Generating Executable Software Requirements through Hazard Analysis

Dr. John Thomas

MIT

Why formalize STPA?

Advantages

- Can provide more guidance for people new to STPA
- Can lead to tools to help automate the process
- Completeness/consistency checks
- Automatically generate requirements
- Requirements are clear and precise, not vague
- Requirements are executable

Formal STPA: Applications

Existing applications to date:

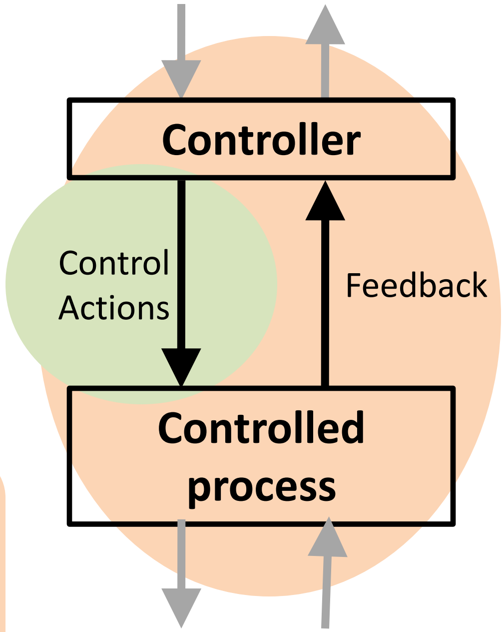
- Paul Scherrer Institute: Radiation Therapy Machine
  - In-depth detailed analysis of very complex machine
- Automated automotive systems
  - Adaptive Cruise Control, Auto Hold, and others
- NextGen In-Trail Procedure
  - New equipment and pilot procedures for oceanic flights
- JAXA: H-II Transfer Vehicle
  - Unmanned cargo vehicle that travels to International Space Station
- JAXA: GPM Satellite
  - Precipitation monitoring with dual band radar
- NRC: New Evolutionary Power Reactor
  - Automated and manual control of Main Steam Isolation
- EPRI: High Pressure Coolant Injection
  - Blind study to test multiple methods – which can identify the accident?
- ILF: Oil Pipeline Emergency Shutdown System
  - Deriving behavioral requirements for digital Integrated Control and Safety System

STPA (System-Theoretic Process Analysis)

STPA Hazard Analysis

STAMP Model

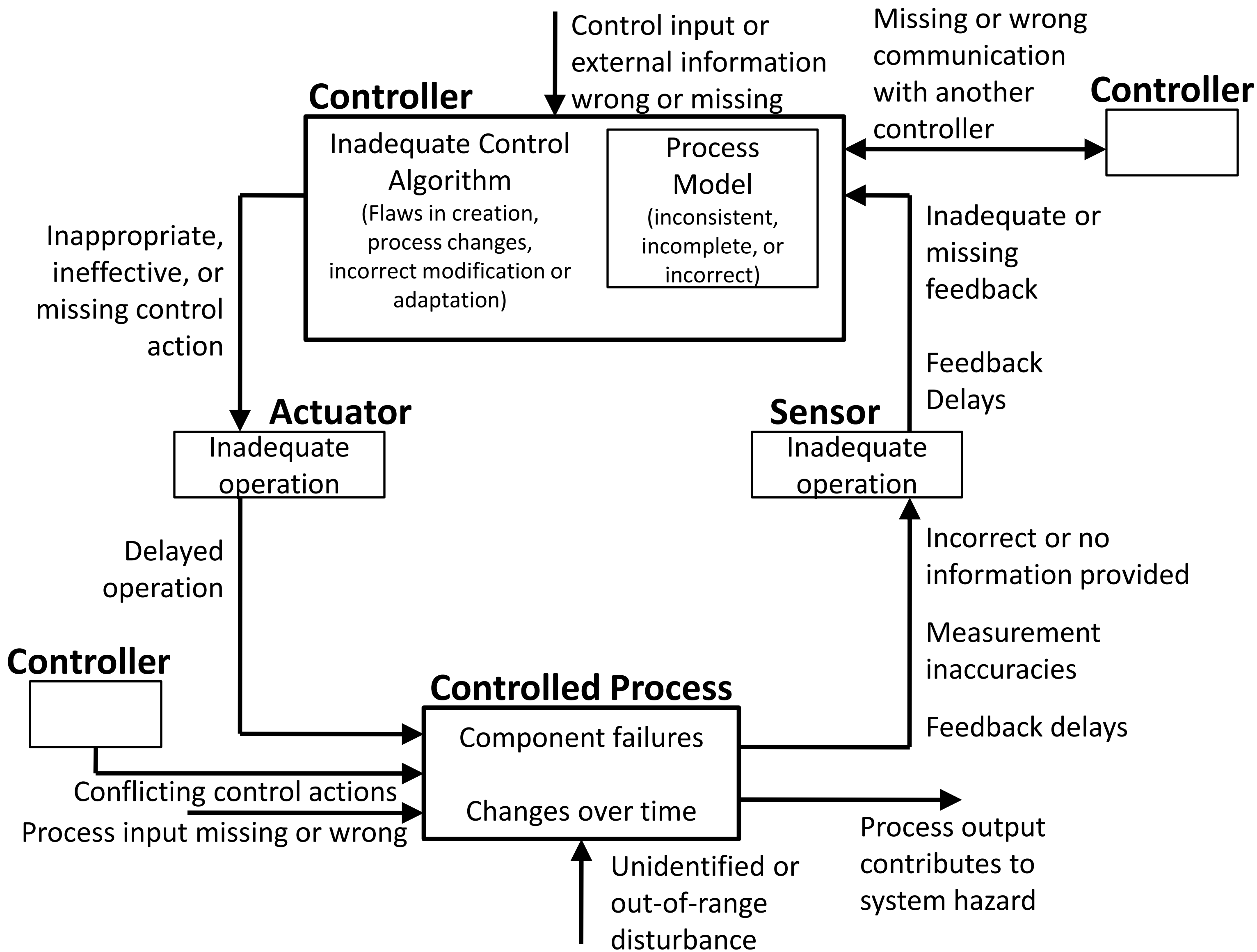
- Built on STAMP model
- Start from hazards
- Identify hazardous control actions and safety constraints
- Identify scenarios that lead to violation of safety constraints



Traditionally applied ad-hoc without systematic procedures

(Leveson, 2011)

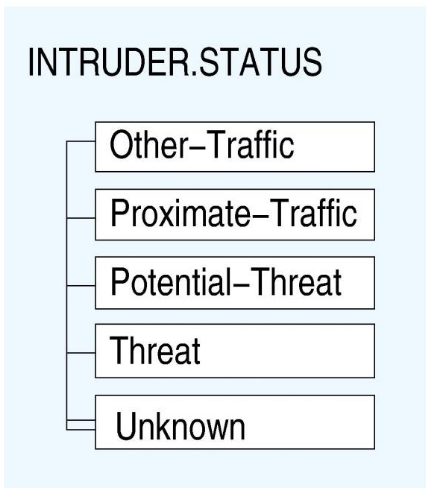
STPA Control Flaws



Need to create requirements specification without control flaws

Formal (model-based) requirements specification language

Example: SpecTRM-RL Model of TCAS II Collision Avoidance Logic



= Other–Traffic

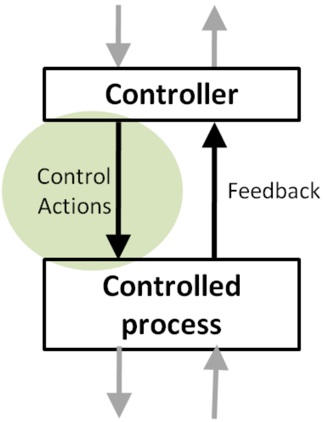
AND		OR			
	Alt–Reporting **in–state** Lost	T	T	T	.
	Bearing–Valid	F	.	T	.
	Range–Valid	.	F	T	.
	Proximate–Traffic–Condition	.	.	F	.
	Potential–Threat–Condition	.	.	F	.
	Other–Aircraft **in–state** On–Ground	.	.	.	T

Formal mathematical representation:

Other-Traffic =
(Alt-Reporting == Lost) ∧ ¬Bearing-Valid ∨ (Alt-Reporting == Lost) ∧ ¬Range-Valid ∨
(Alt-Reporting == Lost) ∧ Bearing-Valid ∧ Range-Valid ∧ ¬Proximate-Traffic-Condition ∧
¬Potential-Threat-Condition ∨ (Other-Aircraft == On-Ground)

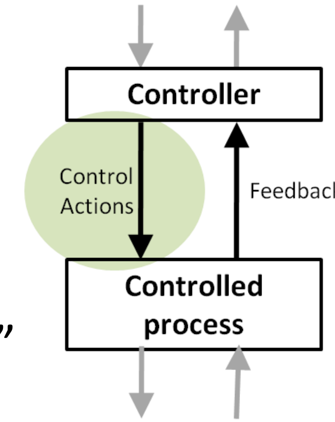
(Leveson, 2000), (Zimmerman, 2002)

Structure of a Hazardous Control Action



Example:
“Operator provides open train door command when train is moving”

Structure of a Hazardous Control Action



Example:
"Operator provides open train door command when train is moving"

Source Controller (SC) — "Operator"
Type (T) — "provides"
Control Action (CA) — "open train door command"
Context (Co) — "train is moving"

Four parts of a hazardous control action

- Source Controller: the controller that can provide the control action
- Type: whether the control action was provided or not provided
- Control Action: the controller's command that was provided / missing
- Context: the system or environmental state in which command is provided

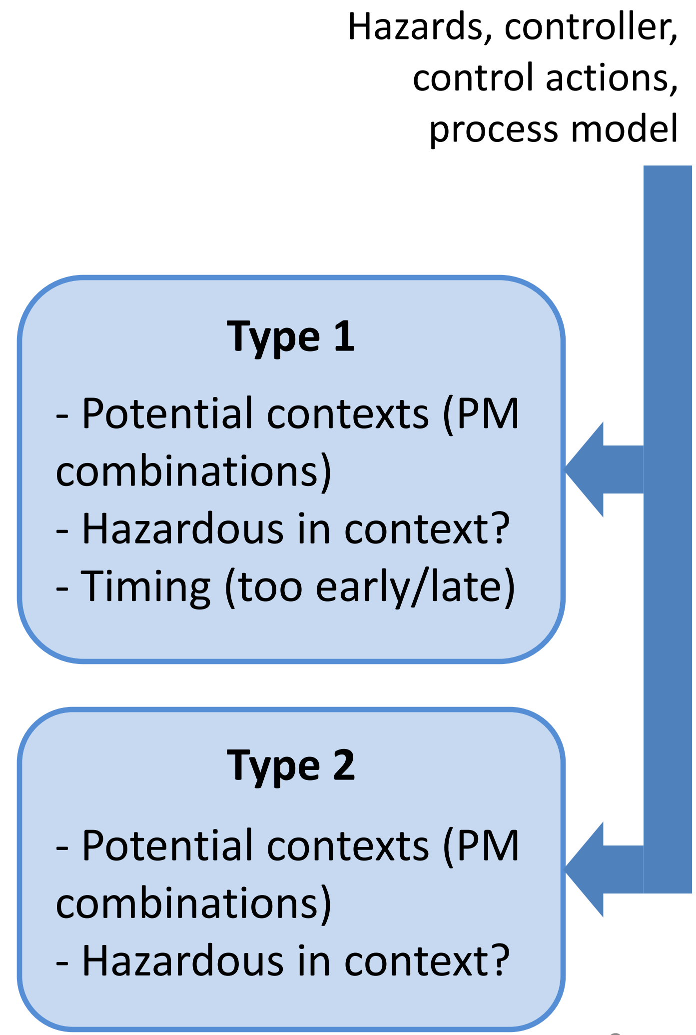
Process Model

Train motion: Stopped / Moving

Train location: At platform / Not Aligned

Identifying Hazardous Control Actions

- Type 1: Providing control action causes hazard
  - 1a) Define potential contexts (combinations of process model values)
  - 1b) Determine whether the control action is hazardous in each context
  - 1c) Determine whether control action can still be hazardous if too early/too late
- Type 2: Not providing control action causes hazard
  - Same as above, but for an absence of the selected control action



Example: Train door controller

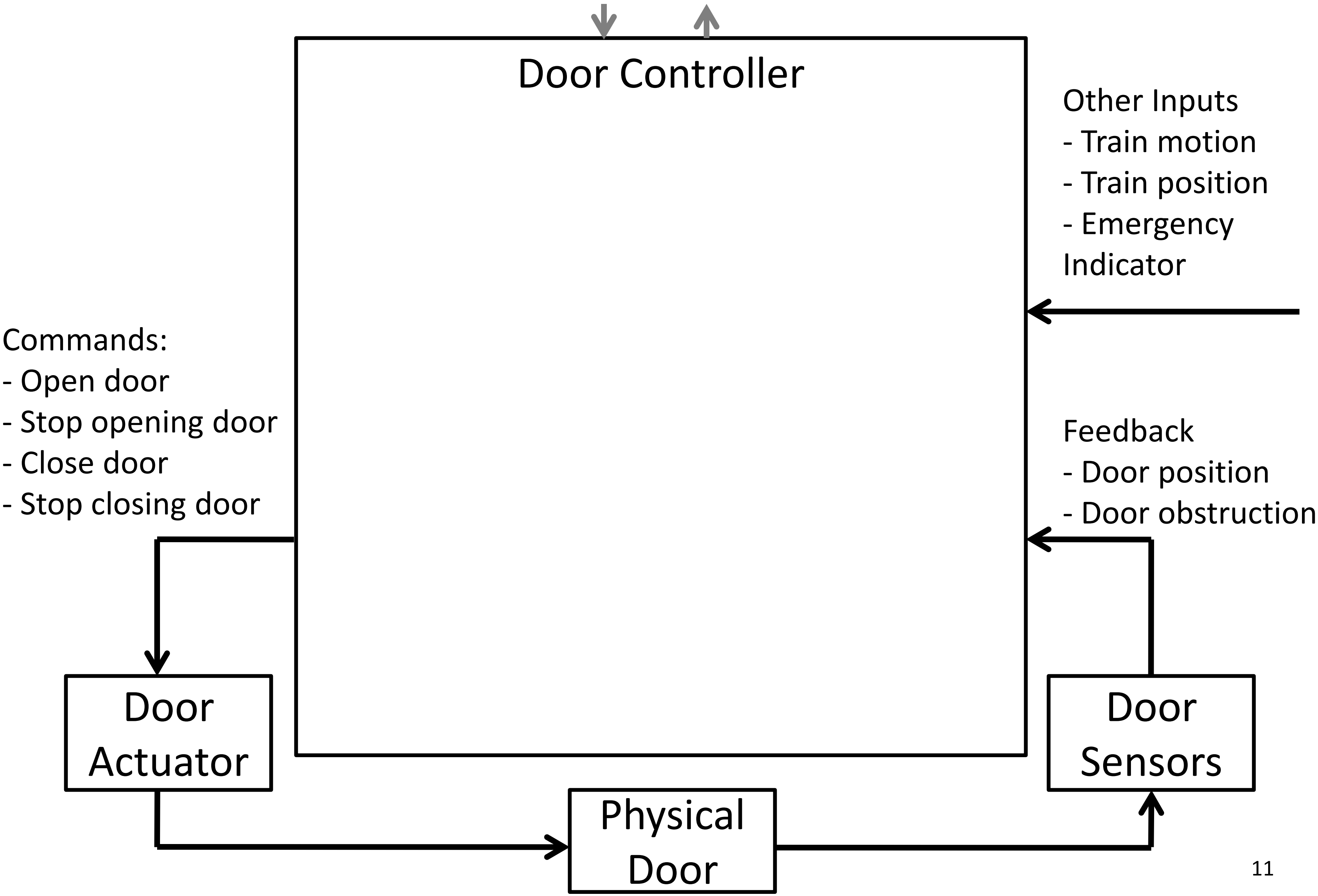
System Hazards

H-1: Doors close on a person in the doorway

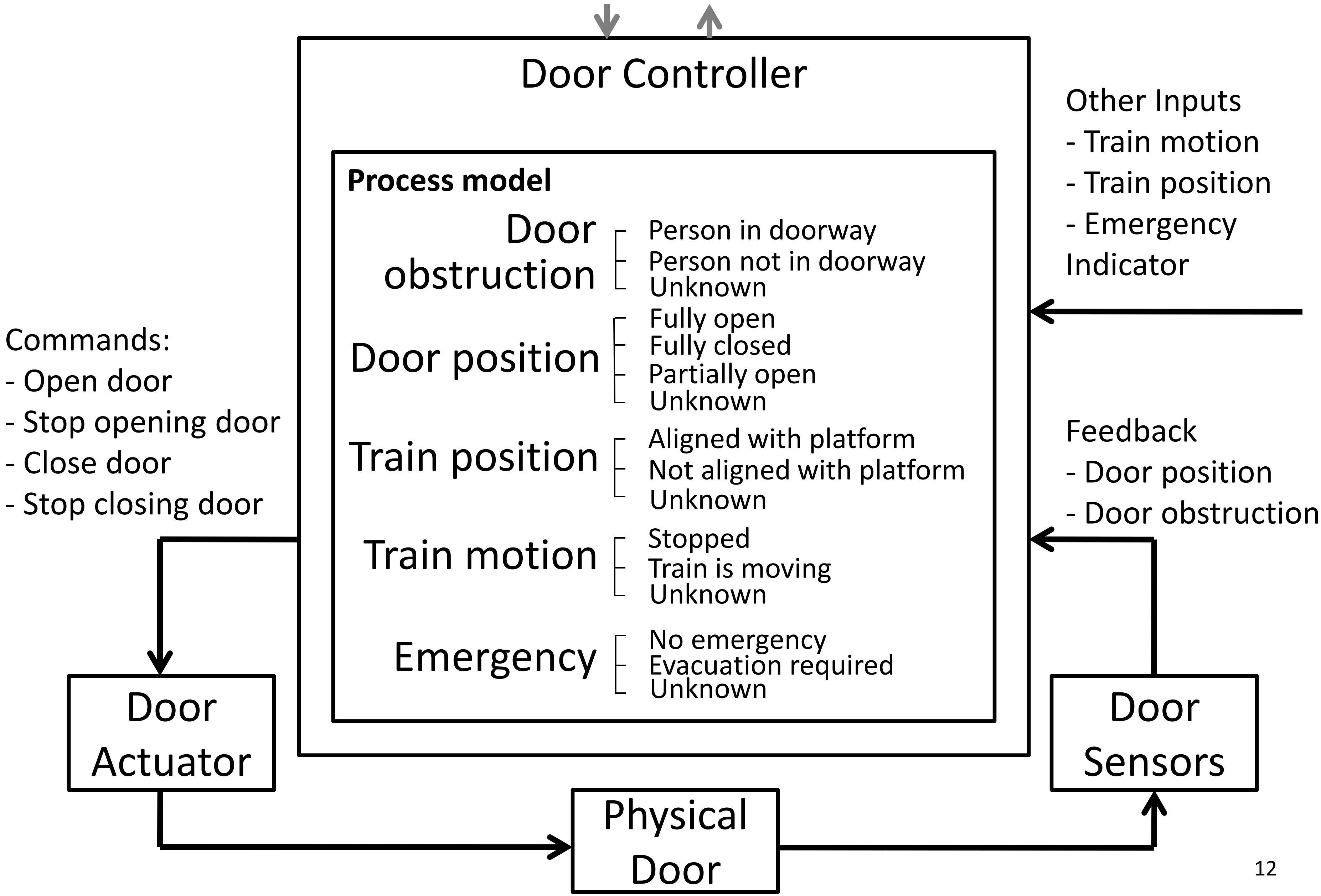
H-2: Doors open when the train is moving or not at platform

H-3: Passengers/staff are unable to exit during an emergency

Example: Control loop

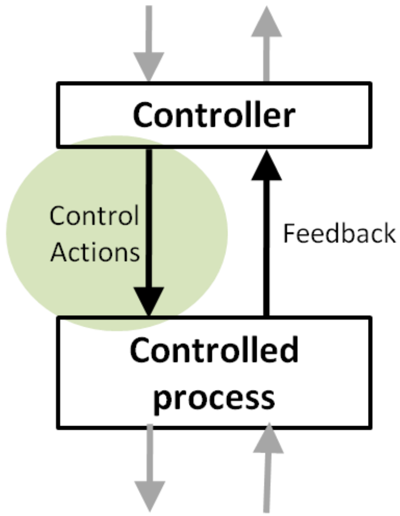


Example: Control loop



STPA Process

- ✓ Identify hazards
- ✓ Create control structure
- ✓ Create process model
- ➡ Identify Unsafe Control Actions
  - For each control action, consider:
  - 1) Providing causes hazard
  - 2) Not providing causes hazard
- Identify Causes of Unsafe Control Actions



1) Control action is provided

- Control action: *Door Open* command
- 1a) Define potential contexts (combinations of process model variables)

Control Action	Train Motion	Emergency	Train Position	Door Obstruction	Door Position
Door open command	Stopped	No	Aligned with platform	Not obstructed	Closed
Door open command	Stopped	No	Aligned with platform	Not obstructed	Open
Door open command	Stopped	Yes	Aligned with platform	Obstructed	Closed
…	…	…	…	…	…

1) Control action is provided

Control action: *Door Open* command

- 1a) Define potential contexts (combinations of process model variables)
- 1b) Determine whether the control action is hazardous in each context

Control Action	Train Motion	Emergency	Train Position	Door Obst. / Position	Hazardous?
Door open command	Moving	No	(doesn't matter)	(doesn't matter)	**Yes**
Door open command	Moving	Yes	(doesn't matter)	(doesn't matter)	**Yes***
Door open command	Stopped	Yes	(doesn't matter)	(doesn't matter)	**No**
Door open command	Stopped	No	Not at platform	(doesn't matter)	**Yes**
Door open command	Stopped	No	At platform	(doesn't matter)	**No**

***Design decision: In this situation, evacuate passengers to other cars. Meanwhile, stop the train and then open doors.**

1) Control action is provided

Control action: *Door Open* command

- 1a) Define potential contexts (combinations of process model variables)
- 1b) Determine whether the control action is hazardous in each context
- 1c) Determine whether control action can still be hazardous if too early/too late

Control Action	Train Motion	Emergency	Train Position	Door Obst. / Position	Hazardous ?	Hazardous if provided too early?	Hazadous if provided too late?
Door open command	Moving	No	(doesn't matter)	(doesn't matter)	Yes	Yes	Yes
Door open command	Moving	Yes	(doesn't matter)	(doesn't matter)	Yes*	Yes*	Yes*
Door open command	Stopped	Yes	(doesn't matter)	(doesn't matter)	No	No	Yes
Door open command	Stopped	No	Not at platform	(doesn't matter)	Yes	Yes	Yes
Door open command	Stopped	No	At platform	(doesn't matter)	No	No	No

2) Control action is <u>not</u> provided

Control action: *Door Open* command

- 2a) Identify process model variables
- 2b) Determine whether the absence of control action is hazardous in each context

Control Action	Train Motion	Emergency	Train Position	Door Obst. / Pos.	Hazardous?
Door open command not provided	Stopped	Yes	(doesn't matter)	(doesn't matter)	**Yes**
Door open command not provided	Stopped	(doesn't matter)	(doesn't matter)	Closing on obstruction	**Yes**
Door open command not provided	(all others)				**No**

Resulting List of Hazardous Control Actions

Hazardous Control Actions
Door open command provided while train is moving and there is no emergency
Door open command provided too late while train is stopped and emergency exists
Door open command provided while train is stopped, no emergency, and not at platform
Door open command provided while train is moving and emergency exists
Door open command not provided while train is stopped and emergency exists
Door open command not provided while doors are closing on someone and train is stopped

Much of this can be automated to assist the safety engineer!

Generating safety requirements

Hazardous Control Actions

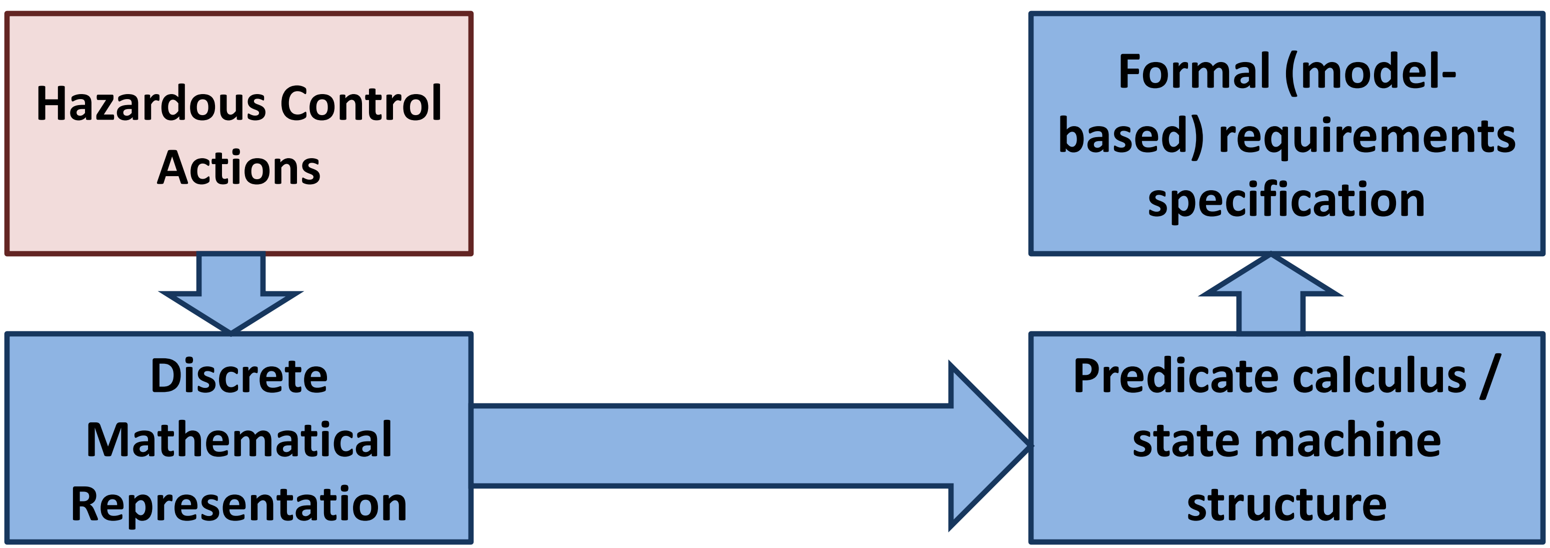
?

Formal (model-based) requirements specification

Alt–Reporting **in–state** Lost	T	T	T	.
Bearing–Valid	F	.	T	.
Range–Valid	.	F	T	.
Proximate–Traffic–Condition	.	.	F	.
Potential–Threat–Condition	.	.	F	.
Other–Aircraft **in–state** On–Ground	.	.	.	T

Generating safety requirements

- Formal requirements can be derived using
  - Discrete mathematical structure for hazardous control actions
  - Predicate calculus to obtain necessary requirements
- Automatically generate formal requirements given these relationships!



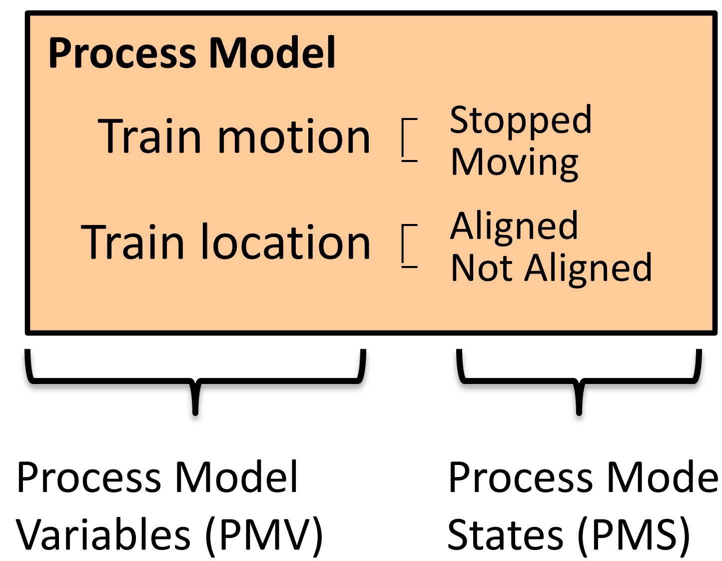
Hazardous control actions: mathematical representation

Example: "Operator provides open train door command when train is moving"

Source Controller (SC) — Type (T) — Control Action (CA) — Context (Co)

Hazardous control action as 4-tuple (SC, T, CA, Co) where:

- SC ∈ Controllers [from control structure]
- T ∈ {Provided, Not Provided}
- CA ∈ ControlActions(SC)
- Co = {V, SC} | (V ∈ PMV) ∧ (SC ∈ PMS) ∧ SC child V



Generating safety requirements

- Example: Generated black-box model for door controller

Provide 'Open Doors' command

		Behavior required for function	Behavior required for safety	Behavior required for safety
Door State =	Doors not closing on person			
	Doors closing on person			T
Train Position =	Aligned with platform	T		
	Not aligned with platform			
Train Motion =	Stopped	T	T	T
	Train is moving			
Emergency =	No emergency			
	Emergency exists		T	

Open Doors =
(Train Position in-state Aligned) ∧ (Train Motion in-state Stopped) ∨ (Train Motion in-state Stopped) ∧ (Emergency in-state exists) ∨ (Door State in-state closing on person) ∧ (Train Motion in-state Stopped)

Detecting conflicts

- Can automatically check consistency using info in context tables

Control Action	Train Motion	Emergency	Hazardous?
Door open command	Moving	Yes	**Yes***

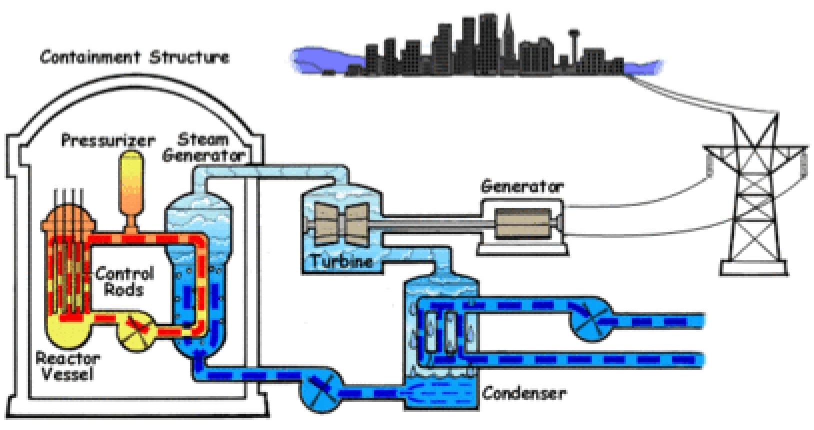
Control Action	Train Motion	Emergency	Hazardous?
Door open command not provided	Moving	Yes	**Yes***

- Example: Conflict between opening the door vs. not opening the door

Nuclear MSIV example

Identify Unsafe Control Actions

- What are the process model variables?
- MSIV remains open during normal plant operation
- MSIV only used to control a few specific abnormal conditions:
  - **Steam generator tube rupture**
    - Can cause uncontrolled SG level increase, release contaminated fluid into secondary system
  - **Steam system piping failure**
    - Can depressurize SG, cause overcooling transient and energy release into containment
  - **Feedwater system piping failure**
    - Can depressurize SG, cause overcooling transient and energy release into containment
- MSIV also controls heat exchange within SG
  - **Other support systems** must be engaged to provide additional cooling if closed



Context table for *Close MSIV* control action not provided

- Automatically generated from control structure and process model
- To identify the UCAs, engineers fill in the last column

	1	2	3	4	5	6
	Control Action	**Condition of Steam Generator Tube**	**Condition of Main Feedwater Pipe**	**Condition of Main Steamline**	**Operation of other support systems**	**Not Providing Control Action is Hazardous?**
1	*Close MSIV*	Not Ruptured	Not Ruptured	Not Ruptured	Adequate	
2		Ruptured	Not Ruptured	Not Ruptured	Adequate	
3		Not Ruptured	Ruptured	Not Ruptured	Adequate	
4		Not Ruptured	Not Ruptured	Ruptured	Adequate	
5		Ruptured	Ruptured	Not Ruptured	Adequate	
6		Not Ruptured	Ruptured	Ruptured	Adequate	
7		Ruptured	Not Ruptured	Ruptured	Adequate	
8		Ruptured	Ruptured	Ruptured	Adequate	
9		Not Ruptured	Not Ruptured	Not Ruptured	Adequate	
10		Ruptured	Not Ruptured	Not Ruptured	Inadequate	
11		Not Ruptured	Ruptured	Not Ruptured	Inadequate	
12		Not Ruptured	Not Ruptured	Ruptured	Inadequate	
13		Ruptured	Ruptured	Not Ruptured	Inadequate	
14		Not Ruptured	Ruptured	Ruptured	Inadequate	
15		Ruptured	Not Ruptured	Ruptured	Inadequate	
16		Ruptured	Ruptured	Ruptured	Inadequate	

Context table for *Close MSIV* control action not provided

- Keeping MSIV open is not hazardous if no rupture (row 1, 9)
- If MSIV kept open during SGTR, will cause all hazards
- If kept open, causes H-2, H-3 during steamline or feedwater rupture

Tools can automatically populate table using these 3 rules

	1	2	3	4	5	6
	Control Action	**Condition of Steam Generator Tube**	**Condition of Main Feedwater Pipe**	**Condition of Main Steamline**	**Operation of other support systems**	**Not Providing Control Action is Hazardous?**
1	*Close MSIV*	Not Ruptured	Not Ruptured	Not Ruptured	Adequate	No
2		Ruptured	Not Ruptured	Not Ruptured	Adequate	H-1, H-2, H-3, H-4
3		Not Ruptured	Ruptured	Not Ruptured	Adequate	H-2, H-3
4		Not Ruptured	Not Ruptured	Ruptured	Adequate	H-2, H-3
5		Ruptured	Ruptured	Not Ruptured	Adequate	H-1, H-2, H-3, H-4
6		Not Ruptured	Ruptured	Ruptured	Adequate	H-2, H-3
7		Ruptured	Not Ruptured	Ruptured	Adequate	H-1, H-2, H-3, H-4
8		Ruptured	Ruptured	Ruptured	Adequate	H-1, H-2, H-3, H-4
9		Not Ruptured	Not Ruptured	Not Ruptured	Adequate	No
10		Ruptured	Not Ruptured	Not Ruptured	Inadequate	H-1, H-2, H-3, H-4
11		Not Ruptured	Ruptured	Not Ruptured	Inadequate	H-2, H-3
12		Not Ruptured	Not Ruptured	Ruptured	Inadequate	H-2, H-3
13		Ruptured	Ruptured	Not Ruptured	Inadequate	H-1, H-2, H-3, H-4
14		Not Ruptured	Ruptured	Ruptured	Inadequate	H-2, H-3
15		Ruptured	Not Ruptured	Ruptured	Inadequate	H-1, H-2, H-3, H-4
16		Ruptured	Ruptured	Ruptured	Inadequate	H-1, H-2, H-3, H-4

Context table for *Close MSIV* control action provided

	1	2	3	4	5	6	7	8
	Control Action	**Condition of Steam Generator Tube**	**Condition of Main Feedwater Pipe**	**Condition of Main Steamline**	**Operation of other support systems**	**Control Action Hazardous?**	**Control Action Hazardous if Too Late?**	**Control Action Hazardous if Too Early?**
1	*Close MSIV*	Not Ruptured	Not Ruptured	Not Ruptured	Adequate	**Yes**	**Yes**	**Yes**
2		Ruptured	*	*	Adequate	No	**Yes**	**Yes**
3		Not Ruptured	Ruptured	Not Ruptured	Adequate	No	**Yes**	No
4		Not Ruptured	Not Ruptured	Ruptured	Adequate	No	**Yes**	No
5		Not Ruptured	Ruptured	Ruptured	Adequate	No	**Yes**	No
6		*	*	*	Inadequate	**Yes**	**Yes**	**Yes**

Summary of UCAs identified

Control Action	Unsafe Control Actions			
	Not Providing Causes Hazard	Providing Causes Hazard	Wrong Timing or Order Causes Hazard	Stopped Too Soon or Applied Too Long
Close MSIV	Close MSIV not provided when there is a rupture in the S/G tube, main feedwater, or main steam line and the support systems are adequate [H-2, H-1, H-3]	Close MSIV provided when there is a rupture and other support systems are inadequate [H-1, H-2, H-3] Close MSIV provided when there is no rupture [H-4]	Close MSIV provided too early (while SG pressure is high): SG pressure may rise, trigger relief valve, abrupt steam expansion [H-2, H-3] Close MSIV provided too late after SGTR: contaminated coolant released into secondary loop, loss of primary coolant through secondary system [H-1, H-2, H-3]	N/A

Conflicts automatically detected

- Rows 10-16
  - Context: rupture is present but other support systems are not operating or inadequate
  - Hazardous to keep MSIV open
    - May contaminate secondary system, cause overcooling transient, etc.
  - Hazardous to close MSIV
    - Isolates the only operational cooling system
  - Conflict should be addressed. For example, may be best to keep MSIV open to provide limited cooling until operators find a solution

Automatically generated model-based requirements

Provide 'Close MSIV valve' command

Steam Generator Tube =	Not Ruptured	T	T	T	F	F	F	F	T	T	T	F	F	F
	Ruptured	F	F	F	T	T	T	T	F	F	F	T	T	T
Condition of Main Feedwater Pipe =	Not Ruptured	T	F	F	T	T	F	F	T	F	F	T	F	F
	Ruptured	F	T	T	F	F	T	T	F	T	T	F	T	T
Condition of Main Steamline =	Not Ruptured	F	T	F	T	F	T	F	F	T	F	F	T	F
	Ruptured	T	F	T	F	T	F	T	T	F	T	T	F	T
Operation of other support systems =	Adequate	T	T	T	T	T	T	T	F	F	F	F	F	F
	Inadequate	F	F	F	F	F	F	F	T	T	T	T	T	T

Derived from H-2, H-3
Derived from H-2, H-3
Derived from H-2, H-3
Derived from H-1, H-2, H-3, H-4
Derived from H-1, H-2, H-3, H-4
Derived from H-1, H-2, H-3, H-4
Derived from H-1, H-2, H-3, H-4
Derived from H-2, H-3
Derived from H-2, H-3
Derived from H-2, H-3
Derived from H-1, H-2, H-3, H-4
Derived from H-1, H-2, H-3, H-4
Derived from H-1, H-2, H-3, H-4

Traceability can also be provided from info in context tables

Summary

- Systematic process for performing STPA
- Method to help automate STPA
- Drives the creation of requirements and definition of control algorithms from the STPA analysis
- Automatically generating formal safety requirements
- Analyze not only safety aspects, but also functional goals
- Consistency checks to detect safety vs. functional conflicts

Thank you!

Questions?