

Verification of Programmable Logic Controllers for Critical Infrastructures

McKenzy Heavlin

CS 584

Spring 2025

Background & Problem Statement

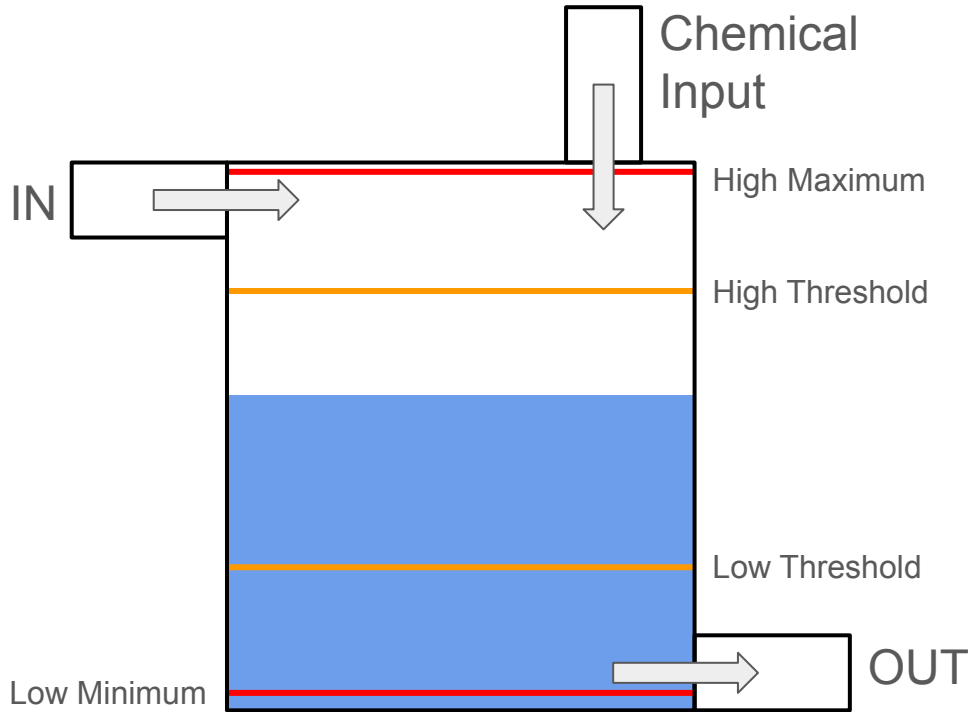
Programmable Logic Controllers (PLCs) are computers specifically designed to **manage and run industrial plant processes** reliably and repeatedly.



PLC programs must **operate safely** and **adhere to strict safety principles** assigned by system operators.

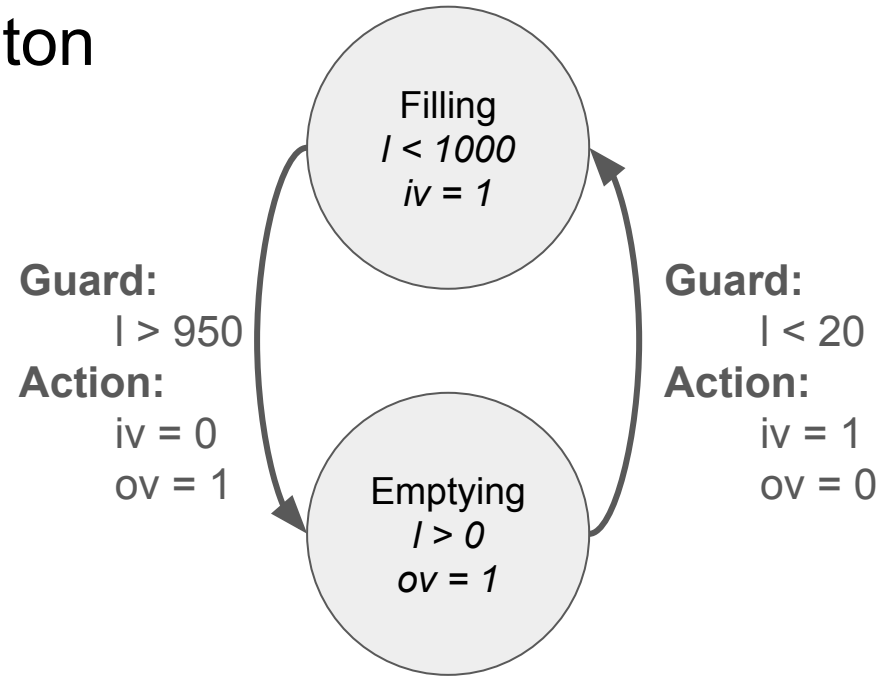
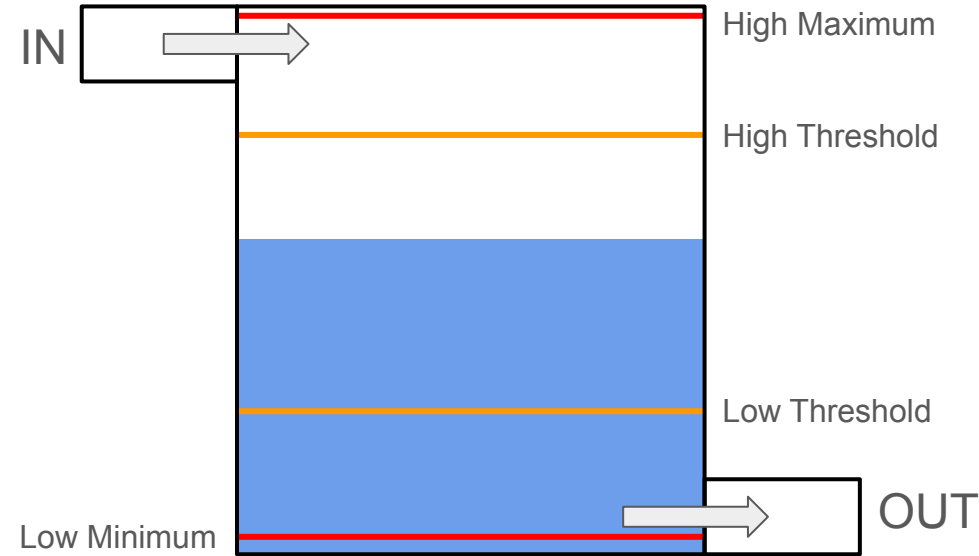
Goal: verify simple PLC controller code in a simplified system.

Problem Setting: Water Treatment Facility



- Sensor's monitor the thresholds, maximums, and chemical input rate
- Valve actuators enable opening/closing of all input and output valves
- Focus on water thresholds, flow rates first; then add in chemical dosing if time allows

Converting to a Simple Automaton



Explored adding an 'Idle' state where the ' $iv=0$ ' and ' $ov=0$ ' but ran into C2E2 difficulties

C2E2 Conversion – Sanity Check

- Most simple automaton with one variable, two modes

simple_plc.hyxml - C

Mode

Name: emptying

ID: 1

Initial: ☐

Flows: -

$\dot{l} = -0.5 * l$

Add Row

Invariants: -

$l \geq 0$

Add Row

Cancel Confirm

simple_plc.hyxml - C

Mode

Name: filling

ID: 0

Initial: ☒

Flows: -

$\dot{l} = 0.1 * l$

Add Row

Invariants: -

$l \leq 1000$

Add Row

Cancel Confirm

simple_plc.hyxml - C2E2

Transition
emptying -> filling

ID: 1

Source: emptying

Destination: filling

Guards: $l \leq 20$

Actions: -

Add Row

Cancel Confirm

simple_plc.hyxml - C2E2

Transition
filling -> emptying

ID: 0

Source: filling

Destination: emptying

Guards: $l \geq 950$

Actions: -

Add Row

Cancel Confirm

C2E2 Conversion – Sanity Check

Requirement

Name: ✓

◆ Safety

Time step: ✓

Time horizon: ✓

K value: ✓

Simulator: —

Refinement: —

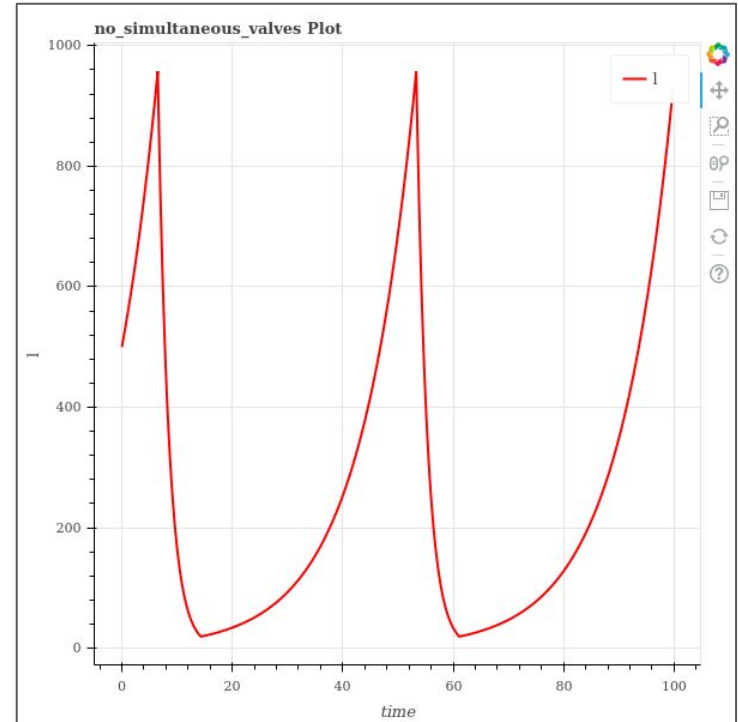
Linear Model: —

Initial set: ✓

filling: $l == 50$

Unsafe set: ✓

$l < 0$



Requirement List		
Requirement	Status	Result
high_level	Verified	Safe
low_level	Verified*	Expired

C2E2 Conversion – Adding in Valves

Updated Modes: 3 variables

model_with_valves.hyxm

Mode

Name: emptying

ID: 1

Initial: ☐

Flows: -

$\dot{l} = iv \cdot 0.1 \cdot l - ov \cdot 0.5 \cdot l$

$\dot{iv} = 0$

$\dot{ov} = 0$

Add Row

Invariants: -

$l \geq 0$

Add Row

Cancel Confirm

model_with_valves.hyxm

Mode

Name: filling

ID: 0

Initial: ☒

Flows: -

$\dot{l} = iv \cdot 0.1 \cdot l - ov \cdot 0.5 \cdot l$

$\dot{iv} = 0$

$\dot{ov} = 0$

Add Row

Invariants: -

$l \leq 1000$

Add Row

Cancel Confirm

model_with_valves.hyxm

Transition
emptying -> filling

ID: 1

Source: emptying

Destination: filling

Guards: $l \leq 20$

Actions: -

$iv = 1$

$ov = 0$

Add Row

Cancel Confirm

model_with_valves.hyxm

Transition
filling -> emptying

ID: 0

Source: filling

Destination: emptying

Guards: $l \geq 950$

Actions: -

$iv = 0$

$ov = 1$

Add Row

Cancel Confirm

C2E2 Conversion – Adding in Valves

- Need initial states for the valves and a new unsafe set
- Checking that 'iv' and 'ov' aren't open at the same time

Requirement

Name: level_w_valves ✓

◆ Safety

Time step: 0.1 ✓

Time horizon: 20.0 ✓

K value: 2000.0 ✓

Simulator: ODEINT: fixed step ☐

Refinement: Default Strategy ☐

Linear Model: Generalized Star ☐

Initial set: ✓

filling: l == 50 && iv == 1 && ov == 0

Unsafe set: ✓

l > 1000 && l < 0

Requirement

Name: valve_states ✓

◆ Safety

Time step: 0.1 ✓

Time horizon: 20.0 ✓

K value: 2000.0 ✓

Simulator: ODEINT: fixed step ☐

Refinement: Default Strategy ☐

Linear Model: Generalized Star ☐

Initial set: ✓

filling: l == 50 && iv == 1 && ov == 0

Unsafe set: ✓

iv == 1 && ov == 1

Requirement List		
Requirement	Status	Result
level_w_valves	Simulated	Safe
valve_states	Simulated	Safe

Requirement List		
Requirement	Status	Result
level_w_valves	Verified	Safe
valve_states	Verified	Safe

C2E2 Simulation

“Sequence of initial states are drawn randomly from the starting set. Once the simulation is computed, the result is checked to see if any guards are hit”

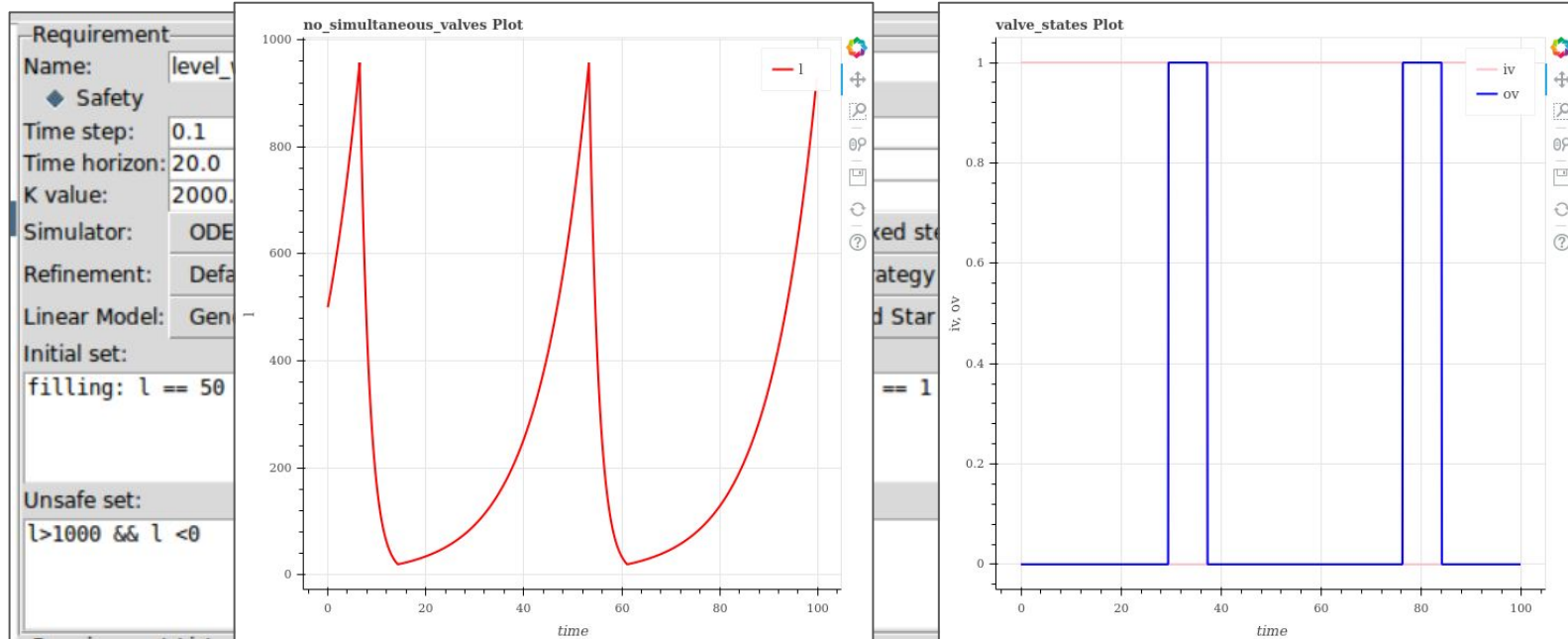
- C2E2 User Manual

C2E2 Conversion – Adding in Valves

- Need initial states for the valves and a new unsafe set
- Checking that 'iv' and 'ov' aren't open at the same time

Requirement List		
Requirement	Status	Result
level_w_valves	Simulated	Safe
valve_states	Simulated	Safe

Requirement List		
Requirement	Status	Result
level_w_valves	Verified	Safe
valve_states	Verified	Safe



C2E2 Conversion – Adding in Flow Rates

- Update model to reflect variable flow rates (5 vars)

flow_rate.hxml - C

Mode

Name: emptying

ID: 1

Initial: ☐

Flows: -

$\dot{l} = iv \cdot i_flow - ov \cdot o_flow$

$\dot{iv} = 0$

$\dot{ov} = 0$

$\dot{i_flow} = 0$

$\dot{o_flow} = 0$

Add Row

Invariants: -

$l \geq 0$

Add Row

Cancel Confirm

flow_rate.hxml - C

Mode

Name: filling

ID: 0

Initial: ☒

Flows: -

$\dot{l} = iv \cdot i_flow - ov \cdot o_flow$

$\dot{iv} = 0$

$\dot{ov} = 0$

$\dot{i_flow} = 0$

$\dot{o_flow} = 0$

Add Row

Invariants: -

$l \leq 1000$

Add Row

Cancel Confirm

flow_rate.hxml - C2E2

Transition
emptying -> filling

ID: 1

Source: emptying

Destination: filling

Guards: $l \leq 20$

Actions: -

$iv = 1$

$ov = 0$

$i_flow = 0.3$

$o_flow = 0$

Add Row

Cancel Confirm

flow_rate.hxml - C2E2

Transition
filling -> emptying

ID: 0

Source: filling

Destination: emptying

Guards: $l \geq 950$

Actions: -

$iv = 0$

$ov = 1$

$i_flow = 0$

$o_flow = 0.7$

Add Row

Cancel Confirm

C2E2 Conversion – Adding in Flow Rates

- Update initial set and check a new condition

Requirement		
Name:	level_w_valves	✓
◆ Safety		
Time step:	0.01	✓
Time horizon:	20.0	✓
K value:	0.0	✓
Simulator:	ODEINT: fixed step	☐
Refinement:	Default Strategy	☐
Linear Model:	Generalized Star	☐
Initial set:		✓
filling: l == 50 && iv == 1 && ov == 0 && i_flow == 0.3 && o_flow == 0		
Unsafe set:		✓
l < 1000		
Requirement List		
Requirement	Status	Result
level_w_valves	Simulated	Unsafe

Requirement		
Name:	valve_states	✓
◆ Safety		
Time step:	0.1	✓
Time horizon:	100.0	✓
K value:	0.0	✓
Simulator:	ODEINT: fixed step	☐
Refinement:	Default Strategy	☐
Linear Model:	Generalized Star	☐
Initial set:		✓
filling: l == 50 && iv == 1 && ov == 0 && i_flow == 0.3 && o_flow == 0		
Unsafe set:		✓
iv == 1 && ov == 1		
Requirement List		
Requirement	Status	Result
level_w_valves	Simulated	Unsafe
valve_states	Simulated	Safe

Requirement		
Name:	flow_rates	✓
◆ Safety		
Time step:	0.1	✓
Time horizon:	100.0	✓
K value:	0.0	✓
Simulator:	ODEINT: fixed step	☐
Refinement:	Default Strategy	☐
Linear Model:	Generalized Star	☐
Initial set:		✓
filling: l == 50 && iv == 1 && ov == 0 && i_flow == 0.3 && o_flow == 0		
Unsafe set:		✓
i_flow - o_flow >= 0		
Requirement List		
Requirement	Status	Result
level_w_valves	Simulated	Unsafe
valve_states	Simulated	Safe
flow_rates	Simulated	Unsafe

C2E2 Conversion – Adding in Flow Rates

- The issue:
 - Values quickly bypass the $l < 1000$ condition
 - Appears to never make the transition from *filling* -> *emptying*
 - Implies there is some error in the transition between modes which I have yet to debug
 - User manual not too helpful
 - ChatGPT suggested it may be an issue with the time step granularity and how C2E2 progresses – not too convinced of this though

The screenshot shows the C2E2 Requirement Editor for a requirement named 'level_w_valves'. The interface includes fields for Name, Safety, Time step, Time horizon, K value, Simulator, Refinement, Linear Model, Initial set, and Unsafe set. The Initial set contains a complex logical expression. The Unsafe set contains the condition $l < 1000$. Below the editor is a Requirement List table.

Requirement	Status	Result
level_w_valves	Simulated	Unsafe

Challenges

- Interaction between transition guards/actions and mode invariants
- Software would buffer *for a while* if too many variables were added at once
- C2E2 documentation was not very helpful—lots of trial and error
- Poor warning/error messages made debugging a pain

Challenges

- Interaction between transition guards/actions and mode invariants
- Software would buffer *for a while* if too many variables were added at once
- C2E2 documentation was not very helpful—lots of trial and error
- Poor warning/error messages made debugging a pain

Next Steps

- Finish integrating the flow rate variables with a possible third '*idle*' state
- Add a *timer* variable to ensure that the valves don't change state too frequently
- Attempt to add chemical dosing [if time allows]

Questions?