OPM-OP

Python In Flow

OPM-OP AS
Heyerdahlsvei 12b
777 Oslo, Norway
T: +(47)-9268-5704
F: +(47)-9268-5704
E: support@opm-op.com

OPM-OP Contents

1. Python and C++ - pybind11
2. Python wrappers
3. Embedding python in flow:
    i. PYINPUT
    ii. PYACTION
4. Simulator in Python

OPM-OP Python And C++

- CPython is a C application
- Interface through shared libraries and a C API.
- Extending and embedding – quite similar
- We use `pybind11` to reduce the boilerplate

OPM-OP Extending Python I

1. Add some boilerplate for Python runtime
2. Compile as shared library
3. => Python module!

```
static inline string_view strip_comments( string_view str ) {
    return { str.begin(),
             find_terminator( str.begin(), str.end(), find_comment() ) };
}

template< typename Itr >
inline Itr trim_left( Itr begin, Itr end ) {
    return std::find_if_not( begin, end, RawConsts::is_separator() );
}

template< typename Itr >
inline Itr trim_right( Itr begin, Itr end ) {

    std::reverse_iterator< Itr > rbegin( end );
    std::reverse_iterator< Itr > rend( begin );

    return std::find_if_not( rbegin, rend, RawConsts::is_separator() ).base();
}

inline string_view trim( string_view str ) {
    auto fst = trim_left( str.begin(), str.end() );
    auto lst = trim_right( fst, str.end() );
    return { fst, lst };
}
```

OPM-OP Extending Python II

```c
#include <Python.h>

static PyObject *
add_method(PyObject *self, PyObject *args)
{
    double arg1;
    double arg2;
    if (!PyArg_ParseTuple(args, "dd", &arg1, &arg2))
        return NULL;

    double result = arg1 + arg2;
    return Py_BuildValue("d", result);
}


static PyMethodDef xmath_methods[] = {
    {"add", add_method, METH_VARARGS, "Add two number
    {NULL, NULL, 0, NULL}   /* sentinel */
};


static struct PyModuleDef xmathmodule = {
    PyModuleDef_HEAD_INIT,
    "xmath",
    "module for basic maths",
    -1,
    xmath_methods
};


PyMODINIT_FUNC
PyInit_xmath(void)
{
    return PyModule_Create(&xmathmodule);
}
```

```
$ gcc -o xmath.so -shared add.c -I /usr/include/python3.7
```

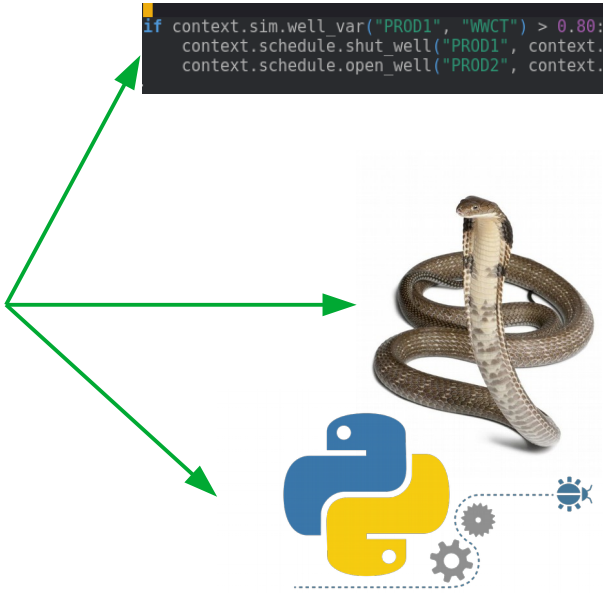
```
hove@ws:~/tmp$ python3
Python 3.7.3 (default, Apr  3 2019, 05:39:12)
[GCC 8.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import xmath
>>> sum = xmath.add(100,200)
>>> print(sum)
300.0
>>>
```

OPM-OP Embedding Python

1. Main application is in C/C++
2. Link with `libpython`
3. Call out to small snippets of Python code

```
   58 Reading WELSPECS in file /home/hove/work/OPM/opm-tests/spe1/SPE1CASE2.DATA, line 384
   59 Reading COMPDAT  in file /home/hove/work/OPM/opm-tests/spe1/SPE1CASE2.DATA, line 393
   60 Reading WCONPROD in file /home/hove/work/OPM/opm-tests/spe1/SPE1CASE2.DATA, line 403
   61 Reading WCONINJE in file /home/hove/work/OPM/opm-tests/spe1/SPE1CASE2.DATA, line 412
   62 Reading TSTEP    in file /home/hove/work/OPM/opm-tests/spe1/SPE1CASE2.DATA, line 420

Warning: Keyword 'ECHO' is not supported by flow.
In file /home/hove/work/OPM/opm-tests/spe1/SPE1CASE2.DATA, line 105


Warning: Keyword 'ECHO' is not supported by flow.
In file /home/hove/work/OPM/opm-tests/spe1/SPE1CASE2.DATA, line 105


===============Saturation Functions Diagnostics===============

System:  Black-oil system.
Relative permeability input format: Saturation Family I.
Number of saturation regions: 1



================ Starting main simulation loop ===============


Report step  0/120 at day 0/3650, date = 01-Jan-2015

Time step 0, stepsize 1 days, at day 0/31, date = 01-Jan-2015
Time step summary: newton its =  3, linearizations =  4 ( 0.003 sec), linear its =   5 ( 0.001 sec)

Time step 1, stepsize 3 days, at day 1/31, date = 02-Jan-2015
Time step summary: newton its =  3, linearizations =  4 ( 0.002 sec), linear its =   8 ( 0.001 sec)

Time step 2, stepsize 9 days, at day 4/31, date = 05-Jan-2015
Time step summary: newton its =  5, linearizations =  6 ( 0.003 sec), linear its =  18 ( 0.002 sec)

Time step 3, stepsize 18 days, at day 13/31, date = 14-Jan-2015

Warning: Keyword 'BASIC' is unhandled for output to file.

Warning: Keyword 'PRES' is unhandled for output to file.

Warning: Keyword 'RS' is unhandled for output to file.

Warning: Keyword 'SGAS' is unhandled for output to file.

Warning: Keyword 'WELLS' is unhandled for output to file.
Time step summary: newton its =  5, linearizations =  6 ( 0.003 sec), linear its =  22 ( 0.002 sec)

Warning: Keyword 'BASIC' is unhandled for output to file.

Warning: Keyword 'PRES' is unhandled for output to file.

Warning: Keyword 'RS' is unhandled for output to file.
```

```
if context.sim.well_var("PROD1", "WWCT") > 0.80:
    context.schedule.shut_well("PROD1", context.report_step)
    context.schedule.open_well("PROD2", context.report_step)
```



OPM-OP Extending Python – Opm Wrappers

Many of the C++ classes in opm-common are available in Python

```
#!/usr/bin/env python
import sys
from opm.io.parser import Parser
from opm.io.ecl_state import EclipseState
from opm.io.schedule import Schedule



fname = sys.argv[1]

parser = Parser()
deck = parser.parse(fname)
ecl_state = EclipseState(deck)
sched = Schedule(deck, ecl_state)
```

OPM-OP Embedding Python - PYINPUT

1. Evaluated *parse time*.
2. Anywhere in the deck.
3. Application objects/state available through `context` object
4. Can manipulate the deck object.

100% normal Python code between `PYINPUT` and `PYEND`

```
START
  31 AUG 1993 /

RUNSPEC

DIMENS
   2 2 1 /

PYINPUT

import numpy as np
dx = np.array([0.25, 0.25, 0.25, 0.25])
active_unit_system = context.deck.active_unit_system()
default_unit_system = context.deck.default_unit_system()

kw = context.DeckKeyword( context.parser['DX'], dx, active_unit_system, default_unit_system )
context.deck.add(kw)

PYEND
```

OPM-OP Embedding Python - PYACTION

1. Evaluated *run time*.
2. In the SCHEDULE section
3. Application objects/state available through a richer `context` object
4. Can manipulate the `Schedule` object.

```
SCHEDULE

TSTEP
  10 /

PYACTION
# This action will close well PROD1 if it has had WWCT > 80
# for more than 100 days.

wwct = context.sim.well_var("PROD1", "WWCT")
if wwct > 0.80:
    if "start" in context.storage:
        duration = datetime.date.today() - context.sim_time
        if duration.days > 100:
            context.schedule.shut_well("PROD1", context.report_step)
            context.schedule.open_well("PROD2", context.report_step)
    else:
        context.storage["start"] = context.sim_time
else:
    context.storage.pop("start", None)

PYEND
```

The Python stuff is *off* by default:

-DOPM_ENABLE_PYTHON=ON # Enable the wrappers
-DOPM_ENABLE_EMBEDDED_PYTHON=ON # Enable PYINPUT / PYACTION
-DOPM_INSTALL_PYTHON=ON # Install with “make install”

Full Simulation In Python

The python wrapper classes are a prerequisite for full simulations(*) in Python.

=> Next presentation

(*): Currently simulations in Python don't support embedding Python with the PYINPUT and PYACTION keywords.