UI Updates

Koichi Murakami (KEK)

Virtual Geant4 Collaboration Meeting 2020

Technology shifts

- C++03 -> C++11 -> C++20
- CVS, SVN -> git, github/gitlab
- make -> CMake
- Parallel computing : multi-thread, task-oriented, GPU, ...
- Container : Docker, Singularity, Podman,…
- Python2 : EOL -> Python3
- IPython -> Jupyter Notebook -> Jupyter Lab
- MPI : MPI v2 -> MPI v3 : C++ binding dropped-off, ZeroMQ
- OpenGL -> Vulkan, Metal (MacOS)

Python

- Python is more popular environment in science.
  - important tool in data science
  - packages : Anaconda (numpy, scipy, …)
  - data management : pandas.DataFrame
  - plot tools : matplotlib, plotly, …

- Language aspects:
  - oop, much easier than C++ : barrier to start
  - multi-language binding (C-API)
  - dynamic binding
    - many third-party modules
    - modularization of software components

Python Bridge Benefits

Improving functionalities of Geant4 UI

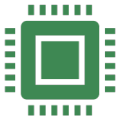
* more powerful scripting environment
* flow control, variables, arithmetic operation

flexibility in the configuration of user applications

* Modularization of user classes with dynamic loading scheme
  (DetectorConstruction, PhysicsList, PrimaryGeneratorAction, UserAction-s)
* quick prototyping and testing

Software component bus

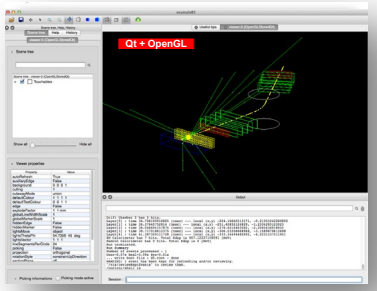
* interconnectivity with many Python external module : analysis tools, e.g. pandas
* middleware for application developers : GUI / web app.

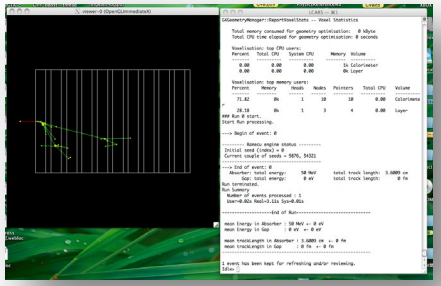


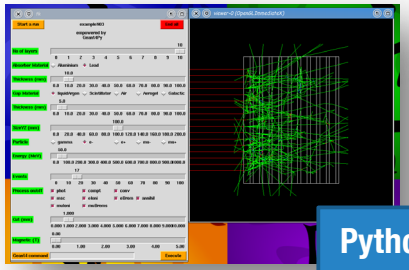
Software development cycle and Performance

* rapid development
* minimize performance loss w/ thin wrapper

Geant4 UI & App.







Python App.

Python as
software component bus

App Drive
w/o C++ API

Geant4 User Applications

No Interface
w/ C++ API

Python Front End

python™

>>> import Geant4

UI Session

GUI | CLI | Batch

Qt *csh, tcsh* *macro file*

Interface

UI commands */run/beamOn, …*

Intercome

Python binding

Geant4 Kernel

Python2 : EOL

- Python2 became End of Life in Apr/2020.
- Geant4Py will drop Python2 codes in the next release.
- Only support Python3 codes

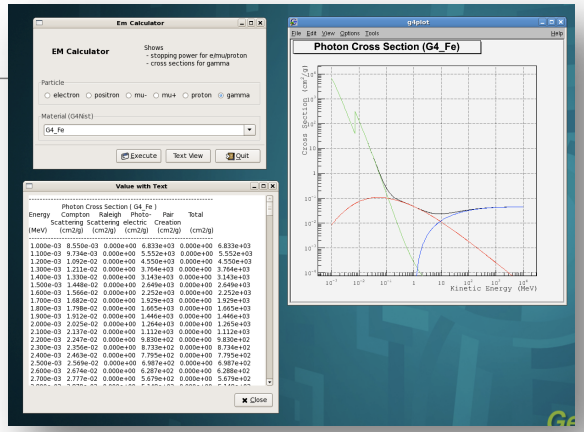
C++ Binding to Python

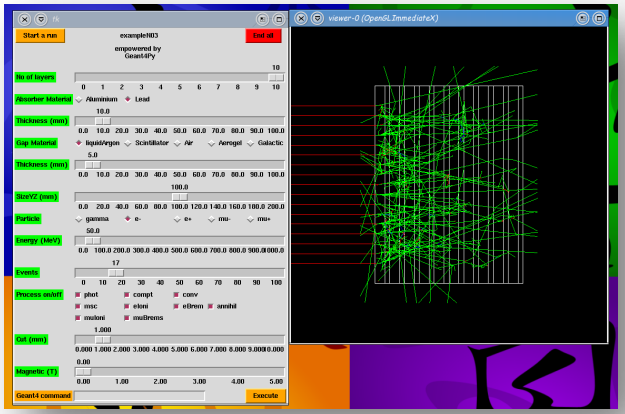
- Building tool : local cmake build -> merge to main CMake build (v10.7)
- currently use boost.python
- pybind11
  - https://github.com/pybind/pybind11
  - Seamless operability between C++11 and Python
  - header only
  - C++11 (modern C++) support
  - STL container support
  - wrapper approach is very similar to boost.python
  - migration to pybind11 planned in 2021 release (v11)

- still some tricky parts exist in Geant4 :
  - global object for singleton
  - object life-time: depends on
    - consider life-time of returned pointer : who has it?
  - copy by value, reference of existing obj (potential danger)

Python Usecases

- Thin wrapper approach
  - parametric runs : changing parameters and runs simulation
- Small tools
  - hacking inside Geant4
  - cross section check
- Educational purpose
  - GUI applications
- Simple app. w/ scripting
  - simple geom + command-line scorer + g4tools + pandas + matplotlib
- will revise examples using modern tools as future development
  - jupyter lab





Jupyter

- http://jupyter.org
- Former IPython notebook
- much more powerful frontend than plain python CLI.
- Notebook / Lab works on web browser
  - inline interactivity : plots, images, …
  - save and share session logs
  - very useful for analysis work, tutorials,…. .
  - We plan to use Jupyter Lab in our tutorial hands-on. (w/ g4tools, anaconda, pandas, matplotlib)
- Other external language (R, Julia, SQL,..) can be run on Jupyter frontend
  - Geant4 UI commands as external language kernel in Jupyter

Jupyter Notebook Sample

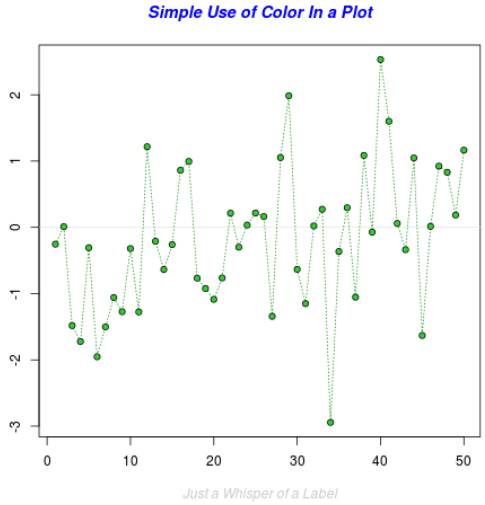
R demo

Here is some code which illustrates some of the differences between R and S graphics capabilities. Note that colors are generally specified by a character string name (taken from the X11 rgb.txt file) and that line textures are given similarly. The parameter "bg" sets the background parameter for the plot and there is also an "fg" parameter which sets the foreground color.

In [1]:
```
require(datasets)

require(grDevices); require(graphics)
```

In [2]:
```
x <- stats::rnorm(50)
opar <- par(bg = "white")
plot(x, ann = FALSE, type = "n") +
abline(h = 0, col = gray(.90)) +
lines(x, col = "green4", lty = "dotted") +
points(x, bg = "limegreen", pch = 21) +
title(main = "Simple Use of Color In a Plot",
      xlab = "Just a Whisper of a Label",
      col.main = "blue", col.lab = gray(.8),
      cex.main = 1.2, cex.lab = 1.0, font.main = 4, font.lab = 3)
```



Out[2]: numeric(0)

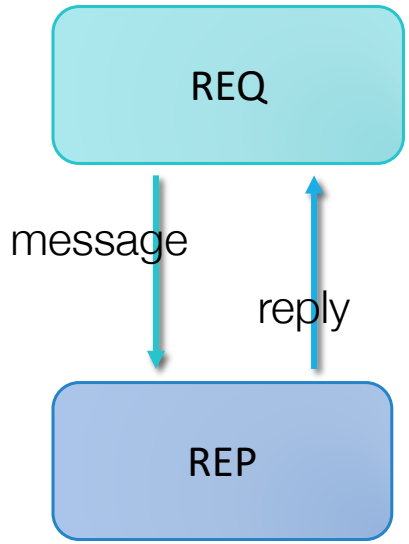
Completion, History, Save

Interactive, Save, Share

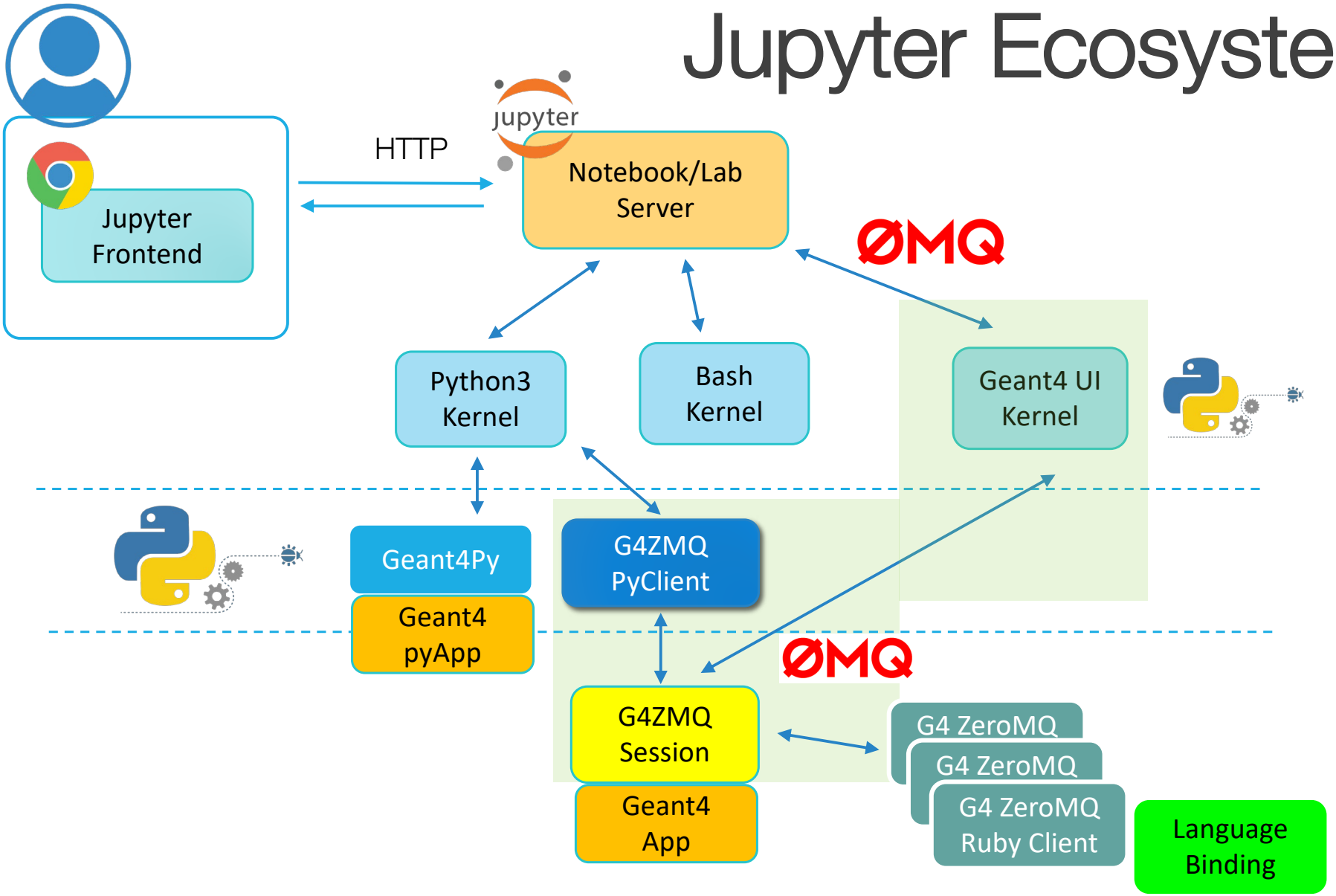
Jupyter Frontend + Language Kernel

ZeroMQ Backend

- light-weight socket API
  - very popular tool : stable, easy to install
- support different message patterns
  - REQ – REP (Request – Reply model)
  - send a message from client to sever
  - reply a message from sever to client
  - …
- Many language bindings
  - C/C++, Python, Ruby, PHP, Perl, Java, ...



Jupyter Ecosystem



G4ZMQServer

- An alternative UI session like
  - UI terminal, Qt session, Batch session, ...
  - /environments/zmq

```
auto zmq_session = new G4ZMQServer();
zmq_session-> SetEndpoint(endpoint);
// endpoint is like tcp://127.0.0.1:5555
zmq_session -> SessionStart();
```

- Waiting a message / Receive a message /
  Execute UI (primitive) commands / Send back an output

ZeroMQ client

- Clients can be in any languages that ZeroMQ is bound.
- Simple pyclient module : *g4zmq (written in Python)*

```
>>> import g4zmq
>>> g4zmq.connect()
>>> g4zmq.ls("/run")
>>> g4zmq.help("/run/beamOn")
>>> g4zmq.apply("/run/beamOn 10")
```

- Language bindings :
  - Python, Julia, Ruby, Perl, PHP, JAVA, ...
  - http://zeromq.org/bindings:_start

Notes on ZMQ interface

- ZMQ interface is an alternative of UI session.
- You can drive Geant4 app in the same way as UI terminal.
  - send a UI command, execute, get a response (output)
  - but more friendly for scripting
- **CANNOT** directly access to Geant4 objects like native Python interface approach (Geant4Py).

IGeant4

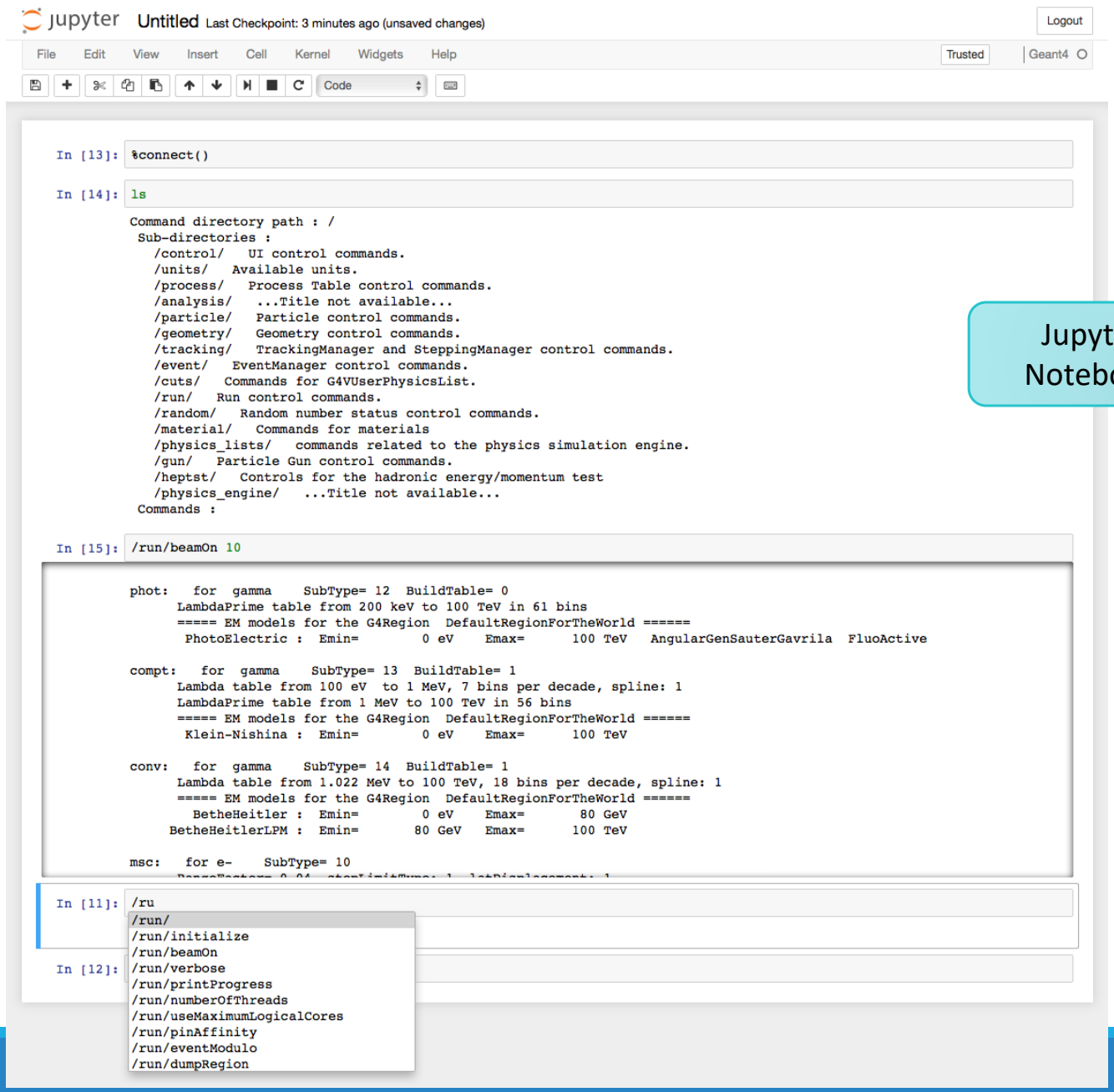
Geant4 UI Kernel for Jupyter

https://github.com/koichi-murakami/igeant4

```
# jupyter console --kernel geant4
```

You can do:

- connect to G4 zmq server
- execute UI / shell commands
- completion
- history
- shell (bash) exec (ex. %shell ls)



Jupyter Notebook

MPI Issue

- MPI: A Message-Passing Interface Standard Version 3.1
  Message Passing Interface Forum
  says

"The C++ bindings were deprecated as of MPI-2.2. The C++ bindings are removed in MPI-3.0. The namespace is still reserved, however, and bindings may only be provided by an implementation as described in the MPI-2.2 standard. "

  - The situation does not change in the 2019 draft specification, hoeplessly in MPI-4

- openMPI v1.10.1 is the latest version of MPI-2, that supports C++ binding.
  - The software status is 'retired'.

MPI vs ZeroMQ

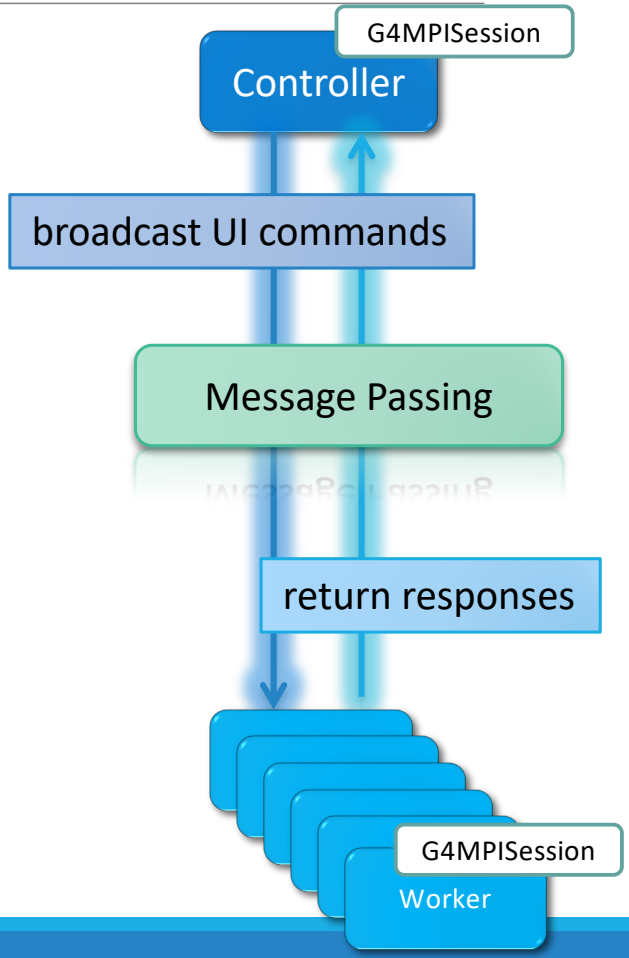
- Alternative approaches
  - Thin wrapper for C-API : extern “C”
  - ~~Boost-MPI~~ : wrapper, backend is necessary
  - Kokkos LAMMPS ???
  - Other message library : ZeroMQ

- MPI pros. / cons.
  - MPI is designed for running in HPC cluster. (pros.)
  - less fault tolerance : fail in single node -> total fail (cons.)

- ZeroMQ
  - simple messaging library. reasonably fast
  - based on more general distributed programming model
  - some side works that are provided in MPI should be prepared for real use. (e.g. *mpirun* command, remote shell)
  - POC done

Message Passing Scheme

- Geant4-MPI interface is based on simple scatter/gather model.
  - Implemented in “interface” layer.
  - Isolated from G4kernel
  - Broadcasting UI commands (from controller to workers)
  - No intercommunications between worker nodes
- ZeroMQ can be used as a backend of message passing.
  - plan to release in v11
  - message distributor is reusable.
  - additional functionalities of controller (node config, remote shell…)
  - communication model of efficient reduce for large scale computing



Summary

- Python interface:
  - Python v2 will be dropped. (EOL)
  - Build package will be merged into main CMake.
  - pybind11 : tool change for python-C++ binding
- Geant4 UI language kernel for Jupyter
  - G4 ZeroMQ Server as a UI session
  - Allows to interface with any languages
- MPI
  - C++ binding is gone in MPI specification.
  - ZeroMQ is adopted as a MPI backend.