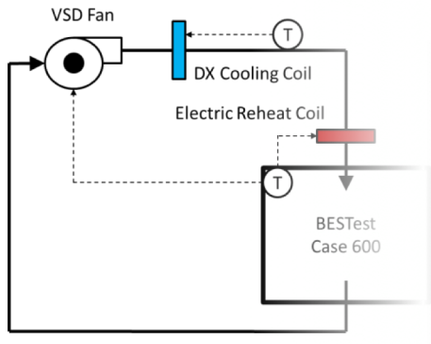
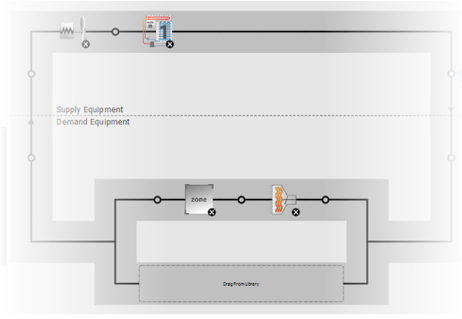
OpenBuildingControl

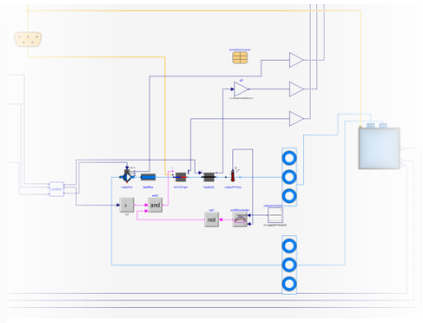
https://obc.lbl.gov

Michael Wetter, Jianjun Hu, Milica Grahovac, Philip Haves, Paul Ehrlich

June 11, 2018







Design ***Model*** ***Simulate*** ***Operate***

BERKELEY LAB

Lawrence Berkeley National Laboratory

Problem Statement

- No generic design tools for control strategies
- No tools to generate sequences of operation from strategies
- No way to ensure, or verify, that sequences get implemented correctly

Other industries have these tools, and depend on them.

Goals and Objectives

Develop a process and a set of tools to enable:

- Design of effective strategies and sequences
- Automated generation of vendor-specific code
- Verification of correct implementation
- Traceability from design to installation and operation, including changes

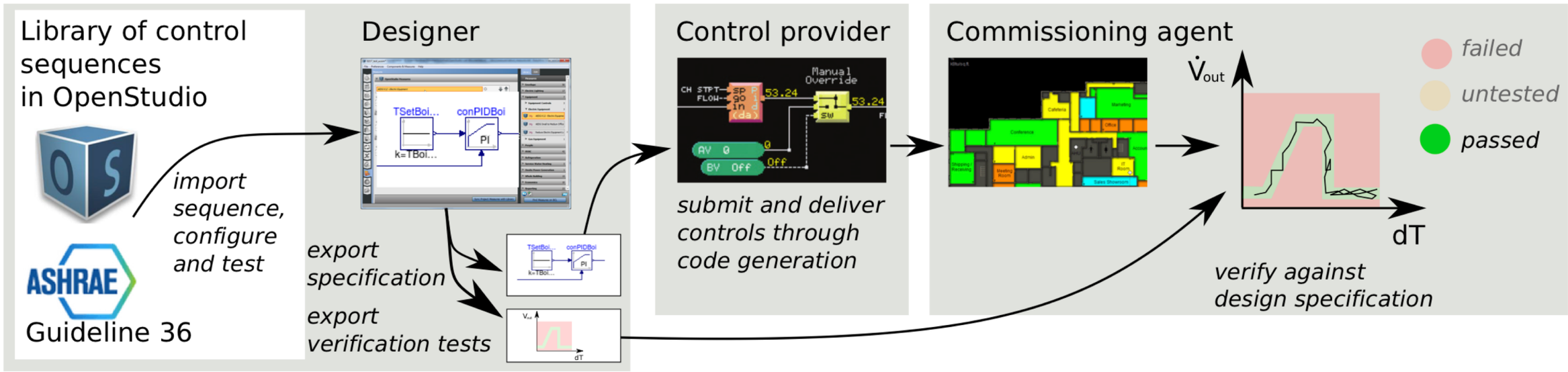
Key characteristics:

- Open standards
- Open source
- Vendor neutral

Long term vision: automated control system design, including:

- Component selection
- Network configuration

OpenBuildingControl: Design and implement control sequences error-free and at lower cost to owner



Codify best practice	Design	Implement	Verify against original design

BACnet standardizes communication.

OpenBuildingControl will standardize:

- basic functional building blocks that are used to compose sequences and tests,
- expressing control sequences,
- expressing functional verification tests,

for bidding, automatic implementation and automated functional testing.

Challenges

- Define a technical process based on viable business model(s) quickly enough to specify tool requirements
- Controls design tool:
  - Computationally efficient modeling of local loop control in a whole building model
  - Projects with no envelope + systems model
- Controls specification language:
  - Vendor-neutral but easily mapped to proprietary products
  - Extensible to support new technology (e.g. MPC) and new control system architectures
- Engage critical mass of vendors, designers, contractors and owners to ensure wide adoption

Team and Structure

LBNL: project management, software development

- **Partners:**
  - Arup
  - CBRE
  - ControlCo
  - (Google)
  - Integral Group
  - kW Engineering
  - Oracle
  - Stanford
- **Subcontractors:**
  - Integral Group: specification and testing of standard sequences
  - Arup: process definition, testing and demonstration, GUI spec, international
  - (PNNL: project organization)
  - Taylor Engineering: specification and testing review, ASHRAE
  - Facility Dynamics Engineering: TAG chair, review

Technical Advisory Group and ASHRAE GPC 36

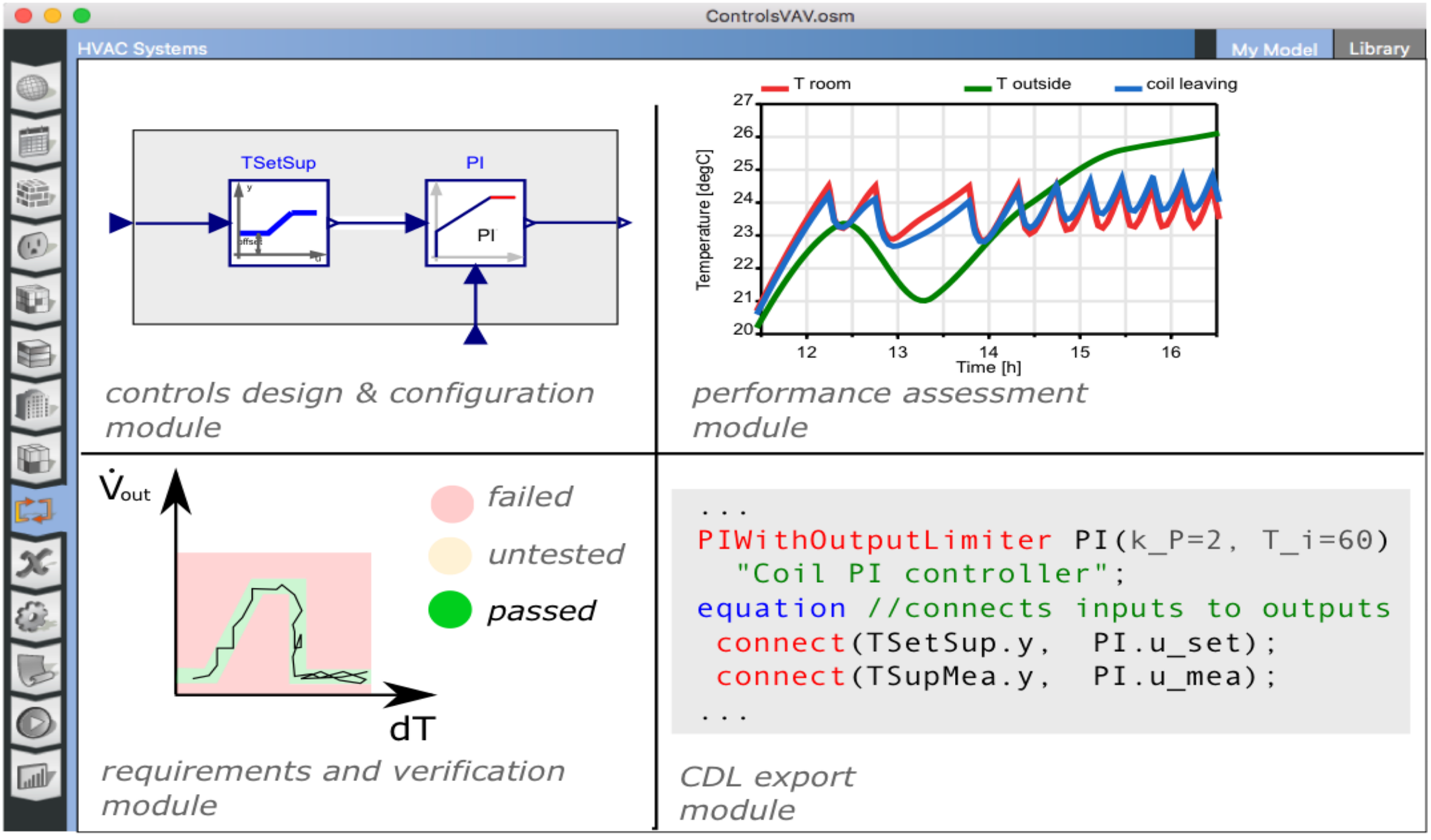
TAG:

- Vendors and other stakeholders
- Technical and business feedback
- Chair: Jay Santos (FDE)
- Quarterly meetings
- 1st meeting: 2/2/2017

ASHRAE GPC 36:

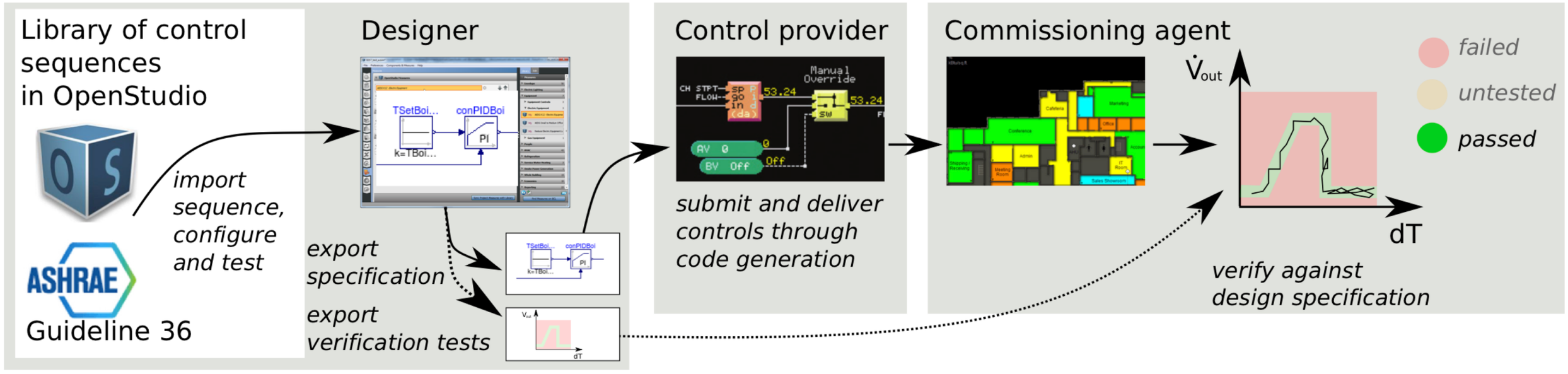
- Chair: Mark Hydeman (Google)
- Multiple members from project
- Two-way feedback→

OpenBuildingControl: Design and implement control sequences error-free and at lower cost to owner



Points list + Bidding documents + Operator manual + …

Control Description Language



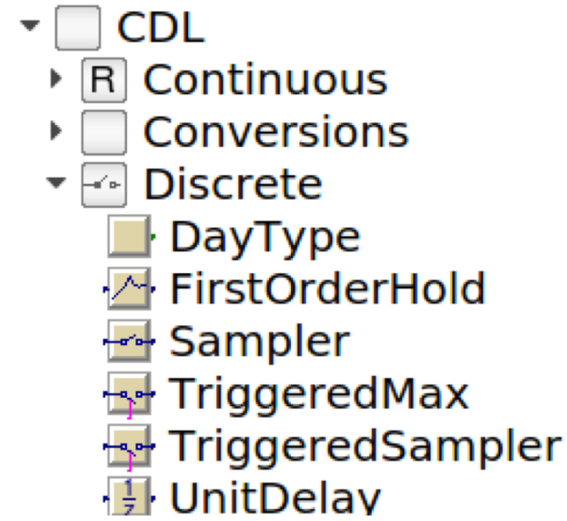
For specifications:
http://obc.lbl.gov/specification/cdl.html#sec-cdl

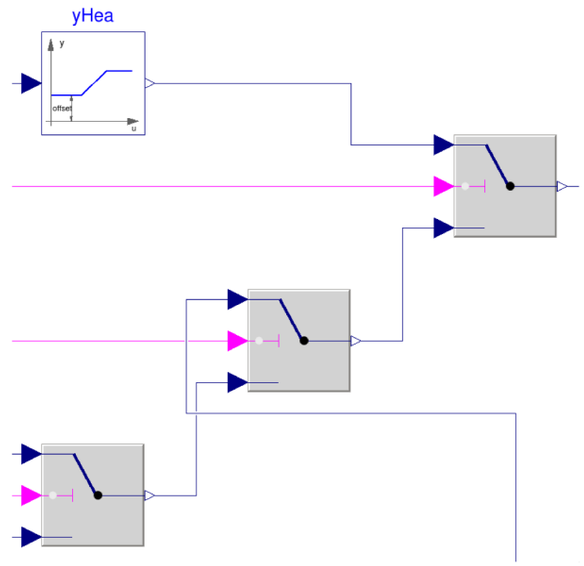
To browse the blocks:
http://simulationresearch.lbl.gov/modelica/releases/latest/help/Buildings_Controls_OBC_CDL.html

Control Description Language, connecting formally design, bidding, implementation & operation

The Control Description Language (CDL) consists of

- A library with elementary input/output blocks that should be supported [through a translator] by CDL-compliant control providers.
- A declarative, open-standard, open-source, non-vendor-specific, language for expressing block-diagrams for control sequences.
- A language for rendering these diagrams.
- A syntax for documenting the control blocks and diagrams.
- A model of computation that describes the interaction among the blocks.





Rather than an ambiguous English Word specification against which one cannot test, we now have (i) English language documentation, (ii) block diagram representation, (iii) code that can be executed and that conforms to an open modeling standard (a subset of Modelica)

Current state of CDL

CDL has been specified

CDL library has been implemented

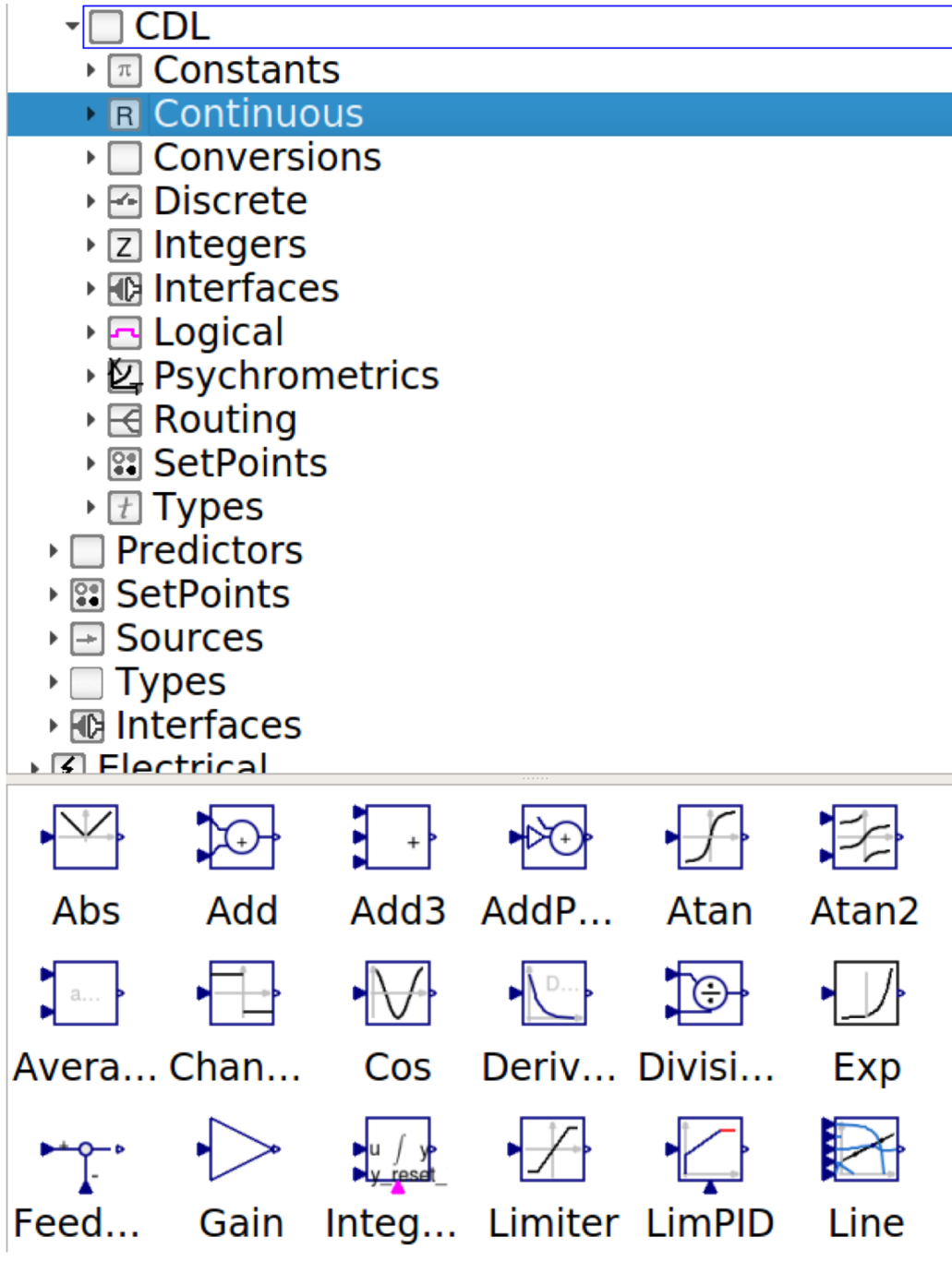
CDL-compliant Guideline 36 sequences have been released and demonstrated

CDL-based code translation to Building Automation System has been shared with control companies to develop translators

In development:

- Refine CDL export to JSON and HTML for code generation and English language documentation
- Verification tool
- Sequences for primary system

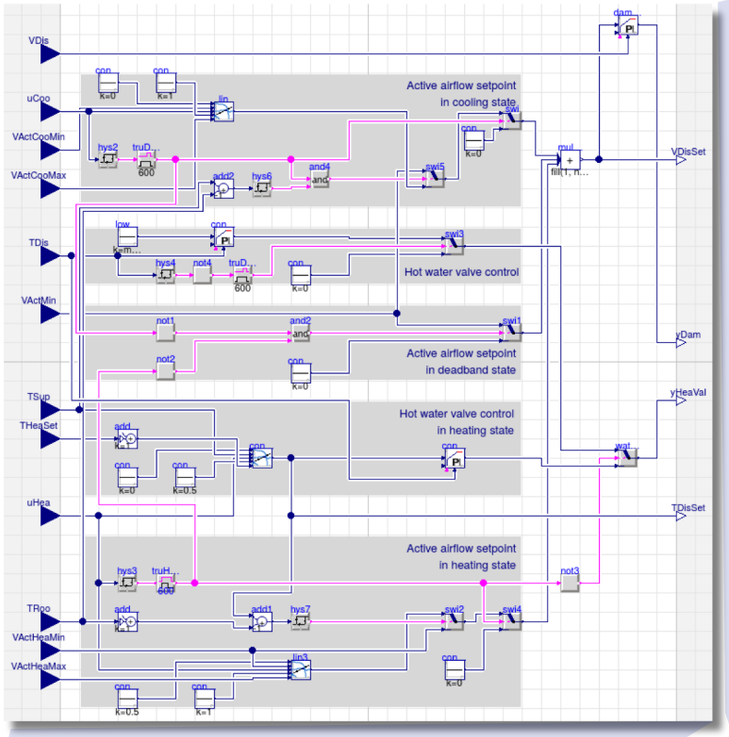
For CDL specification, see
http://obc.lbl.gov/specification/cdl.html



Partial view of CDL library with elementary blocks.

Example implementation of an ASHRAE Guideline 36 sequence

Block-diagram view.



Output signals for controlling VAV reheat box damper and valve position

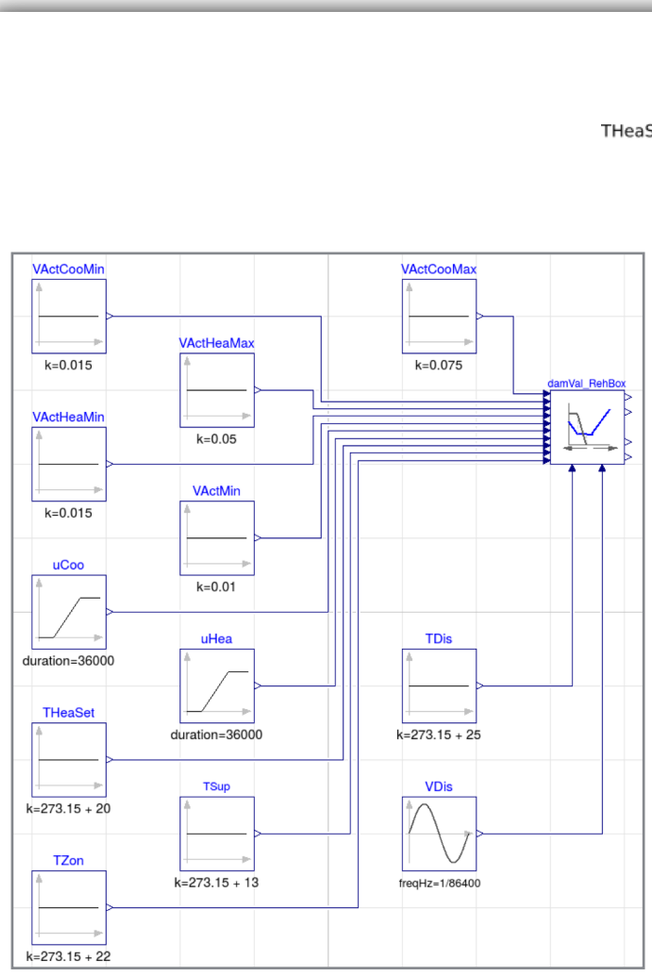
Information

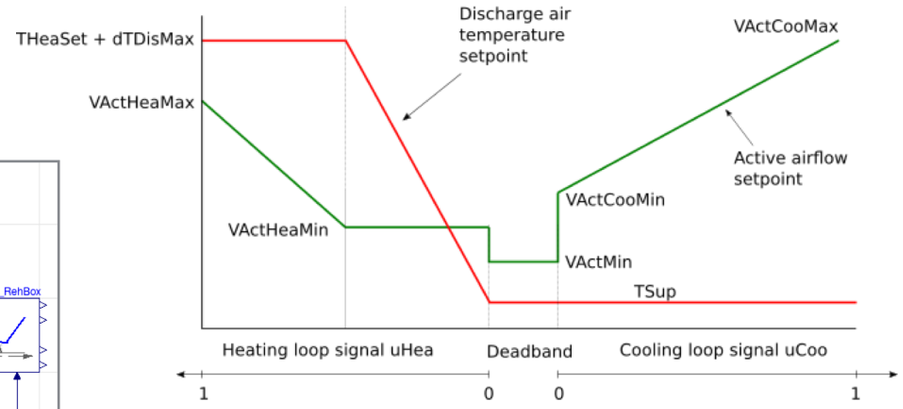
This sequence sets the damper and valve position for VAV reheat terminal unit. The implementation is according to ASHRAE Guideline 36 (G36), PART5.E.6. The calculation is done following the steps below.

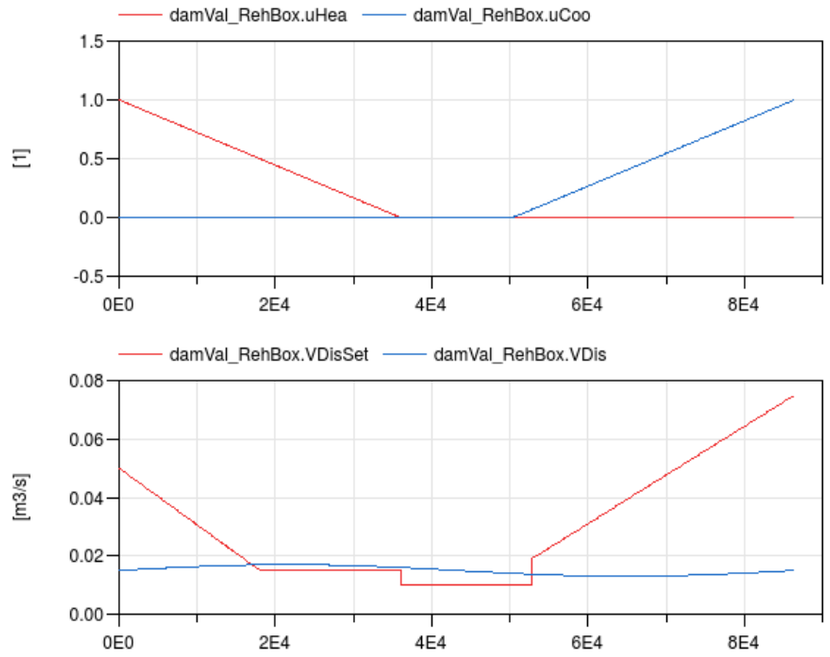
1. When the zone state is cooling (uCoo>0), then the cooling loop output uCoo shall be mapped to the airflow setpoint from the cooling minimum VActCooMin to the cooling maximum VActCooMax airflow setpoints. The hot water valve is closed (yHeaVal=0) unless the discharge air temperature TDis is below the minimum setpoint (10 °C).
2. If supply air temperature TSup from the AHU is greater than room temperature TRoo, cooling supply airflow setpoint shall be no higher than the minimum.
3. When the zone state is Deadband (uCoo=0 and uHea=0), then the active airflow setpoint shall be the minimum airflow setpoint VActMin. Hot water valve is closed unless the discharge air temperature is below the minimum setpoint (10 °C).
4. When the zone state is Heating (uHea>0), then the heating loop shall maintain space temperature at the heating setpoint as follows:
   - From 0-50%, the heating loop output uHea shall reset the discharge temperature setpoint from current AHU SAT setpoint TSup to a maximum of maxDTem above space temperature setpoint. The airflow setpoint shall be the heating minimum VActHeaMin.
   - From 50-100%, if the discharge air temperature TDis is greater than room temperature plus 2.8 Kelvin, the heating loop output uHea shall reset the airflow setpoint from the heating minimum airflow setpoint VActHeaMin to the heating maximum airflow setpoint VActHeaMax.
5. The hot water valve (or modulating electric heating coil) shall be modulated to maintain the discharge temperature at setpoint.
6. The VAV damper shall be modulated by a control loop to maintain the measured airflow at the active setpoint.

The sequences of controlling damper and valve position for VAV reheat terminal unit are described in the following figure below.

Autogenerated documentation.







Examples and validation tests.

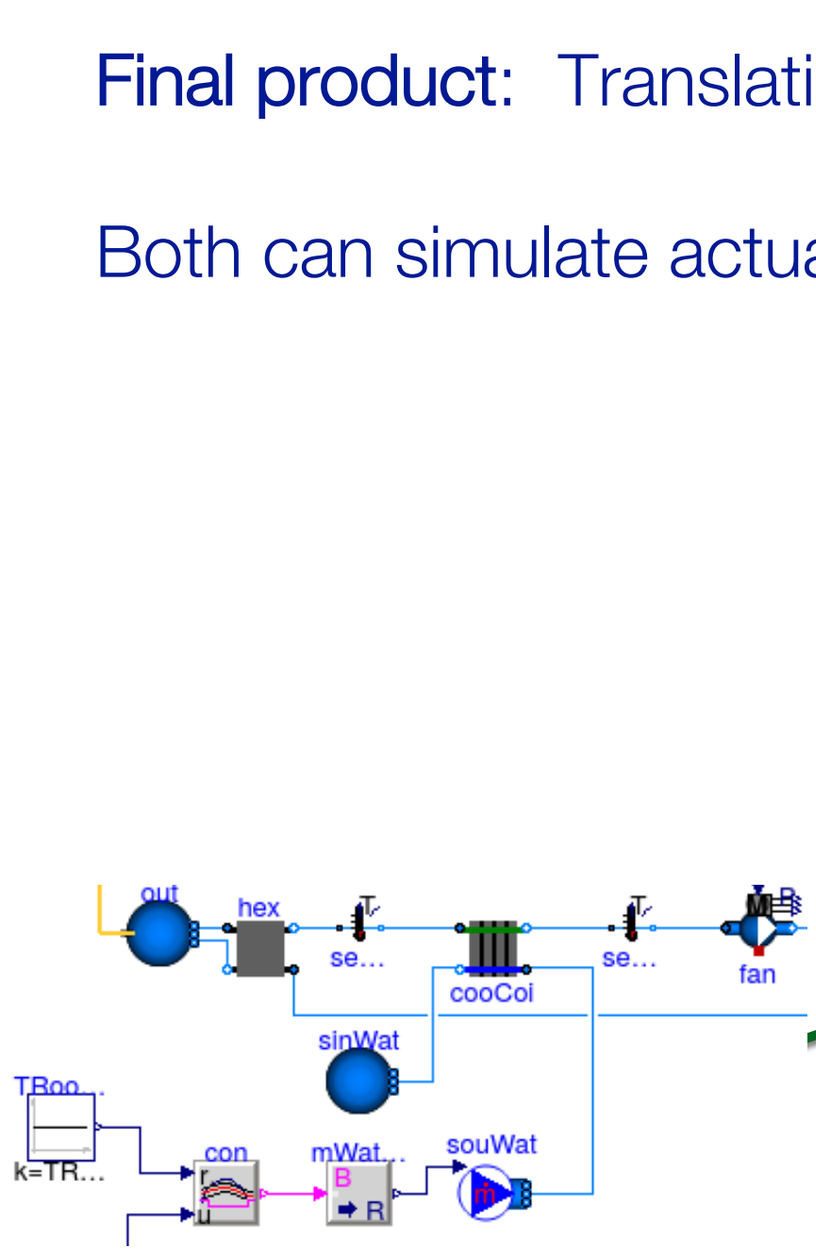
Buildings.Controls.OBC.ASHRAE.G36_PR1.TerminalUnits.Reheat.DamperValve

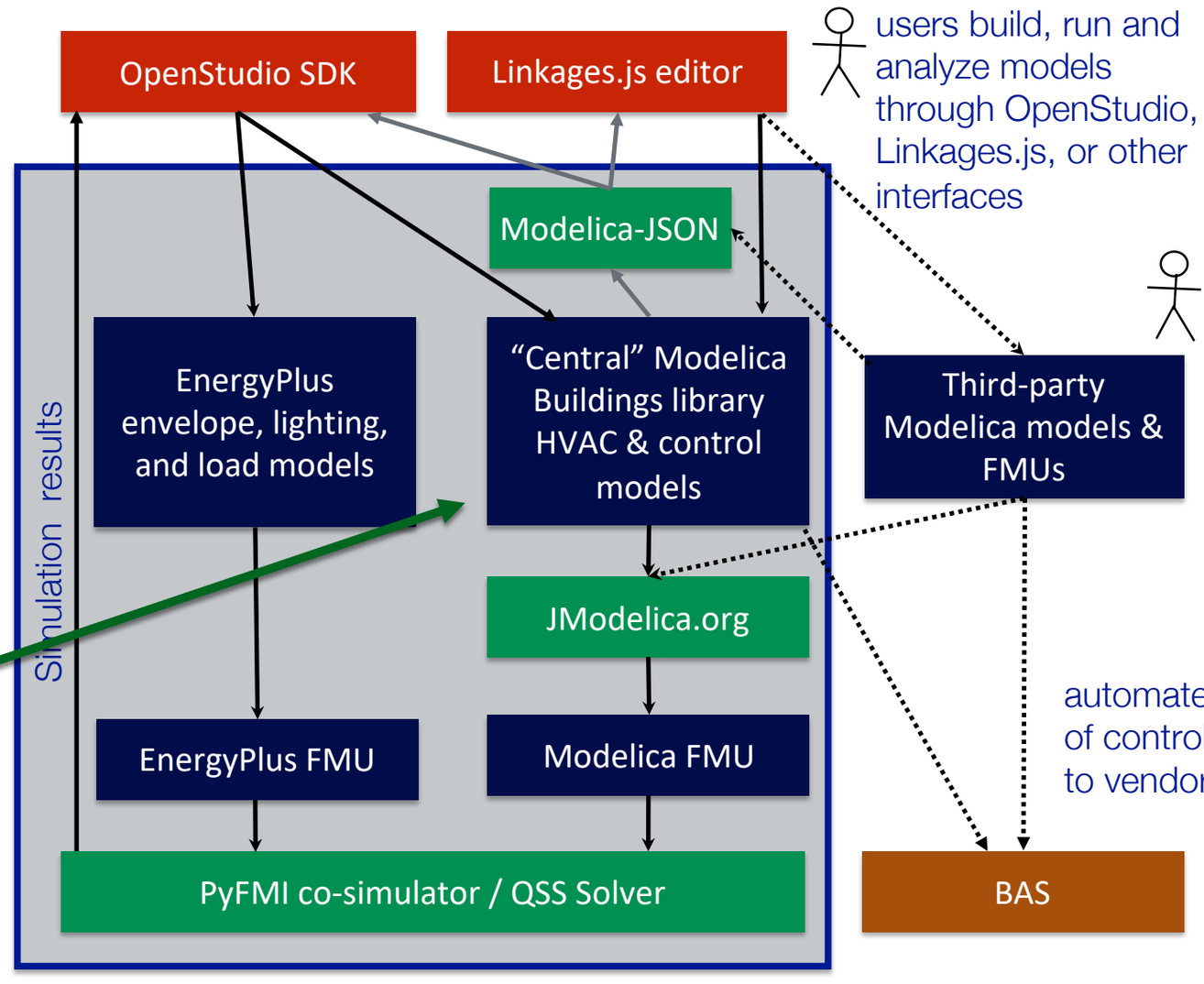
Performance assessment of control sequence with energy simulation model in the loop

Currently: Simulation using JModelica or Dymola

Final product: Translation and simulation through "Spawn of EnergyPlus"

Both can simulate actual control logic, using any time step.



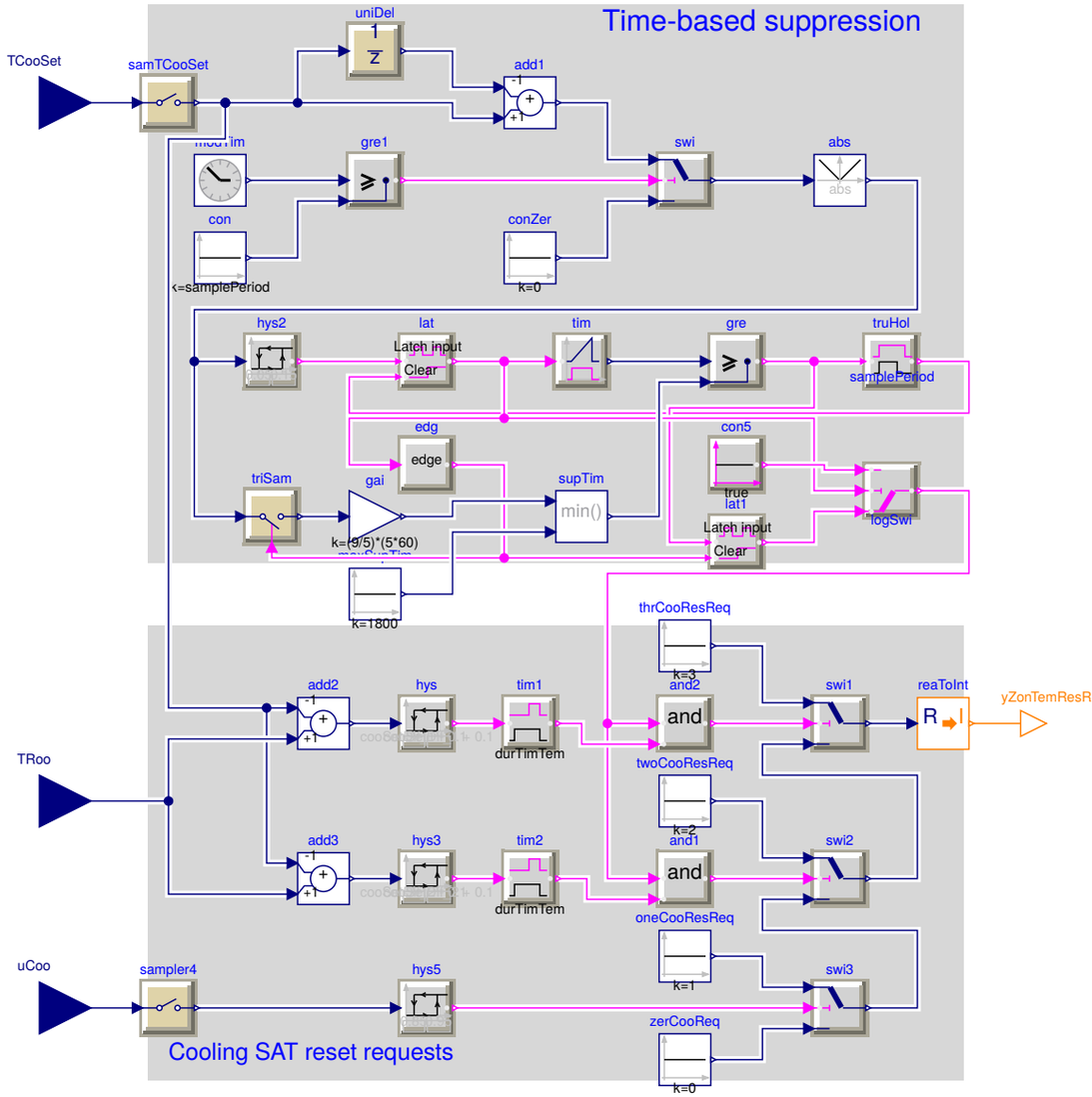


Case study #1

For report, see
http://obc.lbl.gov/specification/example.html or
https://github.com/lbl-srg/obc/tree/master/meetings/2017-11-tag

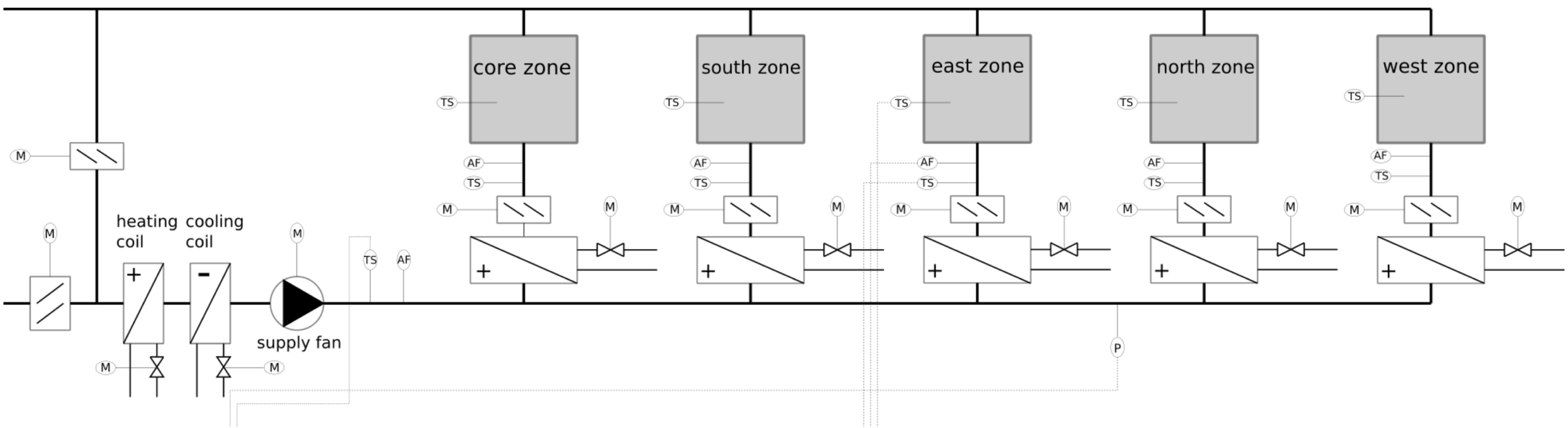
Case study:
Modeling multi-zone VAV controls and equipment

- Full airflow network.
- Wind pressure driven infiltration.
- All flows based on flow friction, damper positions and fan curves.
- 4,000 components, 40,000 variables, adaptive time step, state/time events.



Static pressure reset requests

Hot water reset requests

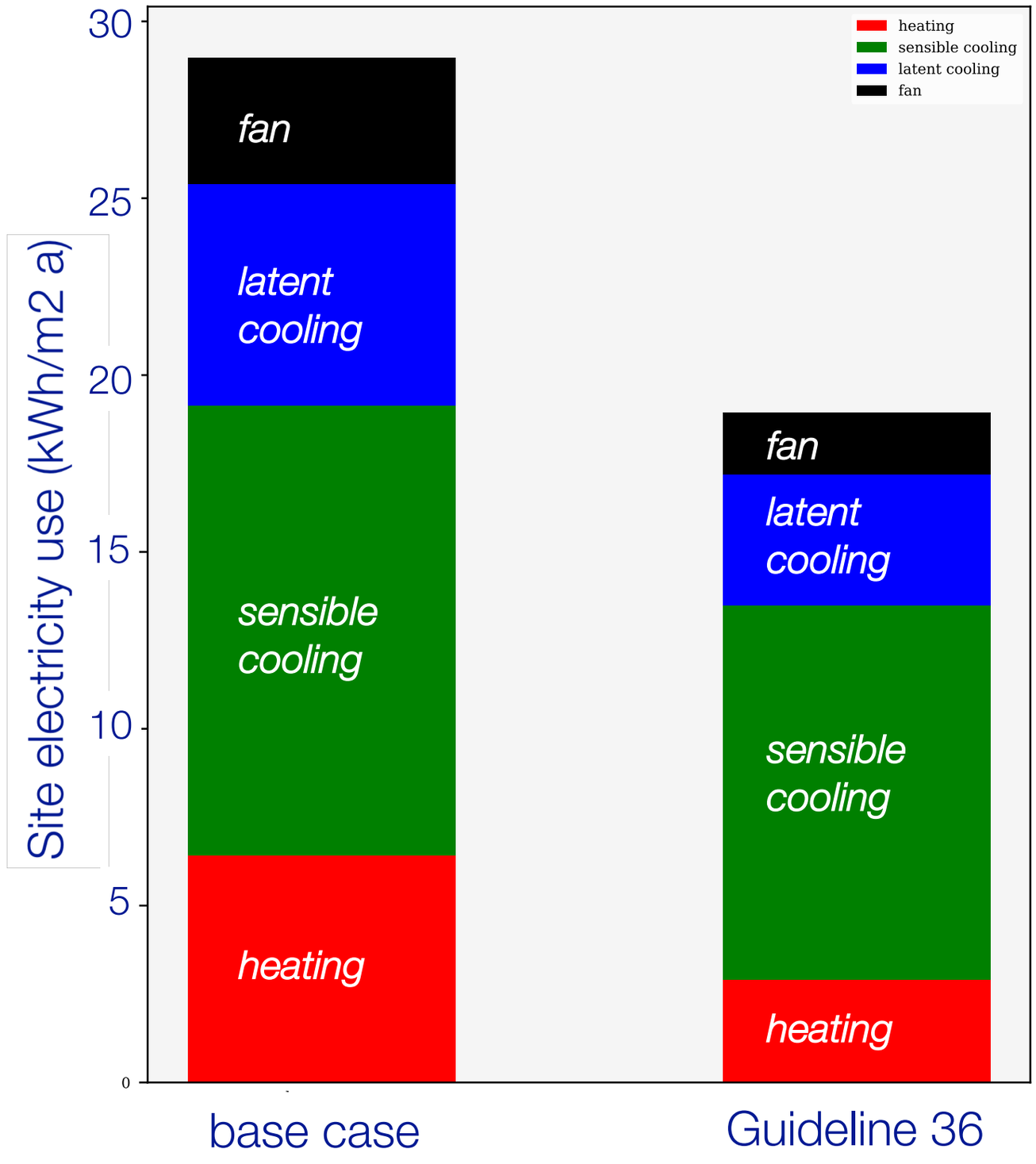


Key take-aways

~30% annual site HVAC energy savings for Chicago, solely due to controls.

Can simulate actual control sequences, with dynamic response.

Packaging of sequences is important, because interpretation and implementation of the sequences was more time-consuming and error-prone than anticipated.



Benefits of OBC approach and tools

Mechanical designer:

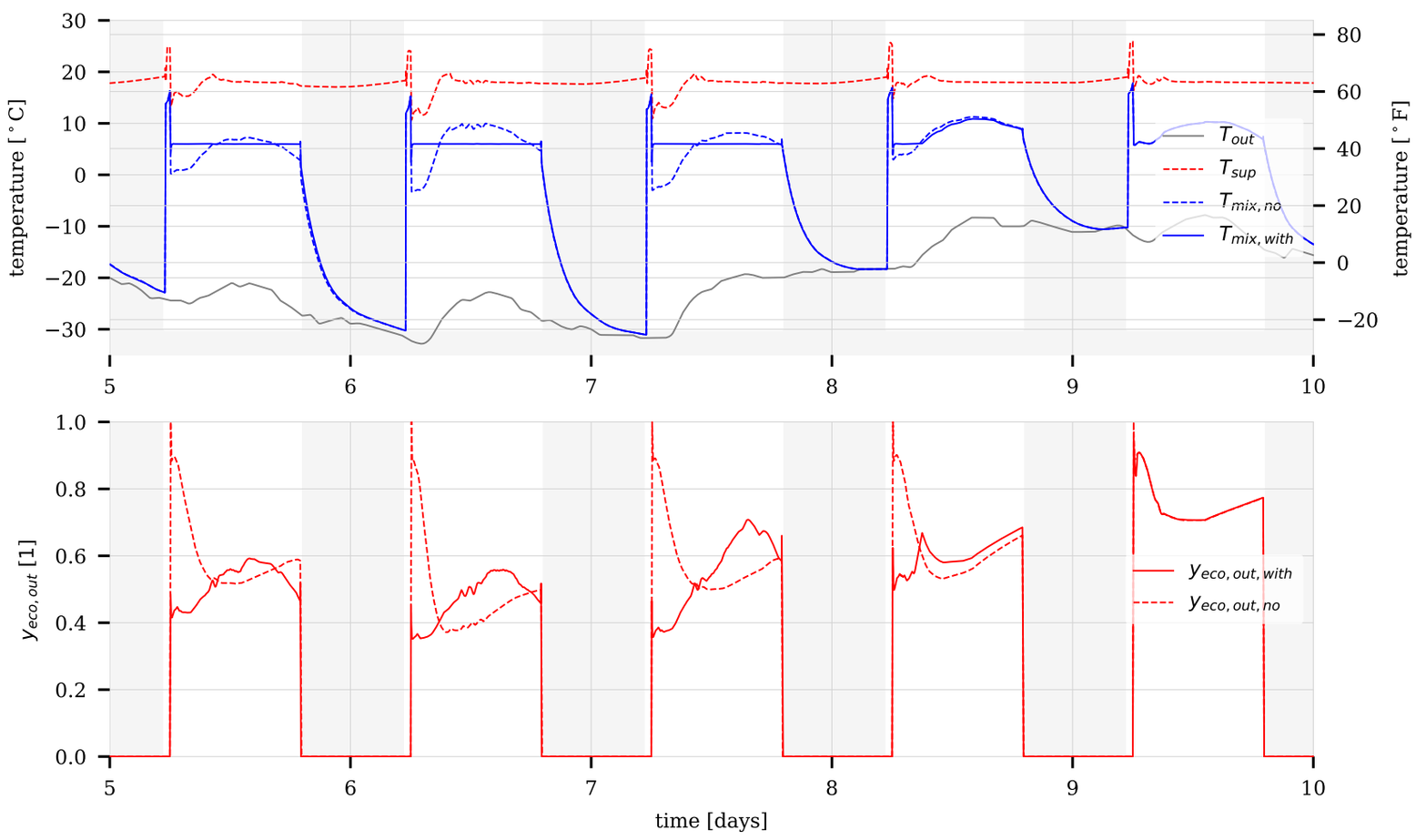
- HVAC design analysis tool
  - Equipment and controls
- Adapt and test sequences for particular building

Controls provider:

- Faster, higher quality, error-free automated implementation

GPC36:

- Formal way to test, compare and deploy sequences



Possible future uses (beyond the current project)

- Publish sequences as **reference implementations** against which vendor implementations can be compared.
- Use as **benchmark for comparing/rating advanced control sequences** such as MPC.
- **Verify G36 savings are robust** through performance assessment across different climates.
- Use OBC to test and prioritize future sequences to be added to G36.
- Potential to work with ASHRAE to move **CDL into a standard**.

Deployment of sequences

Process flow

Mechanical designer

Q&A-based sequence selector (such as CtrlSpecBuilder)

CDL libraries
- G36
- GSA
- custom-libraries

CDL sequences & mechanical system model

CDL representation to bid for job, and for implementation of real control system

Verification tests

Control provider

Control vendor bids for project, and later implements sequence semi-automatically

Commissioning agent

Commissioning agent verifies that design intent is met

Questions?

Comments?

Backup

CDL library

Developed CDL library based on review of industrial control libraries.

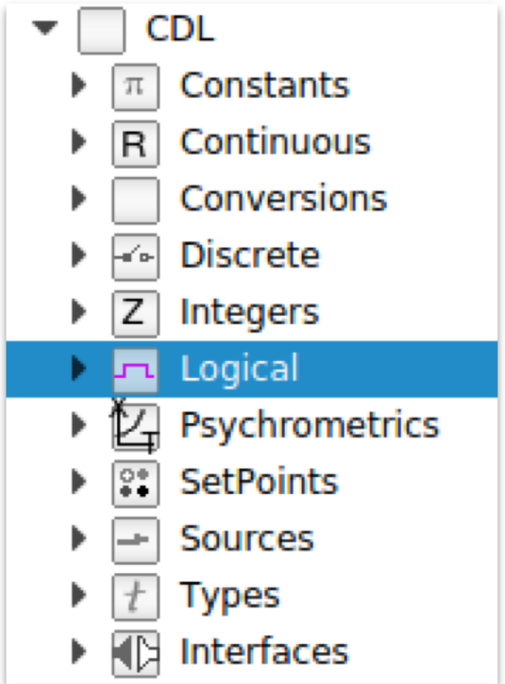
Validated blocks to ensure expected functionalities

- against known results
- across two independent simulators (Dymola and JModelica)

Package name	Description
Constants	Library of constants
Continuous	Library with elementary mathematical functions for continuous variables
Conversions	Library with blocks for type conversion
Discrete	Library of discrete input/output blocks with fixed sample period
Integers	Library with elementary mathematical functions for integer variables
Logical	Library with logical blocks
Psychrometrics	Library with psychrometric blocks
Routing	Package of blocks to combine and extract signals
SetPoints	Package with models for control set points
Types	Package with type definitions
Interfaces	Library with connectors for input and output signals

In CDL library:

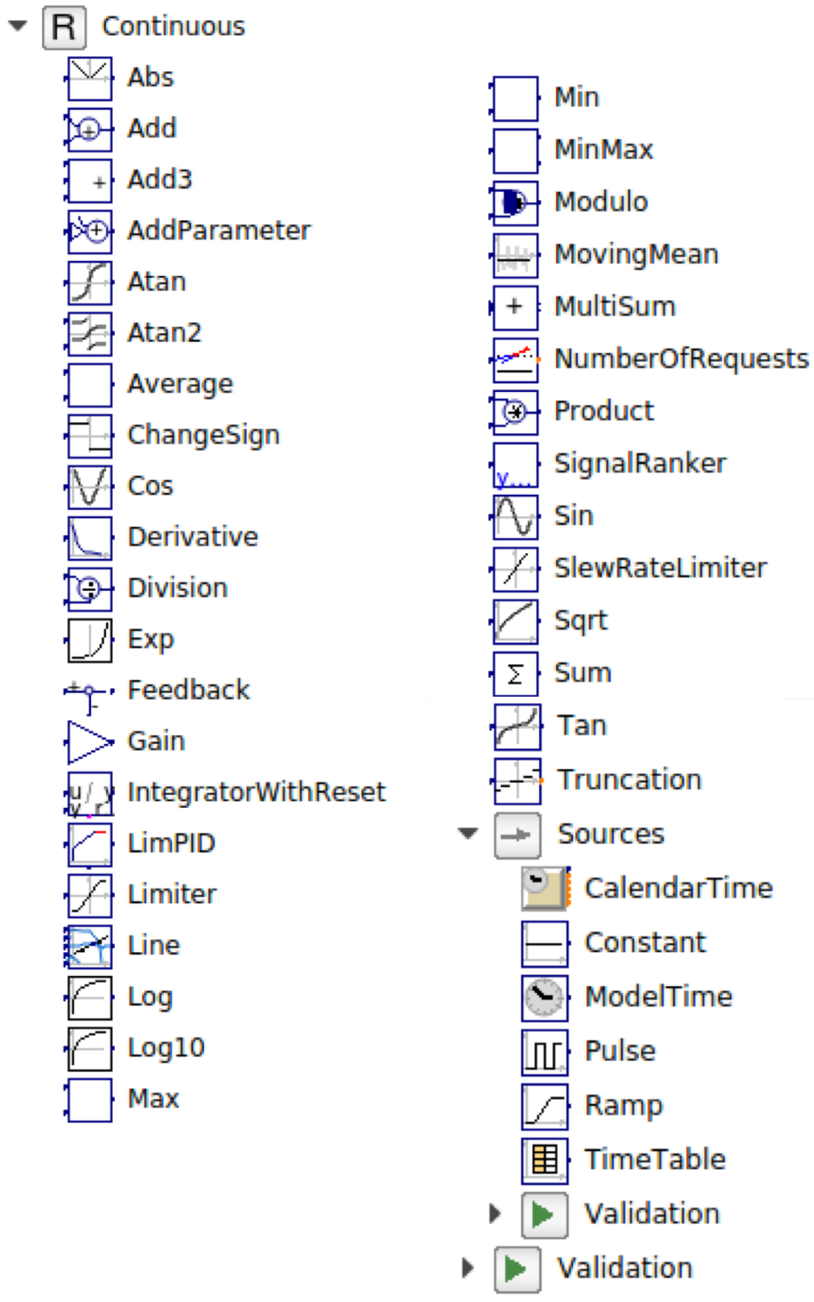
- 11 packages
- 134 basic blocks



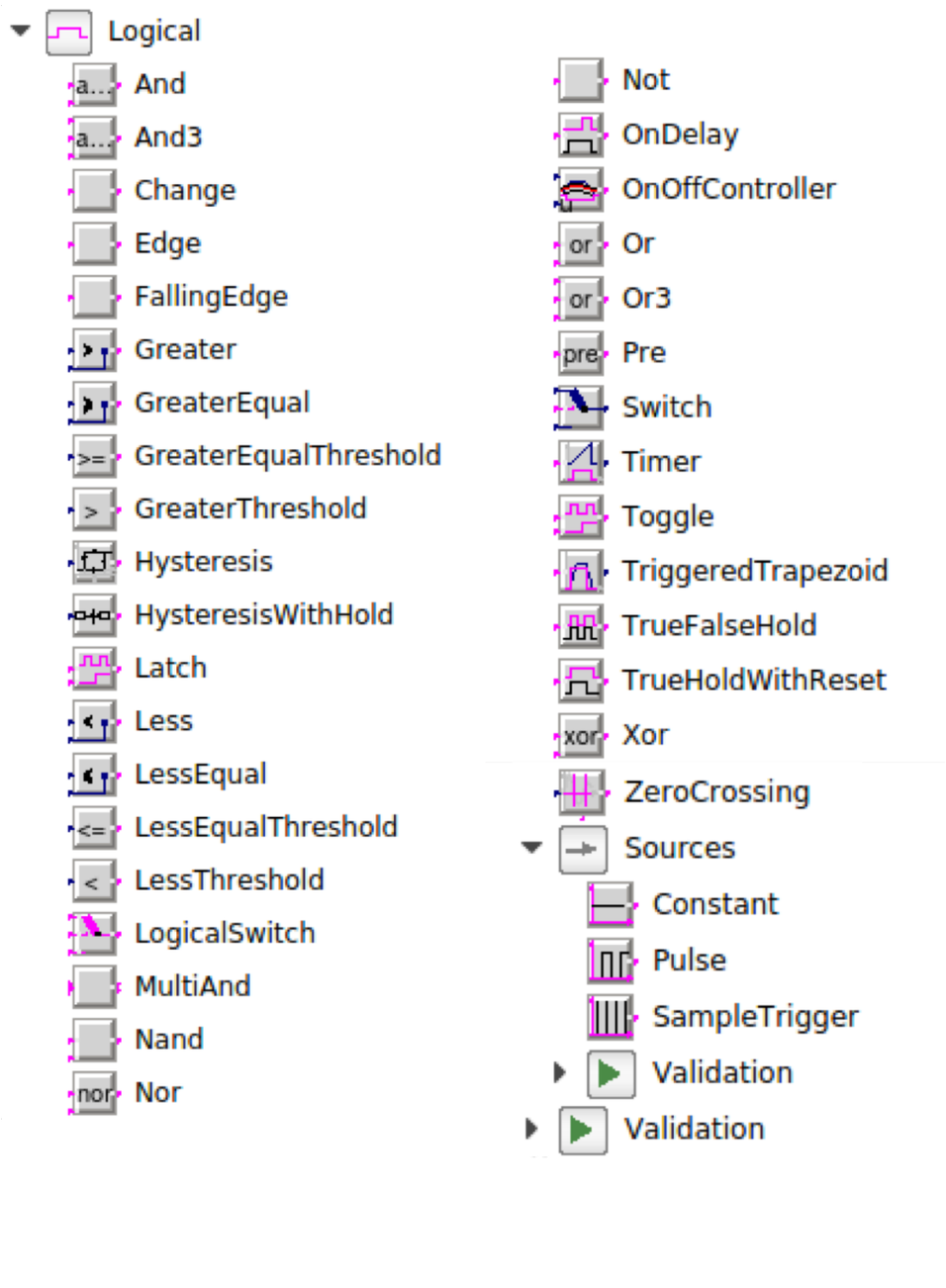
Browse CDL library at
http://simulationresearch.lbl.gov/modelica/releases/v5.0.0/help/Buildings_Controls_OBC_CDL.html

CDL library: Packages

`CDL.Continuous`:
elementary mathematical functions for continuous variables

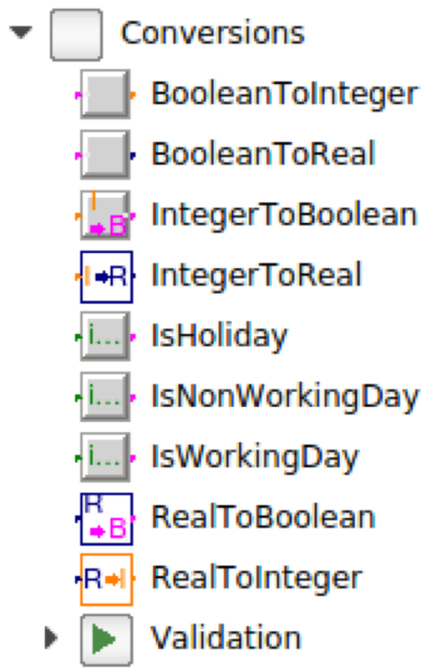


`CDL.Logical`:
elementary mathematical functions for boolean variables

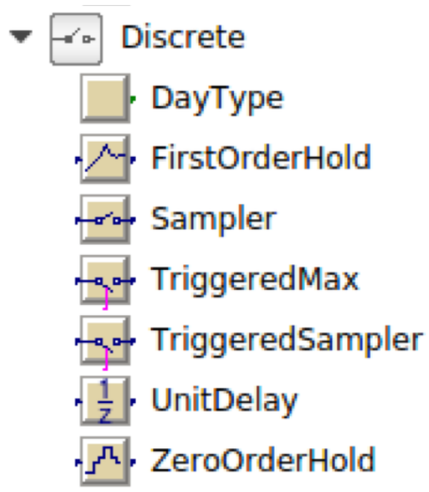


CDL library: Packages

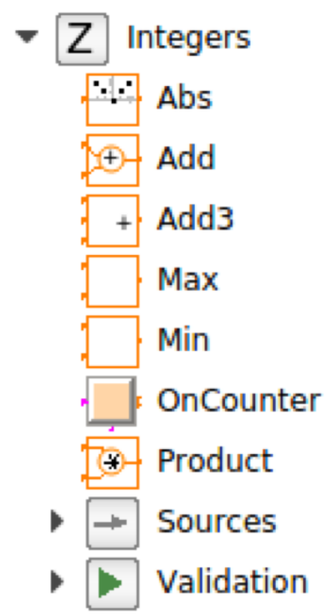
`CDL.Conversions`:
type conversions



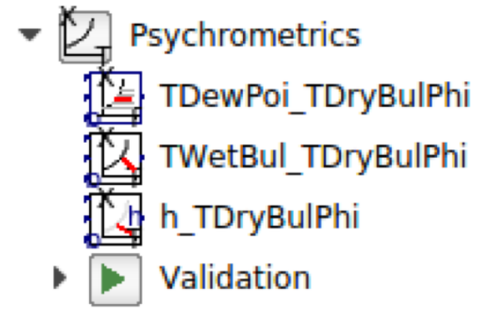
`CDL.Discrete`:
daytype, sample, delay, hold



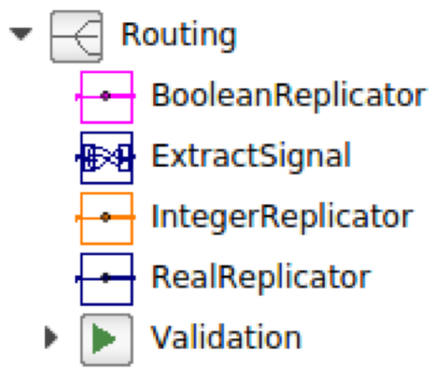
`CDL.Integers`:
mathematical functions for integer variables



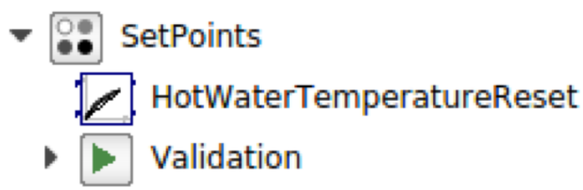
`CDL.Psychrometrics`:
psychrometric calculations



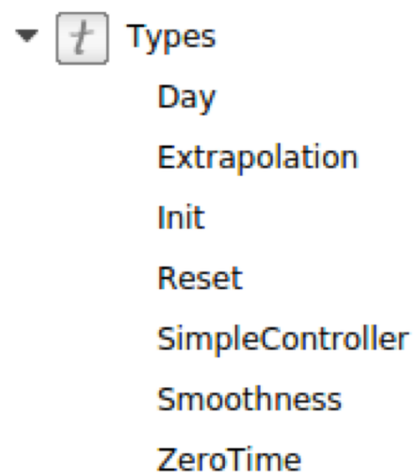
`CDL.Routing`:
combine and extract signals



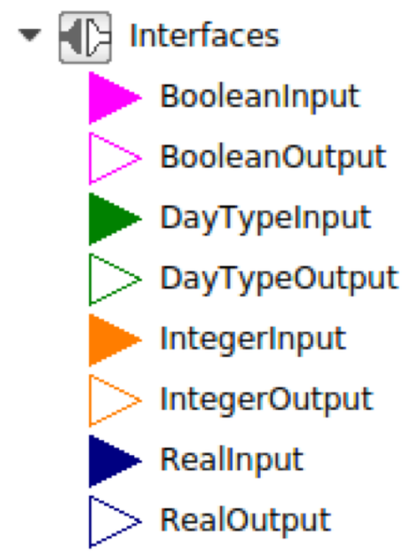
`CDL.Setpoints`:
setpoints for control systems



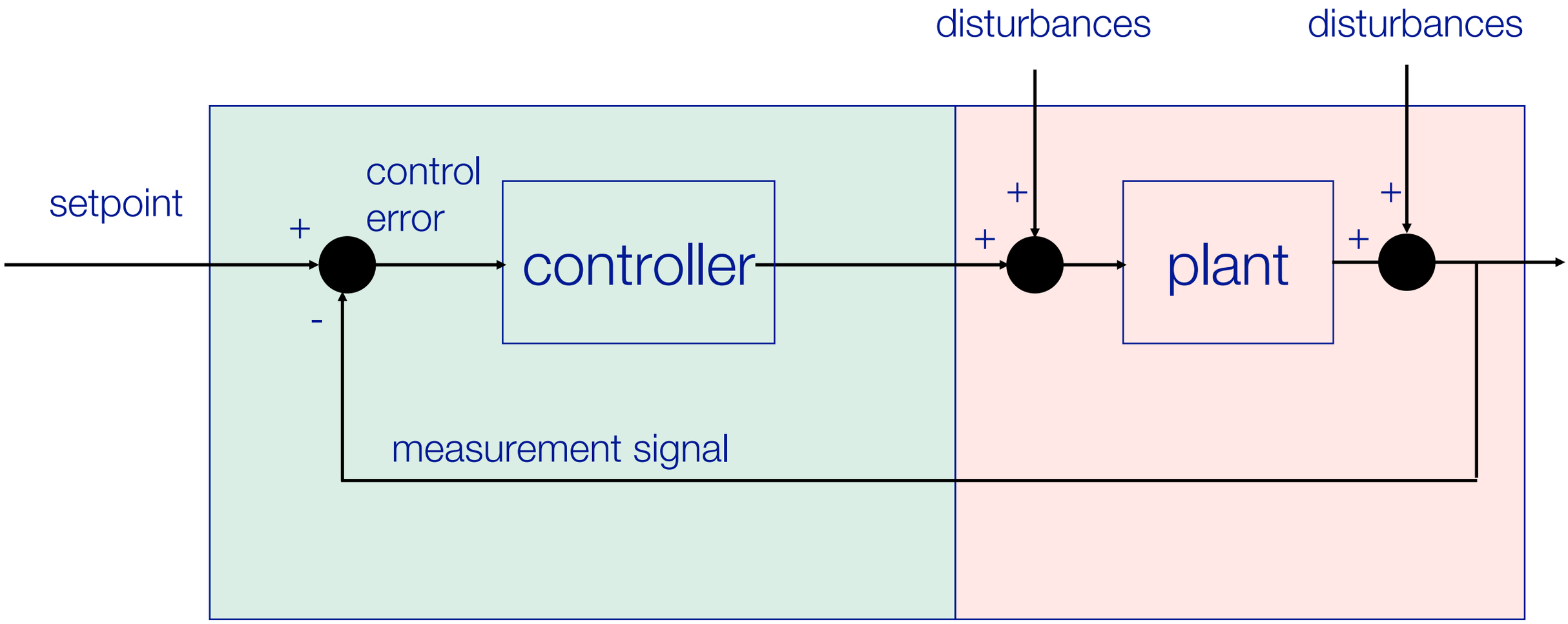
`CDL.Types`:
type definitions



`CDL.Interfaces`:
connectors for input and output signals



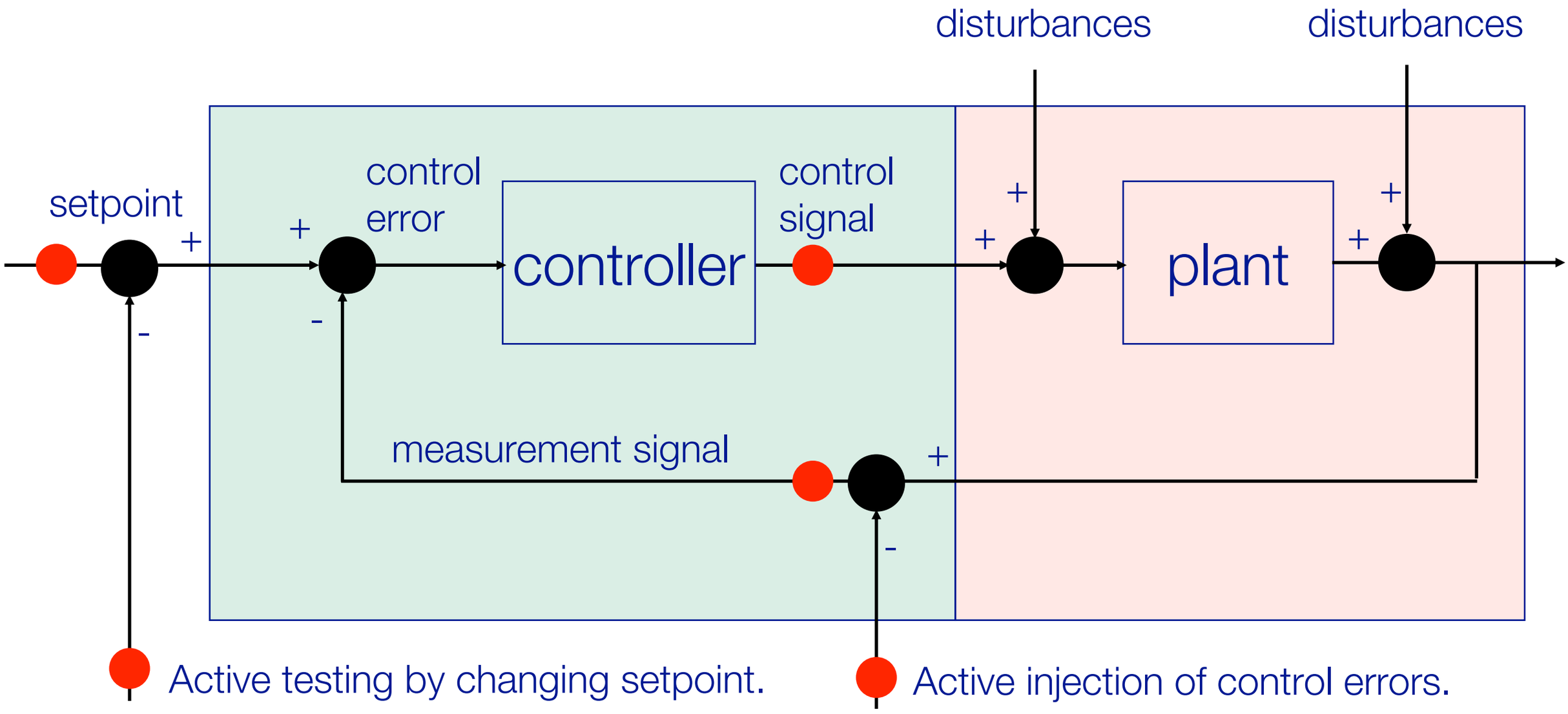
What should be verified?



Disturbances and plant are only approximately known, and hence should be excluded from the verification of the *control delivery*.

But they should be part of an end-to-end verification of the *building delivery*.

How should we verify?



Active testing by changing setpoint.

Active injection of control errors.

- Red points indicate which signals to verify against a CDL generated response.

3-valued logic

Violated: Test condition is **violated** at least once.

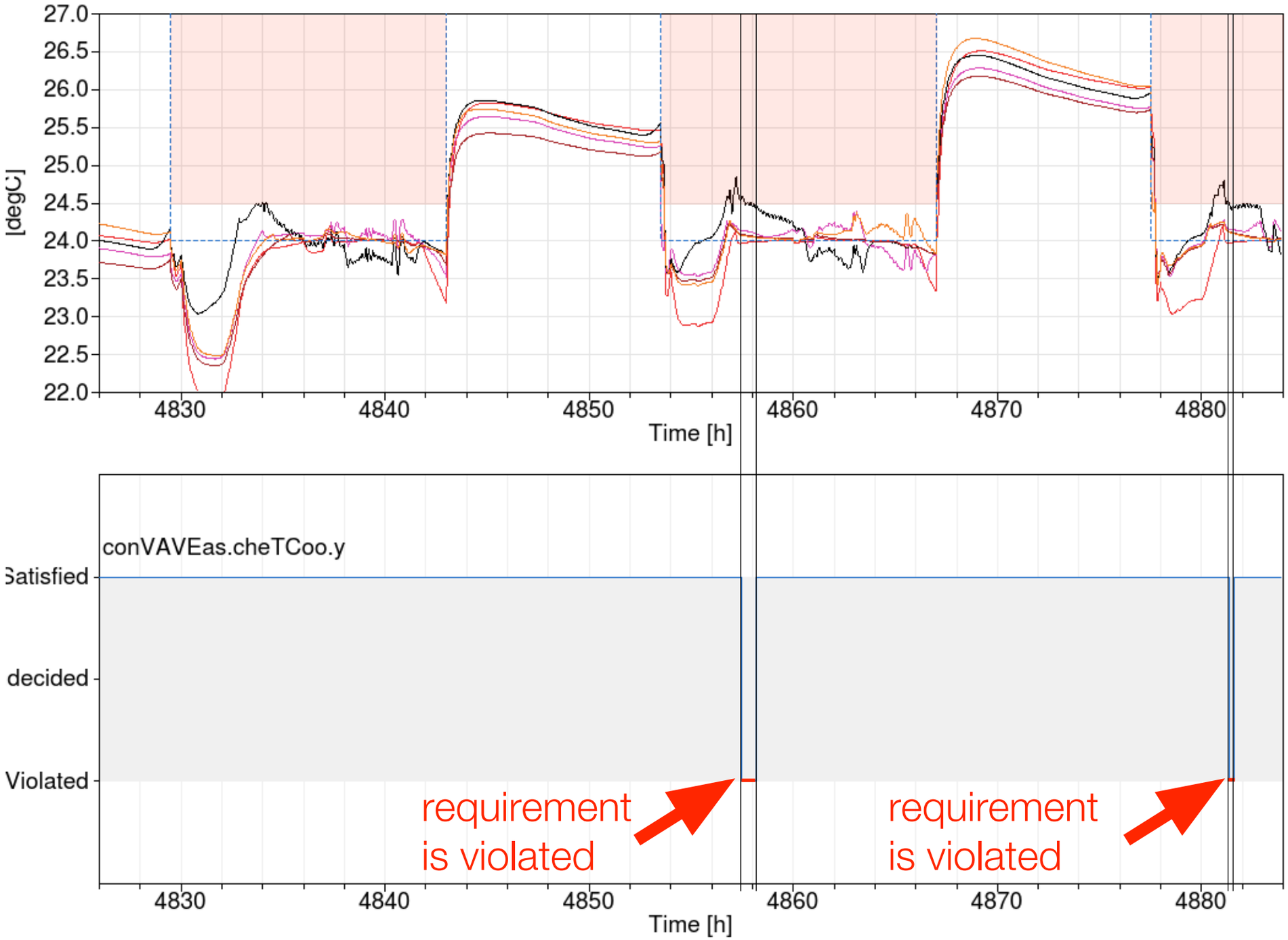
Untested: Test condition is **undecided** for the complete test period.

Satisfied: Test condition is **satisfied** at least once, and is never violated.

Verification Test

Verification of room air temperature of east zone

Requirement: Room air temperature shall be within (TSet + 0.5 K) for at least 45 min within each 60 min window.

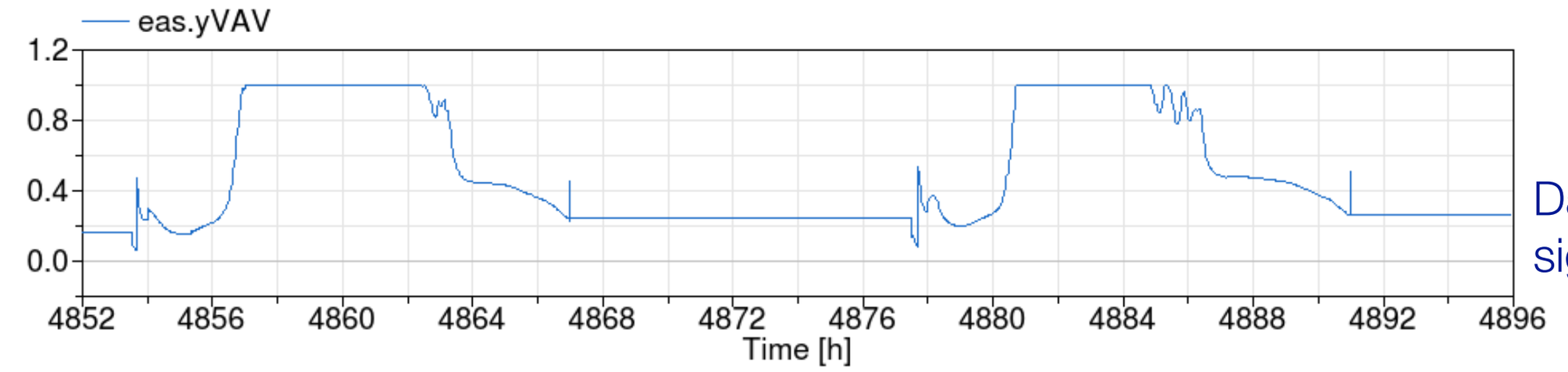


dashed:
cooling set point

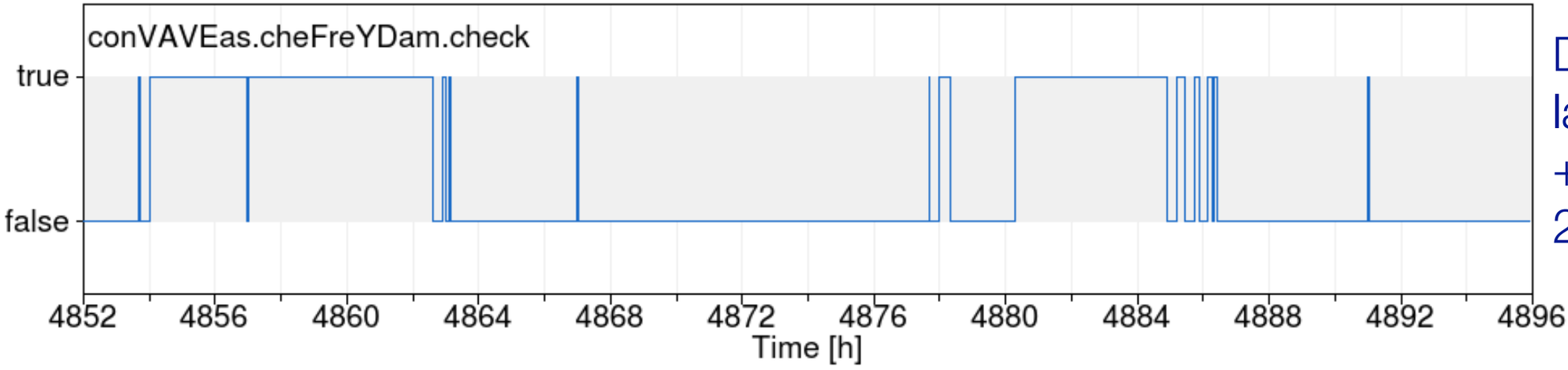
black:
east room
air temperature

Verification of east zone damper signal

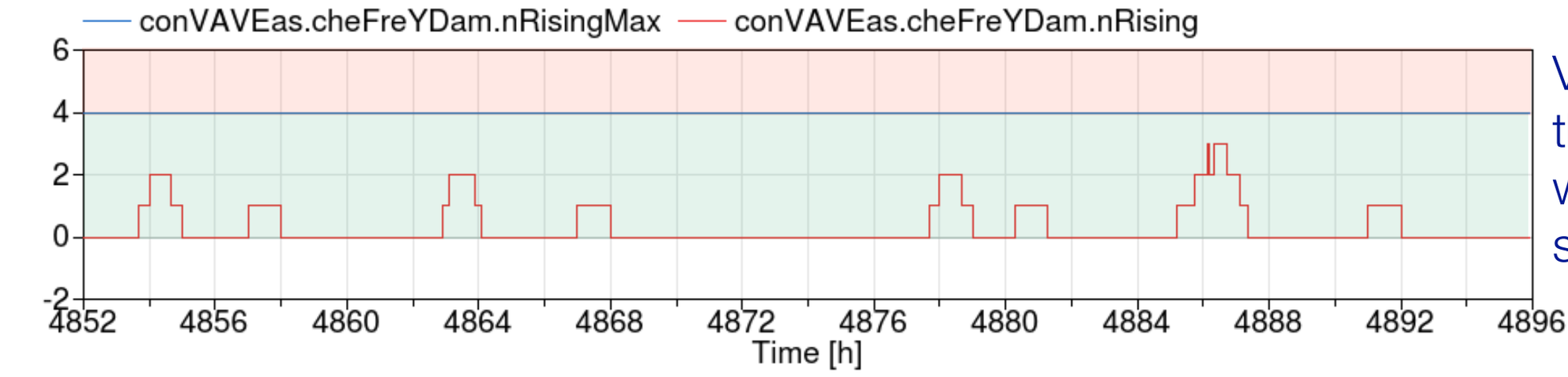
Requirement: Damper signal shall not oscillate more than 4 times per hour between +/- 0.025 (for a 2 minute sample period)



Damper control signal



Detect changes larger than +/- 0.025 between 2 minute samples



Verify no more than 4 changes within 1 hour sliding window

Reports

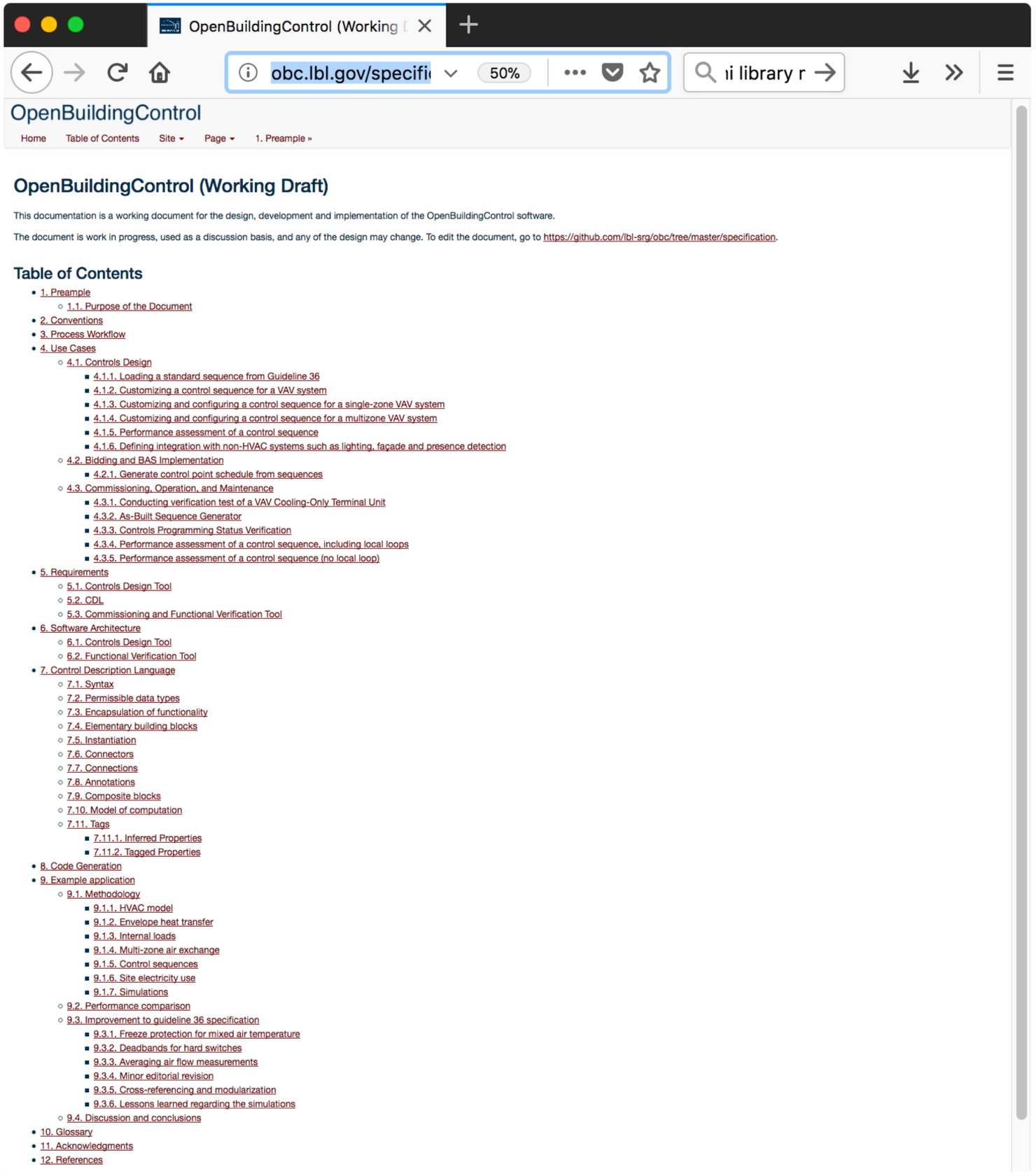
Development requires coordination and compatibility across

- design tool
  - CDL specification
  - CDL library
  - CDL-compliant control sequences
  - CDL-compliant specification for control provider
  - English language representation
- verification tool
  - verification library
  - test specification

To ensure that all elements fit together and that all development is open accessible, reporting with federal agency and stakeholders is done through an integrated document which is continuously updated at:

http://obc.lbl.gov/specification/index.html

Available as html and pdf.

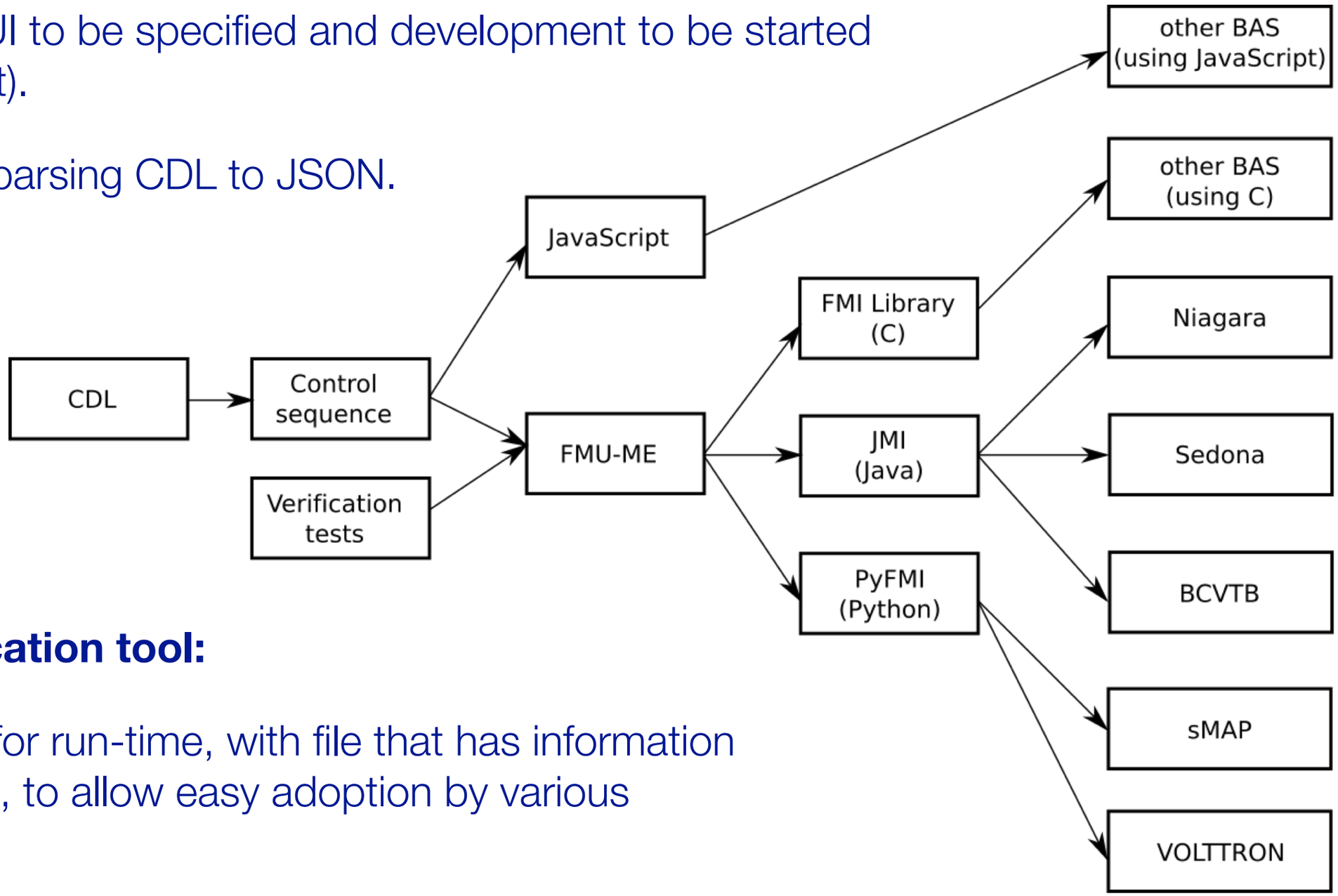


Controls design tool and functional verification tools

Controls design tool:

Schema driven GUI to be specified and development to be started (with SOEP project).

Work ongoing on parsing CDL to JSON.



Functional verification tool:

Use FMI standard for run-time, with file that has information about I/O mapping, to allow easy adoption by various vendors.

http://fmi-standard.org/

Gantt chart

	Year 1												Year 2											
	Q1			Q2			Q3			Q4			Q5			Q6			Q7					
1 Specification			M1.1						M1.2															
2 Controls design tool																								
2.1 Requirements and software architecture						M2.1																		
2.2 Impl. of ctrl seq. (secondary sys.)												M2.2												
2.3 Impl. of ctrl seq. (primary sys., facade & lighting)																								
2.4 Impl. of GUI																								
2.5 CDL export to English language and a product line																								
2.6 OpenStudio integration																				M2.5				
3 Functional verification tool																								
3.1 Requirements and software architecture																								
3.2 Impl. of hardware interface																								
3.3 Impl. of verification test module															M3.1			M3.2						
3.4 Impl. of GUI																								
4 Case studies																								
5 Com. and market transformation plan																								