16th OpenFOAM Workshop - June 11, 2021 - Training session

Flexible & efficient multiphysics simulations with the coupling library preCICE

Gerasimos Chourdakis, Technical University of Munich
chourdak@in.tum.de

Technical University of Munich

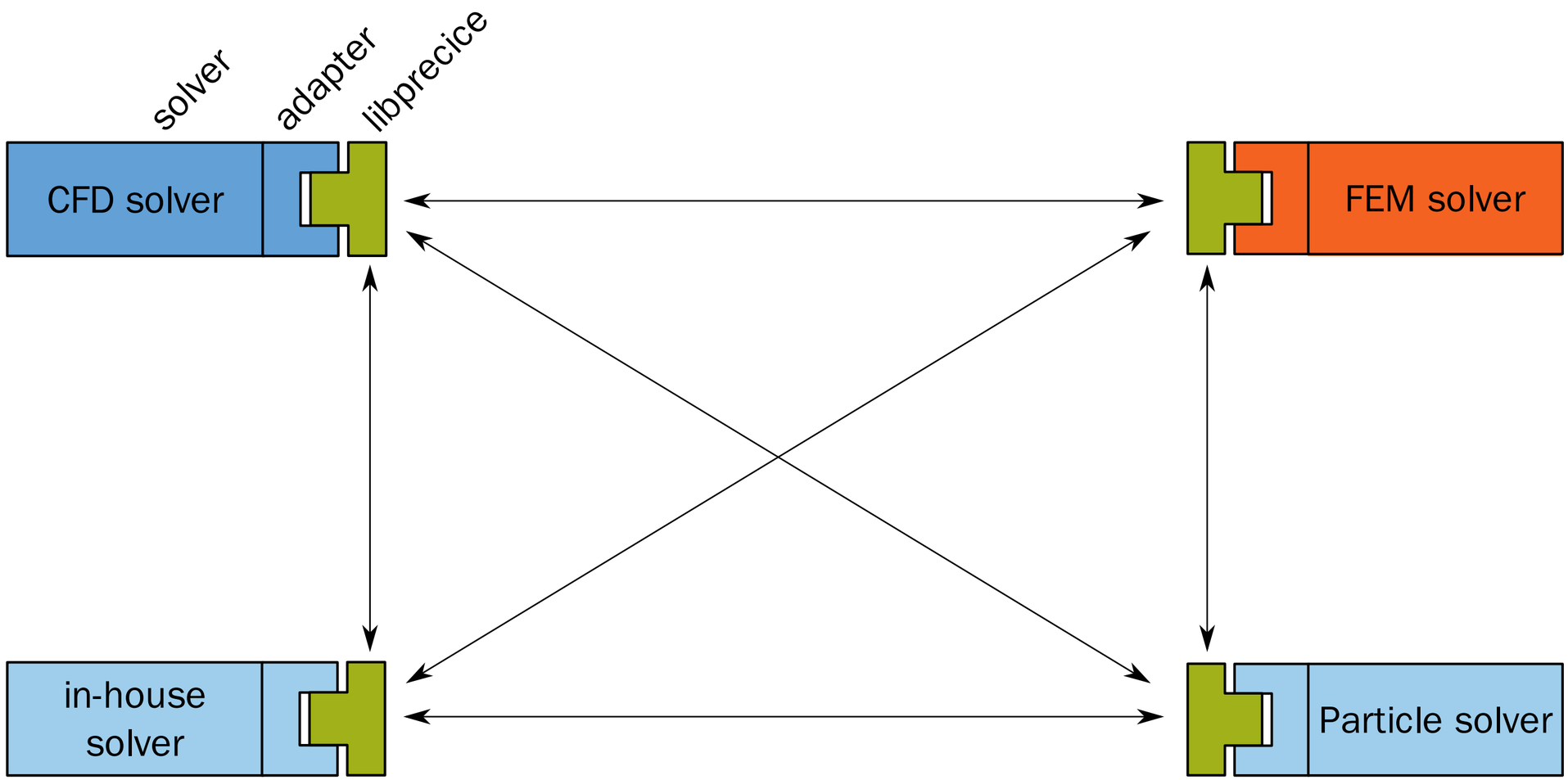
Some statistics

1. How familiar are you with preCICE? (poll)
2. What would you like to couple OpenFOAM with/for? Which version/solver? (chat)

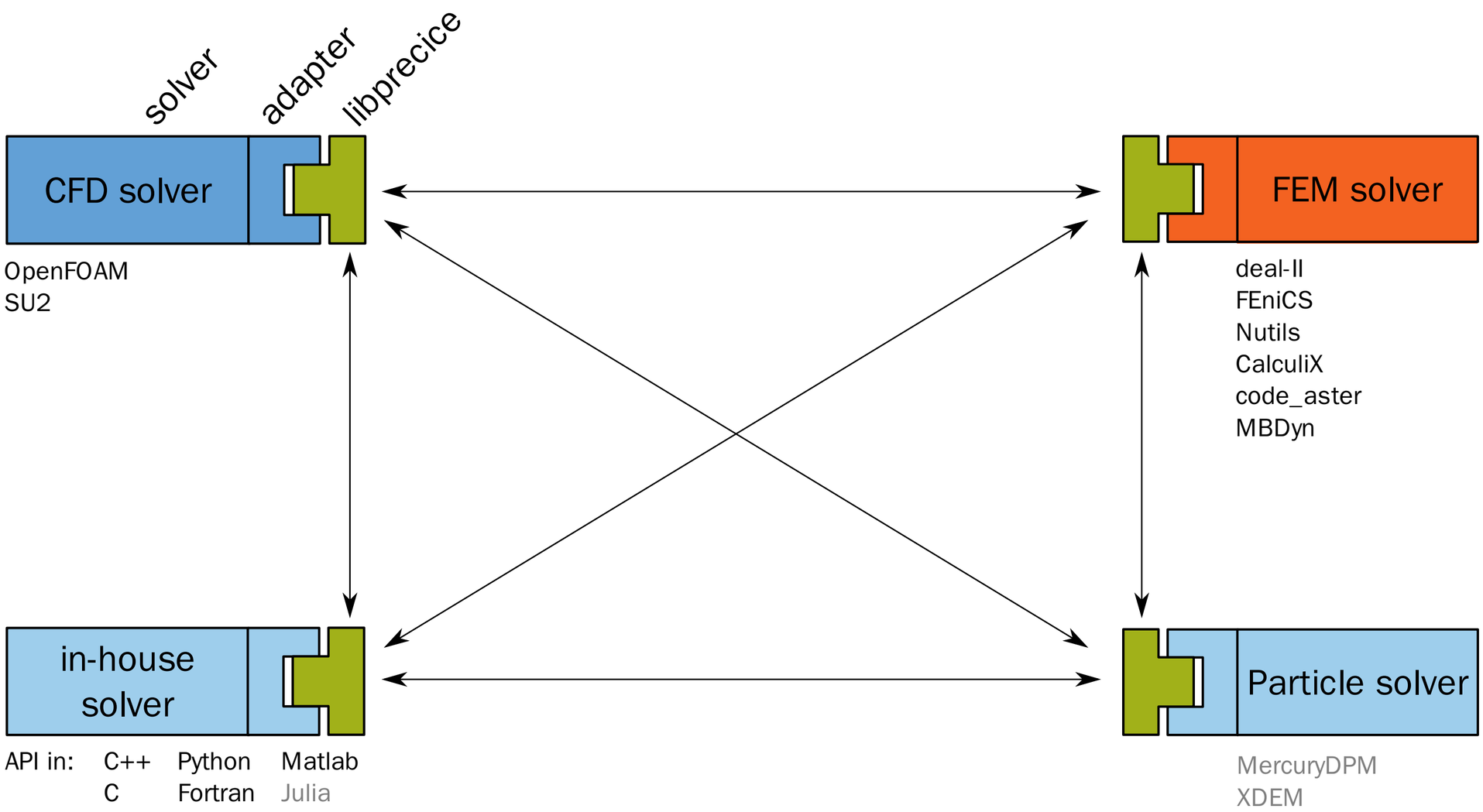
The people behind preCICE

The big picture

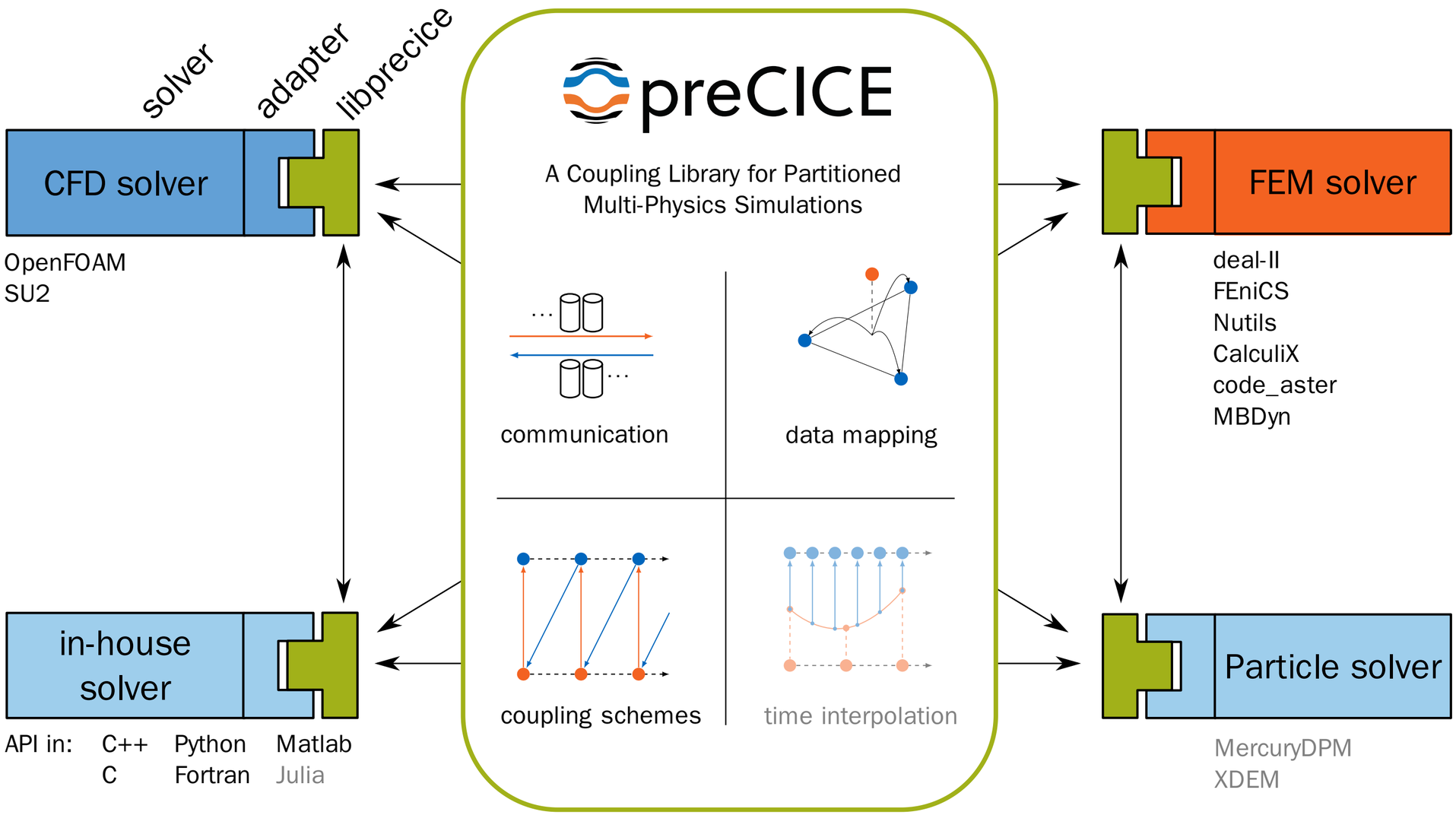
The big picture



The big picture



The big picture



This tutorial

- Level 1: Couple two simple Python solvers
- Level 2: Couple OpenFOAM with OpenFOAM
- Level 3: Couple OpenFOAM with an external solver

Level 1: Call the preCICE API in the code
Levels 2 & 3: Directly use off-the-shelf adapters

Organizational notes

1. You are not expected to try things live.
2. Ask questions in the chat, feel free to interrupt me.
3. Find these slides on GitHub:
   github.com/MakisH/ofw16-training
4. Everything presented here is free software. preCICE and all the adapters are developed publicly on https://github.com/precice/
5. Find all software installed in a demo virtual machine:
   precice.org/installation-vm.html

Level 1: Coupling two simple solvers

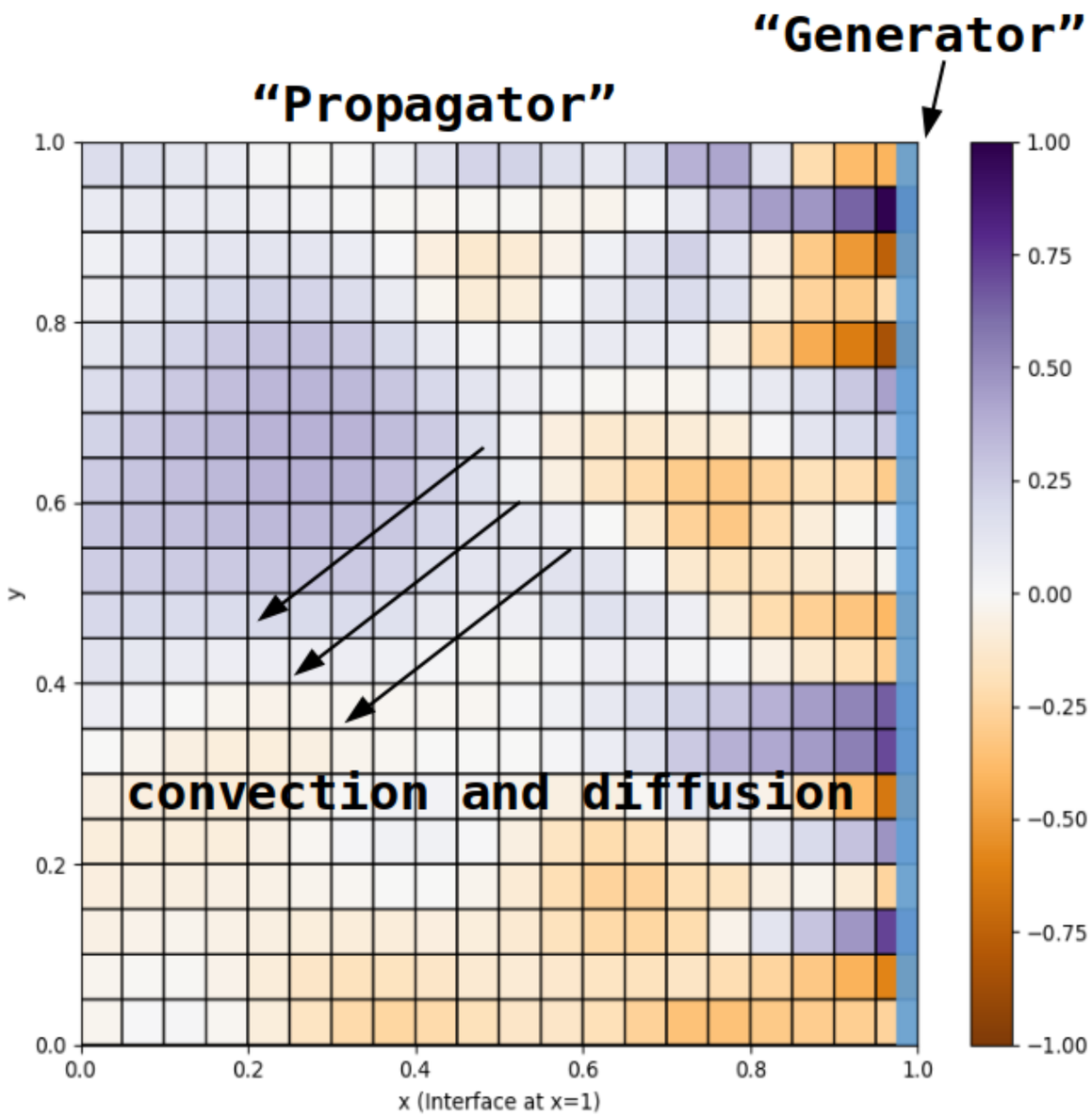
Dependencies

- preCICE v2 (e.g. packages for Ubuntu)
- Python 3
- Python packages `numpy`, `matplotlib`
- preCICE Python bindings:

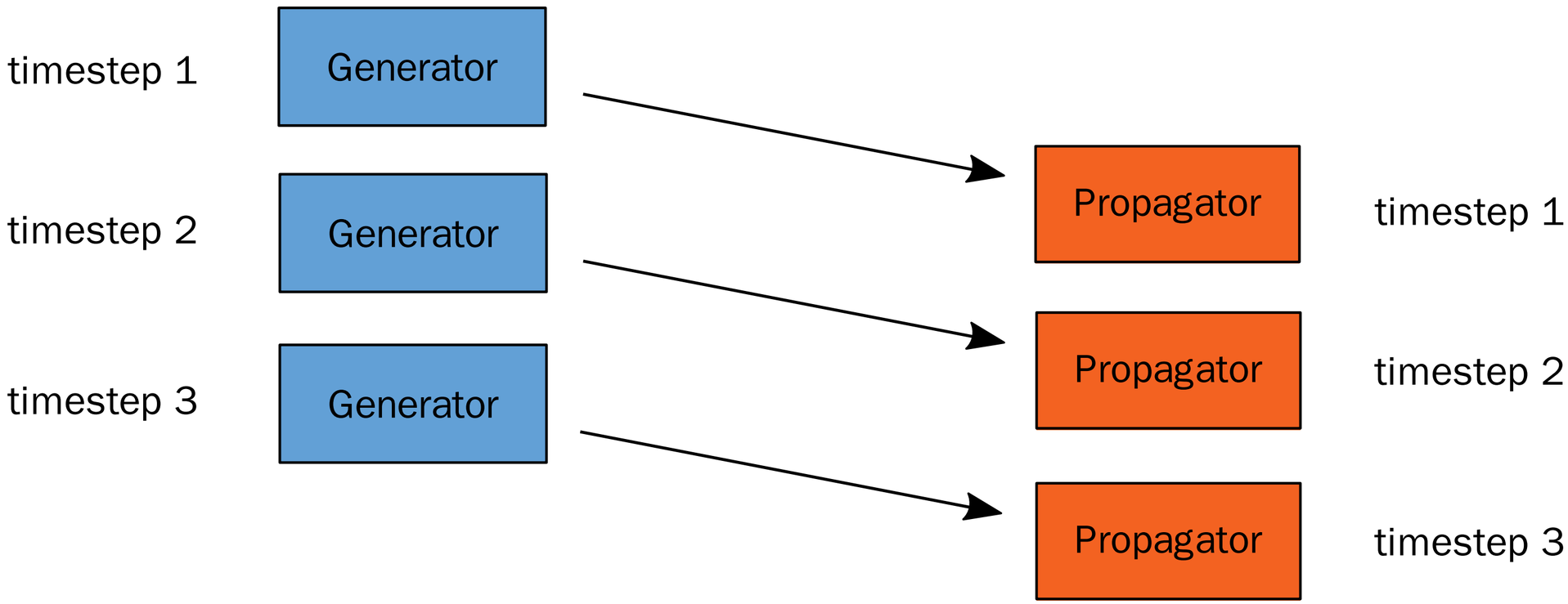
```
pip3 install --upgrade pip
pip3 install --user pyprecice
```

"Generator" and "Propagator"

- `generator.py`: generates random data in a 1D domain
- `propagator.py`: propagates boundary data over a 2D domain



Unidirectional coupling

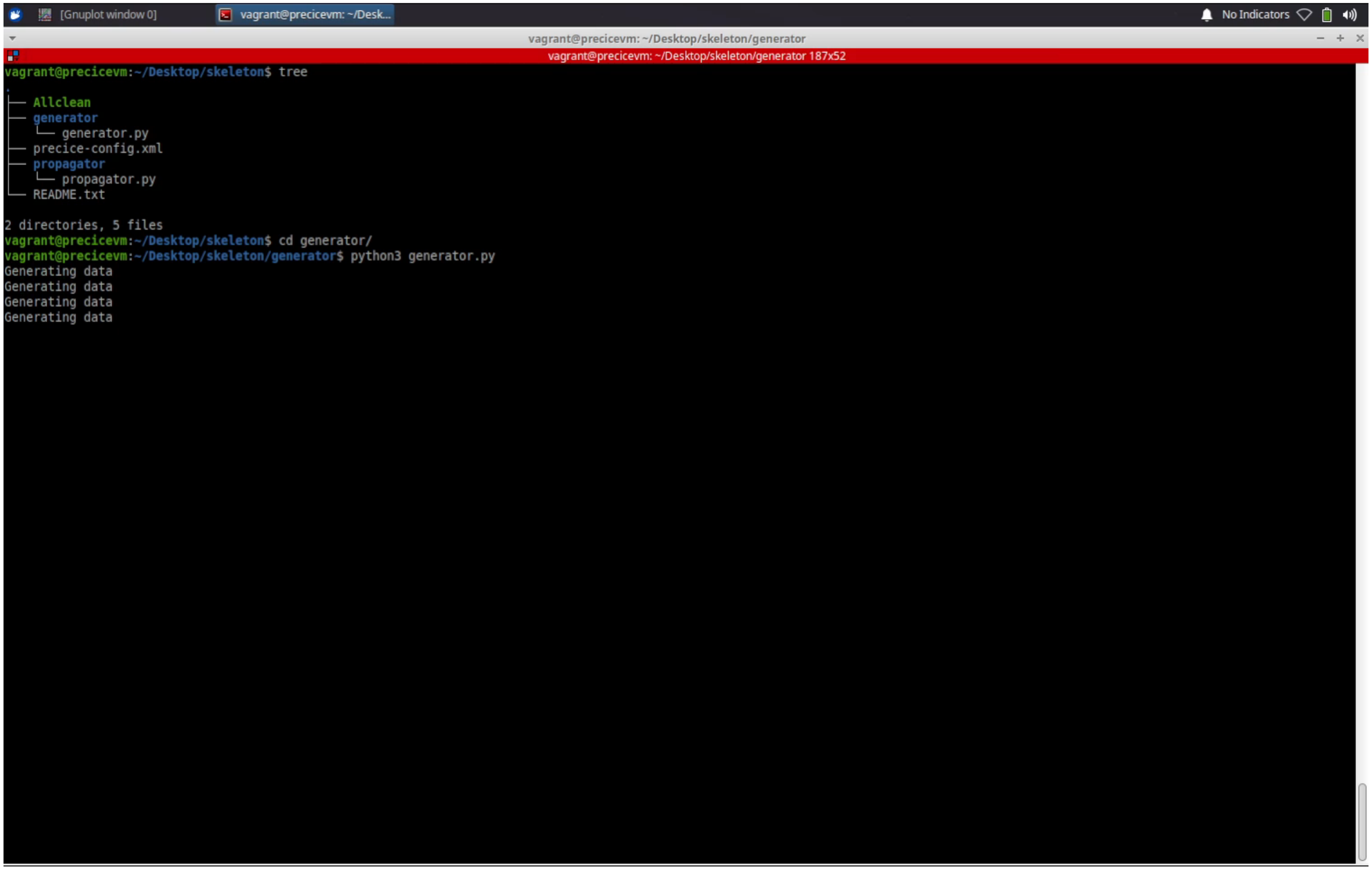


generator.py

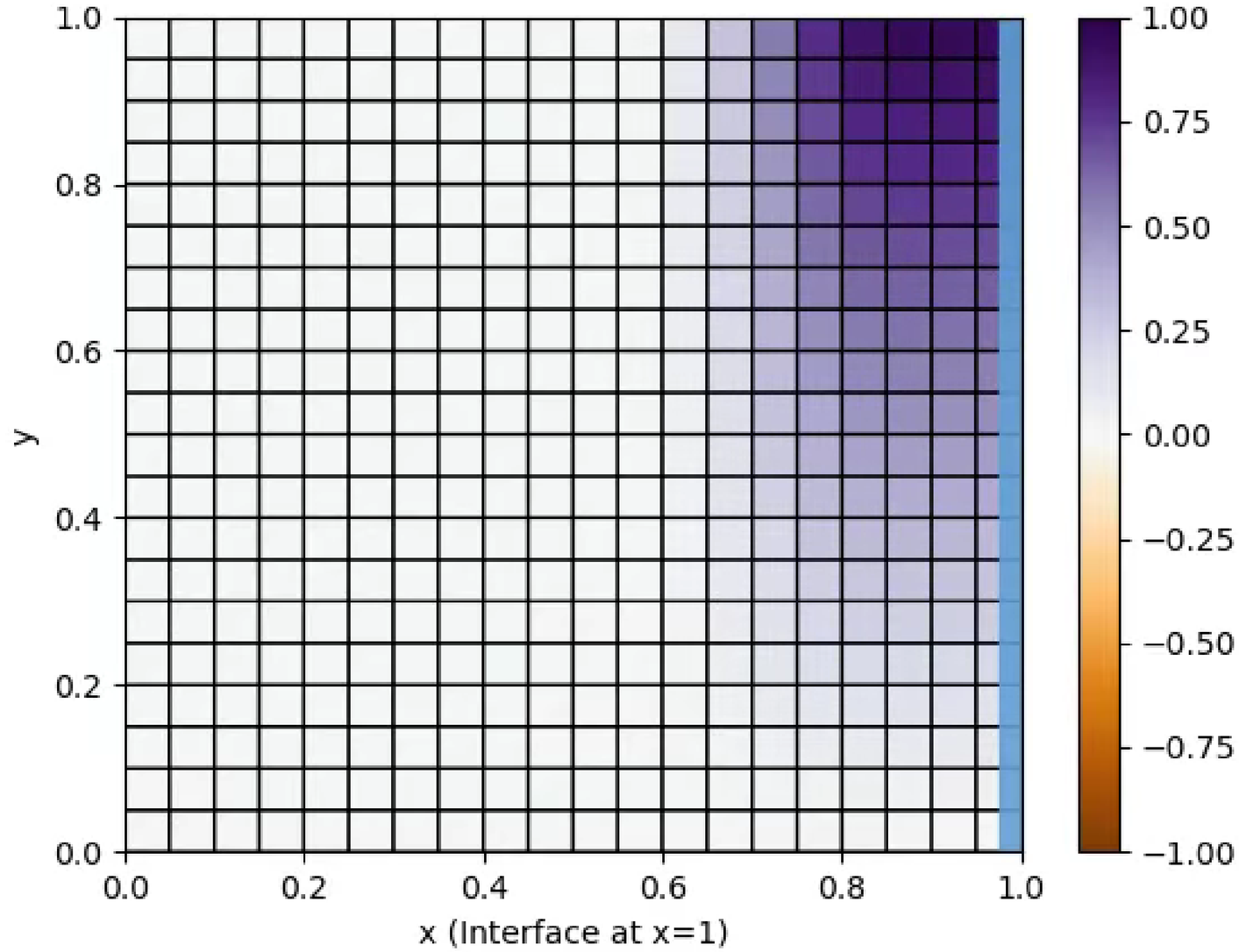
```
import numpy

# generate mesh
n = 20
y = numpy.linspace(0, 1, n + 1)

dt = 0.01
t = 0

while True:
  print("Generating data")
  u = 1 - 2 * numpy.random.rand(n)

  t = t + dt
  if(t > 0.1):
    break
```

propagator.py

```
# generate mesh
n = 20
x = numpy.linspace(0, 1, n+1)
y = numpy.linspace(0, 1, n+1)

# initial data, associated to cell centers
u = numpy.zeros([n, n])

dt = 0.01
t = 0

# boundary condition for u (arbitrary)
u[:,-1] = y[:-1]

while True:

  print("Propagating data")
```


[Gnuplot window 0]
vagrant@precicevm: ~/Desk...
No Indicators
vagrant@precicevm: ~/Desktop/skeleton/generator
vagrant@precicevm: ~/Desktop/skeleton/generator 187x52
vagrant@precicevm:~/Desktop/skeleton$ tree
.
├── Allclean
├── generator
│   └── generator.py
├── precice-config.xml
├── propagator
│   └── propagator.py
└── README.txt
2 directories, 5 files
vagrant@precicevm:~/Desktop/skeleton$ cd generator/
vagrant@precicevm:~/Desktop/skeleton/generator$ python3 generator.py
Generating data
Generating data
Generating data
Generating data

Uncoupled simulation



Import preCICE

```python
import numpy


# generate mesh
n = 20
y = numpy.linspace(0, 1, n + 1)

dt = 0.01
t = 0

while True:
  print("Generating data")
  u = 1 - 2 * numpy.random.rand(n)

  t = t + dt
  if(t > 0.1):
    break
```

Import preCICE

```python
import numpy
import precice

# generate mesh
n = 20
y = numpy.linspace(0, 1, n + 1)

dt = 0.01
t = 0

while True:
  print("Generating data")
  u = 1 - 2 * numpy.random.rand(n)

  t = t + dt
  if(t > 0.1):
    break
```

Configure preCICE

```python
import numpy
import precice

# generate mesh
n = 20
y = numpy.linspace(0, 1, n + 1)

# preCICE setup
participant_name      = "Generator"
config_file_name      = "../precice-config.xml"
solver_process_index = 0
solver_process_size  = 1
interface =
    precice.Interface(
                        participant_name,
                        config_file_name,
                        solver_process_index,
```

Configure preCICE

```
 5 n = 20
 6 y = numpy.linspace(0, 1, n + 1)
 7
 8 # preCICE setup
 9 participant_name      = "Generator"
10 config_file_name      = "../precice-config.xml"
11 solver_process_index = 0
12 solver_process_size  = 1
13 interface =
14     precice.Interface(
15                         participant_name,
16                         config_file_name,
17                         solver_process_index,
18                         solver_process_size
19                       )
20
21 dt = 0.01
22 t = 0
```

Define the coupling mesh

```
 5 n = 20
 6 y = numpy.linspace(0, 1, n + 1)
 7
 8 # preCICE setup
 9 participant_name      = "Generator"
10 config_file_name      = "../precice-config.xml"
11 solver_process_index = 0
12 solver_process_size  = 1
13 interface =
14     precice.Interface(
15                         participant_name,
16                         config_file_name,
17                         solver_process_index,
18                         solver_process_size
19                       )
20
21 # Get the preCICE mesh id
22 mesh_name  = "Generator-Mesh"
```

Define the coupling mesh

```
16                          config_file_name,
17                          solver_process_index,
18                          solver_process_size
19                      )
20
21  # Get the preCICE mesh id
22  mesh_name  = "Generator-Mesh"
23  mesh_id    = interface.get_mesh_id(mesh_name)
24
25  # Define the coupling mesh
26  vertices   = [[1, y0] for y0 in y[:-1]]
27  vertex_ids = interface.set_mesh_vertices(mesh_id, vertices)
28
29  dt = 0.01
30  t = 0
31
32  while True:
```

Initialize and finalize preCICE

```
                                    config_file_name,
                                    solver_process_index,
                                    solver_process_size
                                )

# Get the preCICE mesh id
mesh_name  = "Generator-Mesh"
mesh_id    = interface.get_mesh_id(mesh_name)

# Define the coupling mesh
vertices   = [[1, y0] for y0 in y[:-1]]
vertex_ids = interface.set_mesh_vertices(mesh_id, vertices)

interface.initialize()

dt = 0.01
t = 0
```

Initialize and finalize preCICE

```
25 # Define the coupling mesh
26 vertices   = [[1, y0] for y0 in y[:-1]]
27 vertex_ids = interface.set_mesh_vertices(mesh_id, vertices)
28
29 interface.initialize()
30
31 dt = 0.01
32 t = 0
33
34 while True:
35   print("Generating data")
36   u = 1 - 2 * numpy.random.rand(n)
37
38   t = t + dt
39   if(t > 0.1):
40     break
41
42 interface.finalize()
```

Advance the coupling

```
27 vertex_ids = interface.set_mesh_vertices(mesh_id, vertices)
28
29 interface.initialize()
30
31 dt = 0.01
32 t = 0
33
34 while True:
35   print("Generating data")
36   u = 1 - 2 * numpy.random.rand(n)
37
38
39
40   t = t + dt
41   if(t > 0.1):
42     break
43
44 interface.finalize()
```

Advance the coupling

```
27 vertex_ids = interface.set_mesh_vertices(mesh_id, vertices)
28
29 interface.initialize()
30
31 dt = 0.01
32 t = 0
33
34 while True:
35   print("Generating data")
36   u = 1 - 2 * numpy.random.rand(n)
37
38
39
40   t = t + dt
41   if(t > 0.1):
42     break
43
44 interface.finalize()
```

Advance the coupling

```
27   vertex_ids = interface.set_mesh_vertices(mesh_id, vertices)
28
29 interface.initialize()
30
31 dt = 0.01
32 t = 0
33
34 while True:
35   print("Generating data")
36   u = 1 - 2 * numpy.random.rand(n)
37
38   interface.advance(dt)
39
40   t = t + dt
41   if(t > 0.1):
42     break
43
44 interface.finalize()
```

Advance the coupling

```
27 vertex_ids = interface.set_mesh_vertices(mesh_id, vertices)
28
29 interface.initialize()
30
31 dt = 0.01
32 t = 0
33
34 while interface.is_coupling_ongoing():
35    print("Generating data")
36    u = 1 - 2 * numpy.random.rand(n)
37
38    interface.advance(dt)
39
40    t = t + dt
41
42
43
44 interface.finalize()
```

Advance the coupling

```
27 vertex_ids = interface.set_mesh_vertices(mesh_id, vertices)
28
29 interface.initialize()
30
31 dt = 0.01
32 t = 0
33
34 while interface.is_coupling_ongoing():
35    print("Generating data")
36    u = 1 - 2 * numpy.random.rand(n)
37
38    precice_dt = interface.advance(dt)
39
40    t = t + dt
41
42
43
44 interface.finalize()
```

Advance the coupling

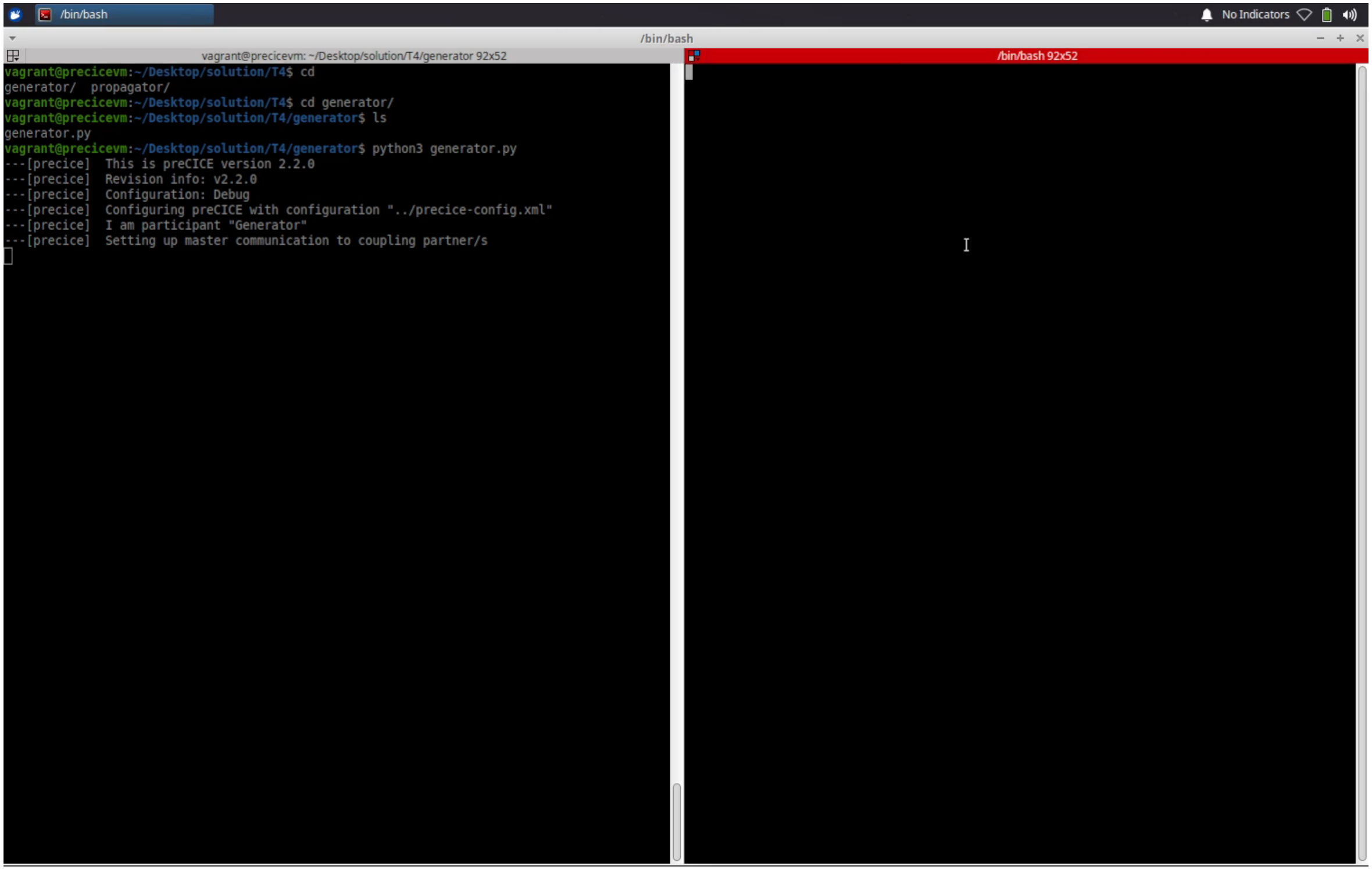
```
27 vertex_ids = interface.set_mesh_vertices(mesh_id, vertices)
28
29 precice_dt = interface.initialize()
30
31 dt = 0.01
32 t = 0
33
34 while interface.is_coupling_ongoing():
35     dt = np.minimum(dt, precice_dt)
36
37     print("Generating data")
38     u = 1 - 2 * numpy.random.rand(n)
39
40     precice_dt = interface.advance(dt)
41
42     t = t + dt
43
44 interface.finalize()
```

Advance the coupling

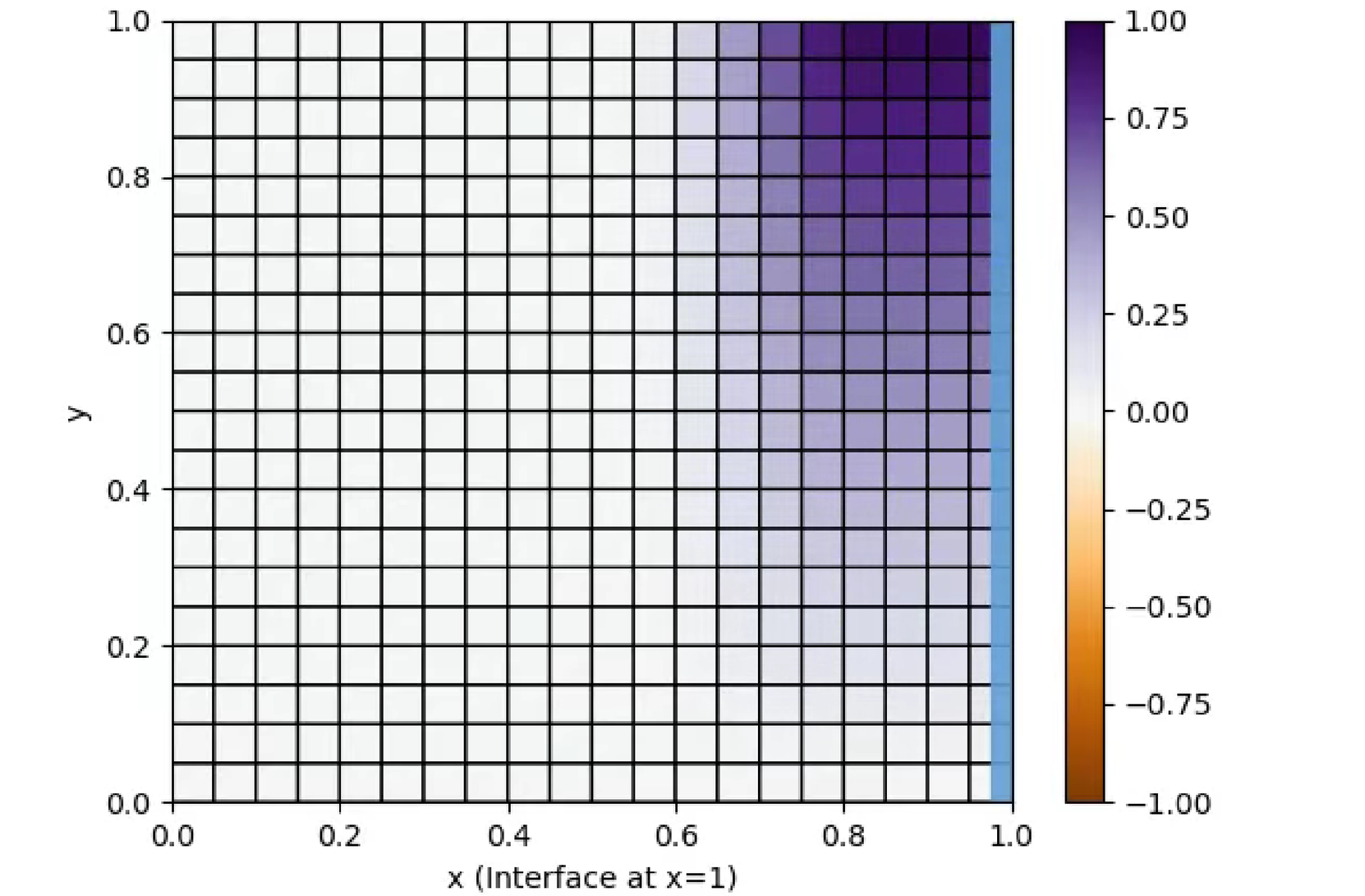
```
27 vertex_ids = interface.set_mesh_vertices(mesh_id, vertices)
28
29 precice_dt = interface.initialize()
30
31 dt = 0.01
32 t = 0
33
34 while interface.is_coupling_ongoing():
35   dt = np.minimum(dt, precice_dt)
36
37   print("Generating data")
38   u = 1 - 2 * numpy.random.rand(n)
39
40   precice_dt = interface.advance(dt)
41
42   t = t + dt
43
44 interface.finalize()
```

Not there yet, but let's run it

(similar changes in propagator.py - try it at home)


/bin/bash
No Indicators
/bin/bash
vagrant@precicevm: ~/Desktop/solution/T4/generator 92x52
/bin/bash 92x52
vagrant@precicevm:~/Desktop/solution/T4$ cd
generator/ propagator/
vagrant@precicevm:~/Desktop/solution/T4$ cd generator/
vagrant@precicevm:~/Desktop/solution/T4/generator$ ls
generator.py
vagrant@precicevm:~/Desktop/solution/T4/generator$ python3 generator.py
---[precice] This is preCICE version 2.2.0
---[precice] Revision info: v2.2.0
---[precice] Configuration: Debug
---[precice] Configuring preCICE with configuration "../precice-config.xml"
---[precice] I am participant "Generator"
---[precice] Setting up master communication to coupling partner/s

Nothing happening?



Did we forget something?

Write & read data

```
20
21 # Get the preCICE mesh id
22 mesh_name  = "Generator-Mesh"
23 mesh_id    = interface.get_mesh_id(mesh_name)
24
25 # Define the coupling mesh
26 vertices   = [[1, y0] for y0 in y[:-1]]
27 vertex_ids = interface.set_mesh_vertices(mesh_id, vertices)
28
29
30
31
32
33 precice_dt = interface.initialize()
34
35 dt = 0.01
36 t = 0
```

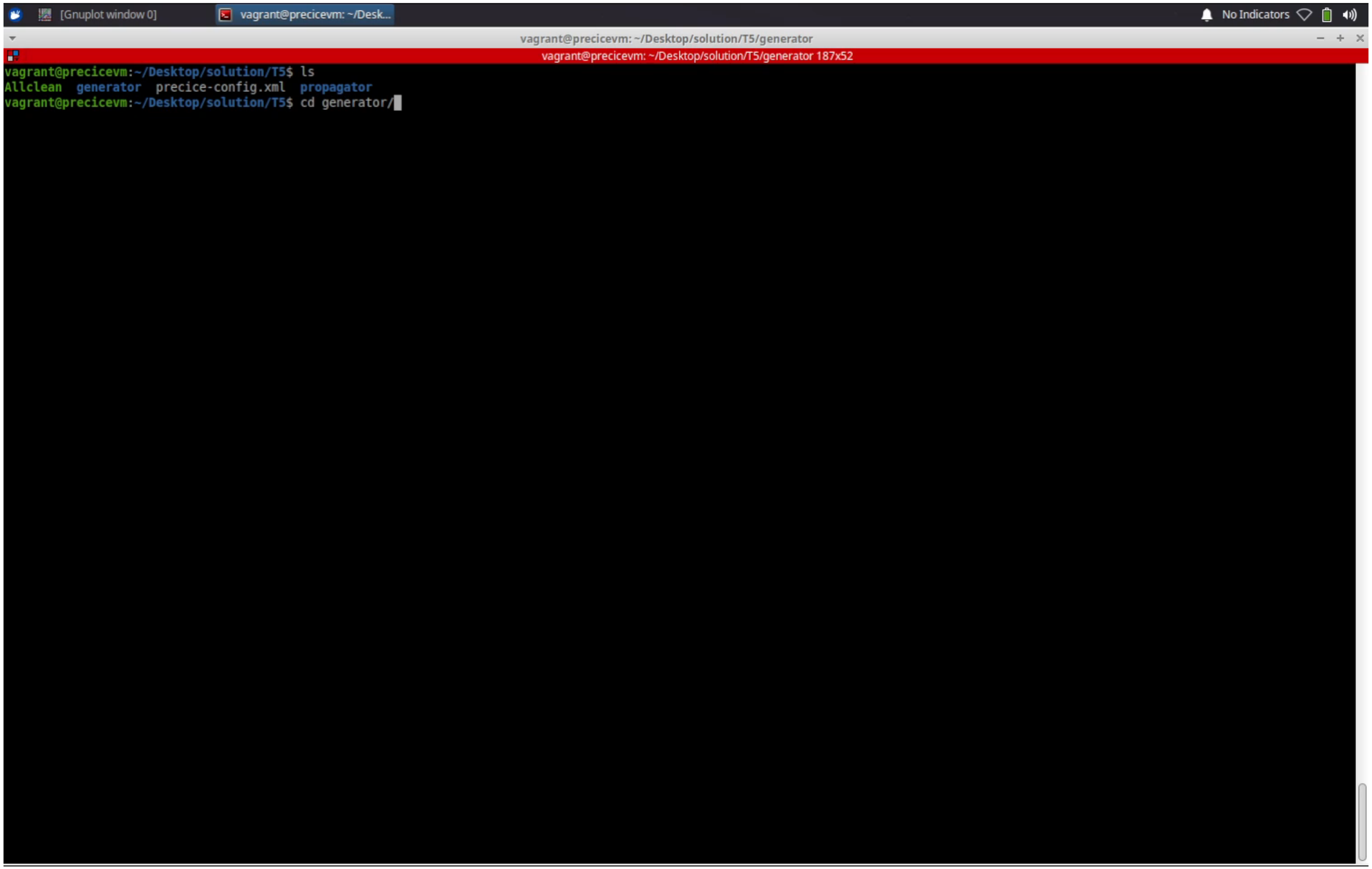
Write & read data

```
20
21 # Get the preCICE mesh id
22 mesh_name  = "Generator-Mesh"
23 mesh_id    = interface.get_mesh_id(mesh_name)
24
25 # Define the coupling mesh
26 vertices   = [[1, y0] for y0 in y[:-1]]
27 vertex_ids = interface.set_mesh_vertices(mesh_id, vertices)
28
29 # Get the exchanged data id
30 data_name = "Data"
31 data_id = interface.get_data_id(data_name, mesh_id)
32
33 precice_dt = interface.initialize()
34
35 dt = 0.01
36 t = 0
```

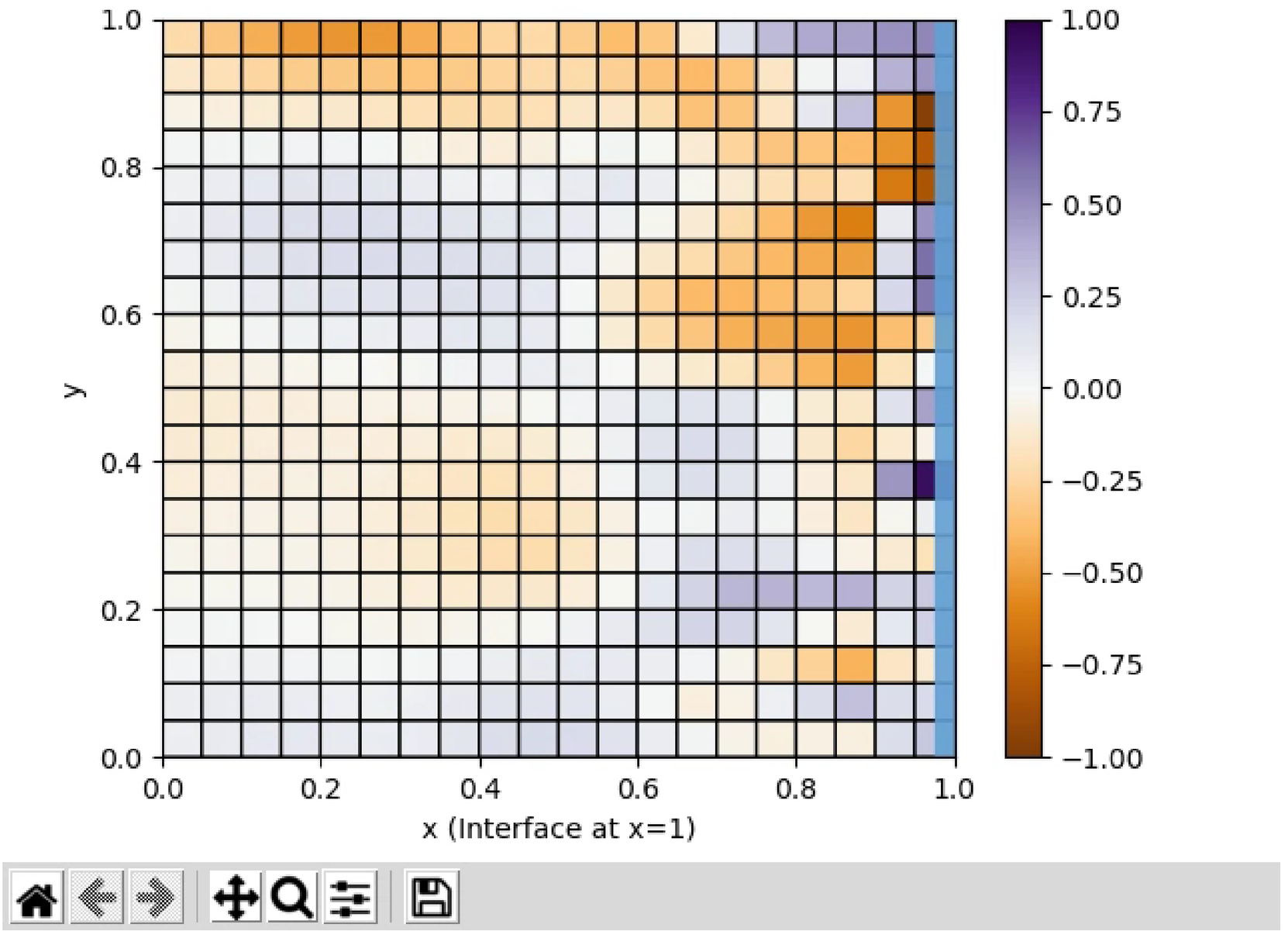
Write & read data

```
precice_dt = interface.initialize()

dt = 0.01
t = 0

while interface.is_coupling_ongoing():
  dt = np.minimum(dt, precice_dt)

  print("Generating data")
  u = 1 - 2 * numpy.random.rand(n)

  interface.write_block_scalar_data(data_id, vertex_ids, u)
  precice_dt = interface.advance(dt)
  # interface.read_block_scalar_data(data_id, vertex_ids, u)

  t = t + dt

interface.finalize()
```

It should work now!

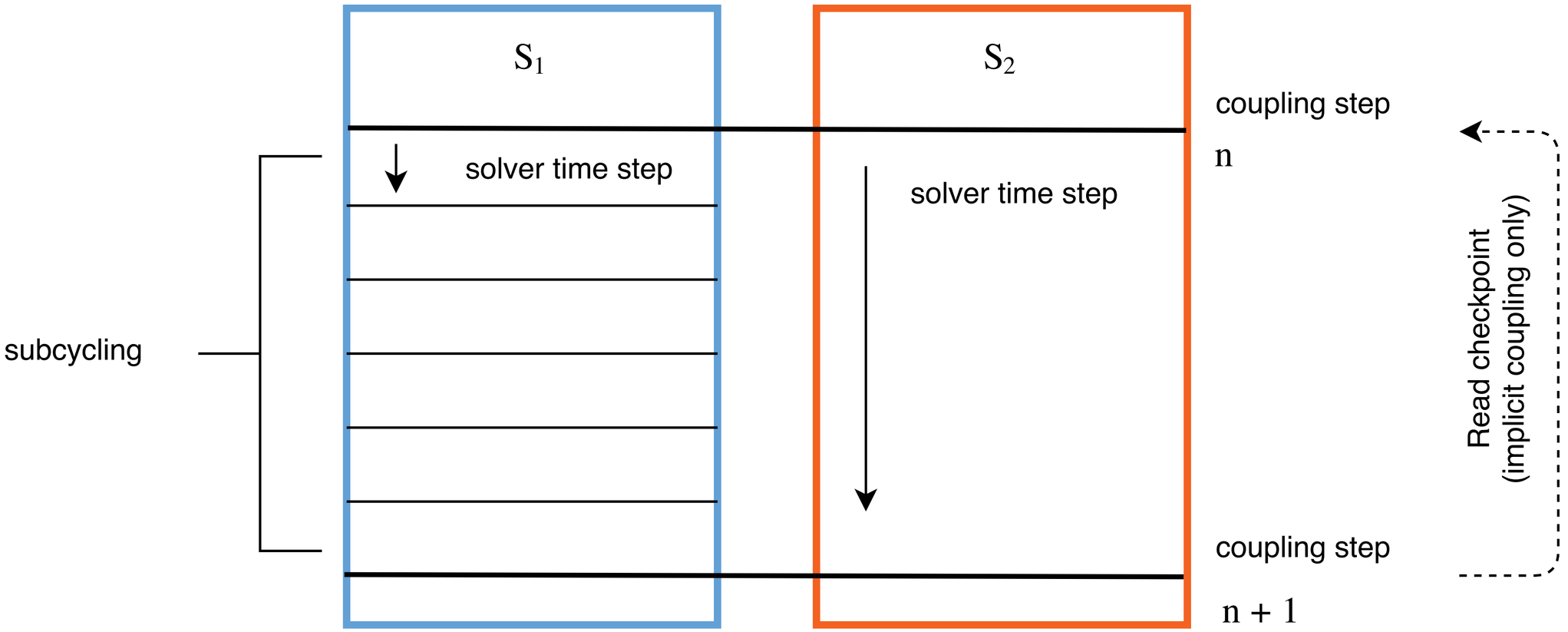

[Gnuplot window 0]
vagrant@precicevm: ~/Desk...
No Indicators
vagrant@precicevm: ~/Desktop/solution/T5/generator
vagrant@precicevm: ~/Desktop/solution/T5/generator 187x52
vagrant@precicevm:~/Desktop/solution/T5$ ls
Allclean generator precice-config.xml propagator
vagrant@precicevm:~/Desktop/solution/T5$ cd generator/

Data is being transfered!



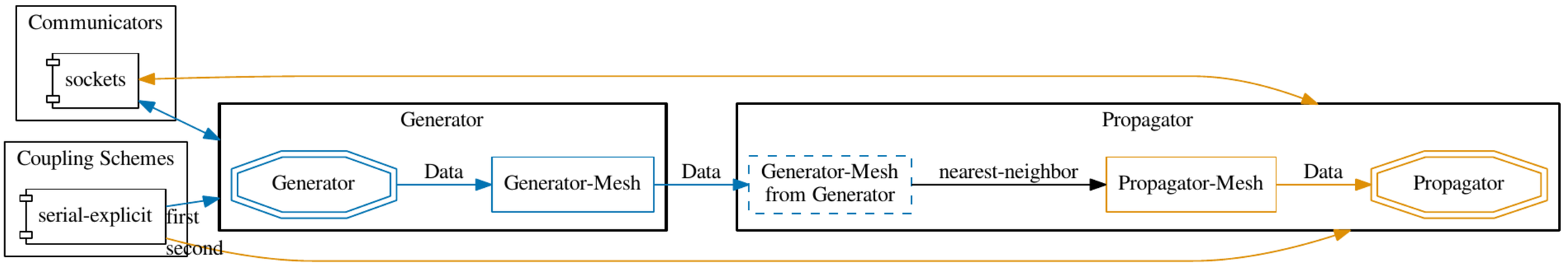
Most basic case: uni-directional, serial-explicit, nearest-neighbor mapping, ...

Side note: Subcycling



preCICE configuration file

Visual representation of precice-config.xml using the config visualizer:



preCICE configuration file

precice-config.xml:

```xml
<precice-configuration>
  <solver-interface dimensions="2">

  <data:scalar name="Data"/>

  <mesh name="Generator-Mesh">
    <use-data name="Data"/>
  </mesh>

  <mesh name="Propagator-Mesh">
    <use-data name="Data" />
  </mesh>

  <participant name="Generator">
    <use-mesh name="Generator-Mesh" provide="yes"/>
    <write-data name="Data" mesh="Generator-Mesh"/>
  </participant>
```

preCICE configuration file

precice-config.xml:

```xml
<precice-configuration>
  <solver-interface dimensions="2">

    <data:scalar name="Data"/>

    <mesh name="Generator-Mesh">
      <use-data name="Data"/>
    </mesh>

    <mesh name="Propagator-Mesh">
      <use-data name="Data" />
    </mesh>

    <participant name="Generator">
      <use-mesh name="Generator-Mesh" provide="yes"/>
      <write-data name="Data" mesh="Generator-Mesh"/>
    </participant>
```

preCICE configuration file

precice-config.xml:

```xml
<precice-configuration>
  <solver-interface dimensions="2">

  <data:scalar name="Data"/>

  <mesh name="Generator-Mesh">
    <use-data name="Data"/>
  </mesh>

  <mesh name="Propagator-Mesh">
    <use-data name="Data" />
  </mesh>

  <participant name="Generator">
    <use-mesh name="Generator-Mesh" provide="yes"/>
    <write-data name="Data" mesh="Generator-Mesh"/>
  </participant>
```

preCICE configuration file

precice-config.xml:

```
 7       <use-data name="Data"/>
 8     </mesh>
 9
10     <mesh name="Propagator-Mesh">
11       <use-data name="Data" />
12     </mesh>
13
14     <participant name="Generator">
15       <use-mesh name="Generator-Mesh" provide="yes"/>
16       <write-data name="Data" mesh="Generator-Mesh"/>
17     </participant>
18
19     <participant name="Propagator">
20       <use-mesh name="Generator-Mesh" from="Generator"/>
21       <use-mesh name="Propagator-Mesh" provide="yes"/>
22       <mapping:nearest-neighbor direction="read"
23         from="Generator-Mesh" to="Propagator-Mesh"
```