Glenn Cavarlé

August 2016

Lab-STICC - University of Brest, France

Dynamic Round-Trip Engineering in the context of FOMDD

Glenn Cavarlé

August 2016

Lab-STICC – University of Brest, France

FOMDD

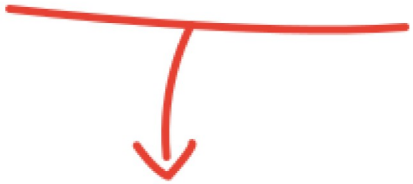
Feature-Oriented Model Driven Development

Glenn Cavarlé

Lab-STICC – University of Brest, France

FOMDD

Feature-Oriented Model Driven Development

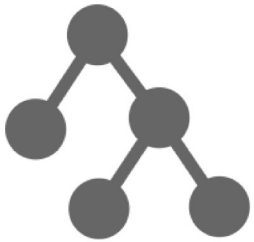


Breaking down software systems
in terms of the features they provide

FOMDD

Feature-Oriented Model Driven Development

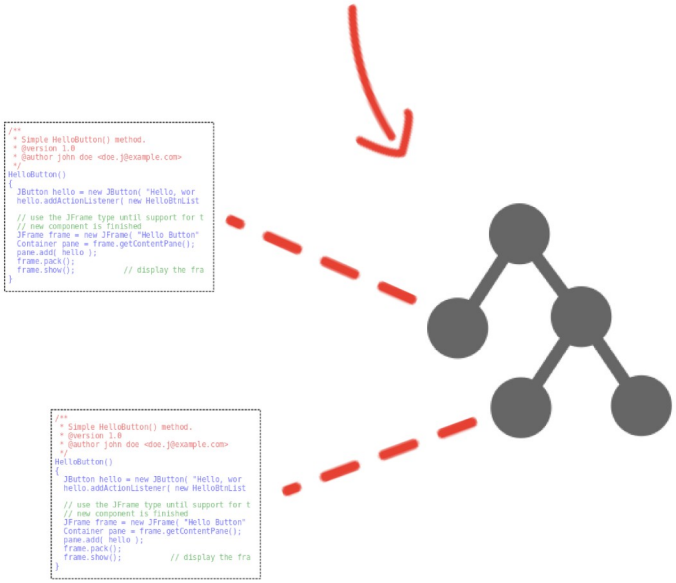
Breaking down software systems
in terms of the features they provide



FOMDD

Feature-Oriented Model Driven Development

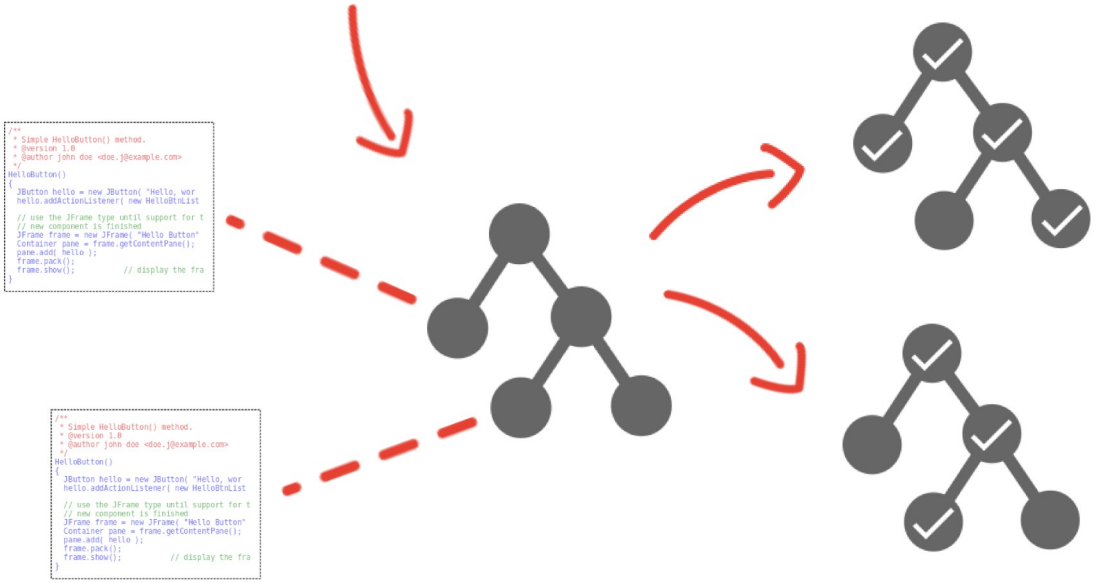
Breaking down software systems in terms of the features they provide



FOMDD

Feature-Oriented Model Driven Development

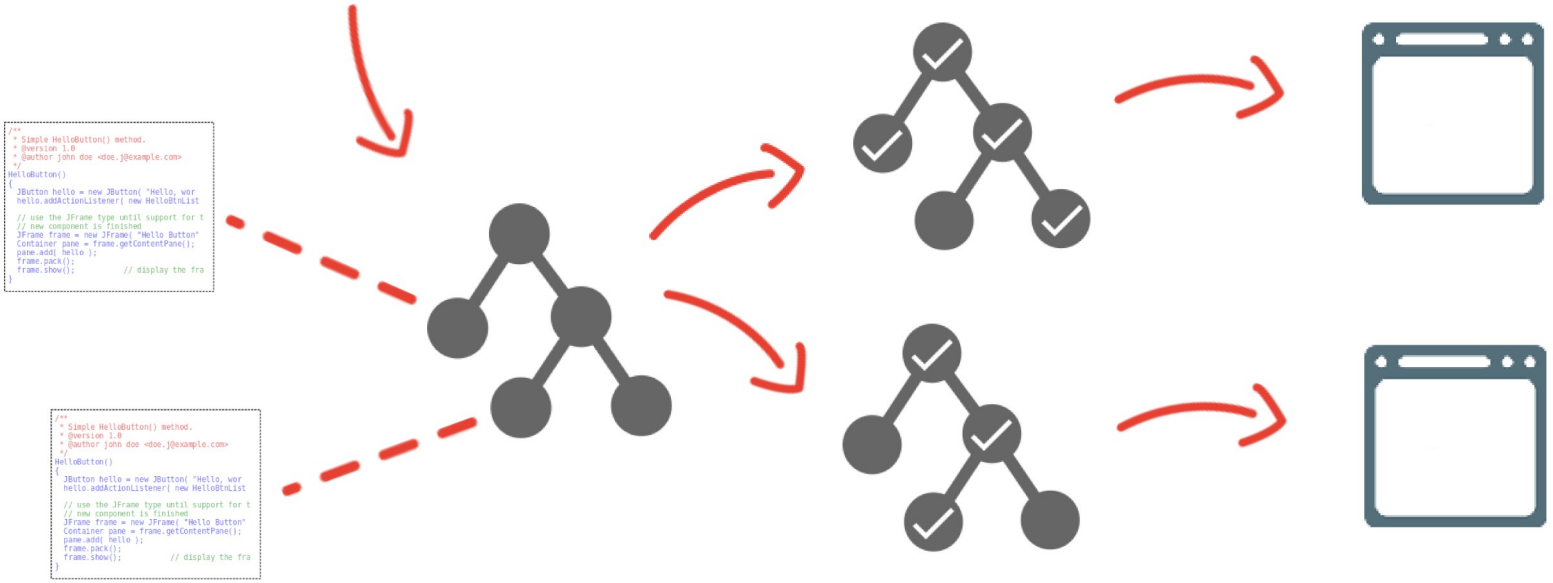
Breaking down software systems in terms of the features they provide



FOMDD

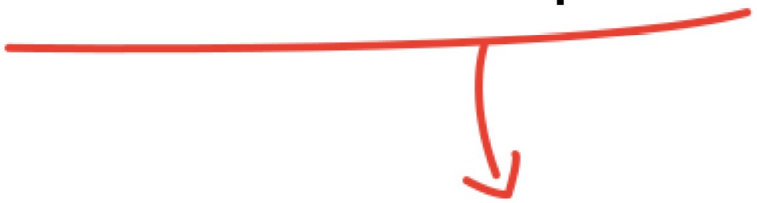
Feature-Oriented Model Driven Development

Breaking down software systems in terms of the features they provide



FOMDD

Feature-Oriented Model Driven Development

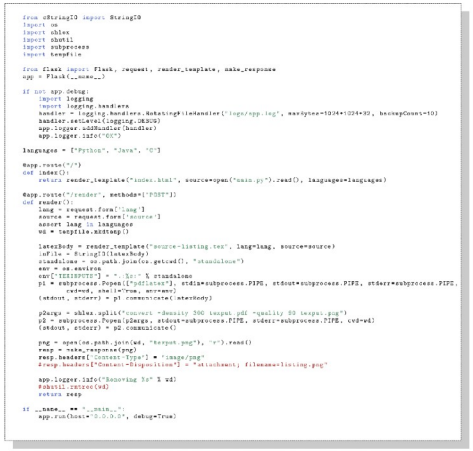


Building abstract representations of software systems

FOMDD

Feature-Oriented Model Driven Development

Building abstract representations of software systems



```
from cStringIO import StringIO
import os
import shlex
import shutil
import subprocess
import tempfile

from flask import Flask, request, render_template, make_response
app = Flask(__name__)

if not app.debug:
    import logging
    import logging.handlers
    handler = logging.handlers.RotatingFileHandler('logs/app.log', maxBytes=1024*1024*32, backupCount=10)
    handler.setLevel(logging.DEBUG)
    app.logger.addHandler(handler)
    app.logger.info("OK")

languages = ["Python", "Java", "C"]

@app.route("/")
def index():
    return render_template("index.html", source=open("main.py").read(), languages=languages)

@app.route("/render", methods=["POST"])
def render():
    lang = request.form['lang']
    source = request.form['source']
    assert lang in languages
    wd = tempfile.mkdtemp()

    latexBody = render_template("source-listing.tex", lang=lang, source=source)
    #file = StringIO(latexBody)
    standalone = os.path.join(os.getcwd(), "standalone")
    env = os.environ
    env['TEXINPUTS'] = ".%s:" % standalone
    p1 = subprocess.Popen(["pdflatex"], stdin=subprocess.PIPE, stdout=subprocess.PIPE, stderr=subprocess.PIPE,
            cwd=wd, shell=True, env=env)
    (stdout, stderr) = p1.communicate(latexBody)

    p2args = shlex.split("convert -density 300 texput.pdf -quality 90 texput.png")
    p2 = subprocess.Popen(p2args, stdout=subprocess.PIPE, stderr=subprocess.PIPE, cwd=wd)
    (stdout, stderr) = p2.communicate()

    png = open(os.path.join(wd, "texput.png"), 'r').read()
    resp = make_response(png)
    resp.headers['Content-Type'] = 'image/png'
    #resp.headers['Content-Disposition'] = "attachment; filename=listing.png"

    app.logger.info("Removing %s" % wd)
    #shutil.rmtree(wd)
    return resp

if __name__ == "__main__":
    app.run(host='0.0.0.0', debug=True)
```

FOMDD

Feature-Oriented Model Driven Development

Building abstract representations of software systems

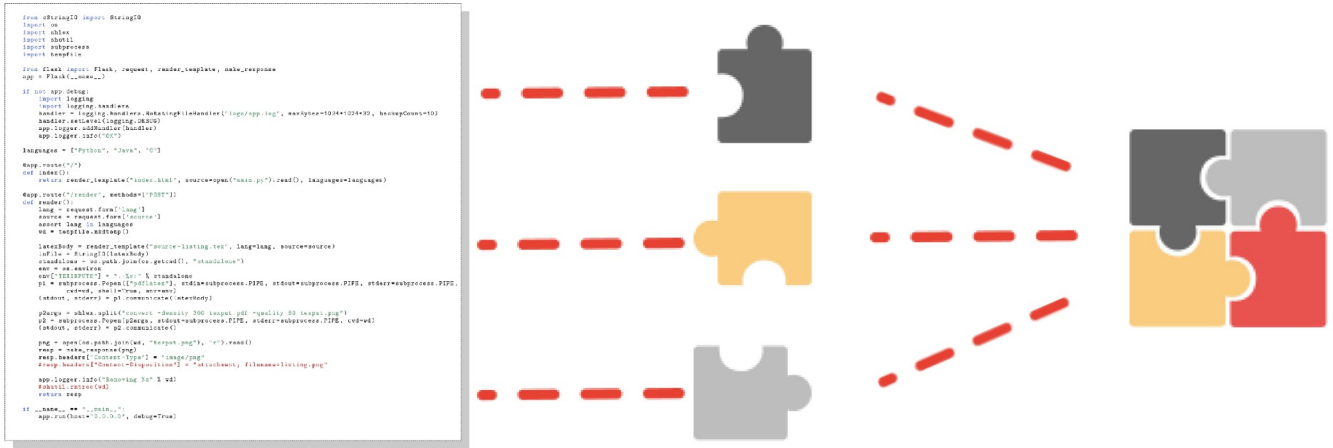
Glenn Cavarlé

Lab-STICC – University of Brest, France

FOMDD

Feature-Oriented Model Driven Development

Building abstract representations
of software systems



Glenn Cavarlé

Lab-STICC – University of Brest, France

FOMDD

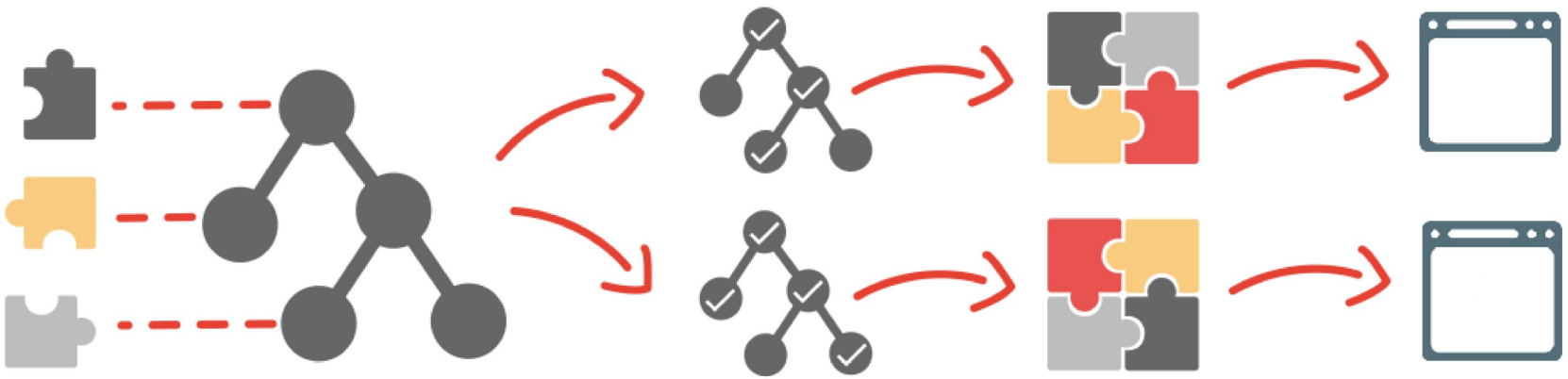
Feature-Oriented Model Driven Development

Glenn Cavarlé

Lab-STICC – University of Brest, France

FOMDD

Feature-Oriented Model Driven Development



Why we use FOMDD?

Why we use FOMDD?

The ***CrossFabrik*** Project

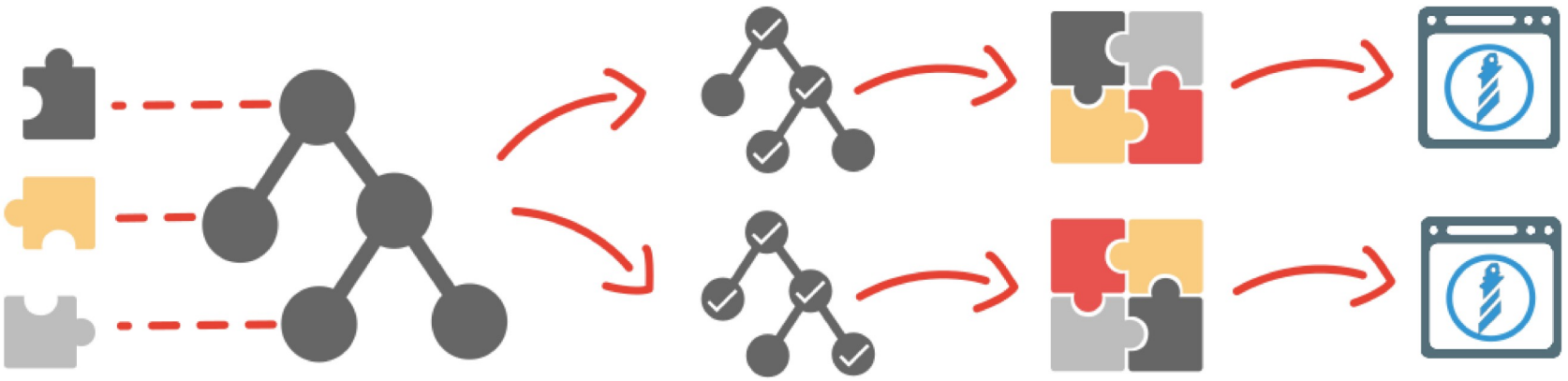
Multi-platform software Prototyping & Assessment

Softwares simulation within the development environment

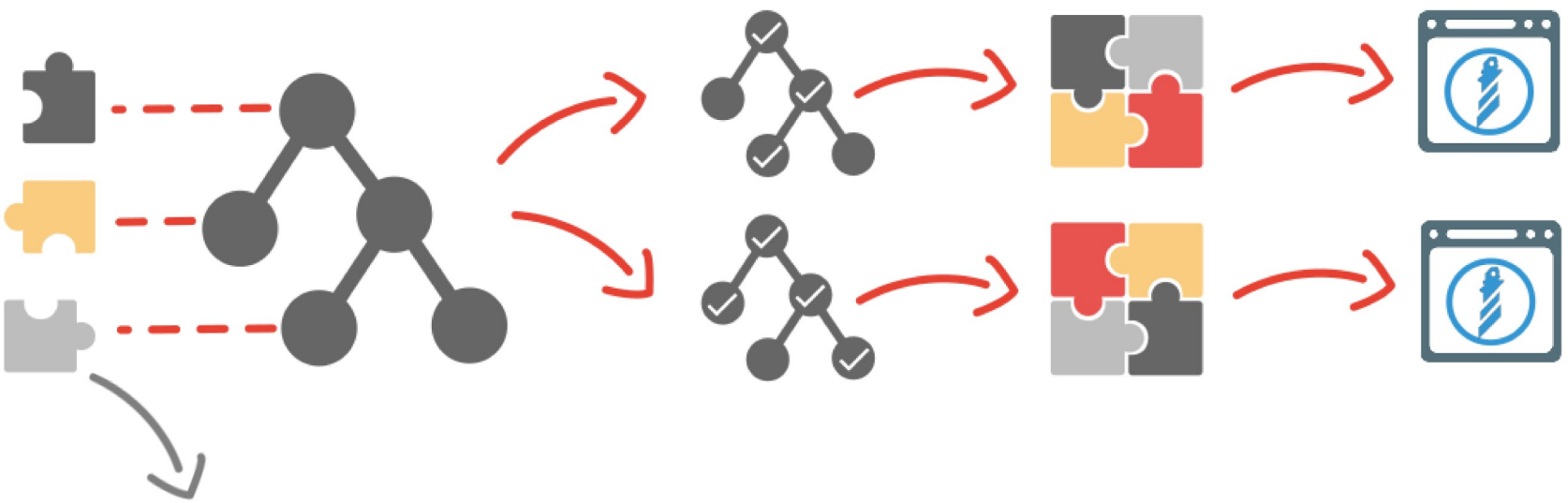
Implemented using ***Pharo*** 

How we use FOMDD?

How we use FOMDD?

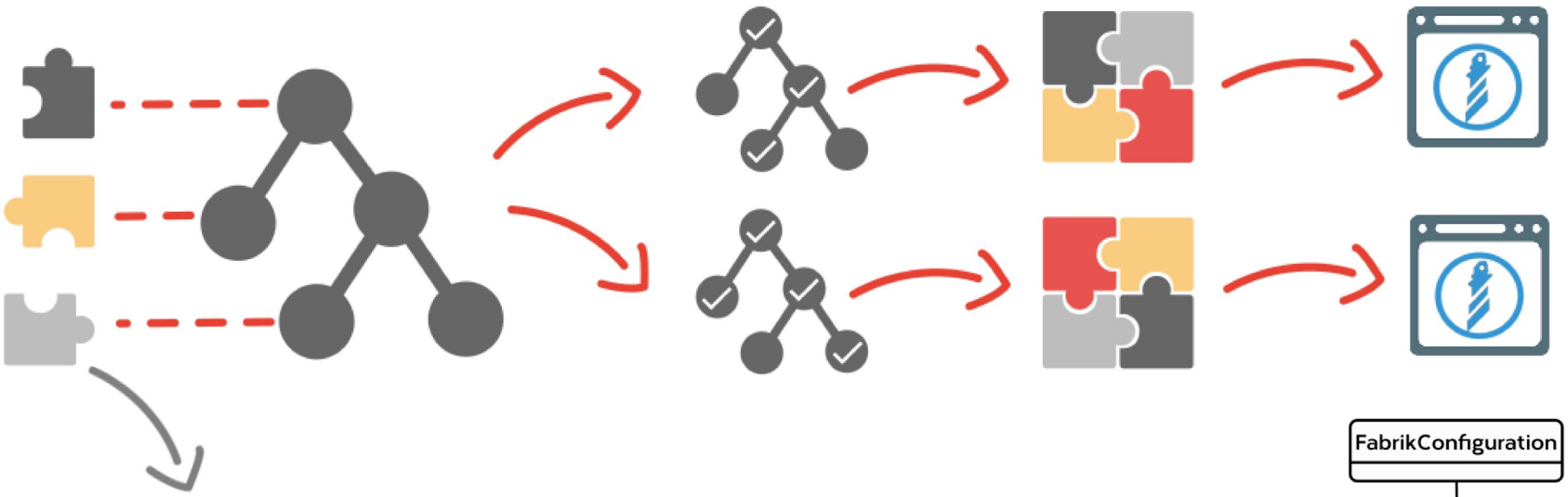


How we use FOMDD?

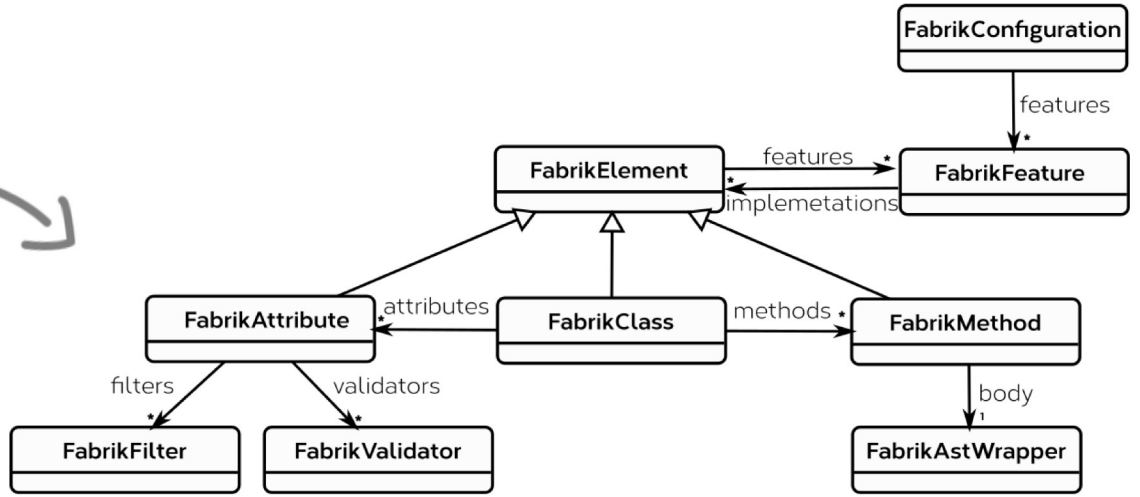


Instances of an EMOF-like meta-model

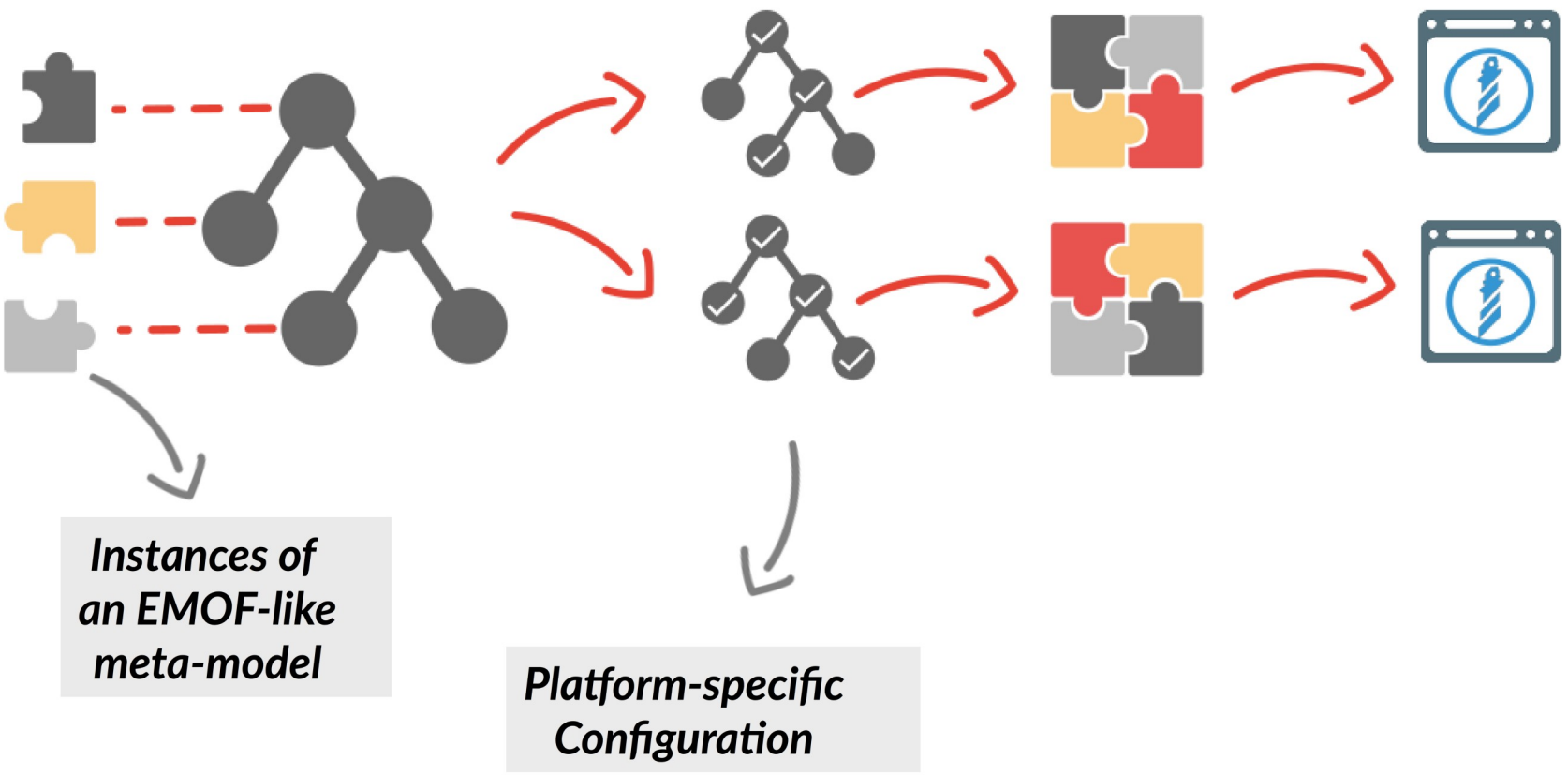
How we use FOMDD?



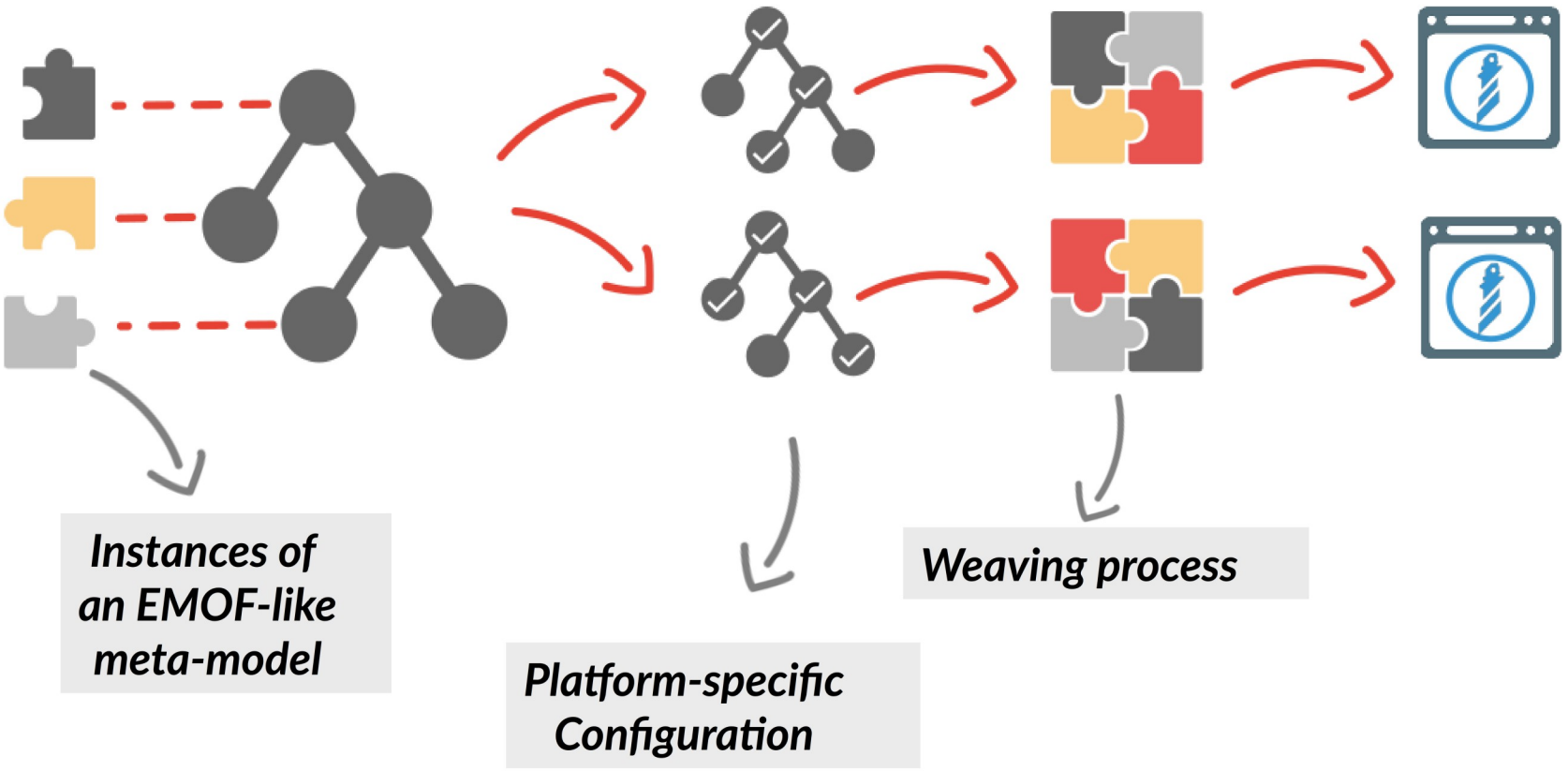
Instances of an EMOF-like meta-model



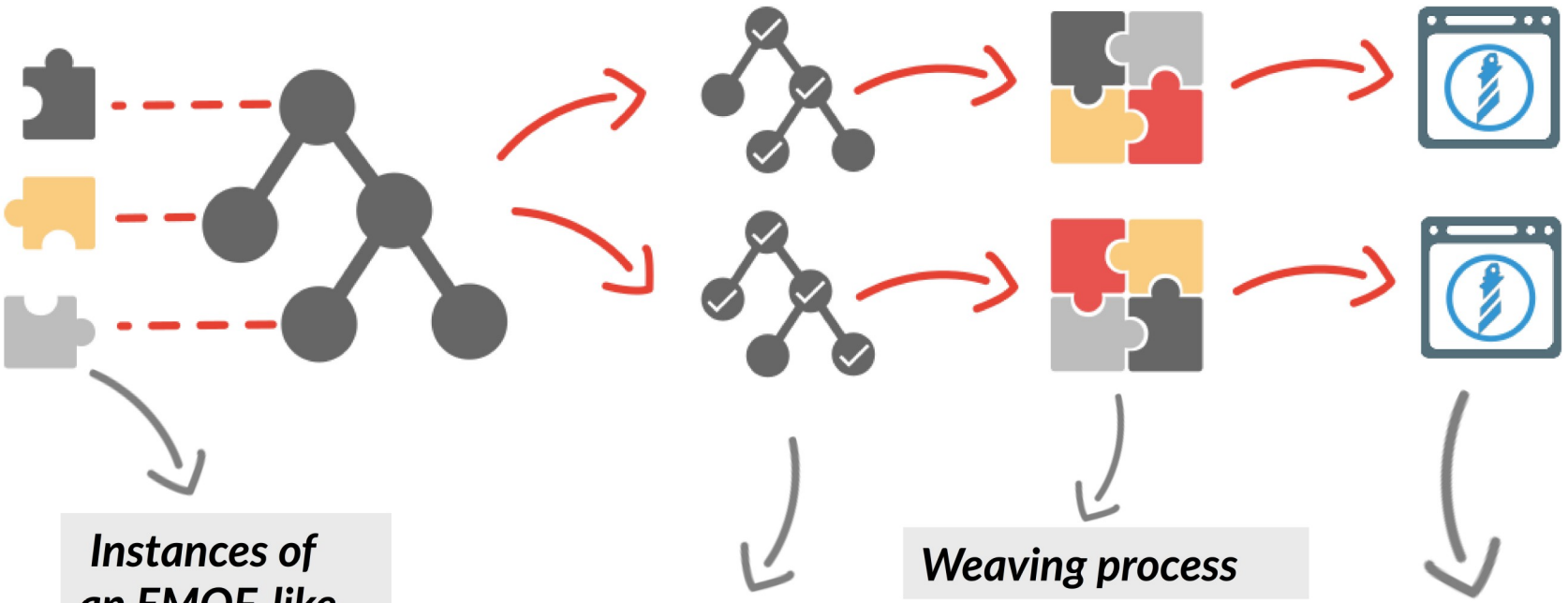
How we use FOMDD?



How we use FOMDD?



How we use FOMDD?



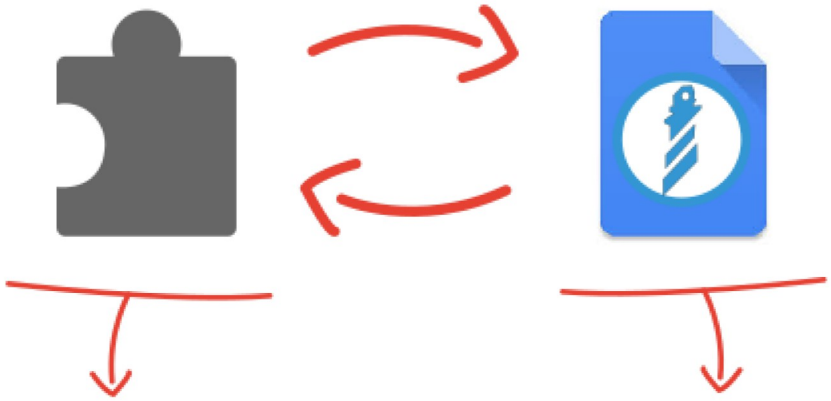
What about **RTE**?

What about **RTE**?

To let developers investigate solutions
from any place

What about **RTE**?

To let developers investigate solutions from any place



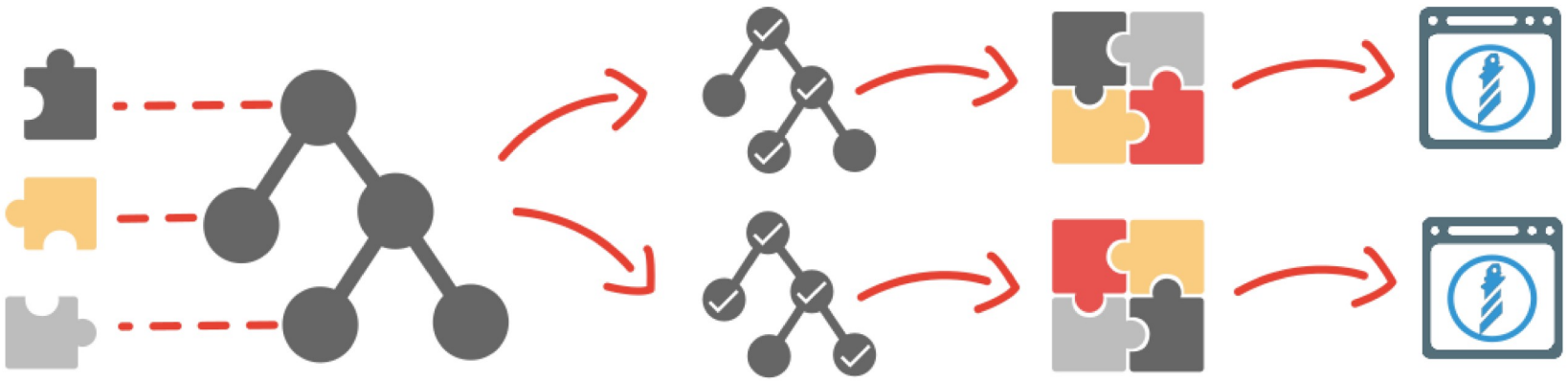
What about **RTE**?

To let developers investigate solutions
from any place

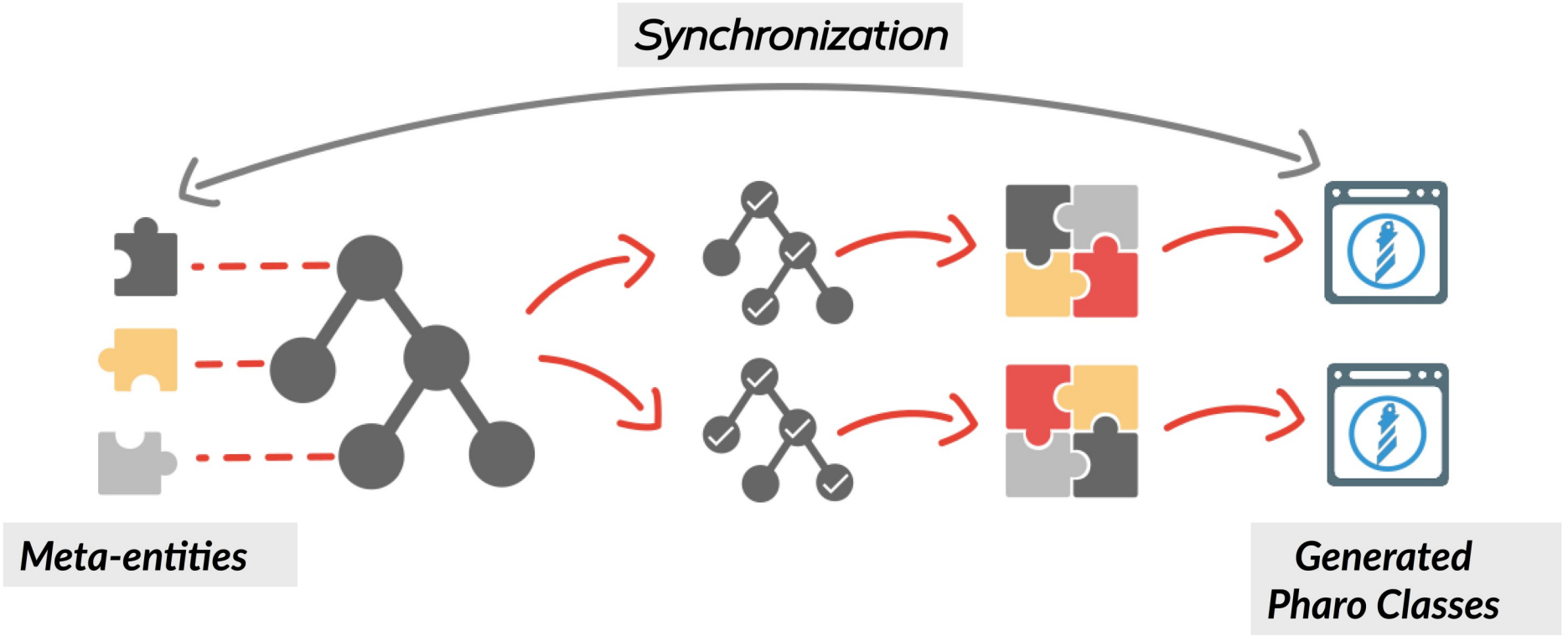
Without having to worry about inconsistency

Without having to stop/restart anything

What about **RTE**?

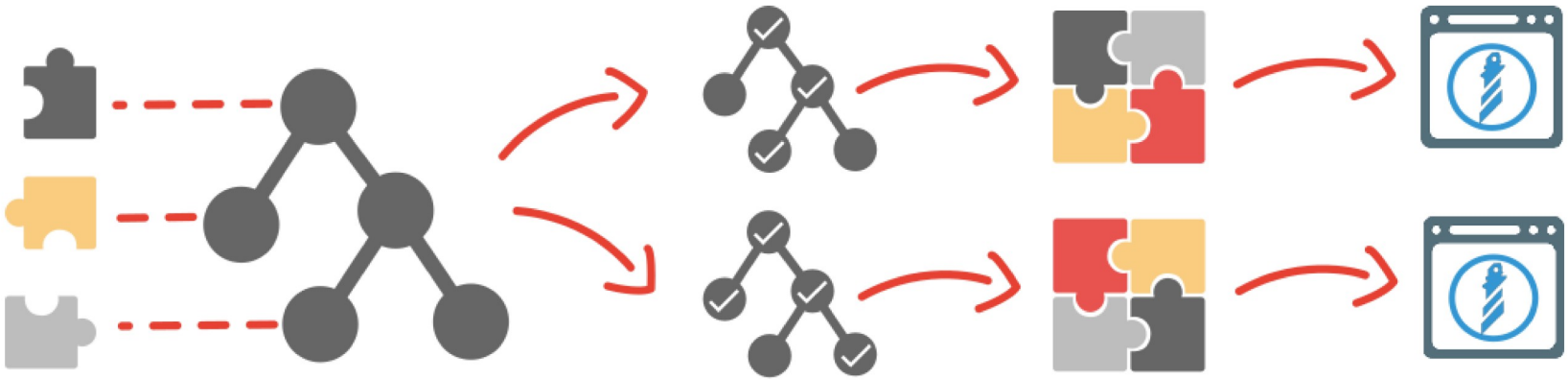


What about **RTE**?

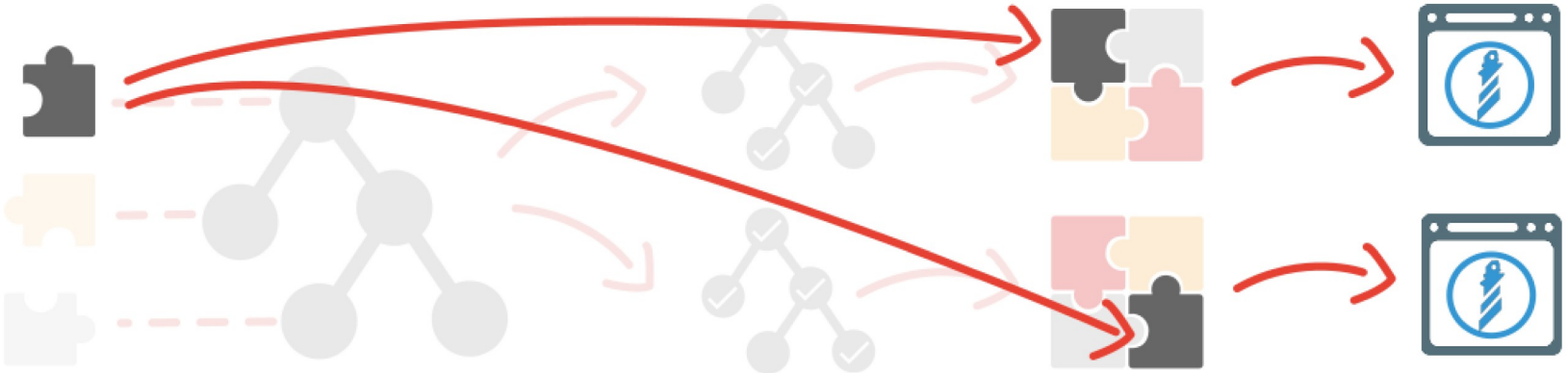


Issues

Issues

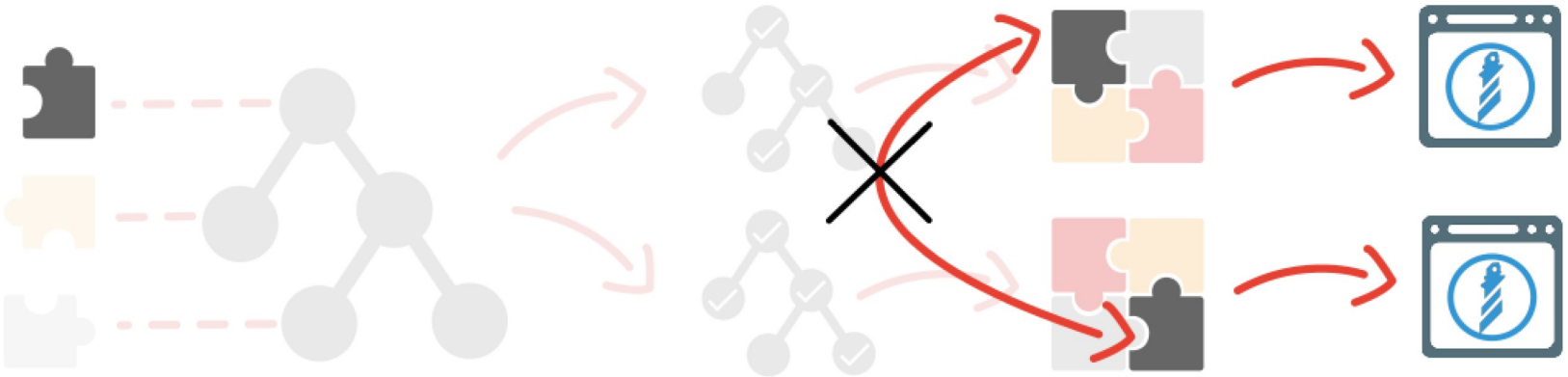


Issues



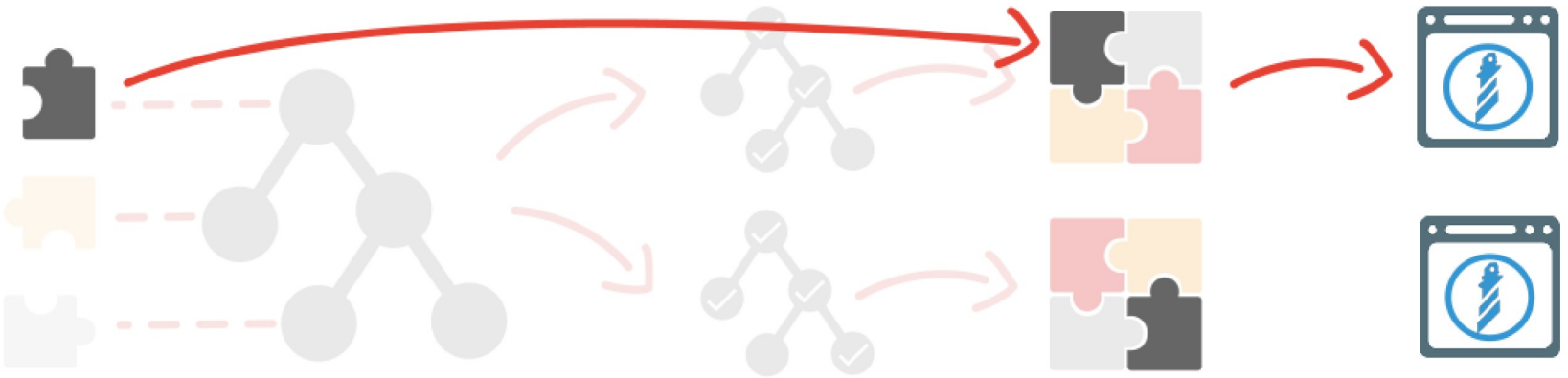
a **one-to-many** relationship

Issues



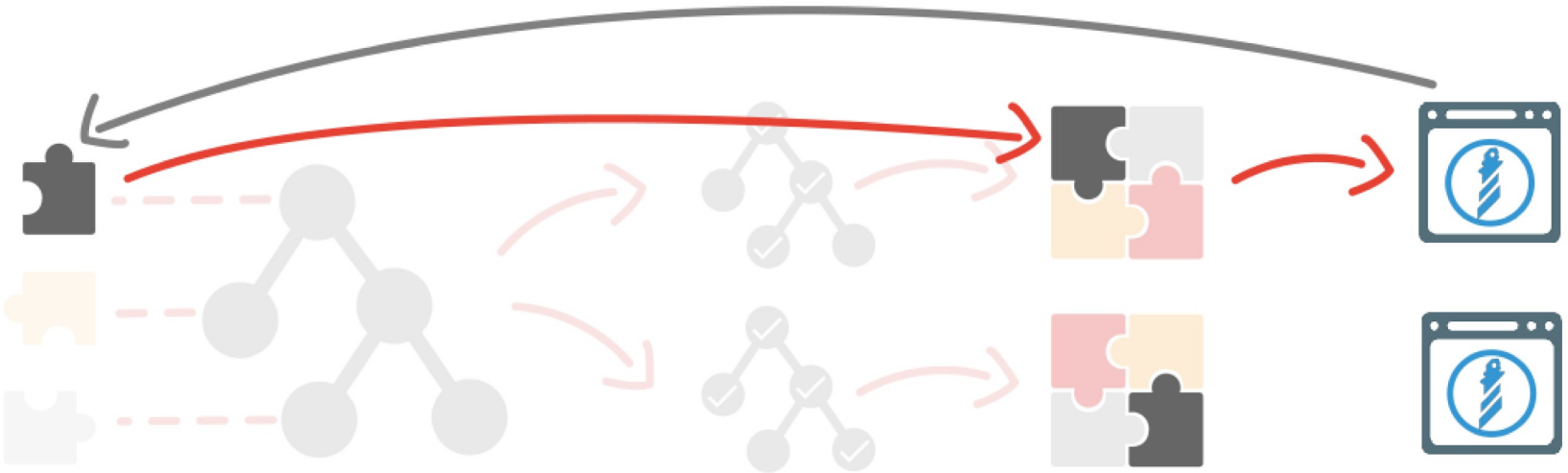
model and code **duplication**

Issues



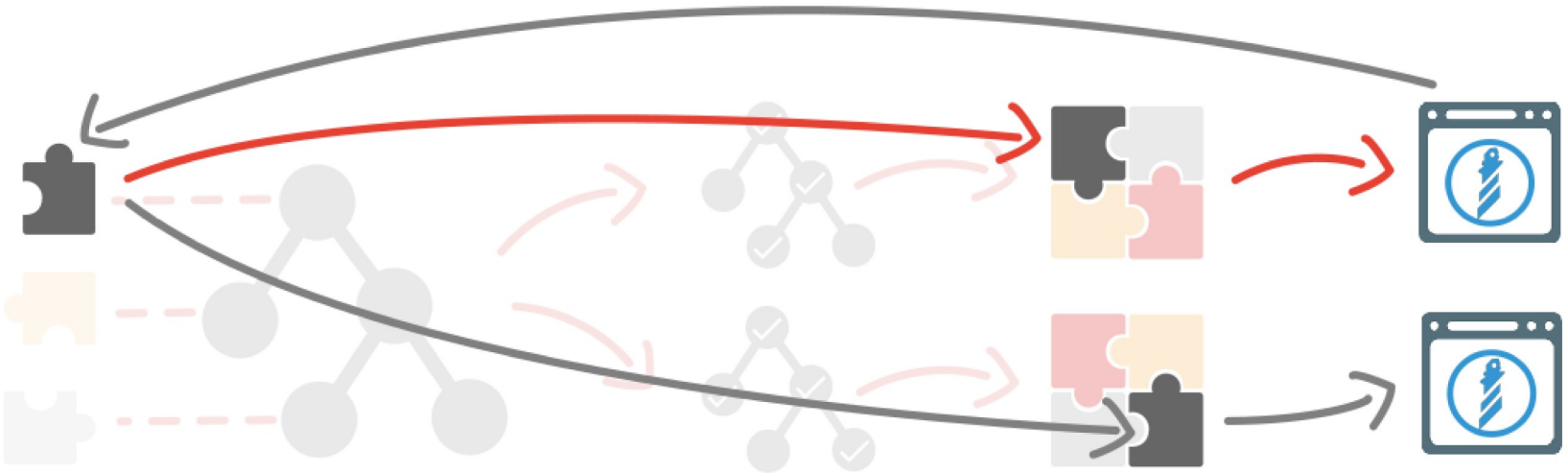
dependencies

Issues



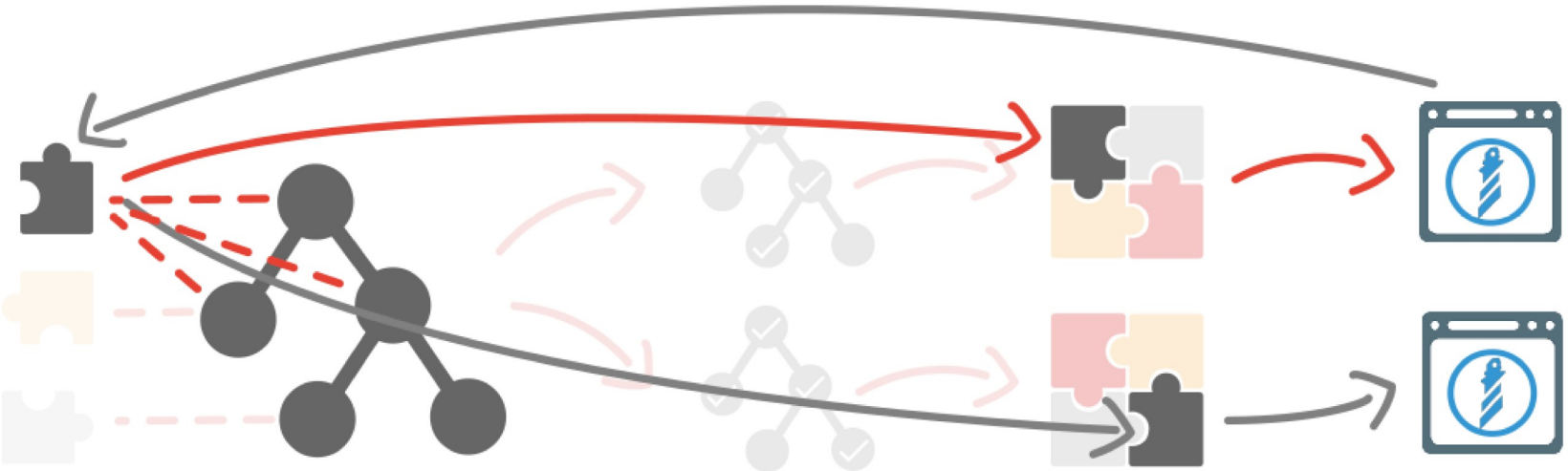
dependencies

Issues



dependencies

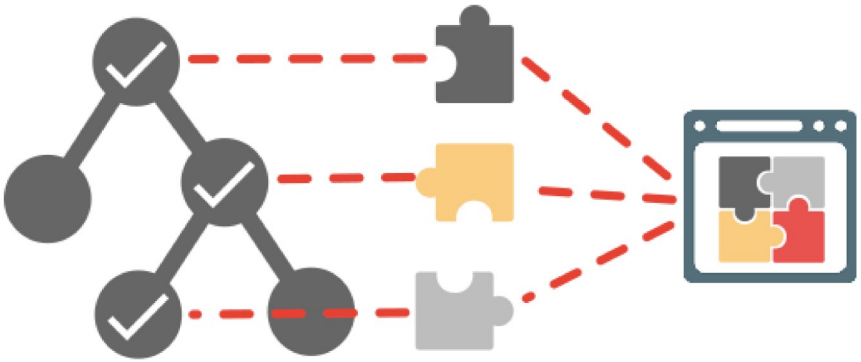
Issues



dependencies

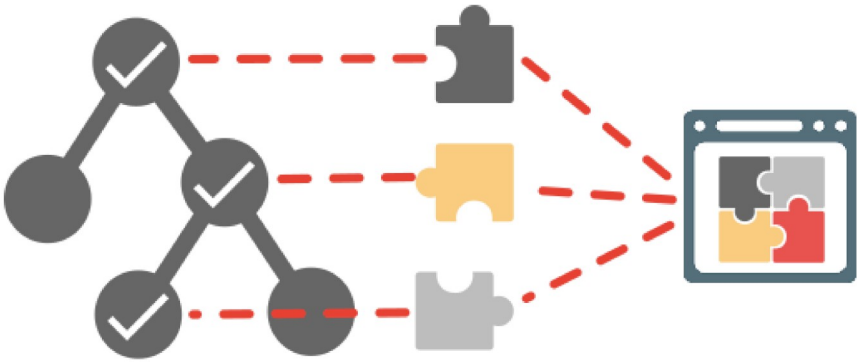
Proposed solution

Custom **Pharo** classes



Pharo classes
as executable containers for models

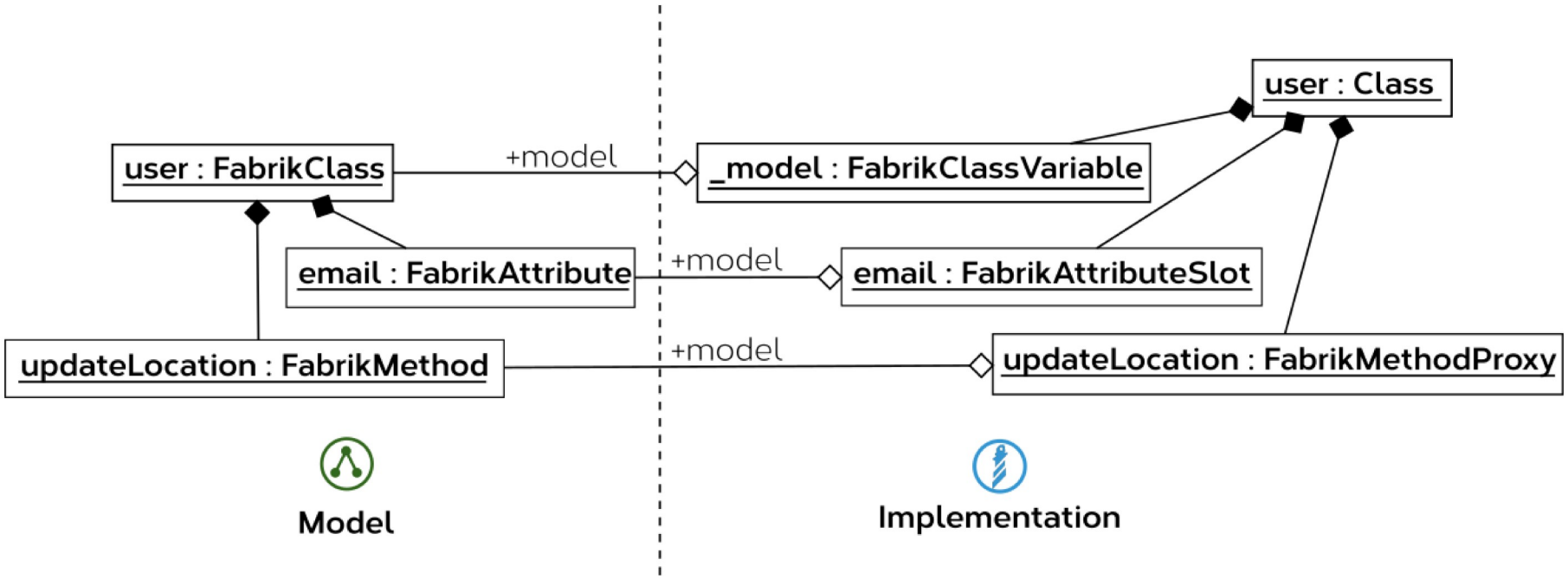
Custom **Pharo** classes



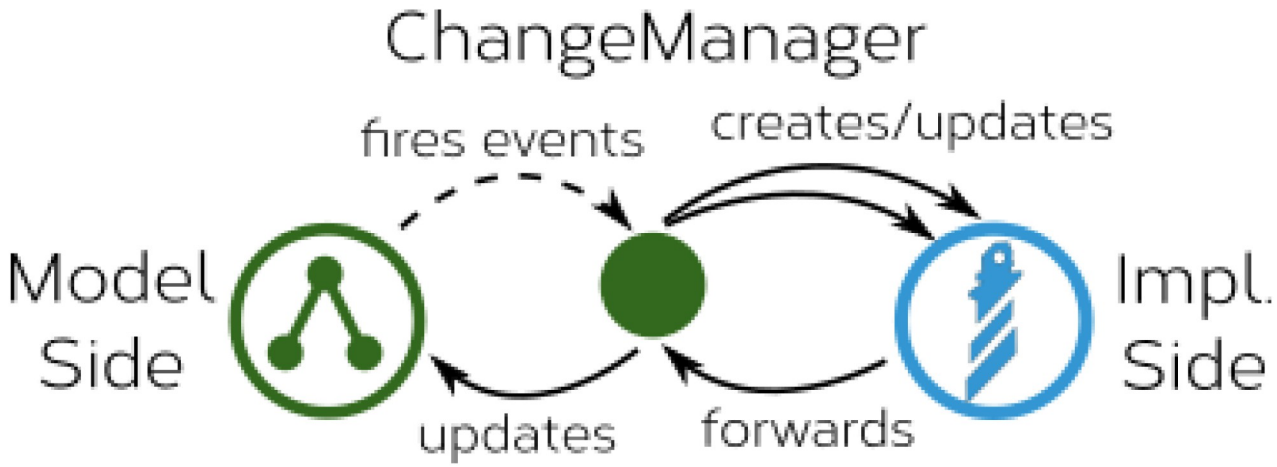
Pharo classes
as executable containers for models

→ hold data
→ delegate the behavior to models

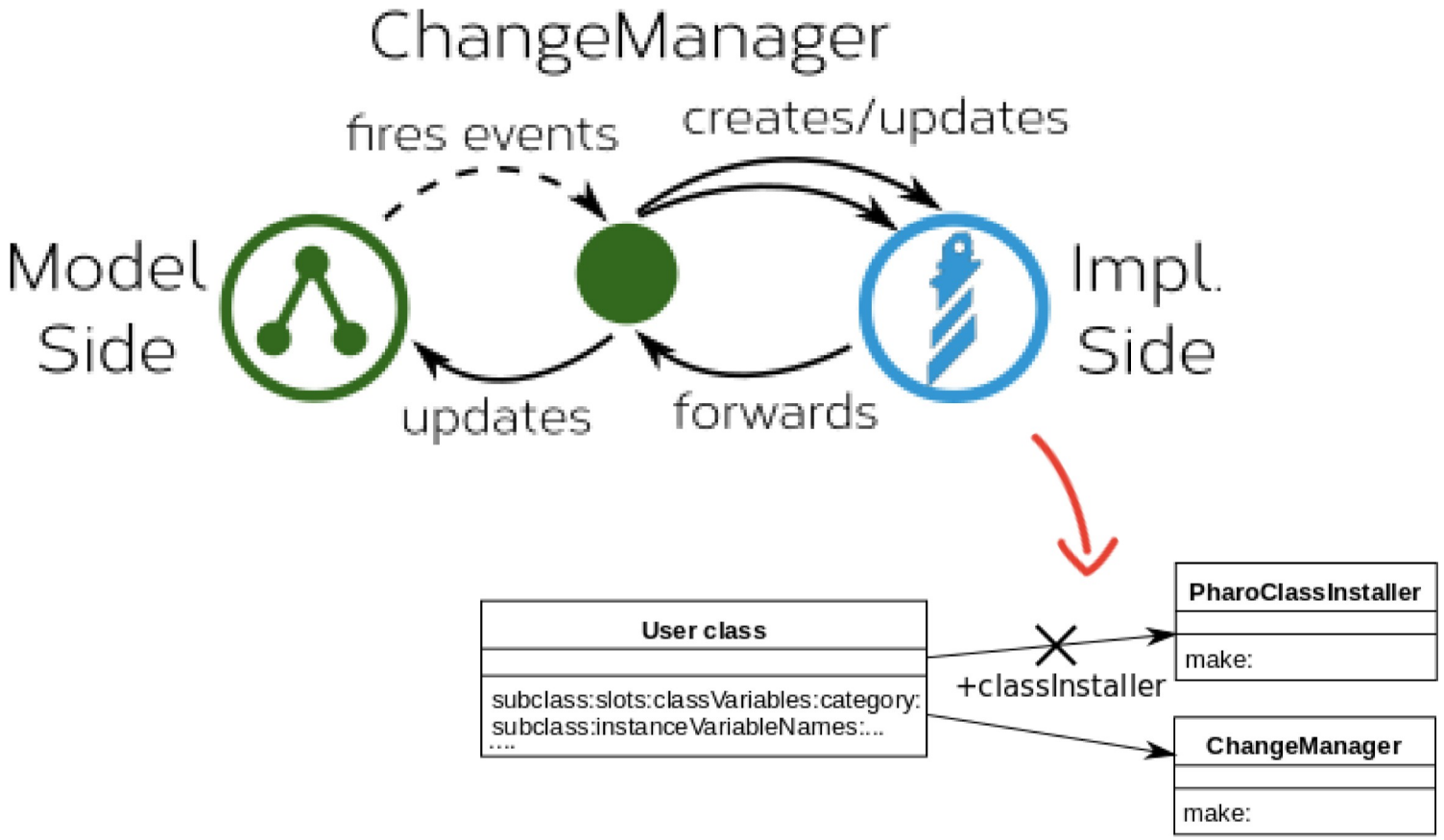
Custom **Pharo** classes



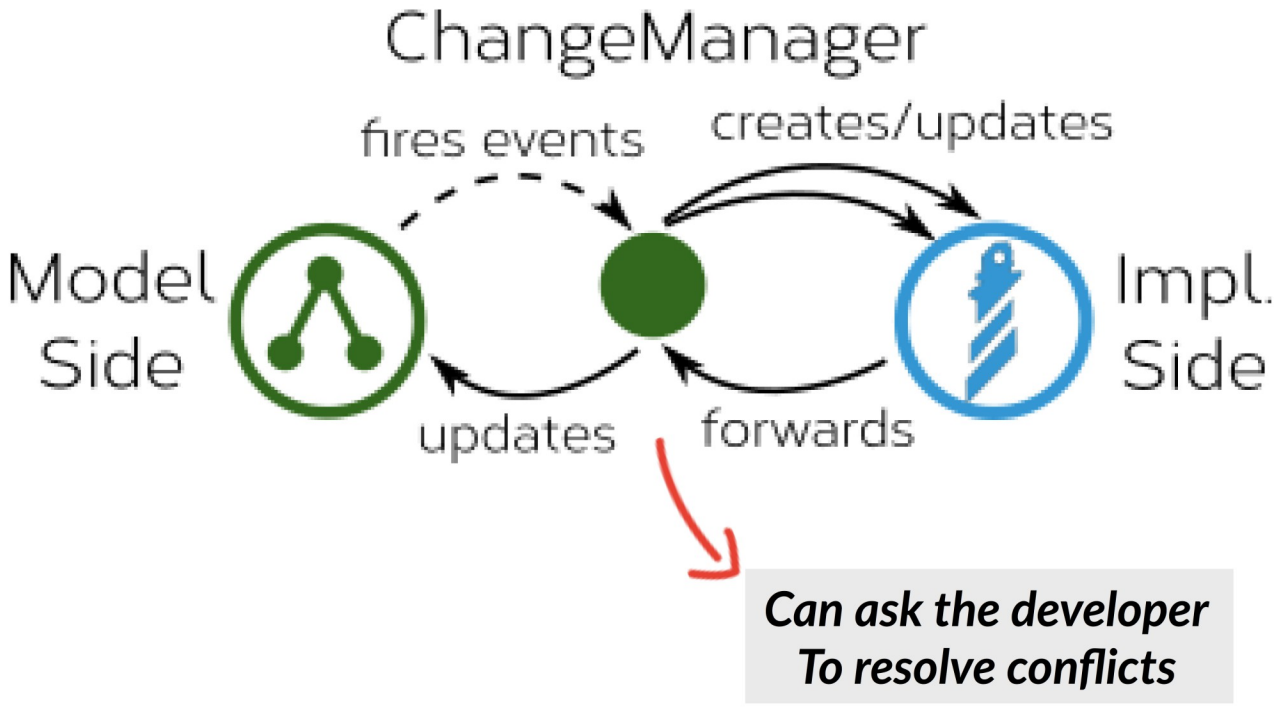
A mediator object



A mediator object



A mediator object



A mediator object

Updating models implies the running of The Forward process

Dynamic Round-Trip Engineering in the context of FOMDD

The *CrossFabrik* project

Multi-platform software prototyping & assessment

Let developers investigate solutions from any place

Without having to stop/restart anything

Implemented using:

Pharo classes as executable containers for models
a mediator object that ensure consistency

Dynamic Round-Trip Engineering in the context of FOMDD

Questions ?