

# TOWARDS GRAPHICAL CONFIGURATION INSIDE THE INTO-CPS APPLICATION

Christian Møldrup Legaard

Casper Thule

Peter Gorm Larsen

# AGENDA

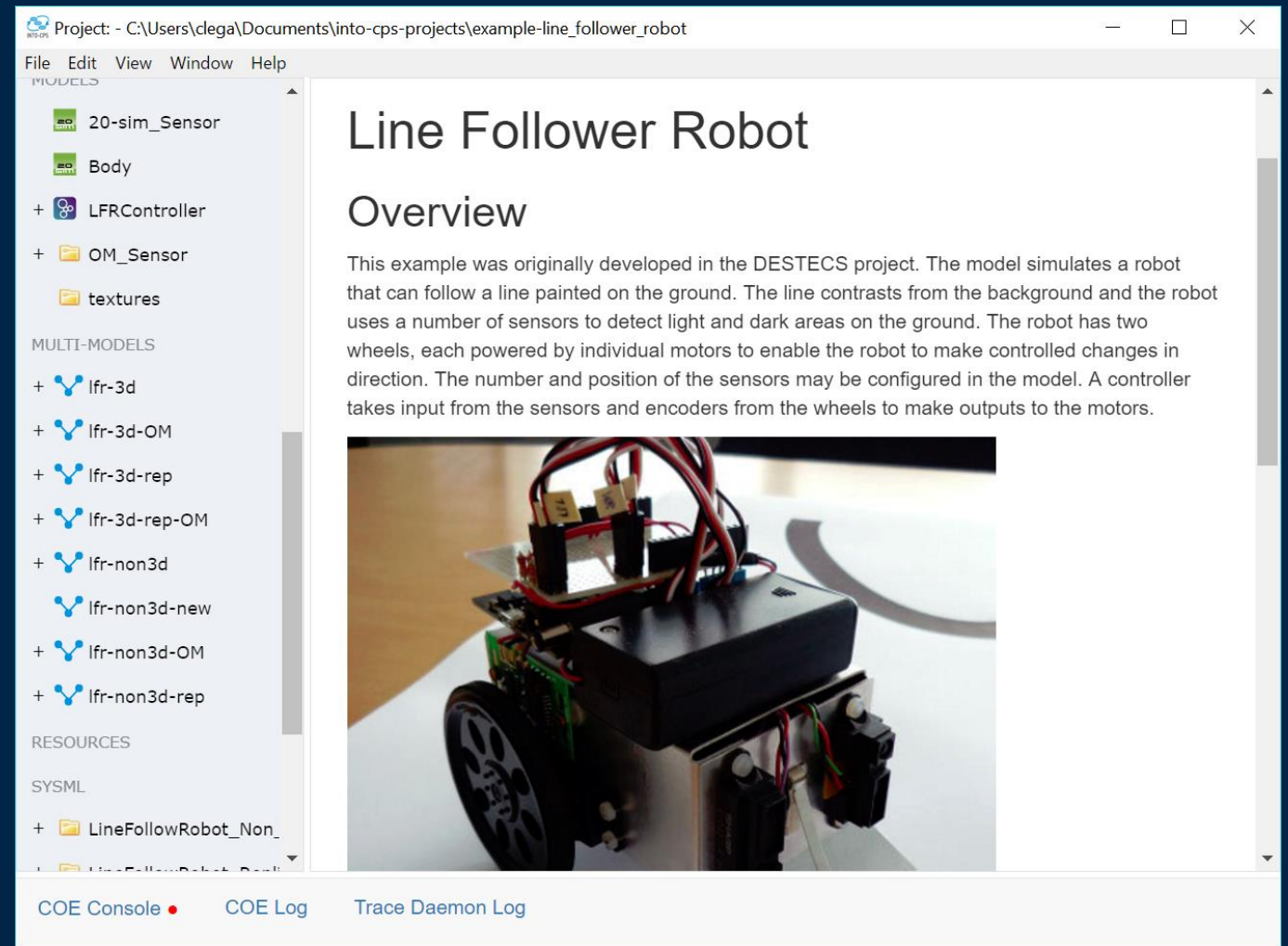
- 
- Background
  - Contribution and Demo
  - Future Work
  - Questions

# BACKGROUND – MOTIVATION

System Engineering Course

Powerful tooling

Frustrating User interface



The screenshot shows a software window titled "Project - C:\Users\clega\Documents\into-cps-projects\example-line\_follower\_robot". The interface is divided into several sections:

- MODELS:** A list of components including "20-sim\_Sensor", "Body", "LFRController", "OM\_Sensor", and "textures".
- MULTI-MODELS:** A list of model variants such as "lfr-3d", "lfr-3d-OM", "lfr-3d-rep", "lfr-3d-rep-OM", "lfr-non3d", "lfr-non3d-new", "lfr-non3d-OM", and "lfr-non3d-rep".
- RESOURCES:** A section for additional resources.
- SYSML:** A section for system modeling language files.

The main content area displays the title "Line Follower Robot" and an "Overview" section. The overview text states: "This example was originally developed in the DESTECs project. The model simulates a robot that can follow a line painted on the ground. The line contrasts from the background and the robot uses a number of sensors to detect light and dark areas on the ground. The robot has two wheels, each powered by individual motors to enable the robot to make controlled changes in direction. The number and position of the sensors may be configured in the model. A controller takes input from the sensors and encoders from the wheels to make outputs to the motors." Below the text is a photograph of a physical line follower robot with two wheels, a black top deck, and various sensors and wires.

At the bottom of the window, there are three tabs: "COE Console", "COE Log", and "Trace Daemon Log".

# BACKGROUND – CONNECTIONS INSIDE APPLICATION

Gets the job done

Limited overview

Find un-connected port?

The screenshot shows a software interface with a sidebar on the left and a main content area on the right. The sidebar is divided into sections: MODEL-CHECKING, MODELS, MULTI-MODELS, and RESOURCES. The MODELS section is expanded, showing a list of components including 20-sim\_Sensor, Body, LFRController, OM\_Sensor, textures, and a list of multi-models under 'lfr-3d'. The main content area is titled 'Connections' and displays a table with four columns: Output instance, Output variable, Input instance, and Input variable. The table lists various components and their associated variables, with some cells highlighted in blue. Below the table is a section for 'Initial values of parameters'.

Output instance	Output variable	Input instance	Input variable
{bodyFMU}.body	encoder_left_output	{bodyFMU}.body	<input checked="" type="checkbox"/> robot_theta
{sensor1FMU}.sensor1	encoder_right_output	{sensor1FMU}.sensor1	<input type="checkbox"/> robot_x
{sensor2FMU}.sensor2	robot_theta	{sensor2FMU}.sensor2	<input type="checkbox"/> robot_y
{3DFMU}.3D	robot_x	{3DFMU}.3D	<input type="checkbox"/> robot_z
{controllerFMU}.controller	robot_y		
	robot_z		
	total_energy_used		
	wheel_left_rotation		
	wheel_right_rotation		

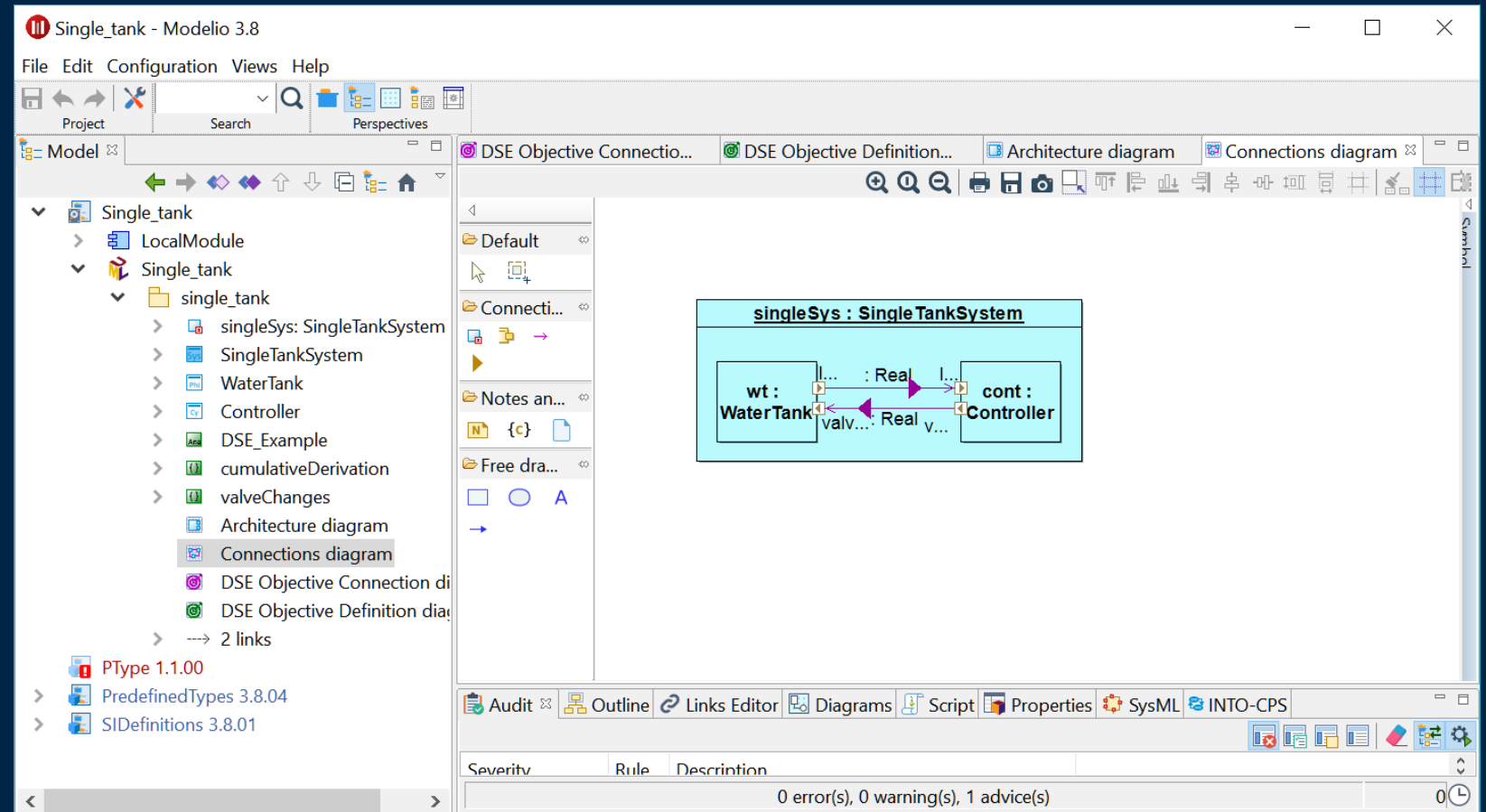
# BACKGROUND – SYSML PROFILE

Better overview

Familiar experience

Ease of use?

External

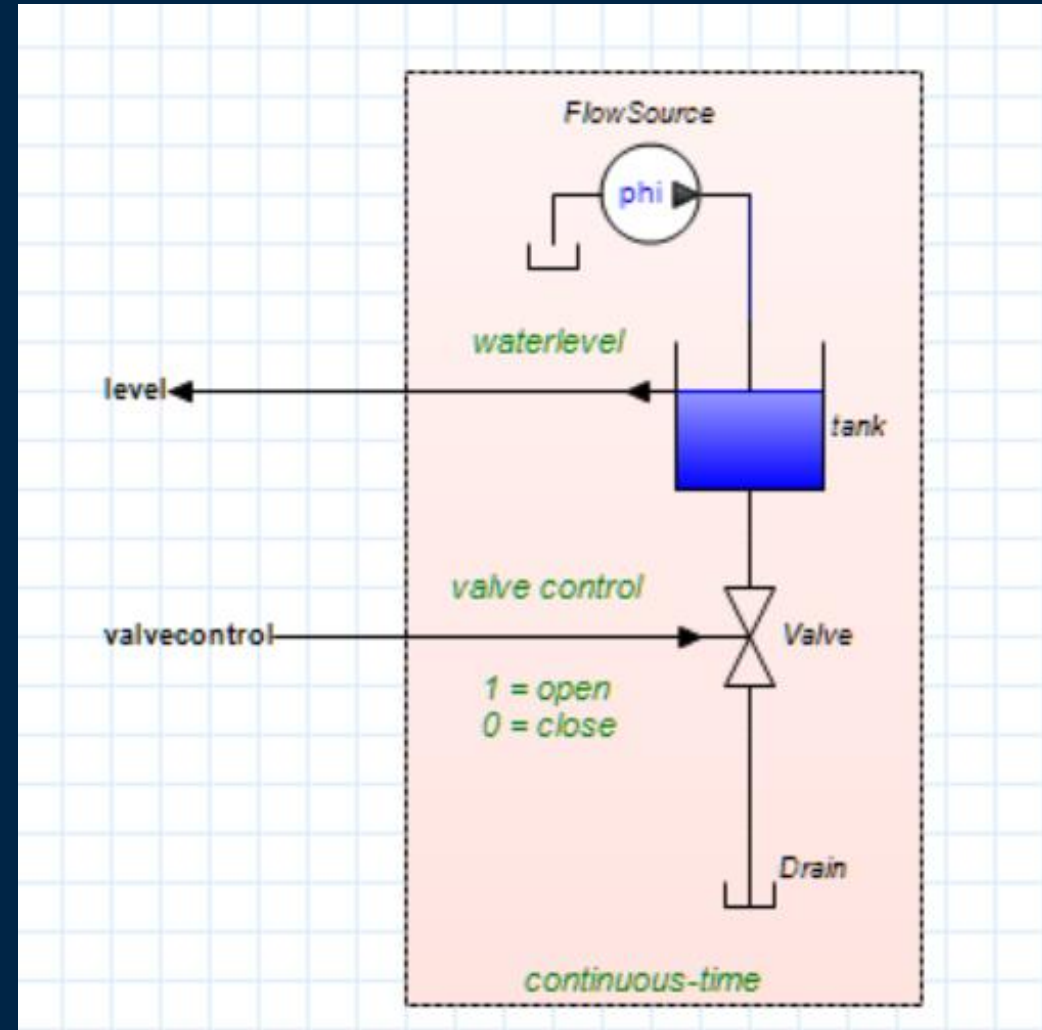


# BACKGROUND – OTHER TOOLS

20Sim, OMEDIT, Simulink ...

Integrated block-based design

Key difference to SysML profile?



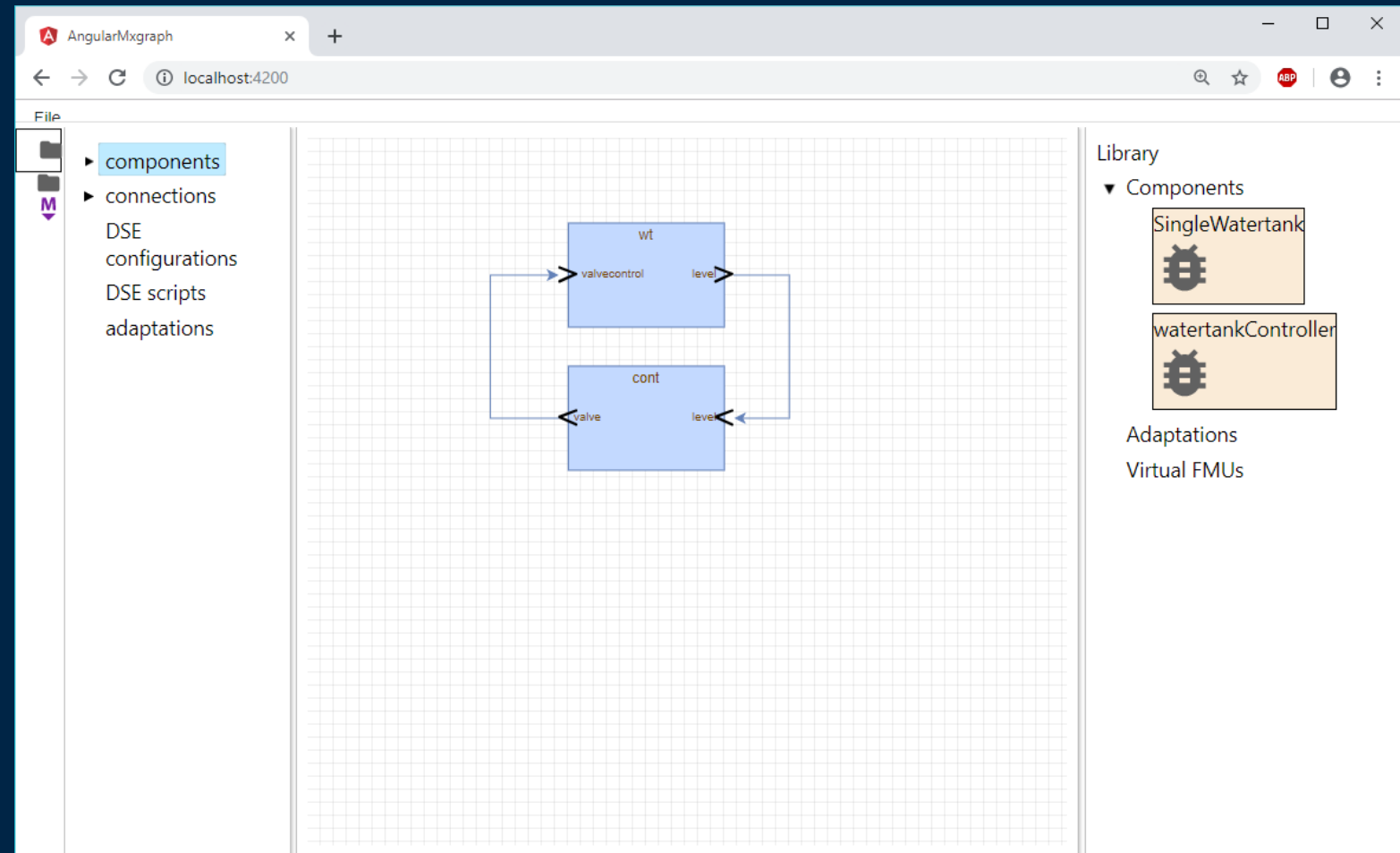
# CONTRIBUTION – GRAPHICAL EDITOR

Integrated Block-based editor

Faster development loop

User assistance

Future proof

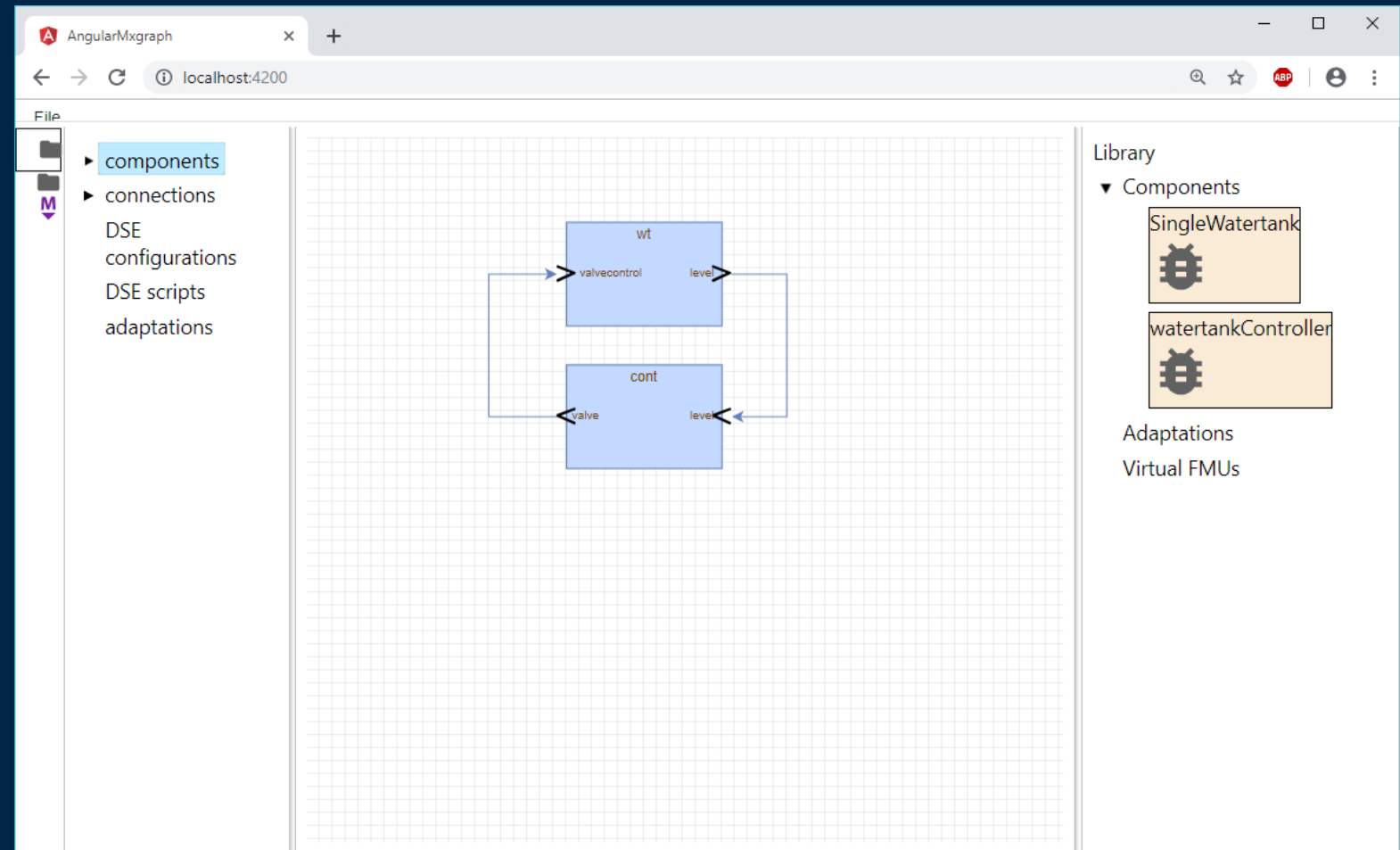


# CONTRIBUTION – GRAPHICAL EDITOR

Project explorer

Canvas

Library





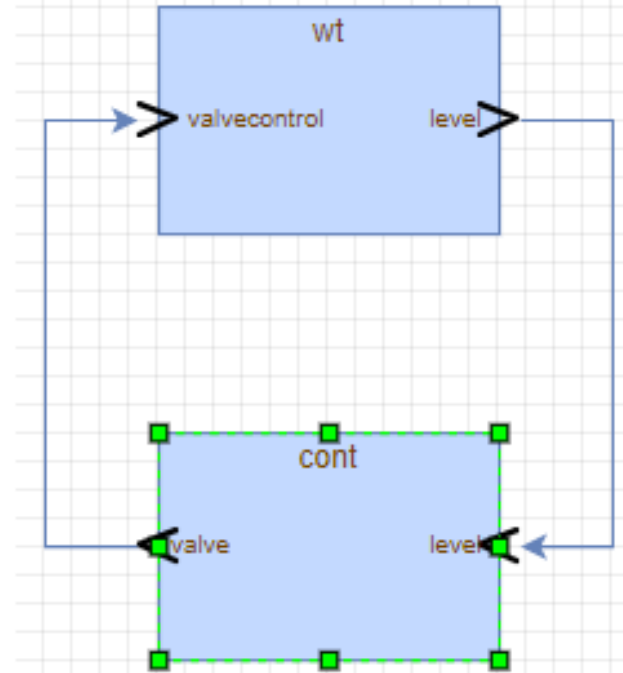
# GRAPHICAL EDITOR – PROJECT EXPLORER

Abstract representation of system

Overview of hierarchy

Also shows artefacts not “drawable” on canvas

- ▼ components
  - ▶ wt
  - ▶ cont
- ▼ connections
  - (wt.level) -> (cont.level)
  - (cont.valve) -> (wt.valvecontrol)
- DSE configurations
- DSE scripts
- adaptations

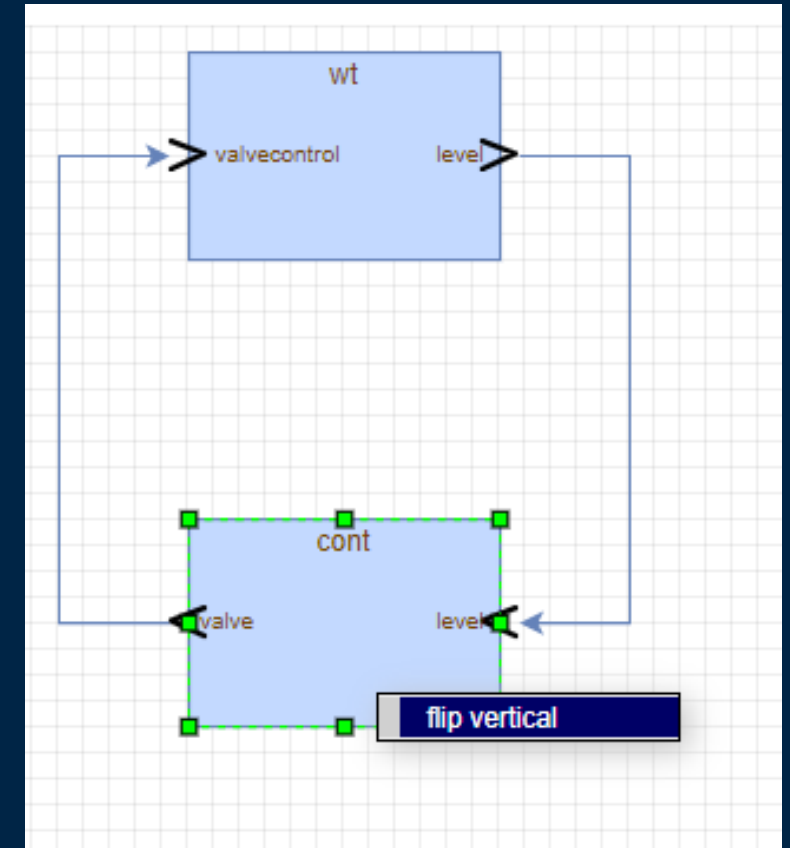
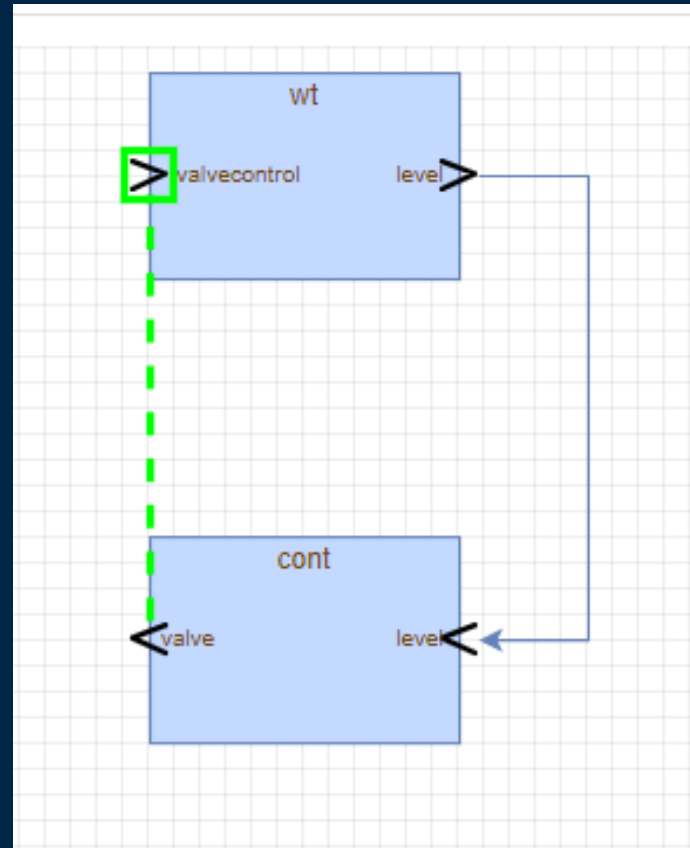


# GRAPHICAL EDITOR – CANVAS

—  
Drag connections

Double click to configure

Context menu



# GRAPHICAL EDITOR – PARAMETERS

Configuration of system

Instance specific vs shared

Cooping with large number of parameters

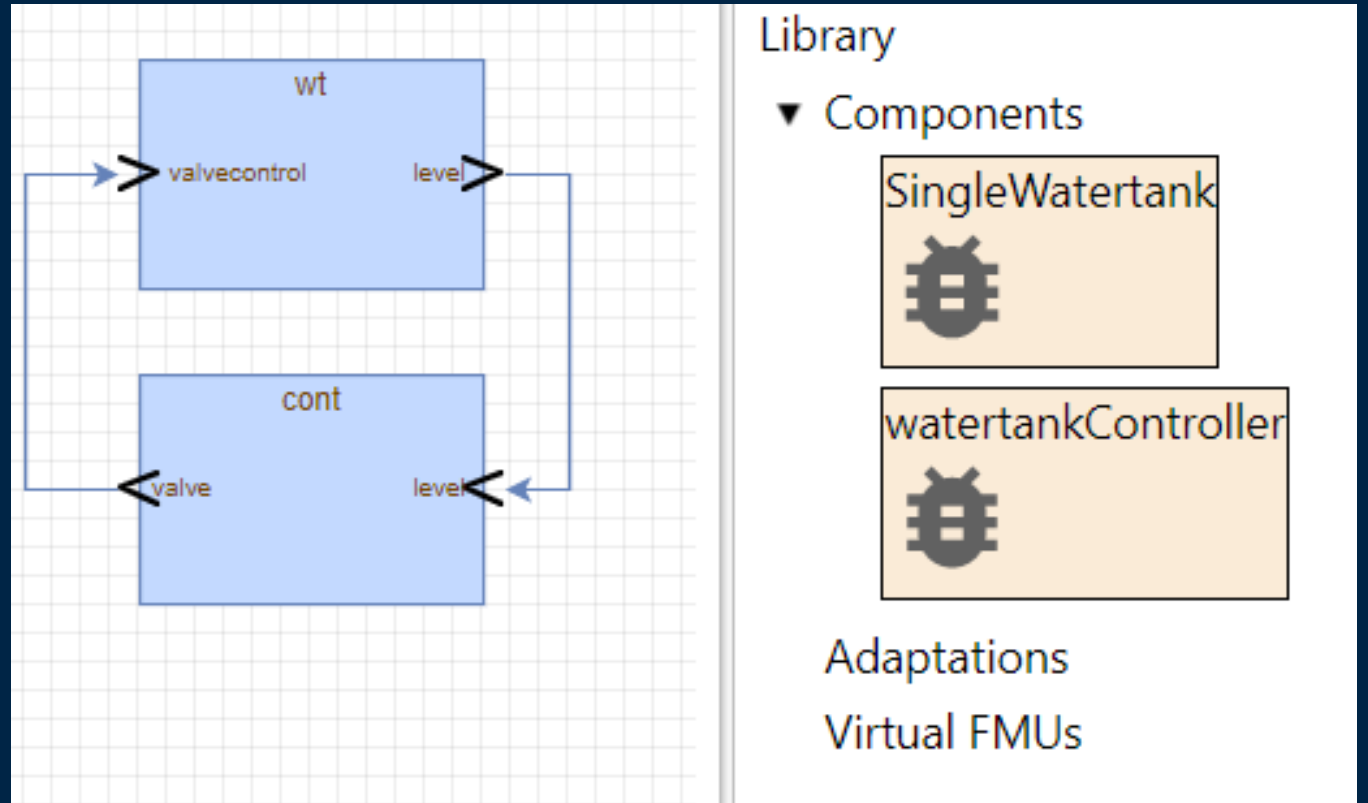
The image shows a graphical editor interface. On the left, a system configuration diagram is displayed on a grid background. It features two main components: a top block labeled 'wt' (water tank) and a bottom block labeled 'cont' (controller). The 'wt' block has a 'valvecontrol' input on the left and a 'level' output on the right. The 'cont' block has a 'valve' input on the left and a 'level' input on the right. Arrows indicate the flow of information: from 'wt' to 'cont' and from 'cont' back to 'wt'. On the right, a 'wt : Configuration' dialog box is open. It has two tabs: 'Information' and 'Parameters'. The 'Parameters' tab is active, showing a list of parameters with their current values. The parameters are: 'Drain.r' (9), 'FlowSource.phi' (1, with a 'flow rate' button), 'tank.Tank.area' (1), 'tank.Tank.gravity' (9.81), 'tank.Tank.liquid\_density' (1), and 'tank.Tank.volume\_initial' (0). Below the list are 'Valve.outflow\_int\_initial' (0) and another '0' value. At the bottom of the dialog are 'Cancel' and 'Save' buttons.

# GRAPHICAL EDITOR – LIBRARY

—  
Drag and Drop

Categories?

Potential sources?



# DEMO

# EXCHANGE FORMAT – LOOKING FORWARD

---

Why is this relevant?

Current format

Future format

# EXCHANGE FORMAT – CURRENT FORMAT

---

Simple JSON format

No geometry, shared parameters, hierarchy ...

Requires drilling into the FMU archives

```
"{"fmus": {"WaterTank": "singlewatertank-20sim.fmu", "Controller": "watertankController-Standalone.fmu"}, "connections": {"WaterTank.wt.level": [{"Controller}.cont.level"}, {"Controller}.cont.valve": [{"WaterTank}.wt.valvecontrol"}], "parameters": {"Controller}.cont.maxlevel": 2, {"Controller}.cont.minlevel": 1}}"
```

# EXCHANGE FORMAT – FUTURE

System Structure And  
Parameterization (SSP)

FMI and SSP

What is an component?

Extension Mechanisms

```
\---single_tank.ssp      : collection of multi models
| SystemStructure.ssd   : default multi model
| VarA.ssd               : alternative multi model
| ParameterSet.ssv      : set of parameters
| ParameterMappings.ssm : binding params to instances
| SignalDictionary.ssb  : mechanism to group signals
|
+---documentation
| index.html
|
+---extra                : extra files extension mechanism
|
\---resources            : implementations of components
  WaterTank.fmu
  Controller.fmu
  subsystem.ssp         : potential subsystem
```



# FUTURE WORK

---

Integration into application(s)

Exchange Format

Adaptations

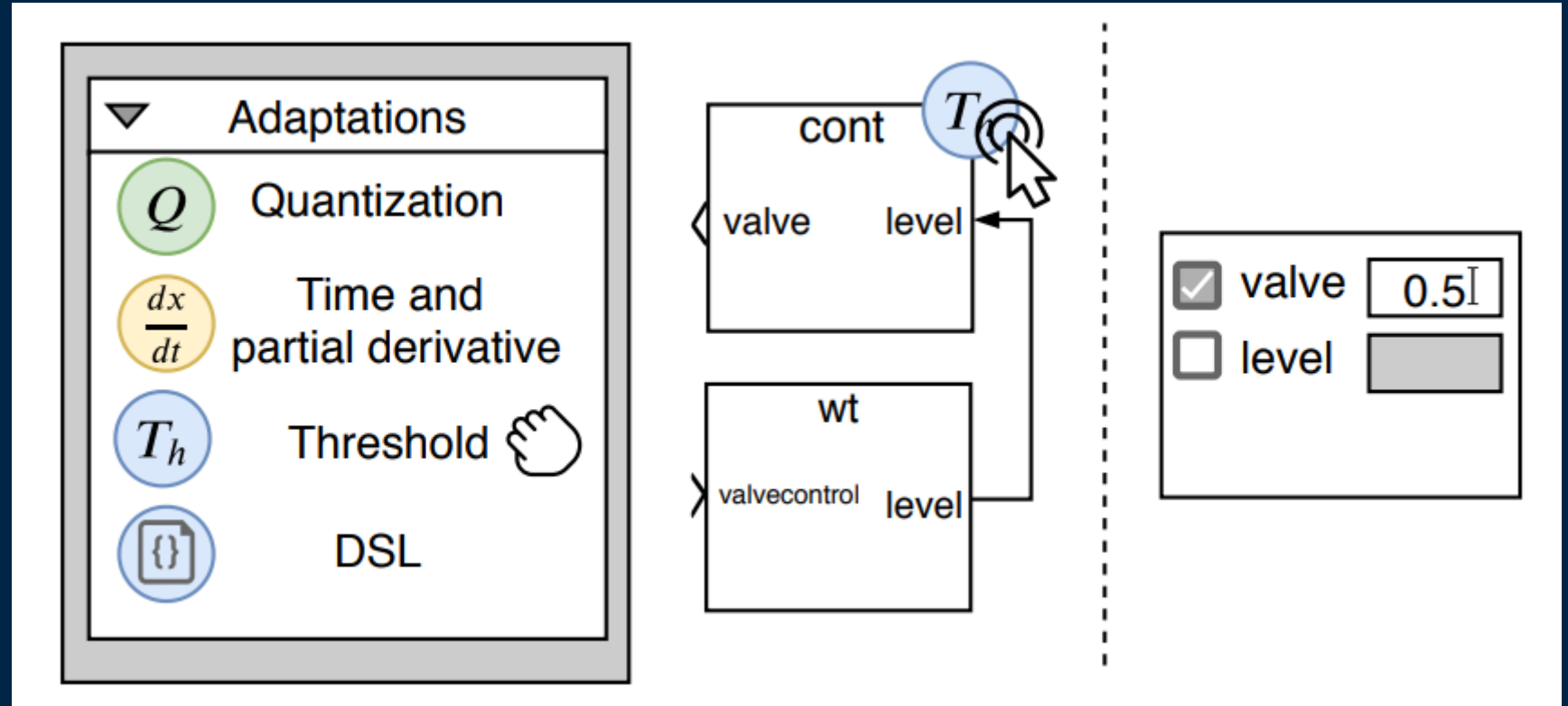
Hierarchy

# FUTURE WORK – SEMANTIC ADAPTATION

Fix mismatches

Wrapper

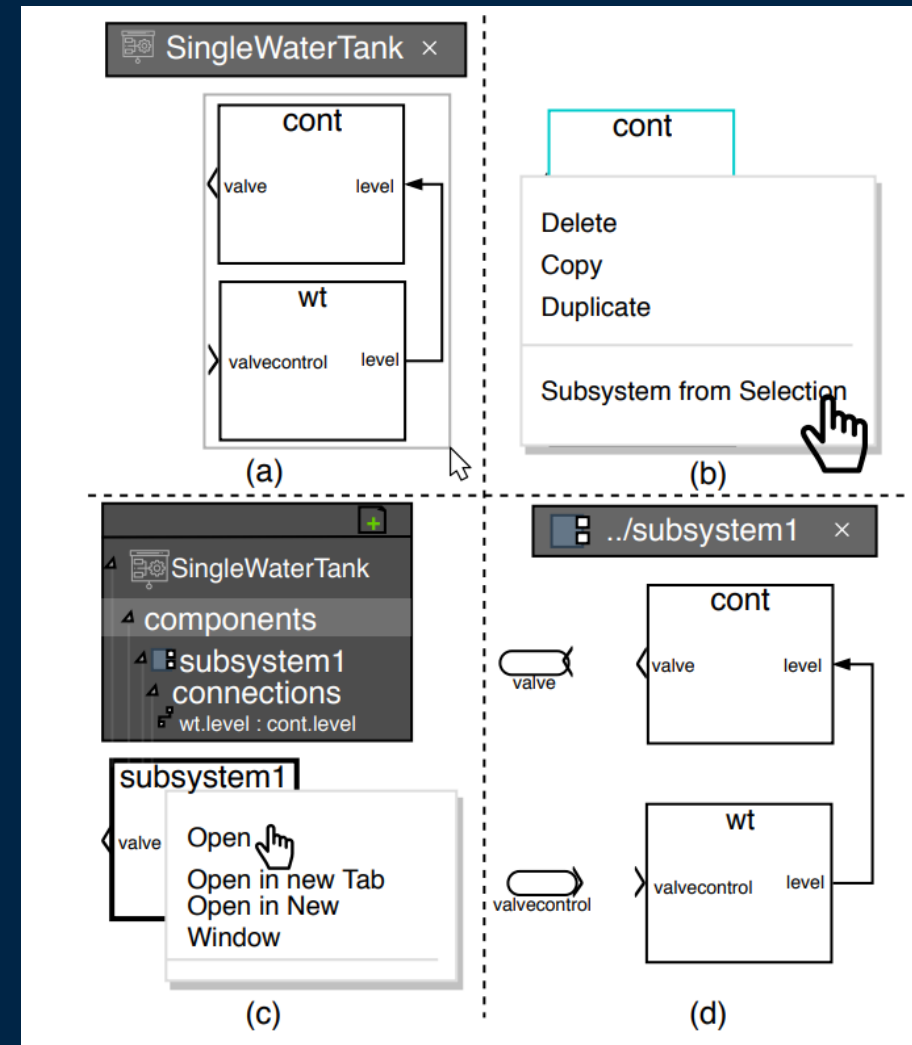
DSL vs graphical?



# FUTURE WORK – HIERARCHY

Complexity Reduction

Relation to adaptations



# ACKNOWLEDGEMENTS

---

Poul Due Jensen Foundation

INTO-CPS Project/Association

System Structure and  
Parameterization



AARHUS  
UNIVERSITY