OtFMI: interfacing OpenTURNS with the leading standard for model exchange

14th June 2024

Sylvain Girard (girard@phimeca.com),
Pascal Borel (pascal.borel@edf.fr)

In three words

fmi, **F**unctional **M**ock-up **I**nterface: free standard defining a container (FMU) and interface to exchange dynamic simulation models.

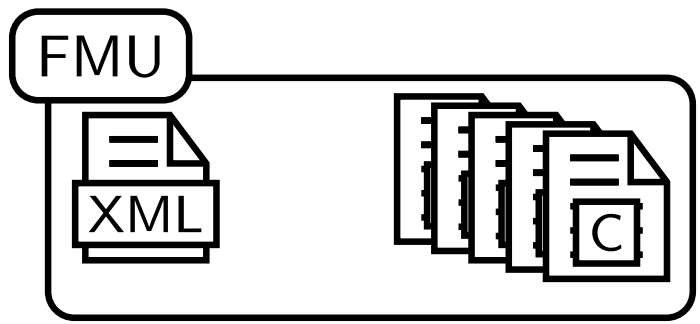
modelica obj. oriented language to model cyber-physical systems.

FMI Python module for

1. Using FMU's within OpenTURNS
2. **[WIP]** Injecting Python functions (e.g. OpenTURNS surrogate model) into a Modelica Model

FMI & FMU

- FMI (Functional Mock-up Interface) aims to simplify creation, storage, exchange and (re-)use of dynamic system models.
- An FMU (Functional Mock-up *Unit*) is a black box container–interface following the standard.



While developed by the Modelica association, it is supposedly tool/language agnostic.

200 tools currently support FMU export and/or import.

---

https://www.fmi-standard.org/

Modelica : equation-based programming

- Model system by differential algebraic equations (DAE)
- Equations are written "naturally" (acausal modelling) and solved by a third party multi-purpose tool (**OpenModelica**, Dymola).
- Modelica is object oriented:
  - Code structure reflects the modelled system
  - Libraries of reusable and combinable modules

Some user : **EDF**, ABB, Siemens,... Audi, BMW, Daimler, Ford, Toyota, ... **Airbus**, **Onera**, NASA, ESA, ... and **Phimeca**!

Born in 1996 (5 years after Python).

Example: the SIR epidemiologic model (equations)

- Soit N individus répartis en 3 groupes :
  - "Susceptible" de contracter la maladie ;
  - "Infecté" ;
  - "Retiré", immunisé ou mort.
- L'évolution de leurs tailles est régie par les équations

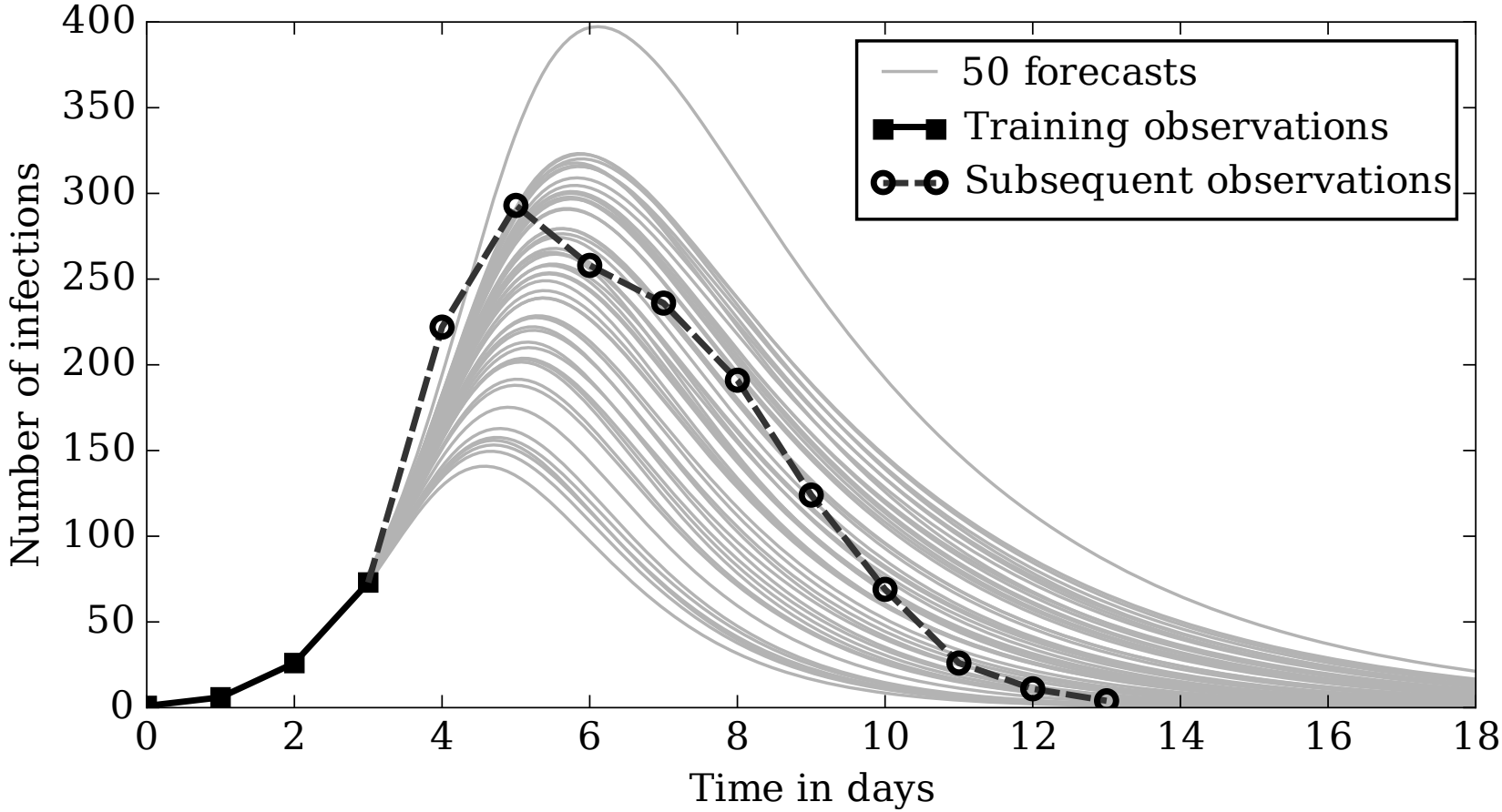
$$\frac{dS}{dt} = -\frac{\beta IS}{N}$$
$$\frac{dI}{dt} = \frac{\beta IS}{N} - \gamma I$$
$$\frac{dR}{dt} = \gamma I$$

β	fréquence caractéristique de rencontre des individus
γ^{-1}	durée entre infection et rémission.

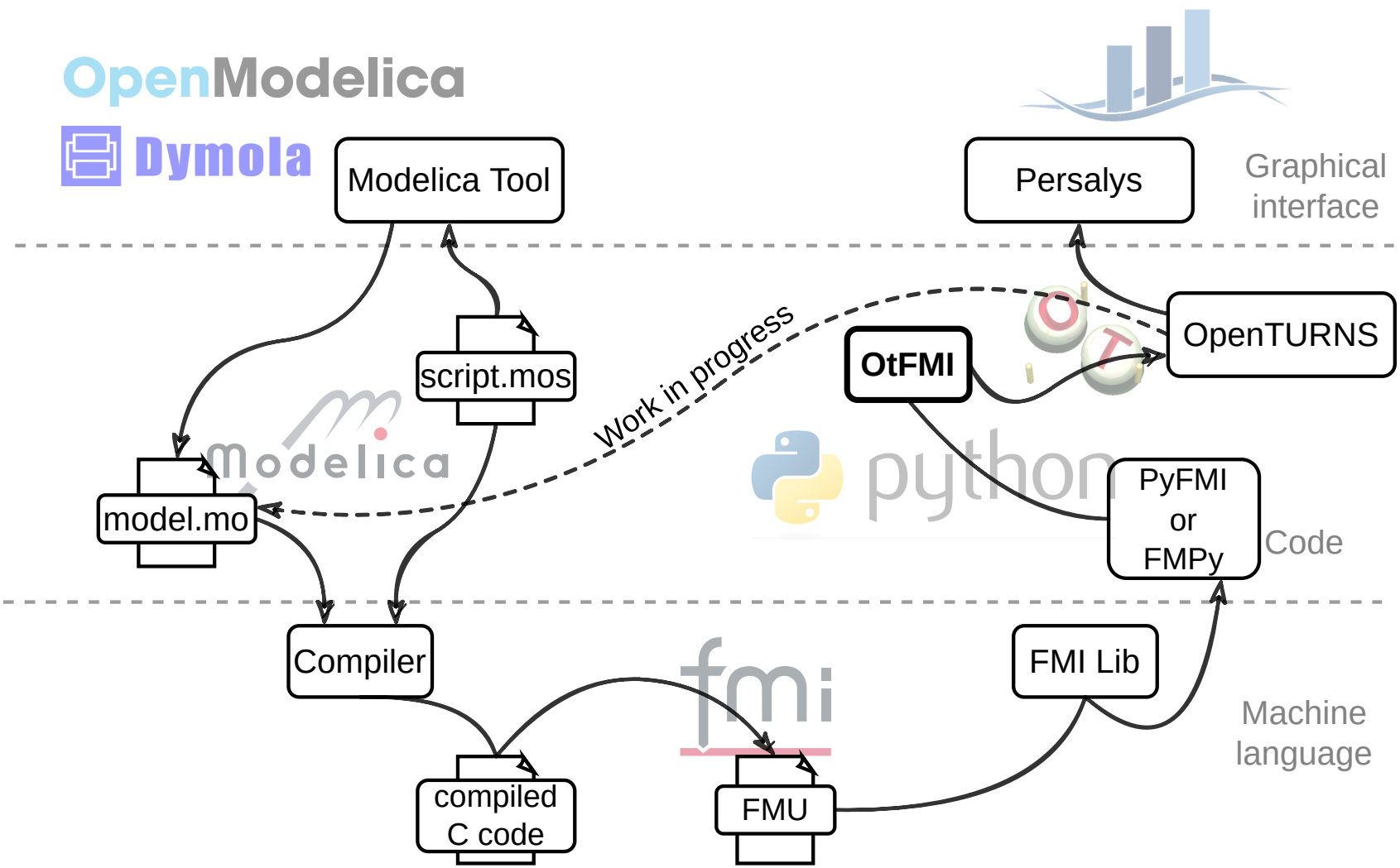
Example: the SIR epidemiologic model (code)

```
model SIR "Susceptible, infected, removed compartment model."
  parameter Real total_population=763;
  Real susceptible;
  Real infected;
  Real removed;
  input Real contact_rate (start=0.5);
  input Real infectious_period (start=2.5, fixed=true);
  parameter Real infected_initial=1;

  initial equation
  total_population = susceptible + infected + removed;
  infected = infected_initial;

  equation

  der(susceptible) = -contact_rate * susceptible * infected /
    total_population;
  der(infected) = contact_rate * susceptible * infected / total_population -
    infected / infectious_period;
  der(removed) = infected / infectious_period;

end SIR;
```

Epidemic Dynamics inferred with OpenTURNS

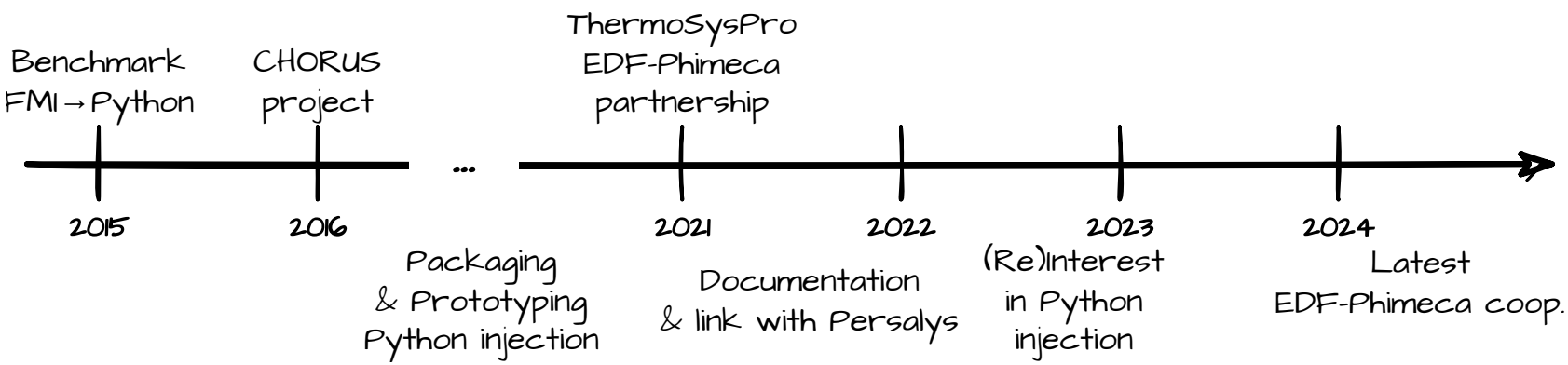


S. Girard, *"A probabilistic take on system modeling with Modelica and Python"*, https://sylvaingirard.net/pdf/girard17-probabilistic_modelica_python.pdf


OpenModelica
Dymola
Modelica Tool
Persalys
Graphical interface
script.mos
Work in progress
OtFMI
OpenTURNS
model.mo
PyFMI or FMPy
Code
Compiler
FMI Lib
Machine language
compiled C code
FMU

OtFMI: *Since 2015!*

A long term cooperation between EDF and PHIMECA L'ingénierie responsable.

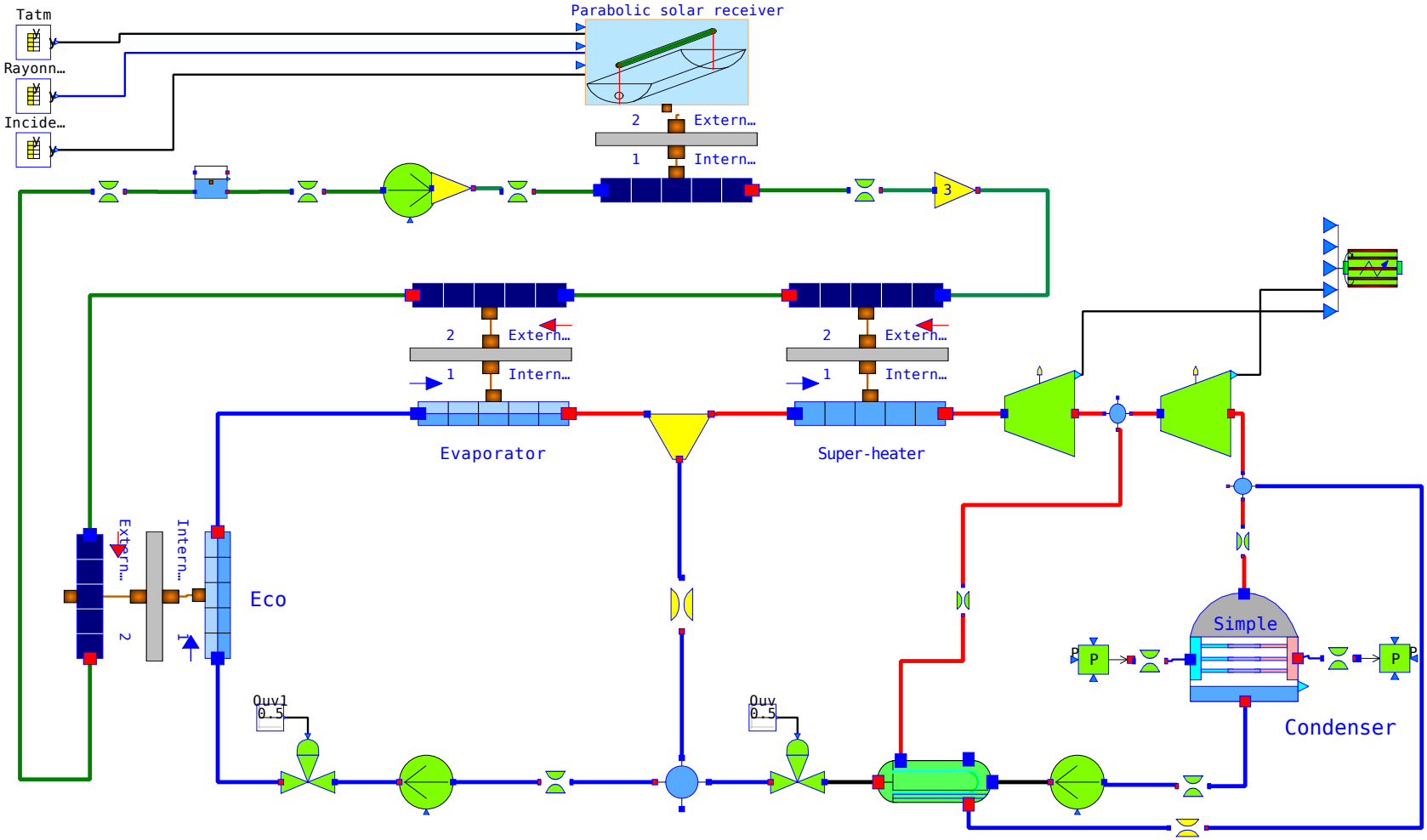


Git: https://github.com/openturns/otfmi

Documentation : http://openturns.github.io/otfmi/master/

Thanks to Audrey Jardin, Michaël Baudin, Anne-Laure Popelin, Eleu Gerrer, Julien Schueller, Pascal Borel & all other contributors!

Concentrated solar power plant model

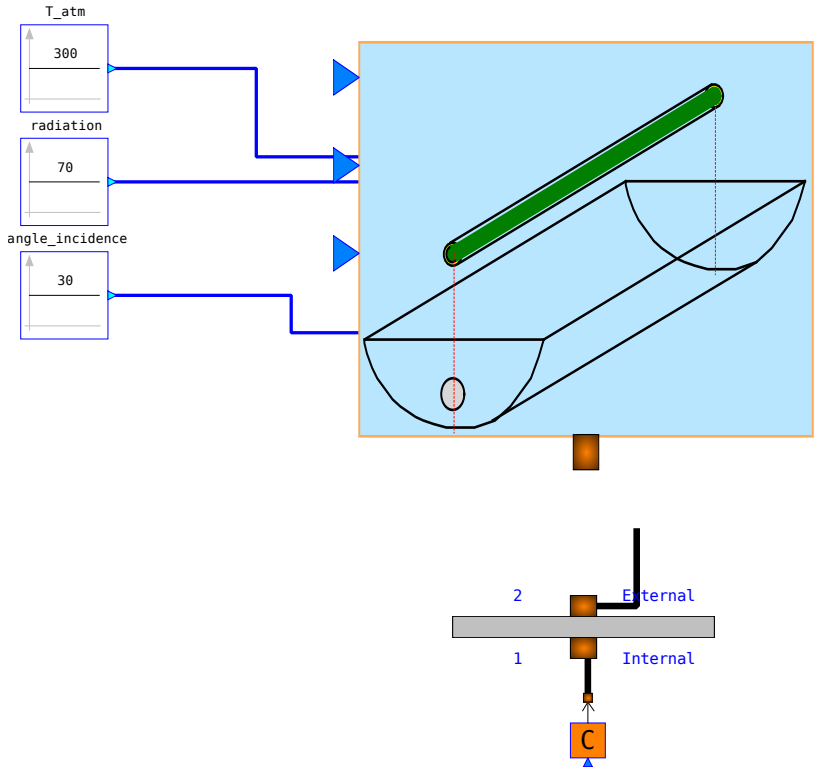


Designed by B. El Hefni & D. Bouskela with thermoSYSPRO → https://thermosyspro.com/.

Design of experiment

```
import openturns as ot, otfmi

path_fmu = "/path/to/model.fmu"

inputs_fmu = ["T_atm.k", "radiation.k", "angle_incidence.k"]
outputs_fmu = ["heatFlow"]

mesh = ot.RegularGrid(0.0, 1.0, 2000)

function = otfmi.FMUPointToFieldFunction(mesh, path_fmu,
    inputs_fmu=inputs_fmu,
    outputs_fmu=outputs_fmu)

dist = ot.ComposedDistribution([
    ot.Normal(300, 20)
    ot.Uniform(10, 1000)
    ot.Uniform(1, 89)])

experiment = ot.LHSExperiment(distribution, size=100)
sample = experiment.generate()
sample.setDescription(inputs_fmu)
out = function(sample)
```

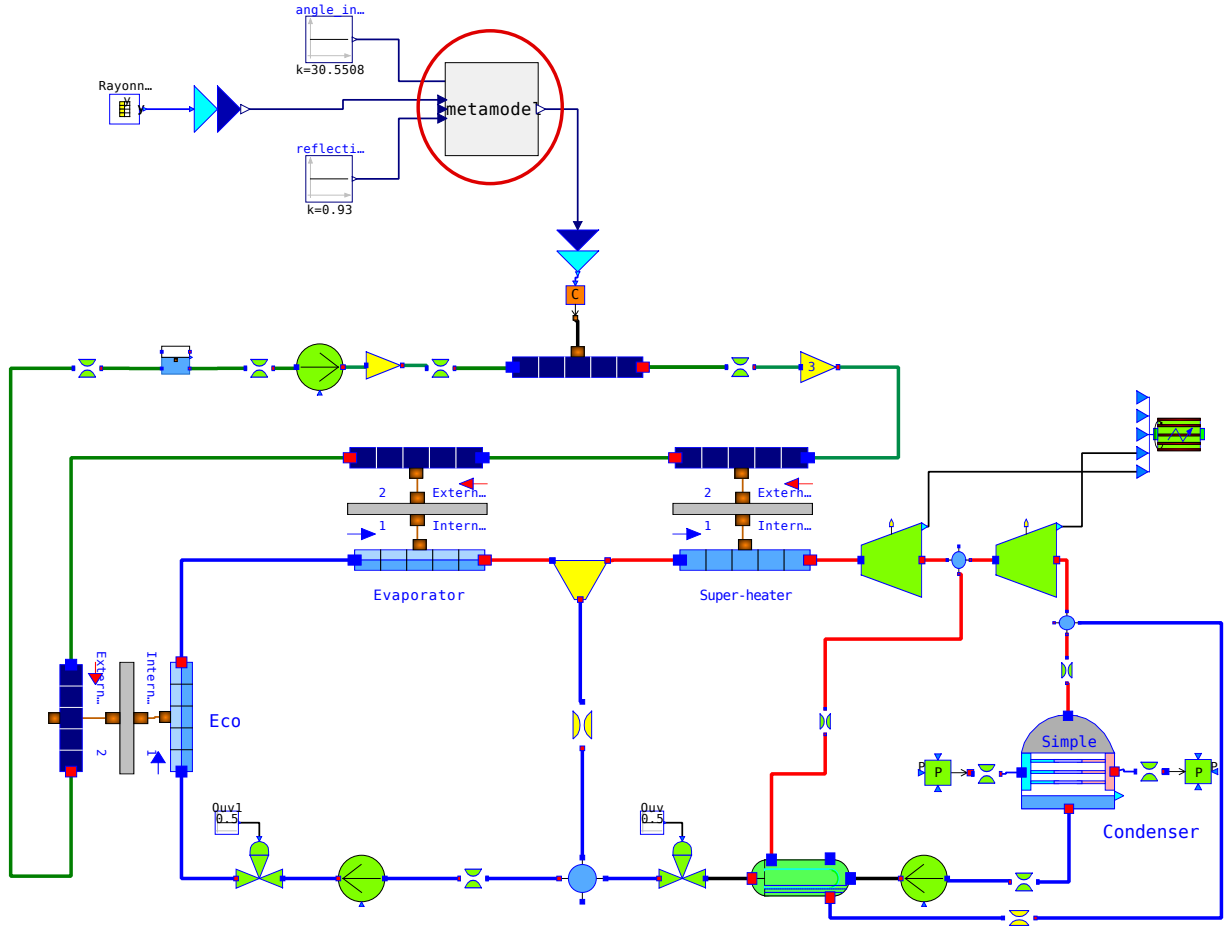
Meta-modelling of the collector



- We built a kriging meta-model of the collector using an FMU

WIP: faster re-implementation of Gerrer et al. “Analysis and reduction of models using persalys,” (2021) https://sylvaingirard.net/pdf/talk/gerrer-modelica21.pdf.

Python injection into Modelica



- The meta-model is wrap as a Modelica component by OtFMI
- Modelica natively include C-code (`external` keyword) ; current implementation uses the C-Python API.

Current state and next steps

OtFMI allows to

1. Use FMU's within OpenTURNS
2. **[Work in Progress]** Inject Python functions into a Modelica Model

- We are currently working out realistic use cases
  - Improve ease of use
  - Streamline interface
  - Revamp documentation

- Please get in touch with us if interested!

  → Sylvain Girard (girard@phimeca.com),

  → Julien Schueller (schueller@phimeca.com),

  → Michaël Baudin (michael.baudin@edf.fr).

Thank you for your attention.

Sylvain Girard : girard@phimeca.com