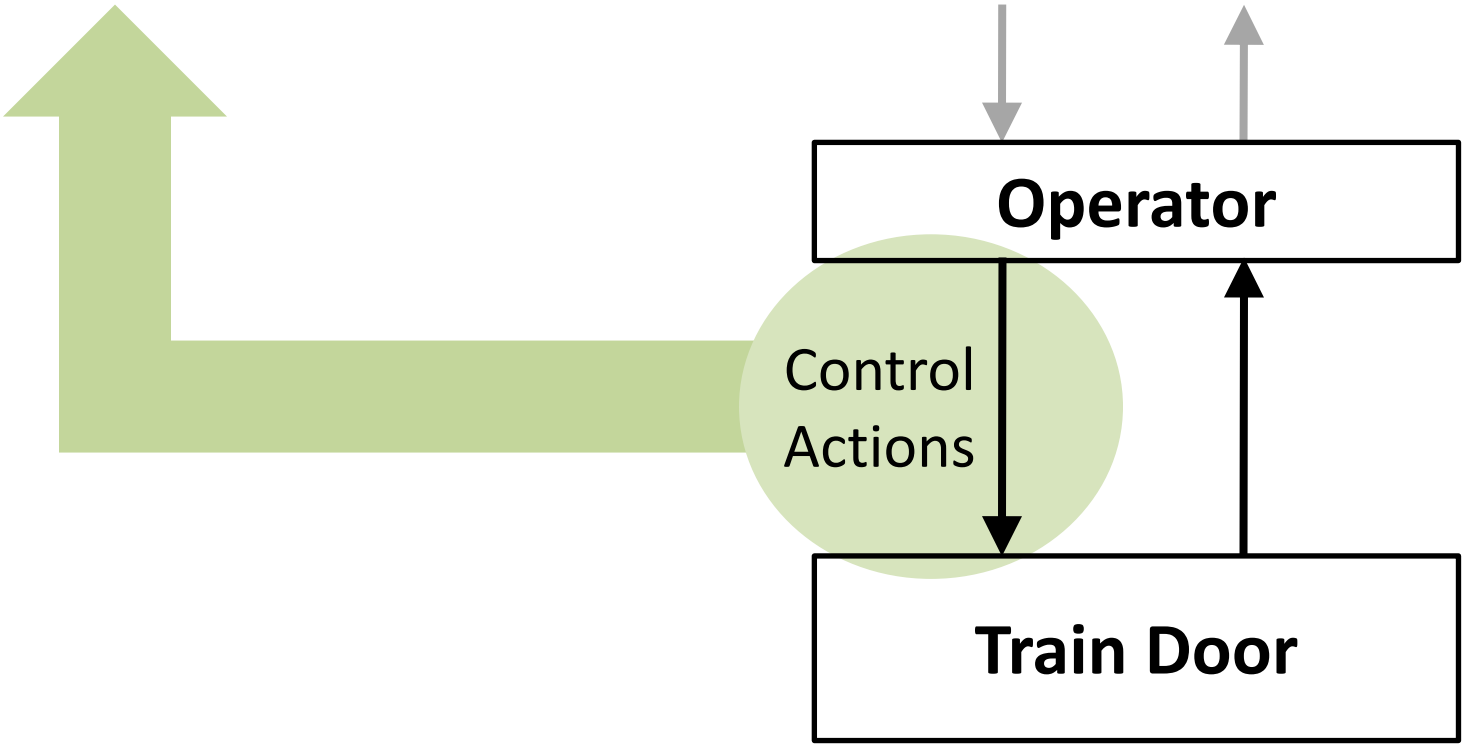
An STPA Tool

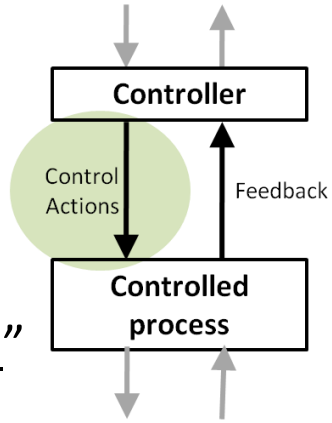
Dajiang Suo, John Thomas

Structure of an Unsafe Control Action

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



Structure of an Unsafe Control Action



Example:

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

- Source Controller: Operator
- Type: provides
- Control Action: open train door command
- Context: 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: conditions for the hazard to occur

Process Model

Train motion	Stopped Moving
Train location	At platform Not Aligned

1) Control action is provided

Example:
"Operator provides open train door command when _____"

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

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

2) Control action is not provided

Example:
"Operator does not provide open train door command when _____"

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	...				**No**

Resulting List of Unsafe Control Actions

Unsafe 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!

1) Control action is provided

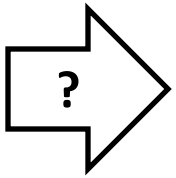
Example:
"Software provides open train door command when train is moving"

Control Action	Train Motion	Emergency	Train Position	Hazardous?
Door open command	Moving	No	Aligned with platform	**Yes**
Door open command	Not Moving	No	Aligned with platform	
Door open command	Not Moving	Yes	Not aligned with platform	
Door open command	Moving	Yes	Not aligned with platform	**Yes**
…	…	…	…	

Automate with Rules

Generating safety requirements

Hazardous Control Actions

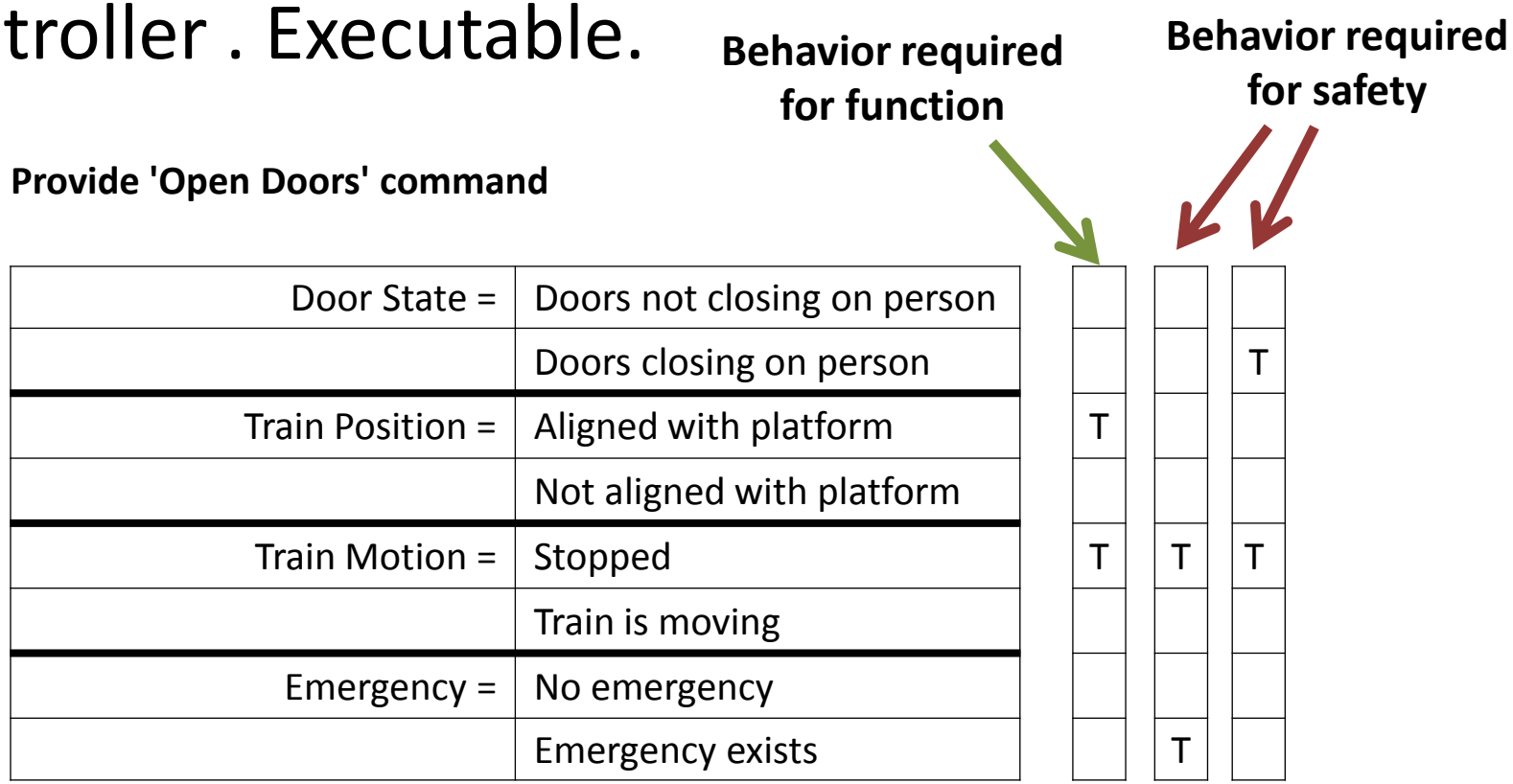


Formal (model-based) requirements specification

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

- Example: Generated black-box model for door controller . Executable.



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***

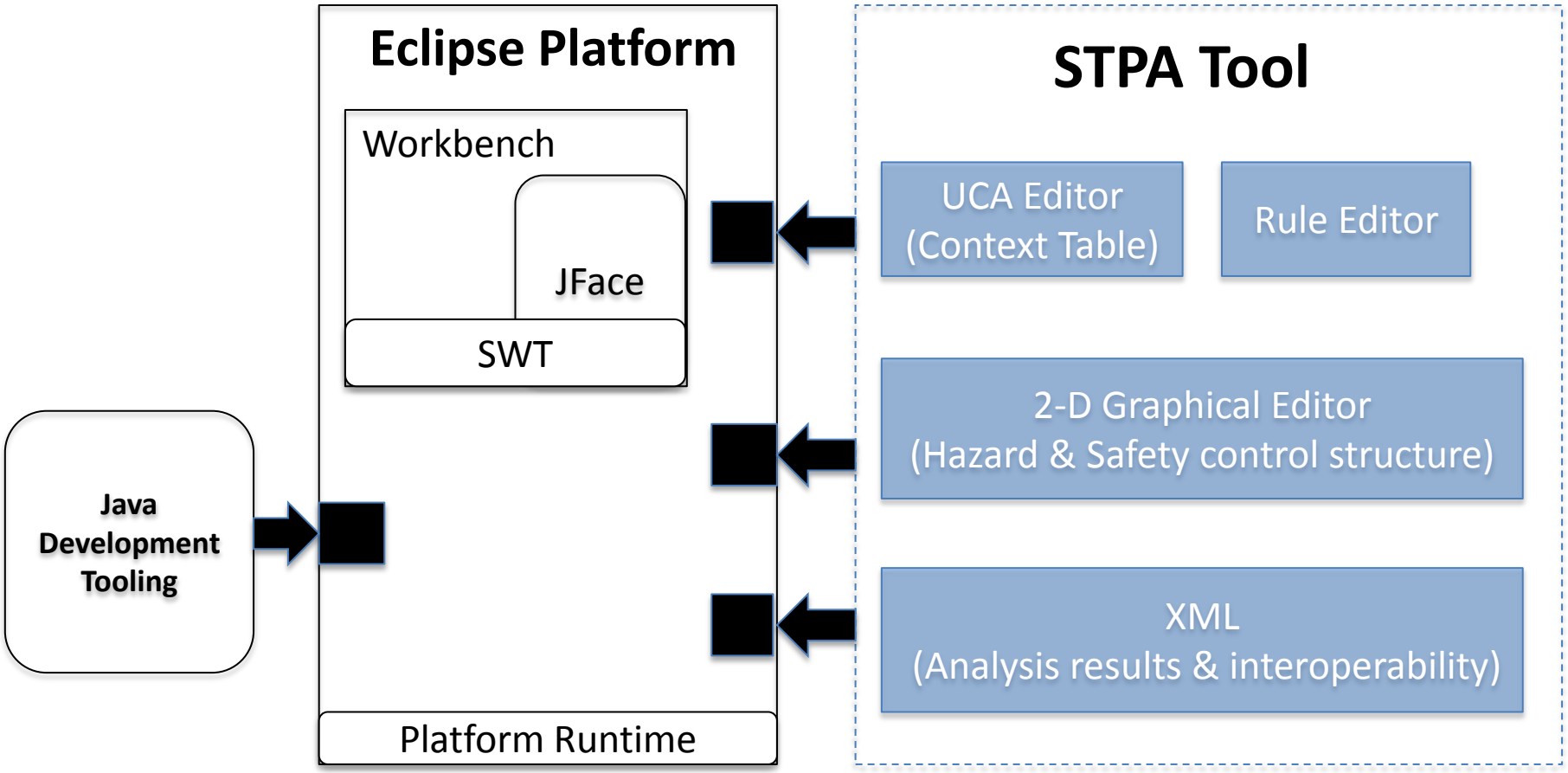
Control Action	Train Motion	Emergency	Hazardous?
Door open command not provided	Moving	Yes	**Yes***

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

Objectives for the STPA tool

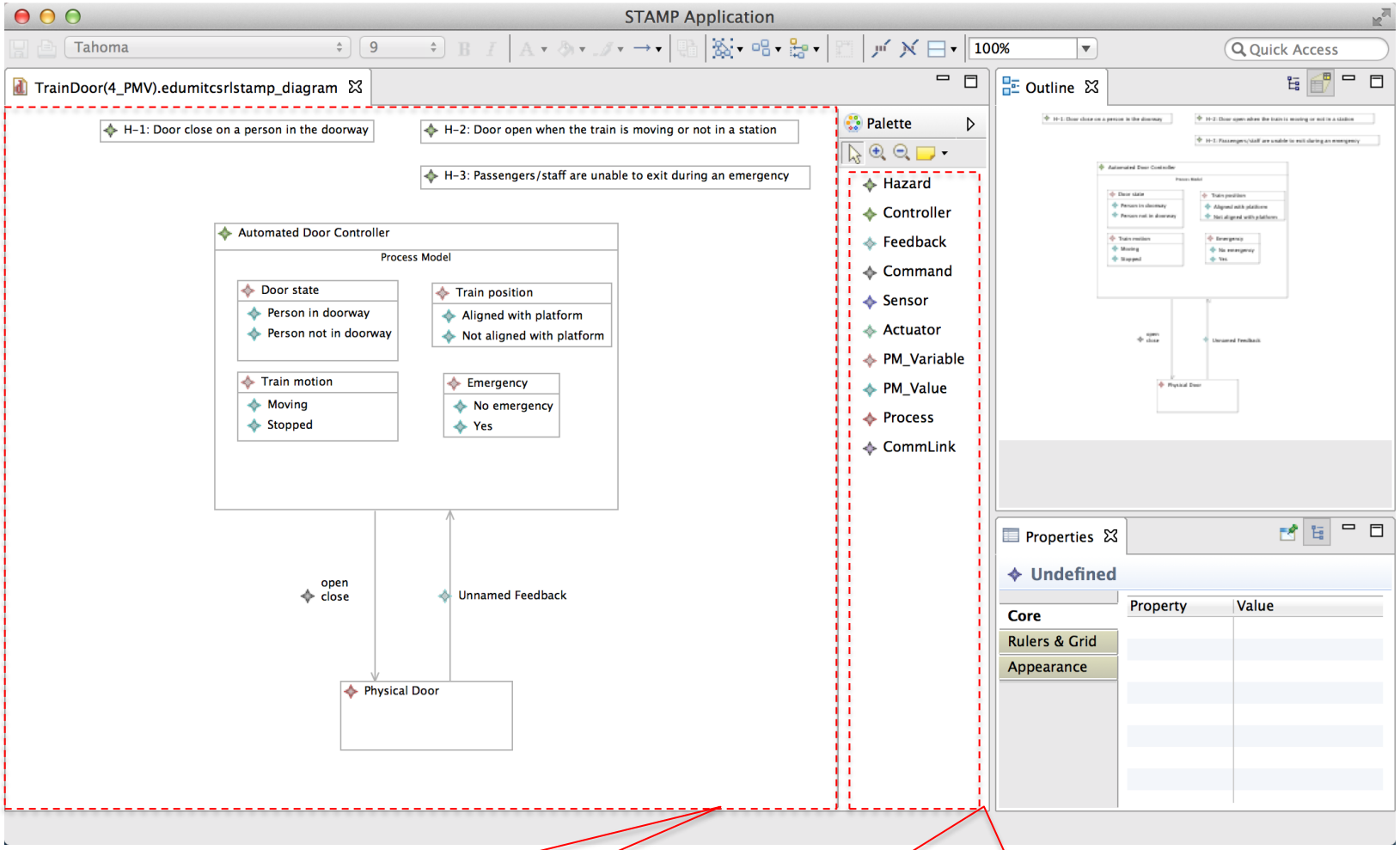
- Allow users to:
  - Specify Hazards
  - Draw the safety control structure
    - Add controllers, controlled processes
    - Add actuators and sensors
    - Add control actions and feedback
    - Add process model variables and values
  - Perform STPA Step 1
    - Generate context table templates automatically based on control structure
    - Allow user to specify which row causes hazards
    - Allow user to define "Rules", used to automatically complete many rows with related hazards
    - Detect conflicts between two rules
    - Calculate and show And/Or tables of executable requirements
  - Generate XML files for storing analysis results and interoperation
- Help users with STPA Step 2
  - TBD

The Architecture of the STPA tool



*The architecture of Eclipse platform is taken from eclipse.org

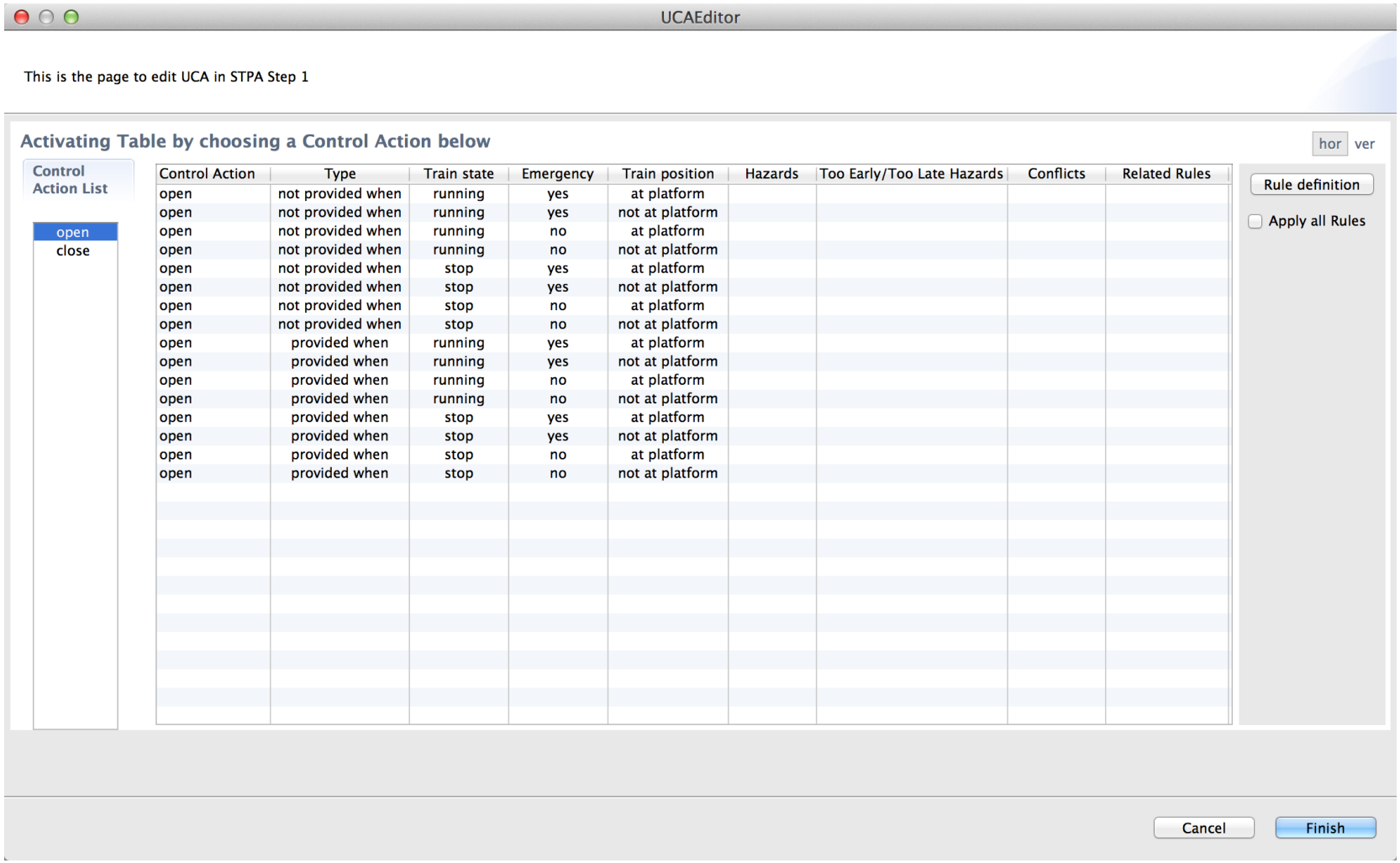
Specify hazards&Draw safety control structure



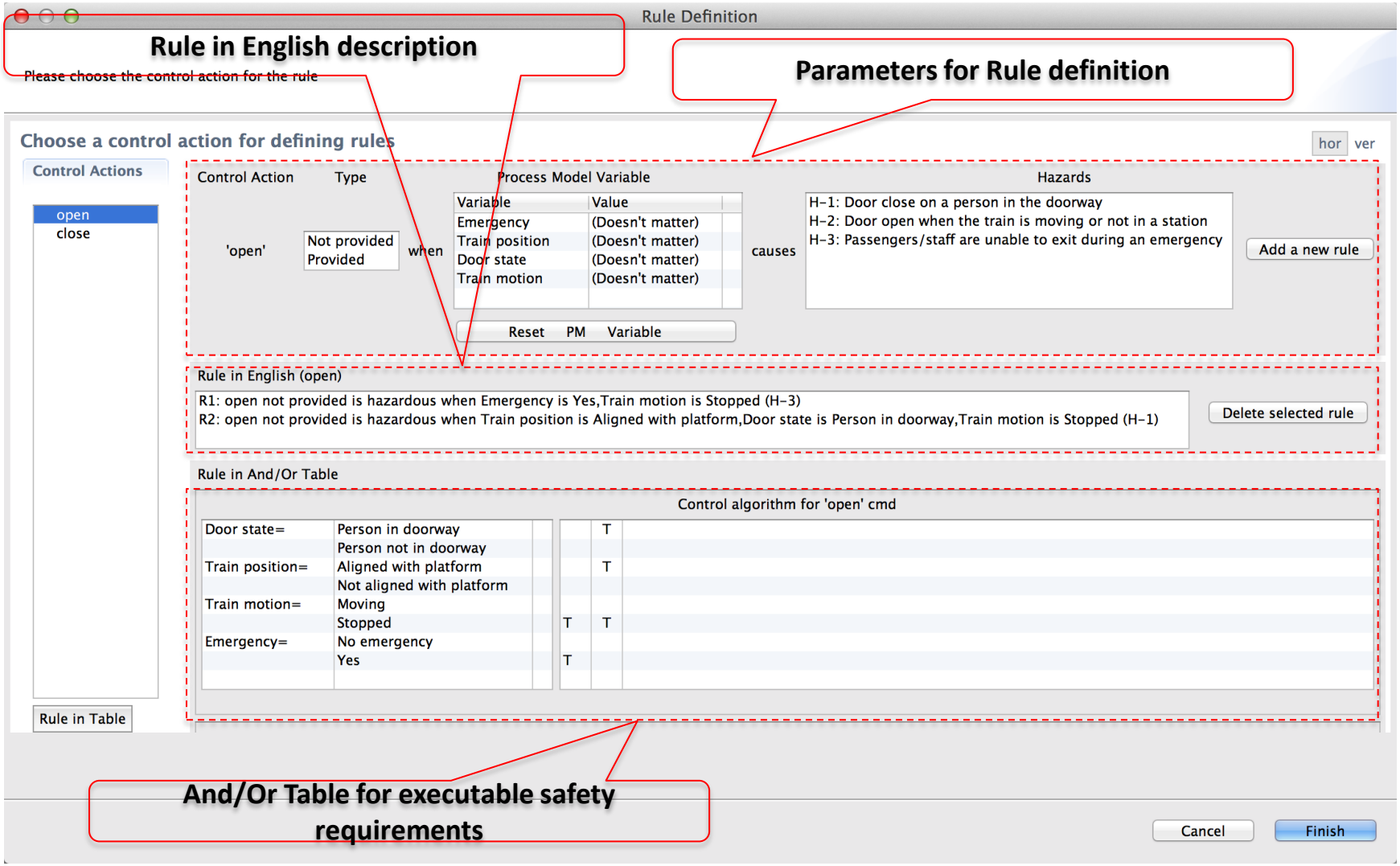
Graphical Editor for Safety Control Structure

Tool Bar for choosing components to add

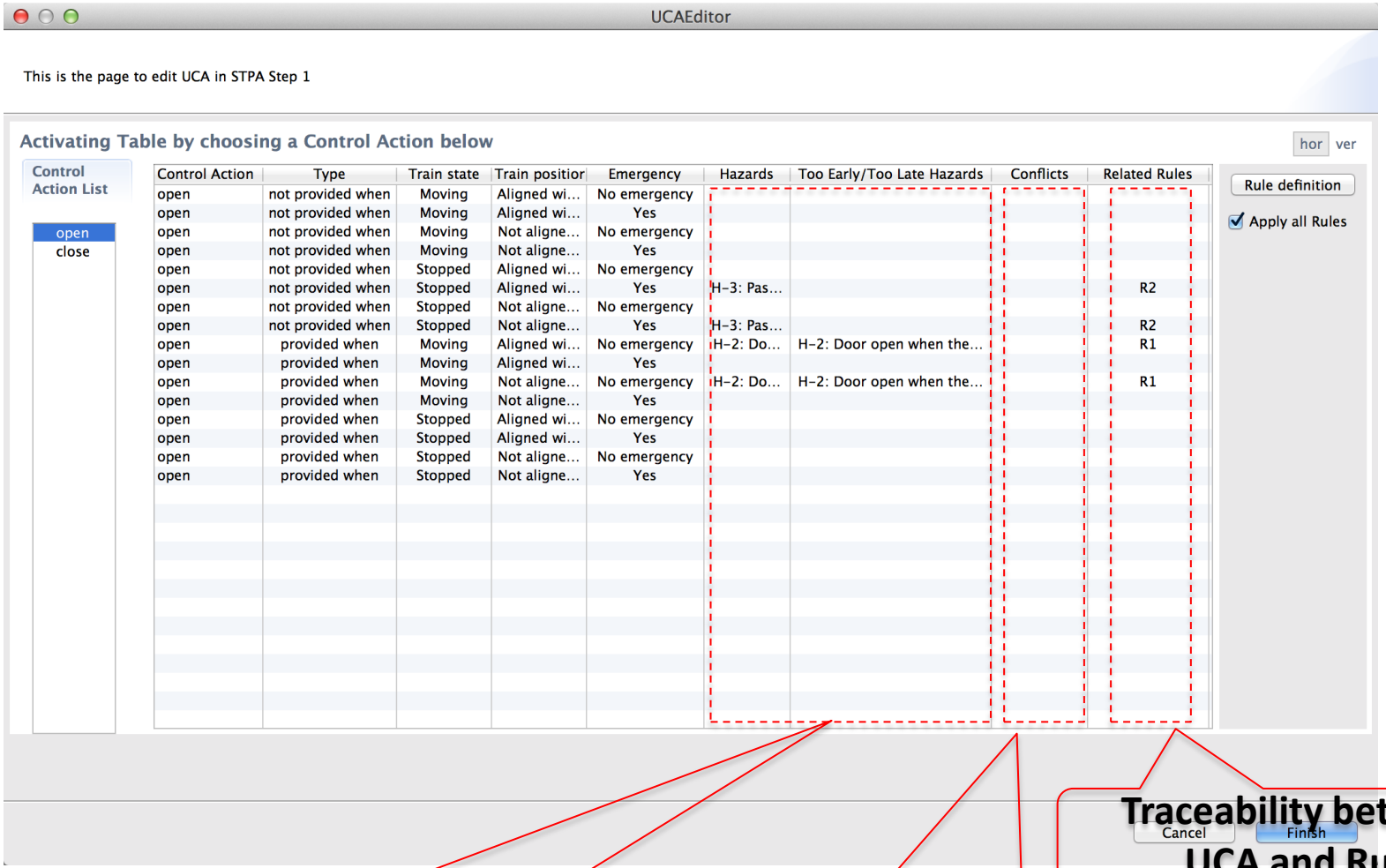
Automatically generate Context Table for UCA



Define rules and calculate related And/or Table



Apply Rules to complete rows with related hazards in context table



Conclusion

- Automatically generate Context Table based on Process model
- Allow the user to define Rules to identify UCA
- Automatically construct And/Or Table for executable safety requirements