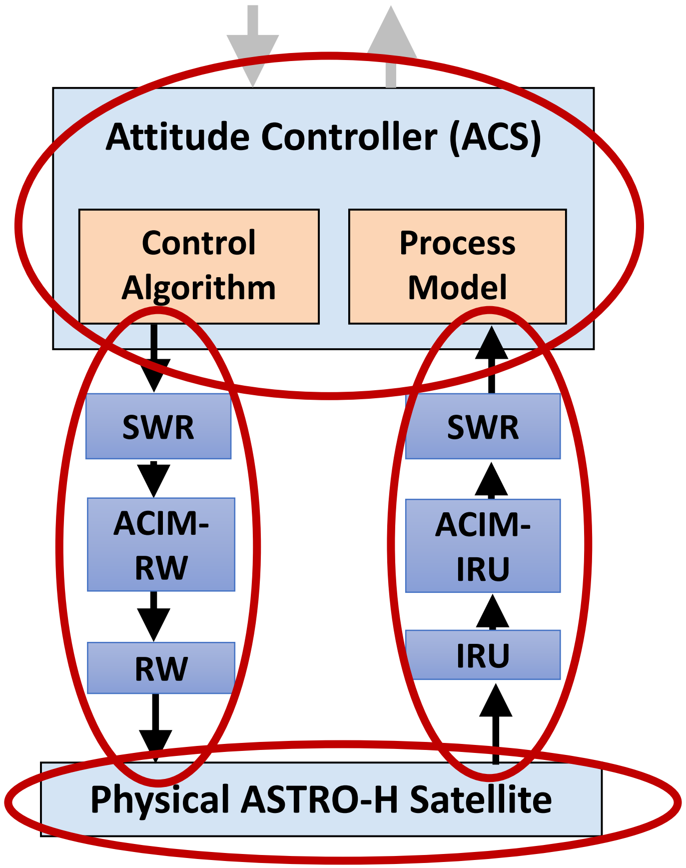
Top-down approach to scenario building

1. Define small number of high-level scenarios
   – Start with few broad, abstract scenarios
   – Consider each scenario type
   – Easy to review, show coverage, completeness, etc.
2. Identify potential solutions
   – Requirements
   – Modify control actions
   – Modify types of feedback
   – Modify responsibilities
   – Etc.
3. Refine high-level scenarios (if solutions not found)
   – Include more design detail
   – Can be done in parallel with development



1) Identify high-level scenarios

2) Inappropriate Decisions
- ACS receives correct vehicle rotation feedback
- ACS provides attitude maneuver commands in the same direction as rotation (UCA-2)



1) Inadequate Control Execution
- ACS provides attitude maneuver commands
- RW does not respond accordingly

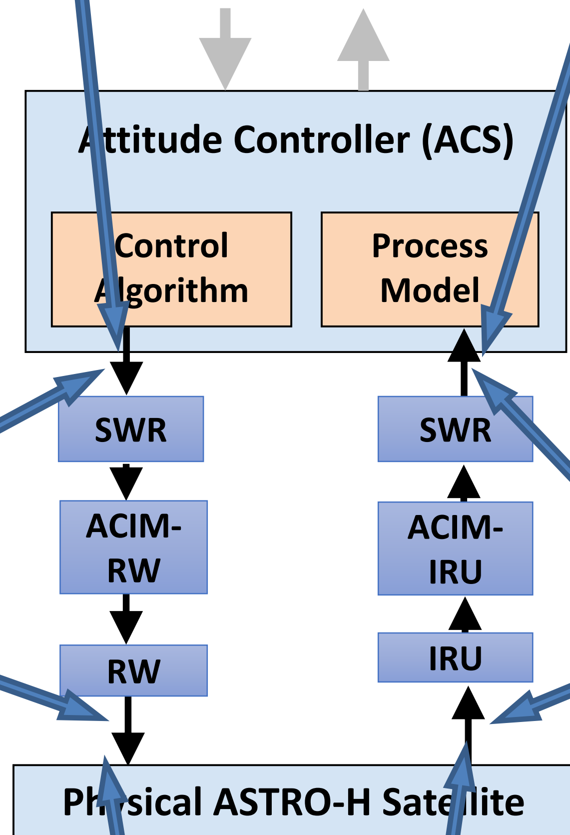
3) Inadequate Feedback & Other Inputs
- ACS receives incorrect feedback that vehicle is rotating
- Vehicle is not rotating

4) Inadequate Process Behavior
- RW momentum changes
- Vehicle attitude does not change accordingly

1) Identify high-level scenarios

2) Inappropriate Decisions

- ACS receives correct vehicle rotation feedback
- ACS provides attitude maneuver commands in the same direction as rotation (UCA-2)



1) Inadequate Control Execution

- ACS provides attitude maneuver commands
- RW does not respond accordingly

3) Inadequate Feedback & Other Inputs

- ACS receives incorrect feedback that vehicle is rotating
- Vehicle is not rotating

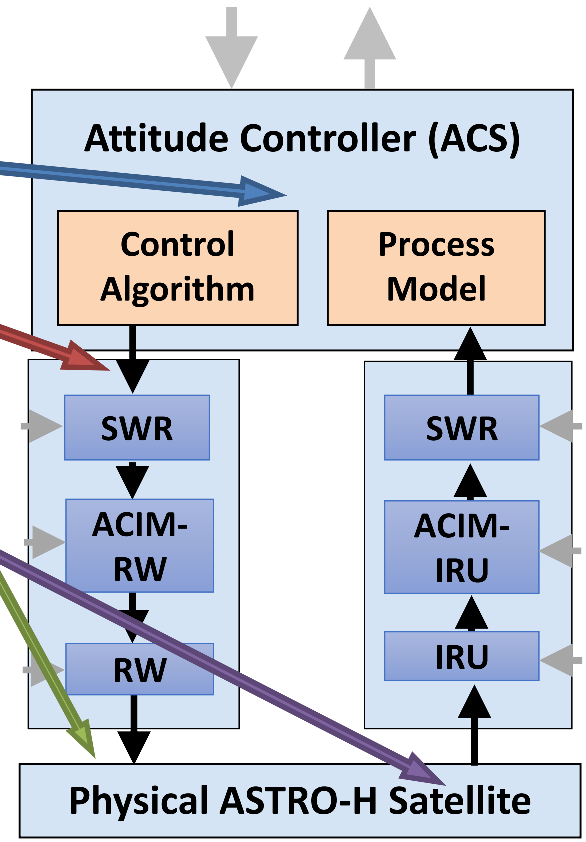
4) Inadequate Process Behavior

- RW momentum changes
- Vehicle attitude does not change accordingly

Show coverage!

Top-down approach to scenario building

UCA:
ACS provides attitude maneuver commands to RW in the same direction as rotation (UCA-4)

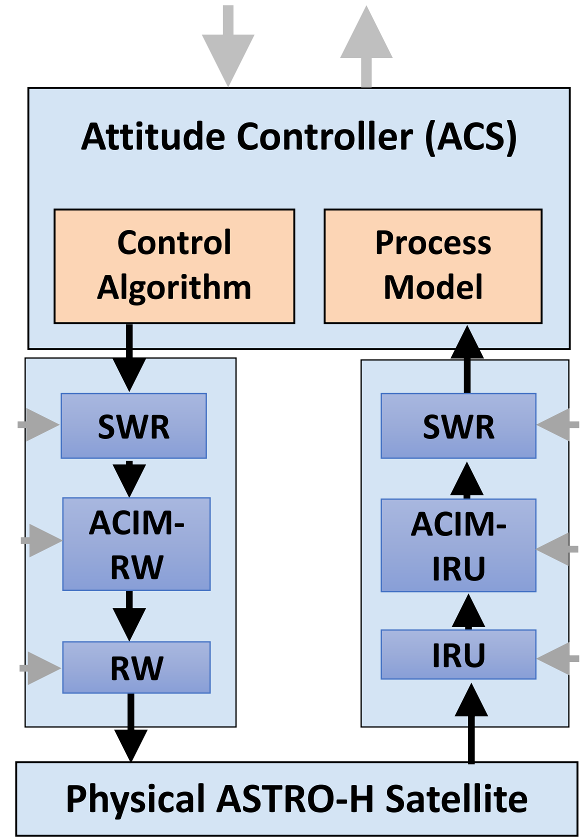


High-level Basic Scenarios

1. Commands not followed / executed
   - ACS provides attitude maneuver commands
   - RW does not respond accordingly
2. Inappropriate Decisions
   - ACS receives correct vehicle rotation feedback
   - ACS provides attitude maneuver commands in the same direction as rotation
3. Inadequate Feedback & Other Inputs
   - ACS receives incorrect feedback that vehicle is rotating
   - Vehicle is not rotating
4. Inadequate Process Behavior
   - RW momentum changes
   - Vehicle attitude does not change accordingly

1) Identify high-level scenarios

UCA:
ACS provides attitude maneuver commands to RW in the same direction as rotation (UCA-2)



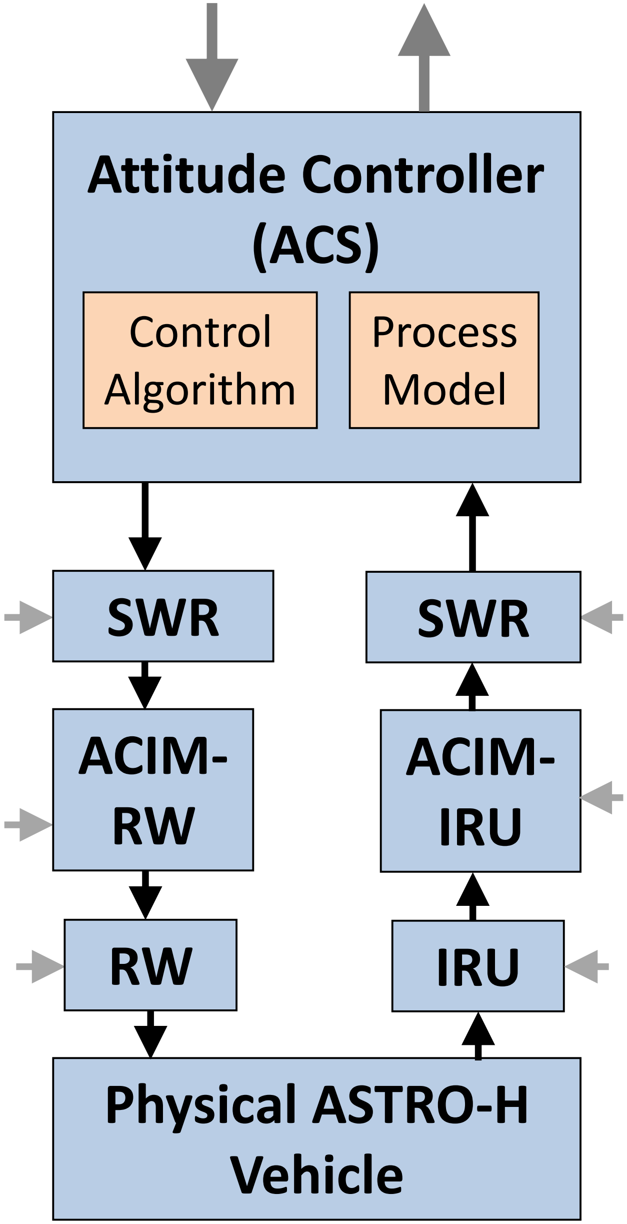
High-level Basic Scenarios

1. Commands not followed / executed
   - ACS provides attitude maneuver commands
   - RW does not respond accordingly
2. Inappropriate Decisions
   - ACS receives correct vehicle rotation feedback
   - ACS provides attitude maneuver commands in the same direction as rotation
3. Inadequate Feedback & Other Inputs
   - ACS receives incorrect feedback that vehicle is rotating
   - Vehicle is not rotating
4. Inadequate Process Behavior
   - RW momentum changes
   - Vehicle attitude does not change accordingly

All of these scenarios can be generated automatically!!

Top-down approach to scenario building

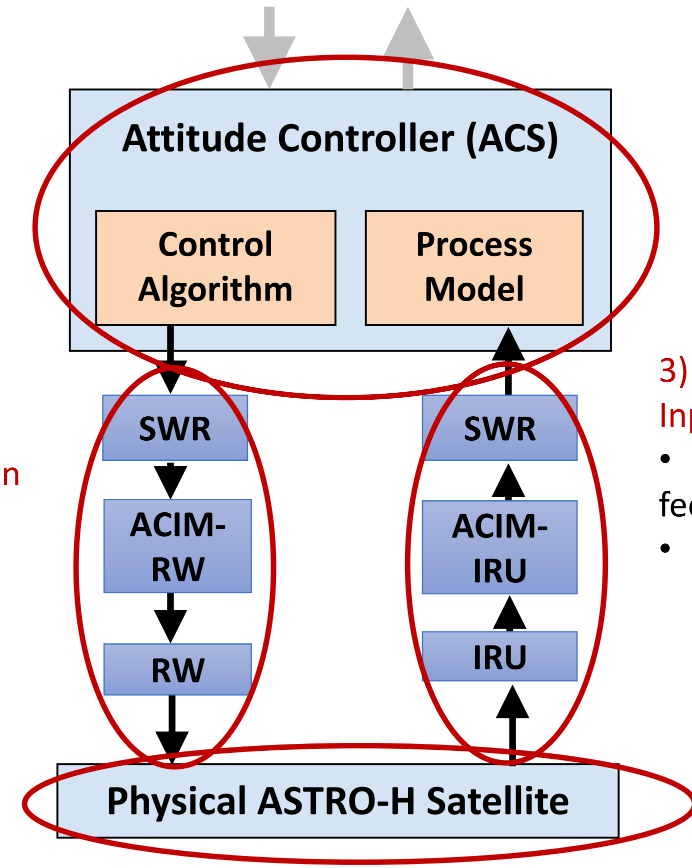
1. Define small number of high-level scenarios
   - Start with few broad, abstract scenarios
   - Consider each scenario type
   - Easy to review, show coverage, completeness, etc.
2. Identify potential solutions (if possible)
   - Requirements
   - Modify control actions
   - Modify types of feedback
   - Modify responsibilities
   - Etc.
3. Refine high-level scenarios (if solutions not found)
   - Include more design detail
   - Can be done in parallel with development



2) Identify potential solutions

2) Inappropriate Decisions

- ACS receives correct vehicle rotation feedback
- ACS provides attitude maneuver commands in the same direction as rotation (UCA-2)



1) Inadequate Control Execution

- ACS provides attitude maneuver commands
- RW does not respond accordingly

3) Inadequate Feedback & Other Inputs

- ACS receives incorrect IRU feedback that vehicle is rotating
- Vehicle is not rotating

Potential solution: Make ACS detect when IRU feedback is incorrect.

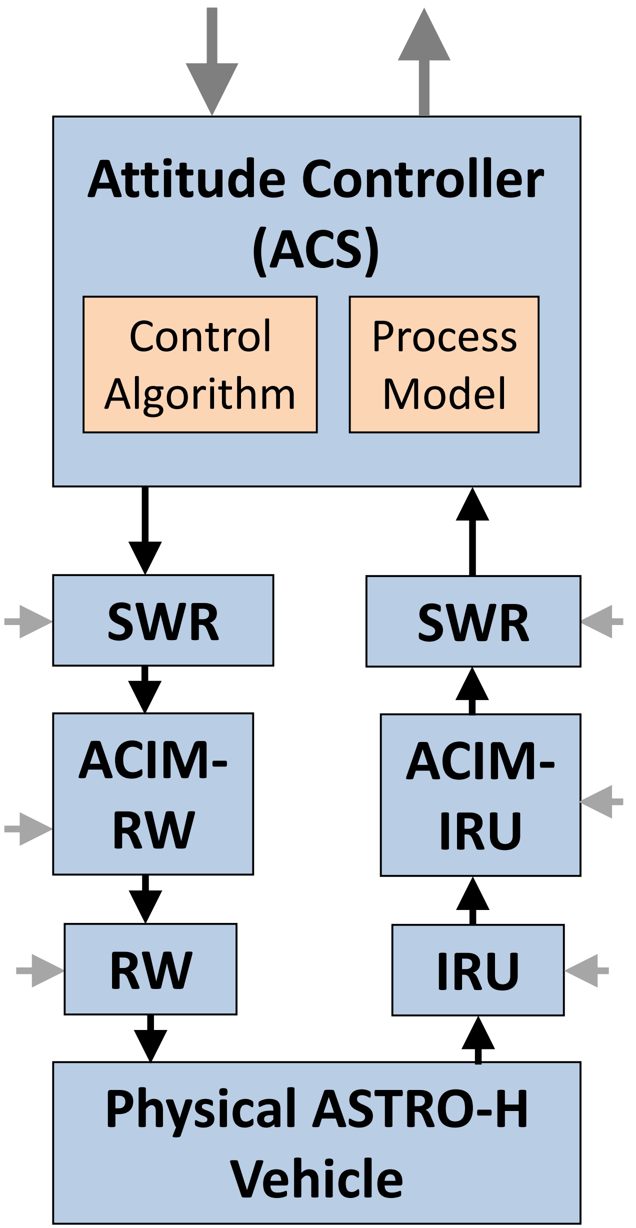
- Must validate IRU data by comparing to other sensors
- If Star Tracker is unavailable, use sun sensor.
- ACS must not use IRU data that is known to be incorrect
- Etc.

4) Inadequate Process Behavior

- RW momentum changes
- Vehicle attitude does not change accordingly

Top-down approach to scenario building

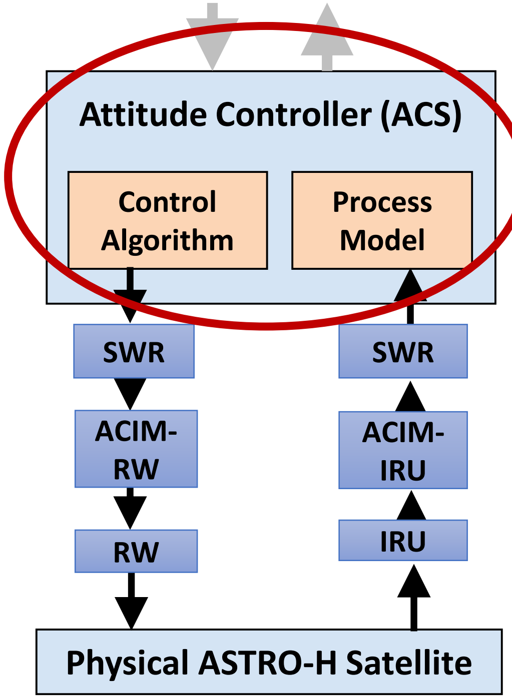
1. Define small number of high-level scenarios
   – Start with few broad, abstract scenarios
   – Consider each scenario type
   – Easy to review, show coverage, completeness, etc.
2. Identify potential solutions (if possible)
   – Requirements
   – Modify control actions
   – Modify types of feedback
   – Modify responsibilities
   – Etc.
3. Refine high-level scenarios (if solutions not found)
   – Include more design detail
   – Can be done in parallel with development



3) Refine high-level scenarios

Type 2 Basic Scenario

- ACS receives correct vehicle rotation feedback from IRU
- ACS provides attitude maneuver commands in wrong direction (UCA-2)



Type 2 Refined Scenarios

Refined Scenario #2.1:

- ACS receives correct vehicle rotation feedback from IRU
- ACS applies an incorrect bias estimate to IRU data
- ACS provides attitude maneuver cmds in the same direction as rotation (UCA-2)

Refined Scenario #2.2:

- ACS receives correct vehicle rotation feedback from IRU
- ACS switches to safe-hold mode and ignores data from IRU
- ACS provides attitude maneuver cmds in the same direction as rotation (UCA-2)

Refined Scenario #2.3:

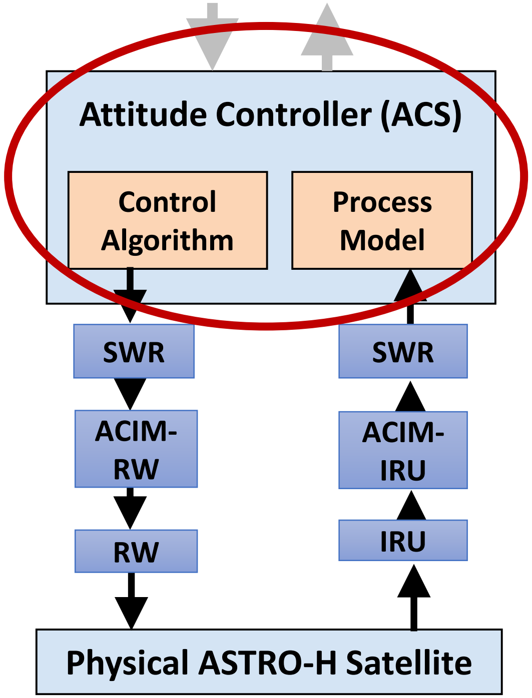
- ACS receives correct vehicle rotation feedback from IRU
- Incorrect control parameters are uploaded to ACS, inverting attitude maneuver calculations
- ACS provides attitude maneuver cmds in the same direction as rotation (UCA-2)

Goal: identify how the basic scenarios might occur

3) Refine high-level scenarios

Type 2 Basic Scenario

- ACS receives correct vehicle rotation feedback from IRU
- ACS provides attitude maneuver commands in wrong direction (UCA-2)



Type 2 Refined Scenarios

Refined Scenario #2.1:

- ACS receives correct vehicle rotation feedback from IRU
- ACS applies an incorrect bias estimate to IRU data
- ACS provides attitude maneuver cmds in the same direction as rotation (UCA-2)

Refined Scenario #2.2:

- ACS receives correct vehicle rotation feedback from IRU
- ACS switches to safe-hold mode and ignores data from IRU
- ACS provides attitude maneuver cmds in the same direction as rotation (UCA-2)

Refined Scenario #2.3:

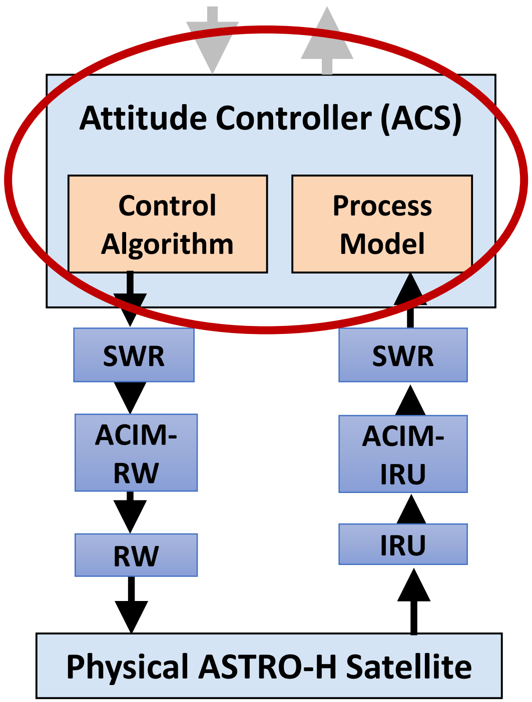
- ACS receives correct vehicle rotation feedback from IRU
- Incorrect control parameters are uploaded to ACS, inverting attitude maneuver calculations
- ACS provides attitude maneuver cmds in the same direction as rotation (UCA-2)

Are these safety or security issues? It’s both!

3) Refine high-level scenarios

Type 2 Basic Scenario

- ACS receives correct vehicle rotation feedback from IRU
- ACS provides attitude maneuver commands in wrong direction (UCA-2)



Type 2 Refined Scenarios

Refined Scenario #2.1:

- ACS receives correct vehicle rotation feedback from IRU
- ACS applies an incorrect bias estimate to IRU data
- ACS provides attitude maneuver cmds in the same direction as rotation (UCA-2)

Refined Scenario #2.2:

- ACS receives correct vehicle rotation feedback from IRU
- ACS switches to safe-hold mode and ignores data from IRU
- ACS provides attitude maneuver cmds in the same direction as rotation (UCA-2)

Refined Scenario #2.3:

- ACS receives correct vehicle rotation feedback from IRU
- Incorrect control parameters are uploaded to ACS, inverting attitude maneuver calculations
- ACS provides attitude maneuver cmds in the same direction as rotation (UCA-2)

This is more than just software verification! This is analyzing software design decisions, requirements, and overall safety and security!

3) Refine high-level scenarios

Example of Type 2 Basic Scenario:

- ACS provides attitude maneuver commands in same direction as vehicle rotation (UCA-2)
- ACS receives correct vehicle rotation feedback

To refine Type 2 scenarios:

- Identify the conditions being described
  - "vehicle rotation"
- Identify the process model variable corresponding to each condition
  - Rotational velocity (x,y,z)
- Case A: Process model is incorrect. Why? Consider:
  - Process model not updated
  - Process model updated incorrectly
  - Default values are incorrect
- Case B: Control Algorithm is incorrect. Why? Consider:
  - Controller ignores process model
  - Controller uses process model, but does so incorrectly
  - Controller does not ignore irrelevant or incorrect process models

- Updates (feedback) received but interpreted incorrectly
  - Information is misidentified as something else
  - Computer not on or doing something else when received (info not properly cached)
  - Error in updating routine
- Controller assumes previous control actions successful and process has changed as expected
- Controller received conflicting information about same process model, resolves the conflict incorrectly

Detailed guidance is provided for each scenario type!

3) Refine high-level scenarios

Example of Type 2 Basic Scenario:

- ACS provides attitude maneuver commands in same direction as vehicle rotation (UCA-2)
- ACS receives correct vehicle rotation feedback

To refine Type 2 scenarios:

- Identify the conditions being described
  - "vehicle rotation"
- Identify the process model variable corresponding to each condition
  - Rotational velocity (x,y,z)
- Case A: Process model is incorrect. Why? Consider:
  - Process model not updated
  - Process model updated incorrectly
  - Default values are incorrect
- Case B: Control Algorithm is incorrect. Why? Consider:
  - Controller ignores process model
  - Controller uses process model, but does so incorrectly
  - Controller does not ignore irrelevant or incorrect process models

- Attacker updates the process model directly
- Attacker provides conflicting information to trigger process model update
- Attacker interferes with previous commands (process model is automatically updated assuming it worked, but doesn't match actual controlled process)
- Attacker causes controller to misinterpret feedback (e.g. by triggering mode change, providing new updating routine, etc.)
- Attacker causes controller to do something else when feedback is received (info not properly cached), Updates (feedback) received but interpreted incorrectly

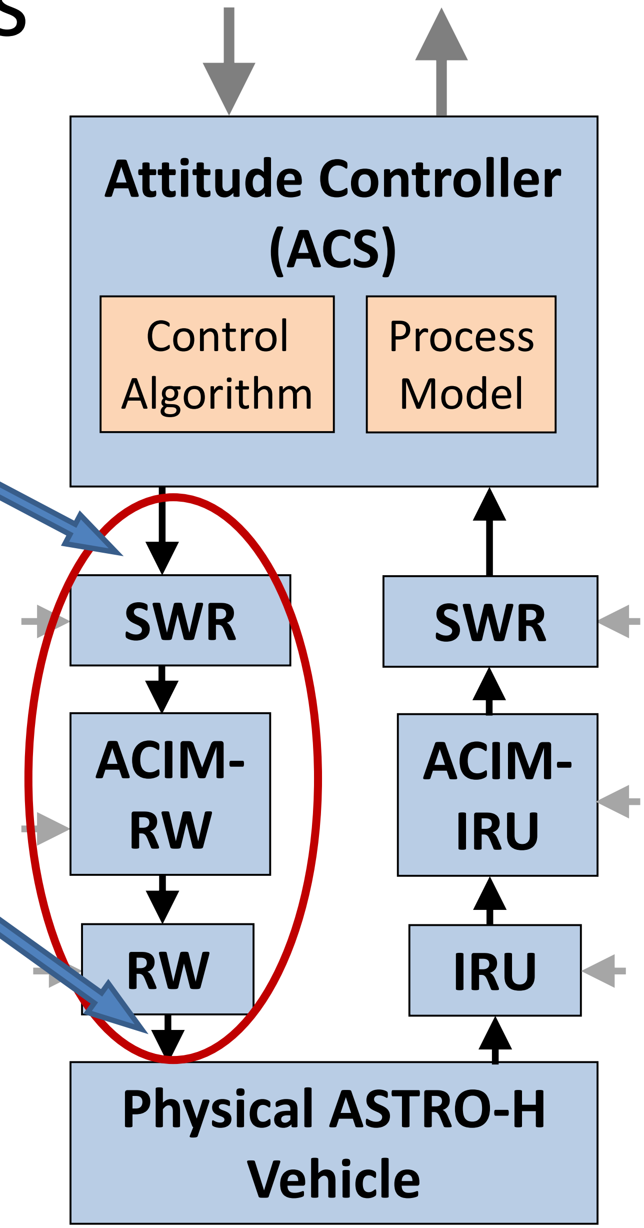
Security-specific guidance provided too!

3) Refine high-level scenarios

ACS provides attitude maneuver commands when vehicle is not rotating (UCA-2)

Basic Scenario #1:

- ACS does not provide attitude maneuver commands
- RW momentum changes



To refine Type 1 scenarios:

Refined Scenario #3.1:

- ACS does not provide attitude maneuver commands
- Previous attitude maneuver cmd buffered, released late
- RW momentum changes

Refined Scenario #3.2:

- ACS does not provide attitude maneuver commands
- Valid cmd is corrupted in transmission, RW sees maneuver cmd
- RW momentum changes

Refined Scenario #3.3:

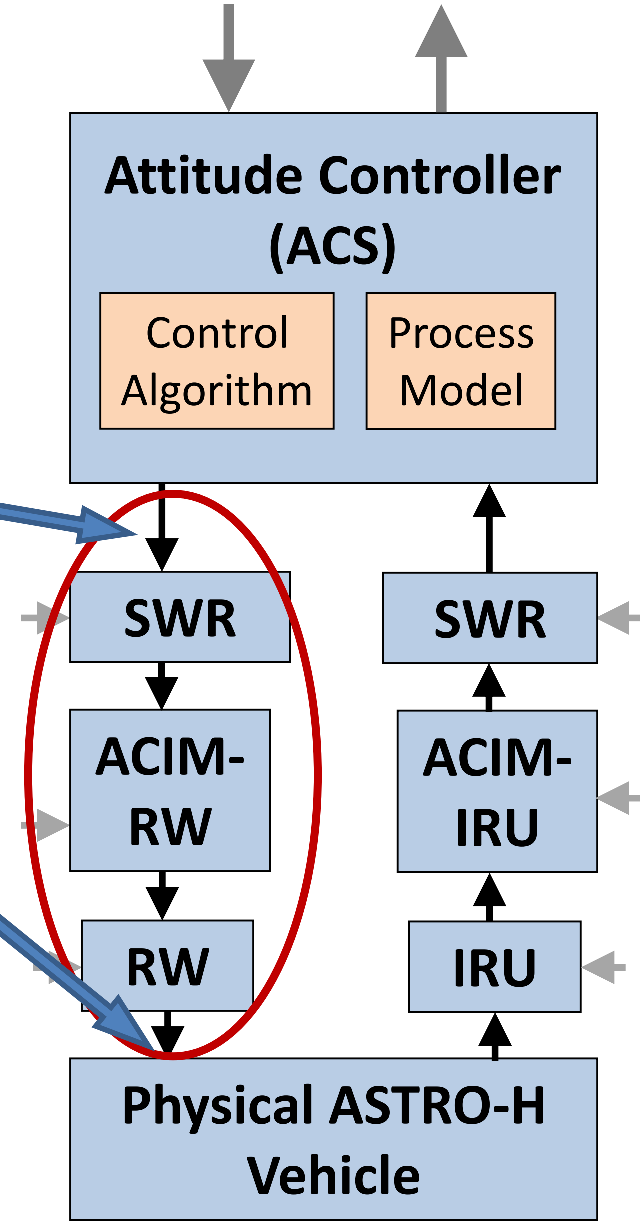
- ACS does not provide attitude maneuver commands
- RW hardware drivers overheat or fail shorted
- RW momentum changes

Top-down approach to scenario building

ACS provides attitude maneuver commands when vehicle is not rotating (UCA-2)

Basic Scenario #1:

- ACS does not provide attitude maneuver commands
- RW momentum changes



<u>To refine Type 1 scenarios:</u>

Explain how this could this happen:

- Identify the command being described
  - Attitude maneuver command
- Identify the control paths and actuators that execute the command
  - Reaction Wheels (RW)
- Case 1: Existing control paths cannot accept this command
  - The design is missing necessary control paths
- Case 2: Incorrect values (commands) transmitted
  - Transmission error or corruption
  - Delay in transmission
  - Communication link failure
  - Actuator failure (violates specification)
  - Actuator inaccuracy
  - Actuator error, misbehavior, or degradation
  - Delay in actuator response
  - Information received in a different order than sent
  - Insufficient resolution
- Case 3: Command is overridden or ignored
  - All of the above
  - Conflicting control actions are provided
  - Conditions required for transmission/operation not met (e.g. loss of power)

See complete process for all scenarios types in the process handbook

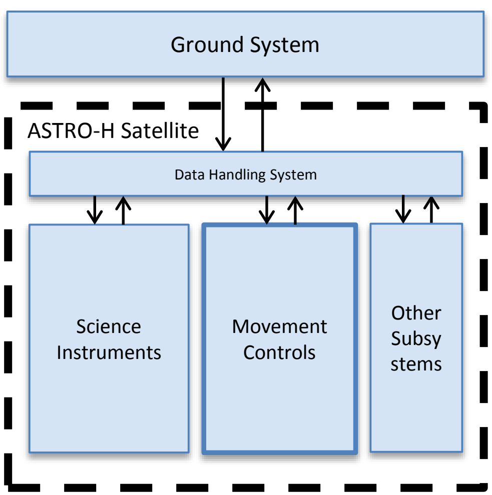
System-Theoretic Engineering Process Overview

Accidents

- A-1: Scientific mission is not performed (mission loss)

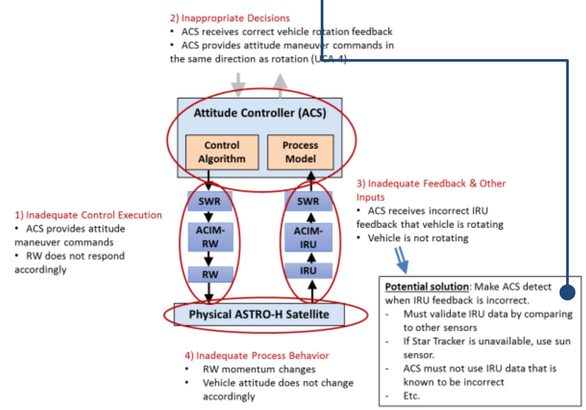
System Hazards

- H-1: ASTRO-H unable to collect scientific data
- H-2: ASTRO-H unable to communicate scientific data



Not Providing causes hazard	Providing causes hazard	Too early, too late, out of order	Stopped too soon, Applied too long
UCA-1: ACS does not provide attitude maneuver commands when ASTRO-H is rotating [H-1,H-2]	UCA-2: ACS provides attitude maneuver commands in the wrong direction (when satellite is rotating in same direction as maneuver cmd) [H-1,H-2] UCA-3: ACS provides attitude maneuver commands when ASTRO-H is not rotating [H-1,H-2] UCA-4: ACS provides attitude maneuver commands that are insufficient to slow ASTRO-H quickly [H-1,H-2]	UCA-5: ACS provides attitude maneuver commands too late after satellite attitude rate is high [H-1,H-2]	UCA-6: ACS stops providing attitude maneuver commands too soon before satellite stops rotating [H-1,H-2] UCA-7: ACS continues providing attitude maneuver commands too long after satellite stopped rotating [H-1,H-2]

Example:
"Computer provides close water valve command when catalyst open"
Source Controller — Type — Control Action — Context



Unsafe Control Action (UCA)	Safety Constraint (SC)
UCA-1: ACS does not provide attitude maneuver commands when ASTRO-H is rotating [H-1,H-2]	SC-1: ACS must provide attitude maneuver commands when ASTRO-H is rotating [H-1,H-2]
UCA-2: ACS provides attitude maneuver commands when ASTRO-H is not rotating [H-1,H-2]	SC-2: ACS must not provide attitude maneuver commands when ASTRO-H is not rotating [H-1,H-2]
UCA-3: ACS provides attitude maneuver commands that are insufficient to slow ASTRO-H quickly [H-1,H-2]	SC-3: ACS must provide attitude maneuver commands that are sufficient to slow ASTRO-H quickly [H-1,H-2]
UCA-4: ACS provides attitude maneuver commands in the same direction as rotation [H-1,H-2]	SC-4: ACS must not provide attitude maneuver commands in the same direction as rotation [H-1,H-2]
UCA-5: ACS provides attitude maneuver commands too late after ASTRO-H has rotated too far [H-1,H-2]	SC-5: ACS must not provide attitude maneuver commands too late after ASTRO-H has rotated too far [H-1,H-2]
UCA-6: ACS provides attitude maneuver commands too early to achieve desired attitude [H-1,H-2]	SC-6: ACS must not provide attitude maneuver commands too early to achieve desired attitude [H-1,H-2]
UCA-7: ACS stops providing attitude commands too soon before attitude has stabilized [H-1,H-2]	SC-7: ACS must not stop providing attitude commands too soon before attitude has stabilized [H-1,H-2]
UCA-8: ACS continues providing attitude maneuver commands too long after attitude has stabilized [H-1,H-2]	SC-8: ACS must not continue providing attitude maneuver commands too long after attitude has stabilized [H-1,H-2]

Refined Scenario #2.1:

- ACS receives correct vehicle rotation feedback from IRU
- ACS receives applies an incorrect bias estimate to IRU data
- ACS provides attitude maneuver cmds in the same direction as rotation (UCA-2)

Refined Scenario #2.2:

- ACS receives correct vehicle rotation feedback from IRU
- ACS switches to safe-hold mode and ignores data from IRU
- ACS provides attitude maneuver cmds in the same direction as rotation (UCA-2)

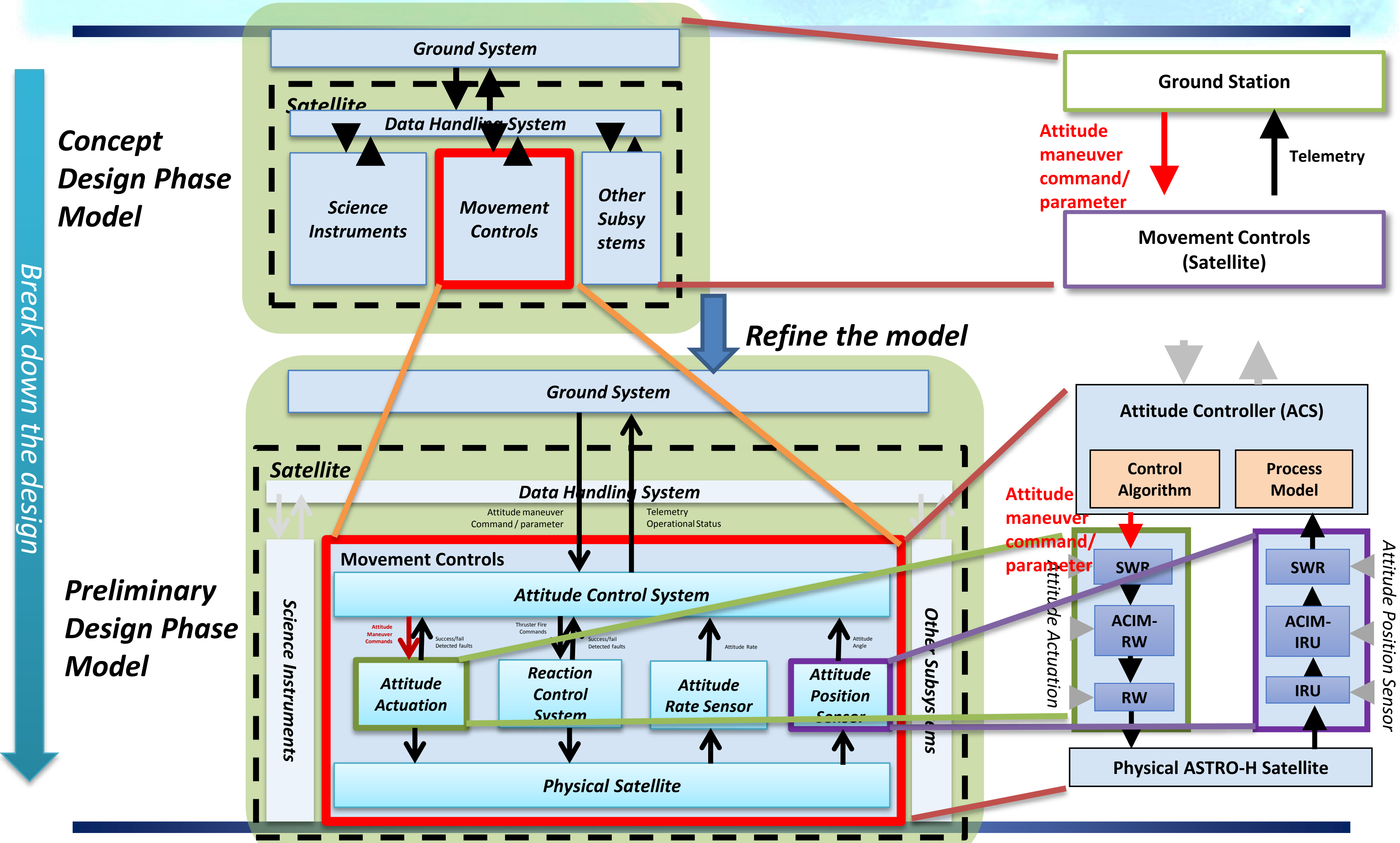
Refined Scenario #2.3:

- ACS receives correct vehicle rotation feedback from IRU
- Incorrect control parameters are uploaded to ACS, inverting attitude maneuver calculations
- ACS provides attitude maneuver cmds in the same direction as rotation (UCA-2)

Implementation

Apply to real development project

- Breakdown and refine the model and analysis -



Massachusetts Institute of Technology

JAXA Explore to Realize

JAMSS Japan Manned Space Systems Corporation

Future Plan
- (1) Collaboration with Safety Review and STAMP/STPA approach -

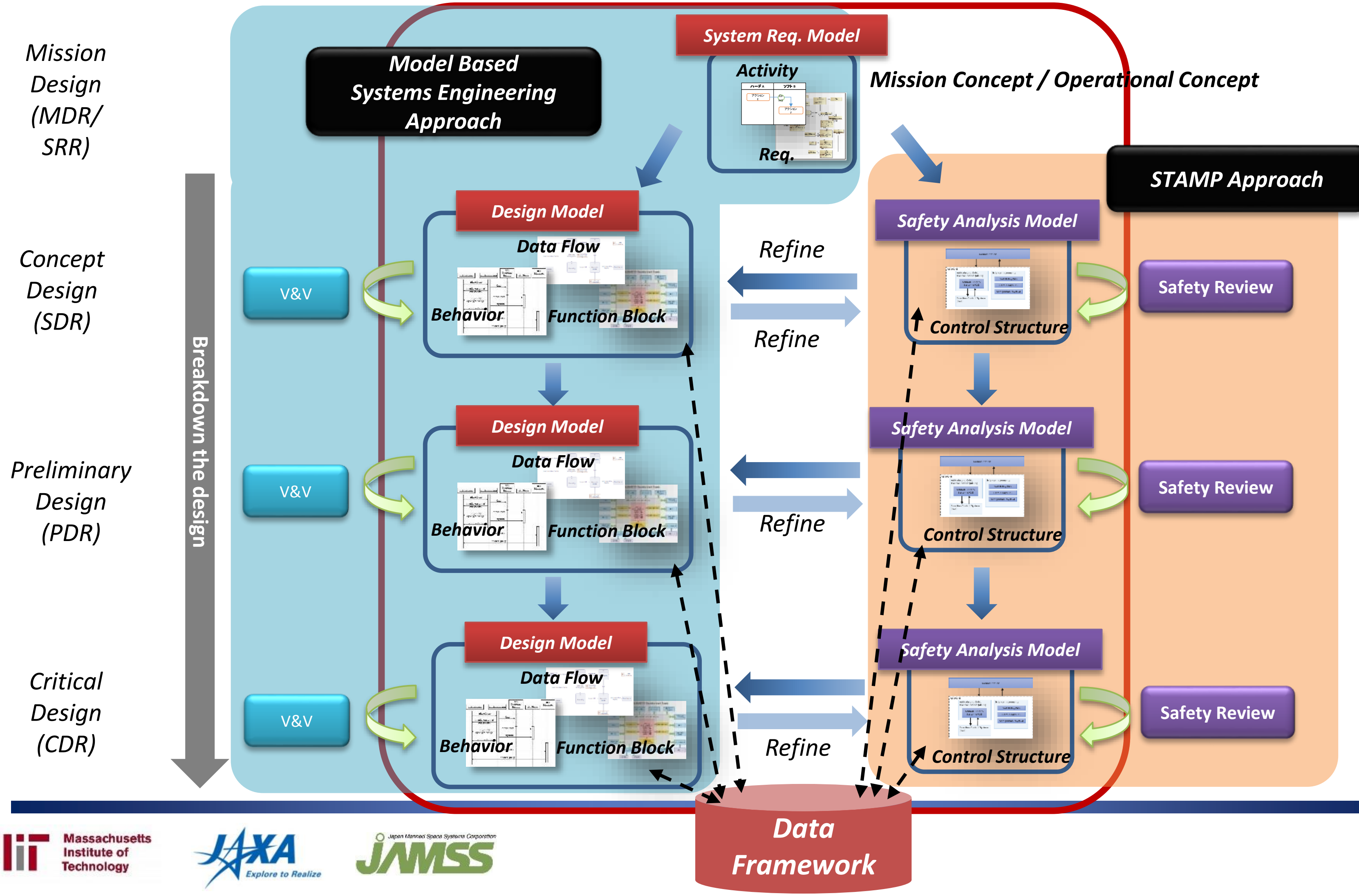
- Process in accordance with Safety Review milestone

	Concept Design	*Preliminary Design*	*Critical Design*
Phase	**MDR/SRR/SDR (Phase0)**	**PDR (Phase1)**	**CDR (Phase2)**
Purpose of Safety Review	• **Identification of hazards and hazards causes**	• **Defining the hazards and hazards causes** • **Evaluating preliminary hazard controls and verification methods**	• **Concurring the hazard control to be implemented in the final design, and verification methods**
STAMP Modeling Process	• **Identification of hazards and hazards causes at system level by System level model STEP0,1,2** • Safety Constraint for *System function level*	• **Identifying interface hazards and requirement inconsistencies by System/Subsystem/Component level model STEP0,1,2** • Safety Constraint *for Subsystem/Component function level*	• Refine the Phase1 model as needed • Finalized

Future Plan

- (2) Collaboration between MBSE and STAMP/STPA -

- MBSE top down approach



Conclusions

- Structured way to build scenarios
- Top-down approach
  - Start with basic scenarios, add detail later
  - Quicker than 100s of detailed scenarios
  - Focuses on fundamental issues first
- Easy to review
- Comprehensive, ensures coverage
- High-level scenarios are broadly applicable
  - These apply to almost every satellite
  - Only the detailed scenarios will change
- High-level scenarios can be automatically generated from UCAs!
- Can still leverage human creativity and expertise to refine scenarios, help identify UCAs, etc.

Any questions? Email me! JThomas4@mit.edu