BERKELEY LAB
Bringing Science Solutions to the World

U.S. DEPARTMENT OF ENERGY

In Situ Analysis and Visualization with SENSEI

E. Wes Bethel, Junmin Gu, Burlen Loring, Dmitriy Morozov, *Gunther H. Weber*, John Wu (LBNL). *Nicola Ferrier*, Silvio Rizzi (ANL). Dave Pugmire, James Cress, Matthew Wolf (ORNL). Earl Duque, Brad Whitlock (Intelligent Light). Utkarsh Ayachit, David Thompson, Andrew Bauer, Patrick O'Leary (Kitware)

BERKELEY LAB | Argonne NATIONAL LABORATORY | OAK RIDGE National Laboratory | Intelligent Light | Kitware

Why in situ?

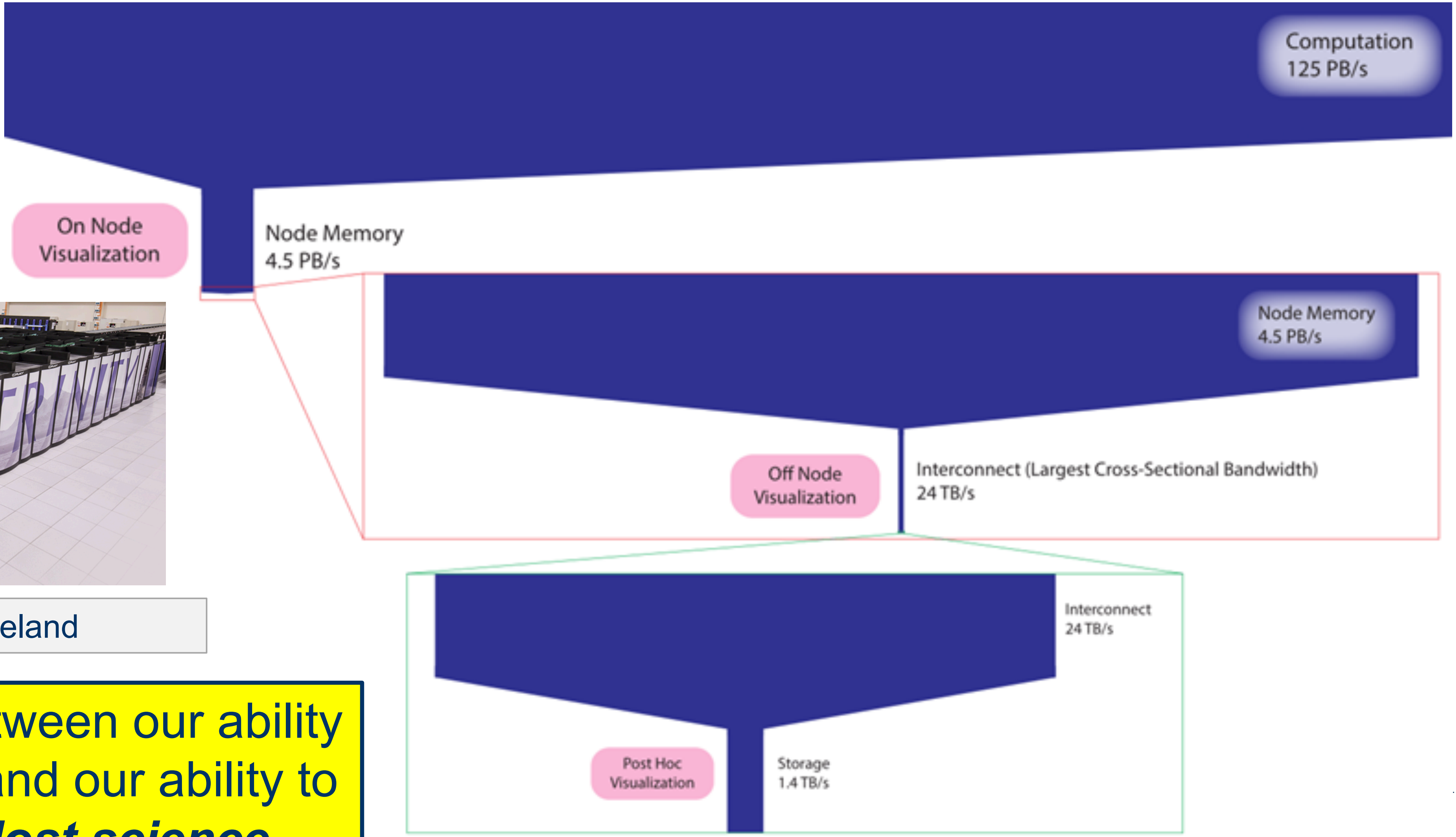


Image courtesy Ken Moreland

The difference between our ability to generate data and our ability to move it results in ***lost science***.

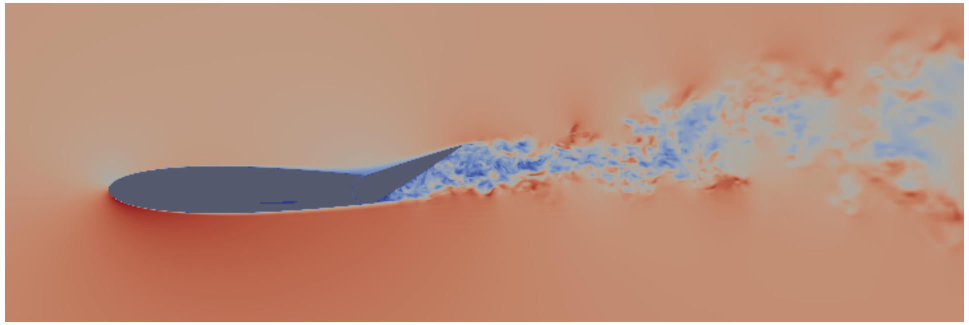
SENSEI : Scalable Environments for Scientific Explorations In Situ

4 Project Pillars

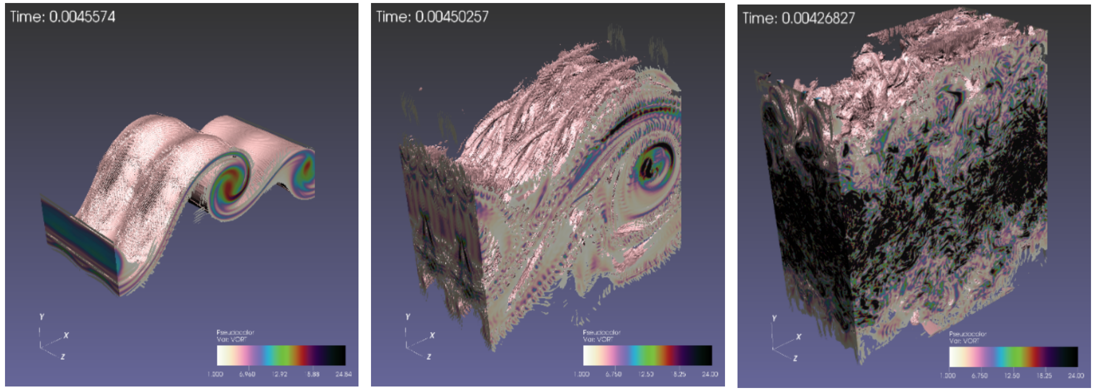
- R&D for scalable infrastructure and methods to work around FLOPS-I/O bottleneck
- Generic infrastructure maximizes portability and preserves investment in DOE codes
- Science code team partnerships focus and prove R & D
- Outreach and community engagement make the technology accessible

Scalability	Portability	Science	Community

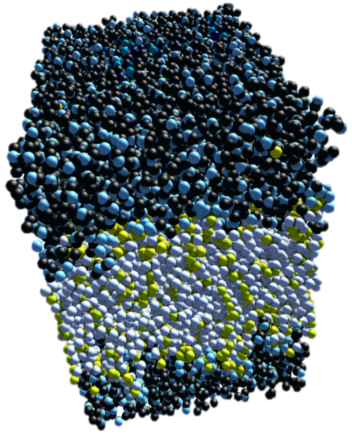
Science Engagements



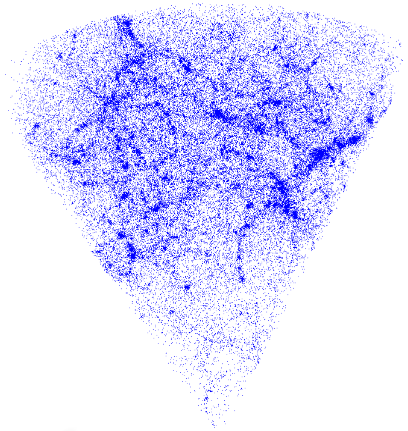
Phasta + Catalyst, 1M Cores, Mira



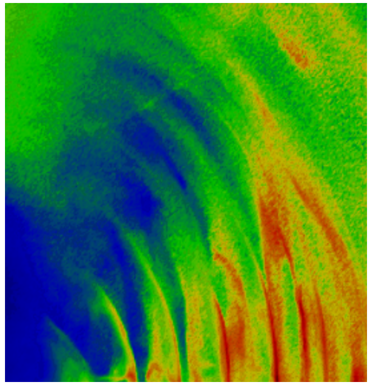
AVF Leslie + Libsim, 131K Cores Cori



LAMMPS + OSPray, interactive, Theta



Henson Gadget, 8192 Cores Edison



Warp + Libsim, 16k Cores Edison

Where do these codes come from?
DOE Office of Science: HEP, BES, BER
High Energy Physics (HEP)
Basic Energy Sciences (BES)
Biological and Environmental Research (BER)
Academic research community, then picked up and extended/used by DOE, DoD, others.

SENSEI In situ Infrastructure

Write once run everywhere - use any simulation with any visualization/analysis and easily swap back-ends at run time

SENSEI enables connection of simulation data sources to visualization and analysis back ends through a data model and API

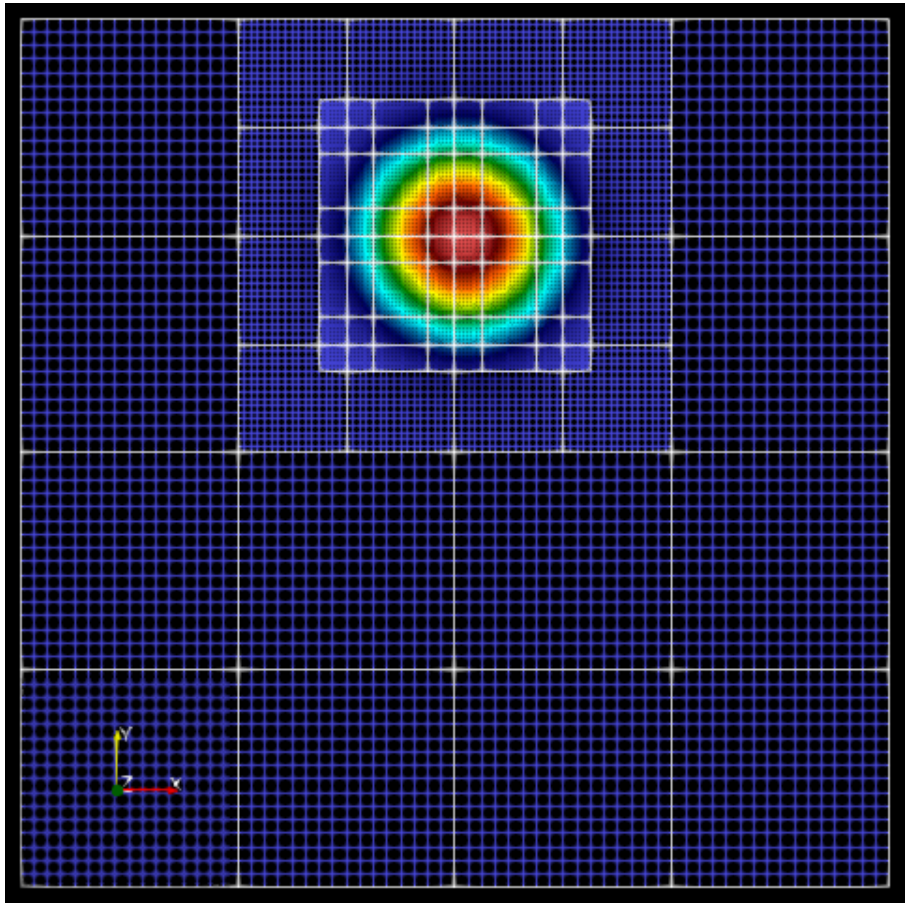
Simulations get run-time interchangeability of analysis/vis codes

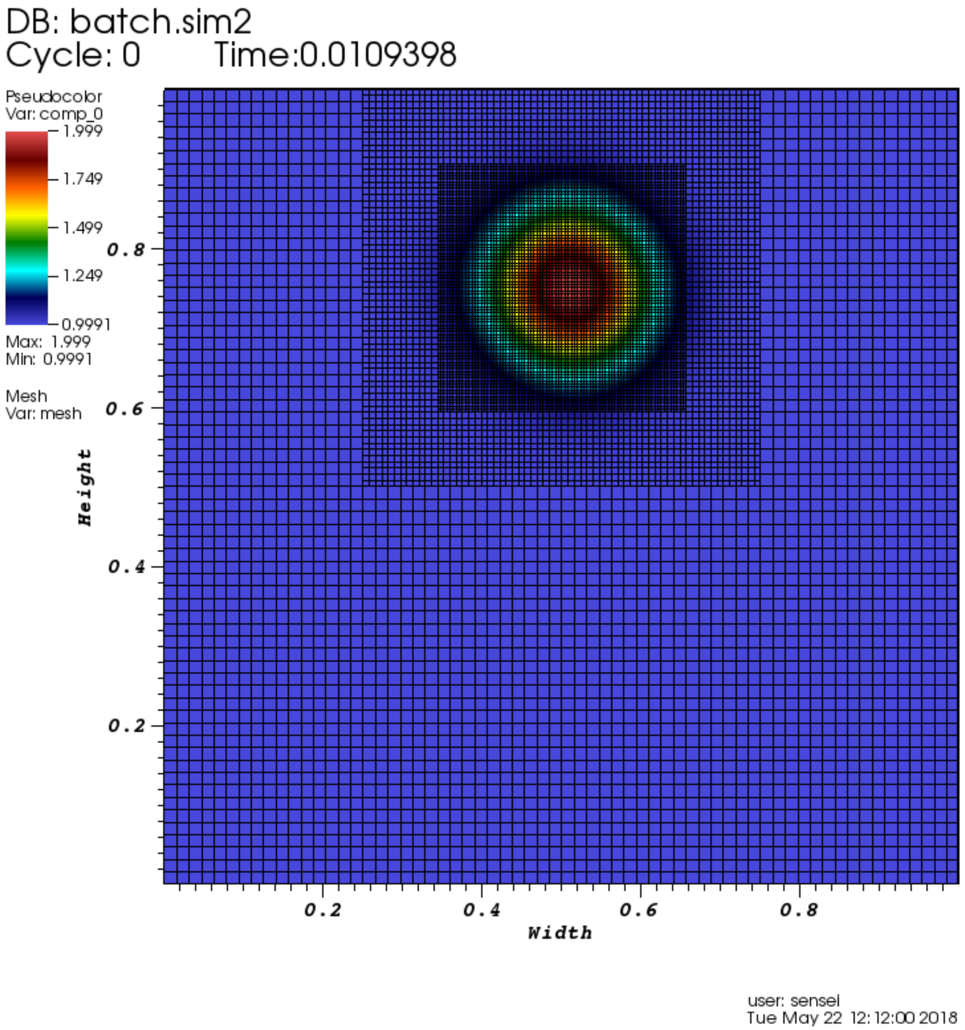
analysis/vis codes can consume data from any simulation

Delivarables

Focus Area	Co-leads
expanding the data model	Andrew Bauer (Kitware), Brad Whitlock (Intelligent Light)
bidirectional data movement	Patrick O'Leary (Kitware), Matthew Wolf (ORNL)
design/execution patterns	Dmitriy Morozov (LBNL), Dave Pugmire (ORNL)
scaling to next-generation systems	Nicola Ferrier (ANL), John Wu (LBNL)
outreach, code team partnerships, cookbook, workshops	Gunther Weber (LBNL), Matthew Wolf (ORNL)
software products, distribution, releases	Earl Duque (Intelligent Light), Patrick O'Leary (Kitware)

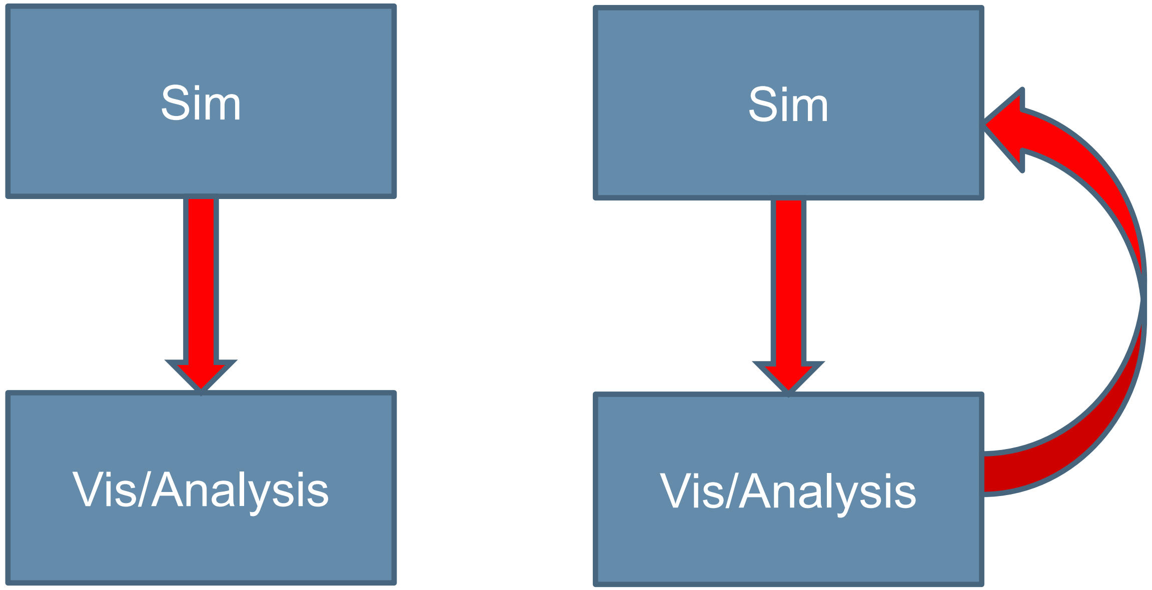
Expanding the data model



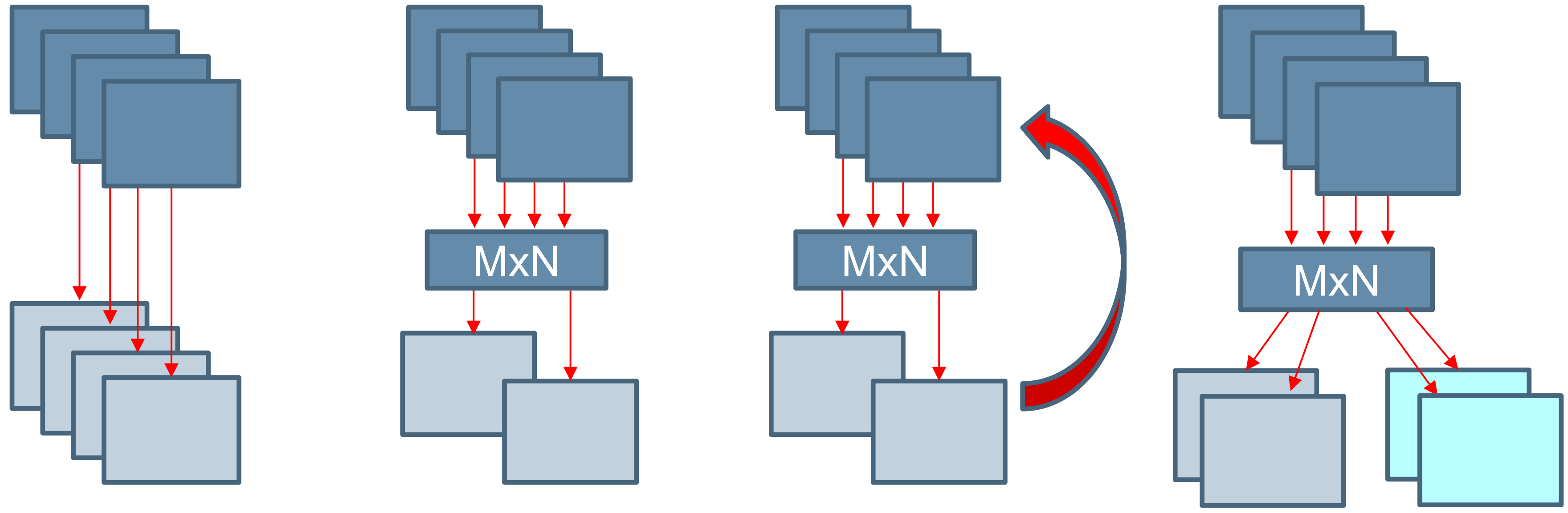


Bidirectional data movement

Bidirectional data movement supports more science use case scenarios; eg, computational steering



Design and execution patterns



N producer ranks,
N consumer ranks
Unidirectional data movement/control (N:N:1)

M producer ranks,
N consumer ranks
Unidirectional data movement/control (M:N:1)

M producer ranks,
N consumer ranks
<u>Bidirectional</u> data movement/control (M:N:2)

M producer ranks,
N1 and N2 consumer ranks,
Unidirectional data movement/control (M:<N1, N2>:1)

Research focus areas:

- MxN data redistribution
- Depth of copies
- Bidirectional: interface, pipeline management
- Leveraging arch features like NVRAM for staging
- Leveraging 3rd party tools like TensorFlow for ML-based analytics
- Specific science app use case drivers

data model

What simulation data types does SENSEI support?

vtkDataObject

- AMR
- Multi-"block"
- Uniform Cartesian
- Stretched Cartesian
- Curvilinear (logically Cartesian)
- Unstructured/FEM
- PIC/Point cloud
- Molecular
- Tabular
- Graphs
- Array Collection (no geometry)

- many more purpose specific and esoteric data types are supported by VTK
- **no explicit dependence on other parts of VTK such as i/o, filters, renderering, etc etc**

www.vtk.org

Speed & Efficiency

zero copy layouts provide pointer equivalent performance

- Array of Structures (AOS)
  - single array with components interleaved

v=	x1	y1	z1	x2	y2	z2	…	xn	yn	zn

```
// VTK's default is AOS, no need to use
vtkAOSDataArrayTemplate
vtkDoubleArray *aos = vtkDoubleArray::New();
aos->SetNumberOfComponents(3);
aos->SetArray(v, 3*n, 0);
```

- Structure of Arrays (SOA)
  - each component in its own arrays

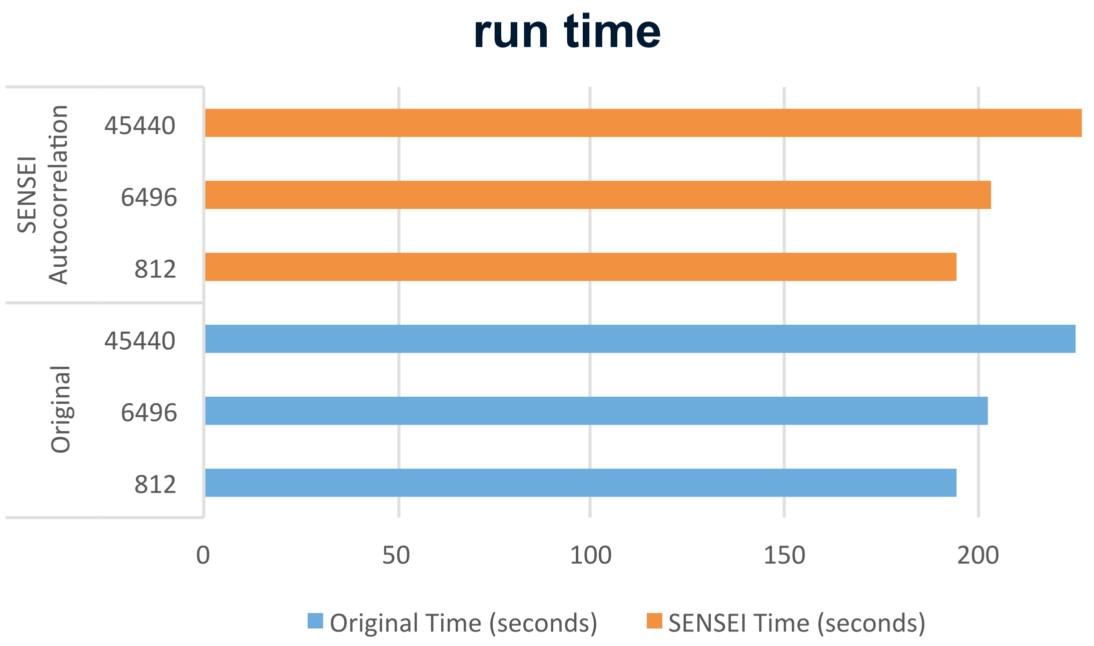
vx=	x1	x2	x3	…	xn
vy=	y1	y2	y3	…	yn
vz=	z1	z2	z3	…	zn

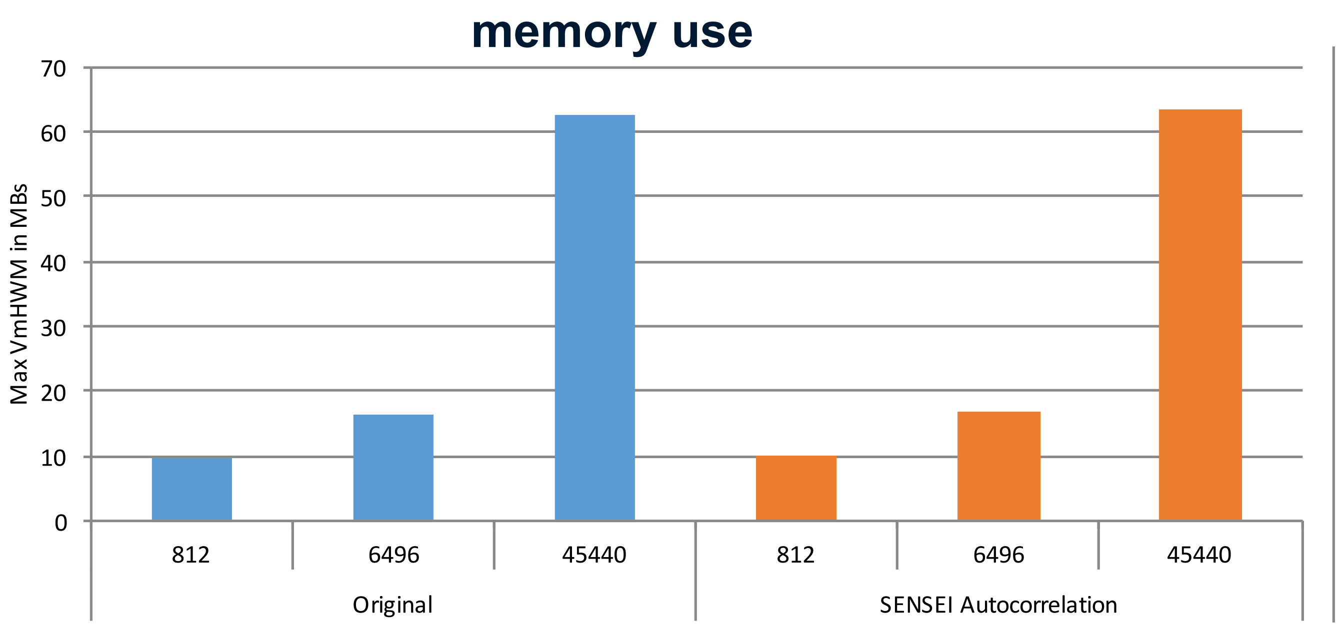
```
// use the new SOA class
vtkSOADataArrayTemplate<double> *soa =
   vtkSOADataArrayTemplate<double>::New();
soa->SetNumberOfComponents(3);
soa->SetArray(0, vx, n, true);
soa->SetArray(1, vy, n);
soa->SetArray(2, vz, n);
```

SENSEI Overhead

Run *Original* and *Baseline* configs, 3 levels of concurrency: 1K, 6K, 45K

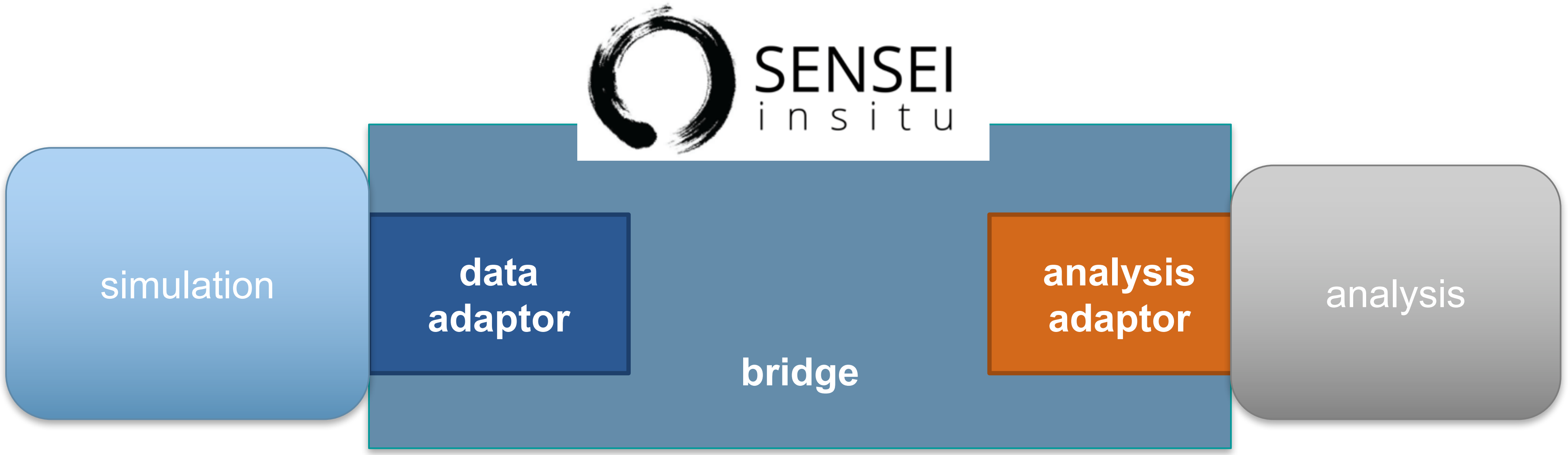
- Original: subroutine called, Baseline: through SENSEI bridge



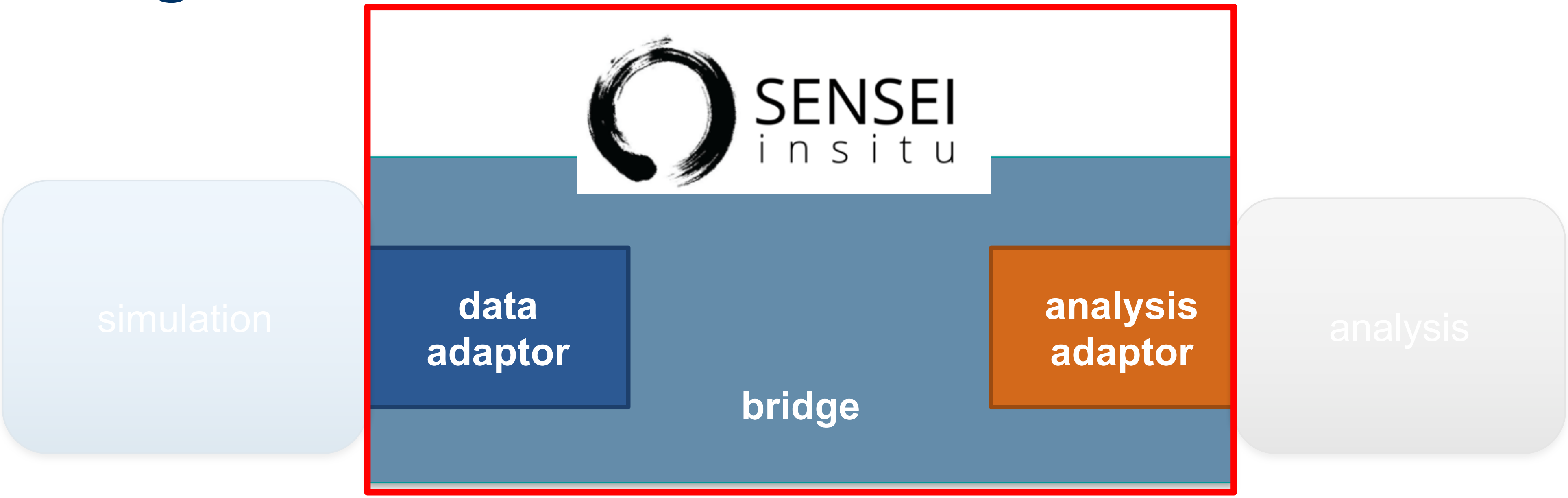


SENSEI architecture

In situ Architecture



The bridge



Manages data and analysis adaptors, periodically pushes data to the analysis

- Typically 3 functions: Initialize, Update and Finalize

The data adaptor

DataAdaptors – API giving analyses access to simulation data and metadata

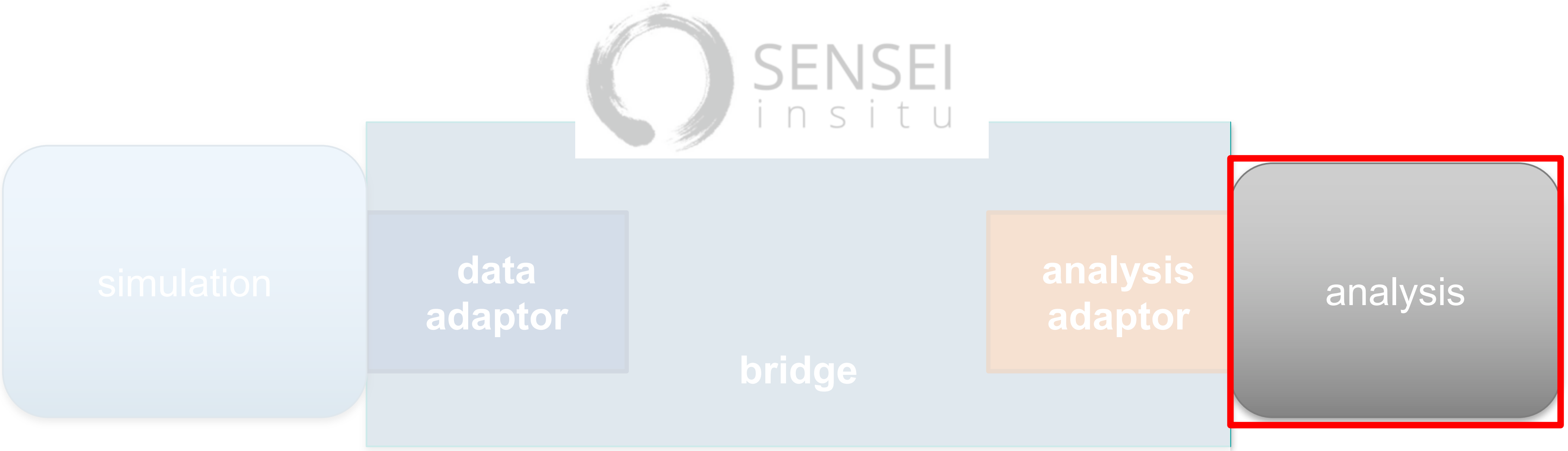
- Convert simulation data to/from the data model

The analysis adaptor

AnalysisAdaptor – API for simulation to invoke vis & analysis

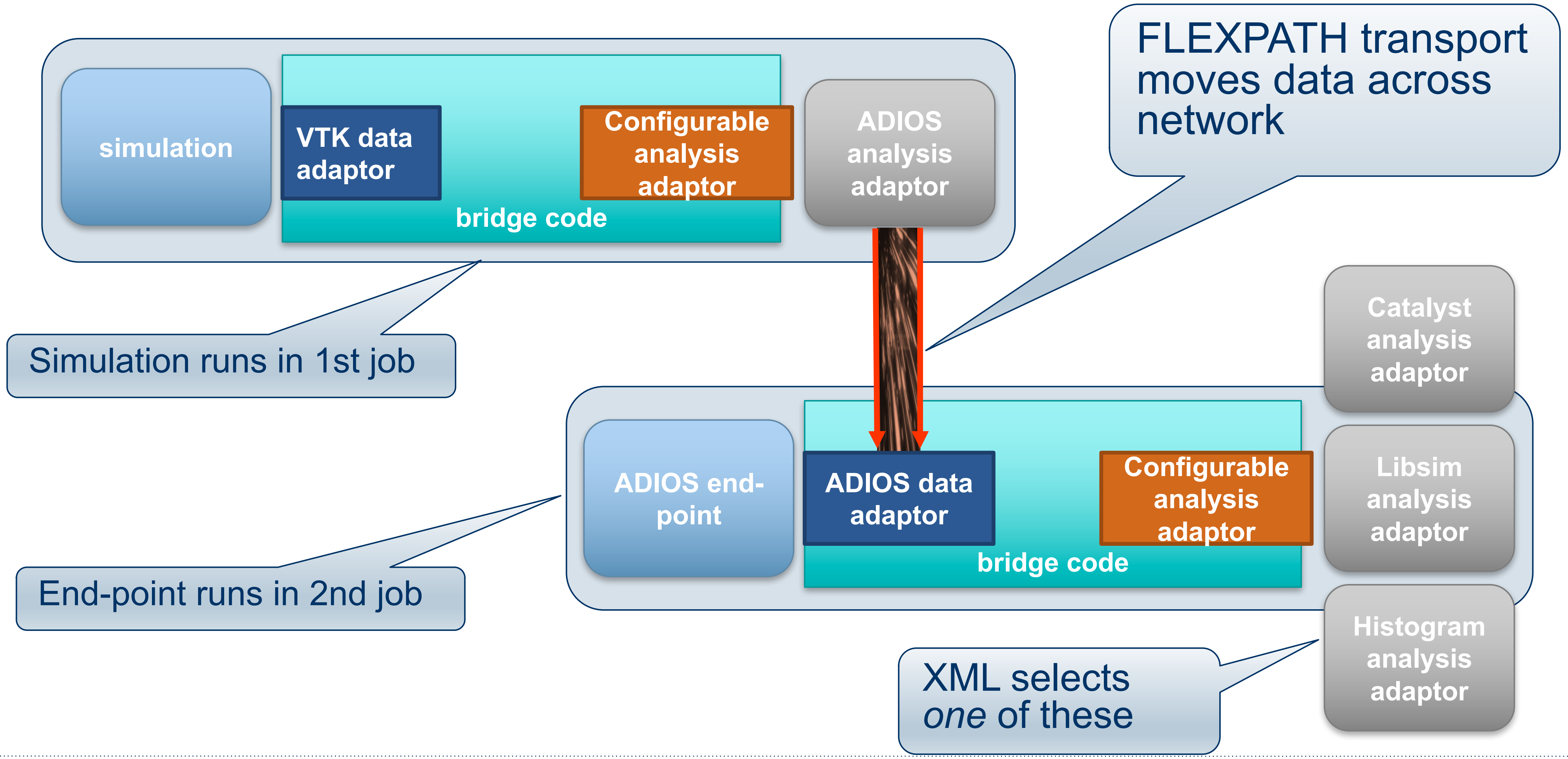
- Consume/process data

Analyses

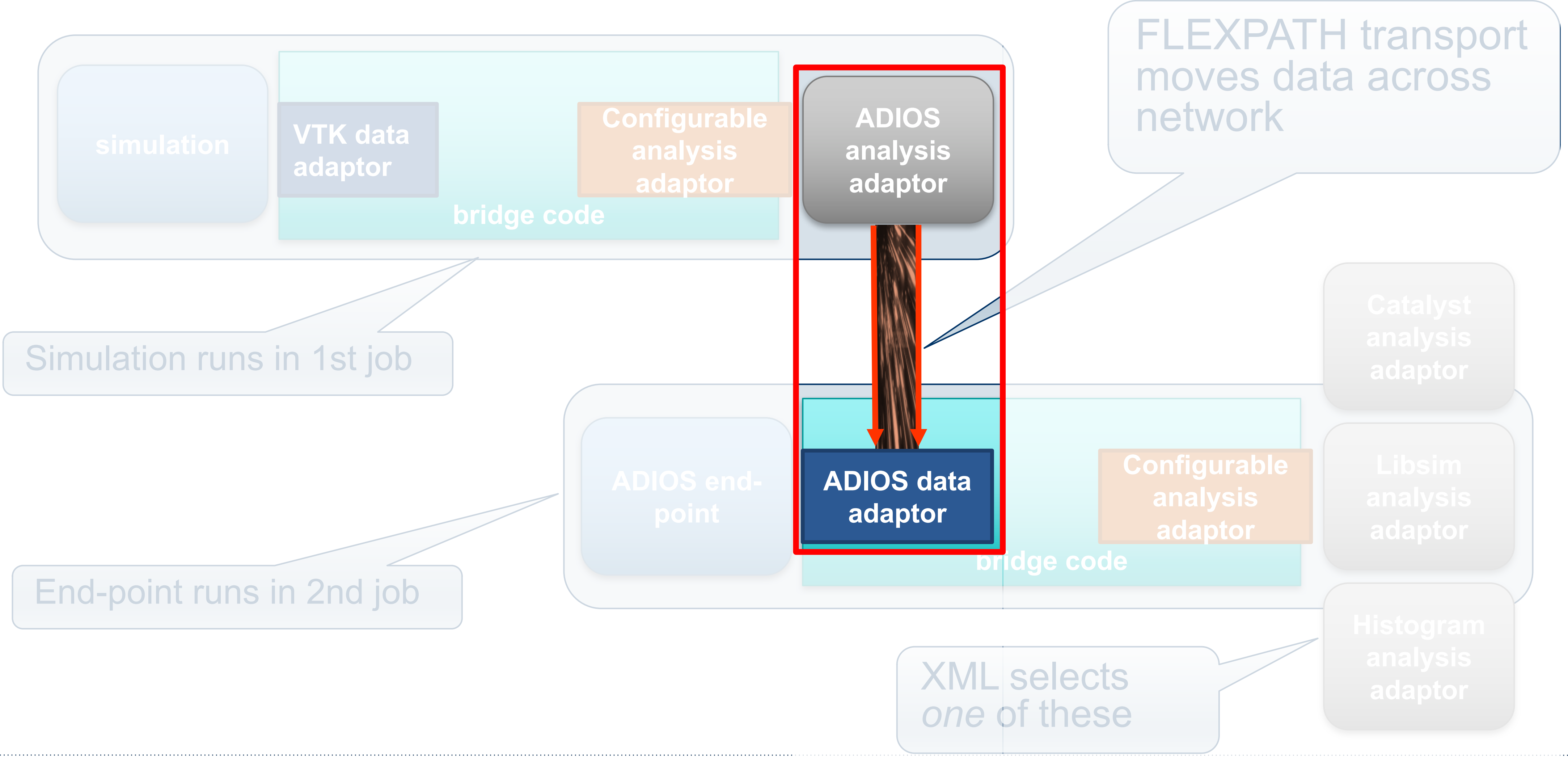


ConfigurableAnalysisAdaptor – select an analysis at run time via an XML config file

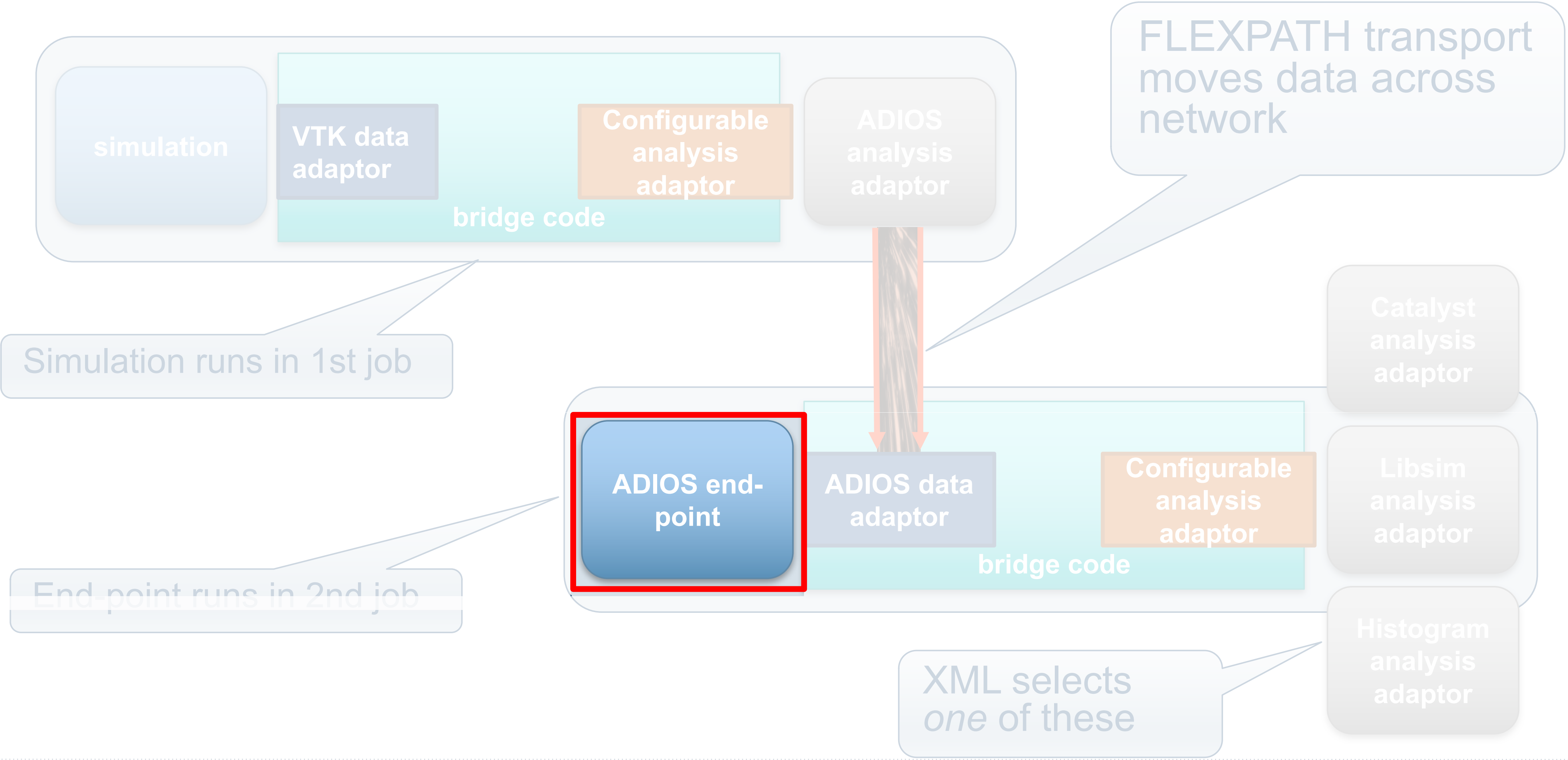
In transit Architecture



ADIOS Adaptors



End-Point


FLEXPATH transport moves data across network
simulation
VTK data adaptor
Configurable analysis adaptor
ADIOS analysis adaptor
bridge code
Simulation runs in 1st job
Catalyst analysis adaptor
ADIOS end-point
ADIOS data adaptor
Configurable analysis adaptor
bridge code
Libsim analysis adaptor
End-point runs in 2nd job
Histogram analysis adaptor
XML selects *one* of these

In situ demo

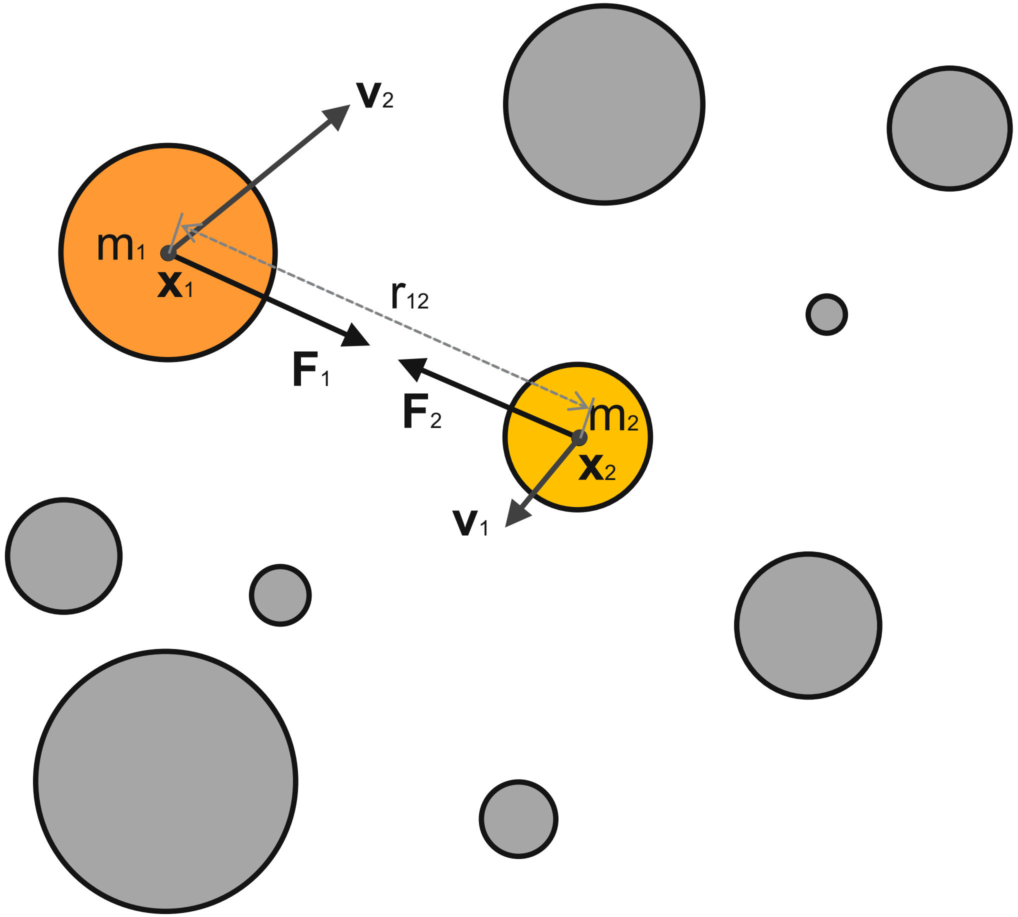
Newton mini-app

N-body Gravitational Simulation. A single file, <400 lines.

Solves Newton's law of gravitation

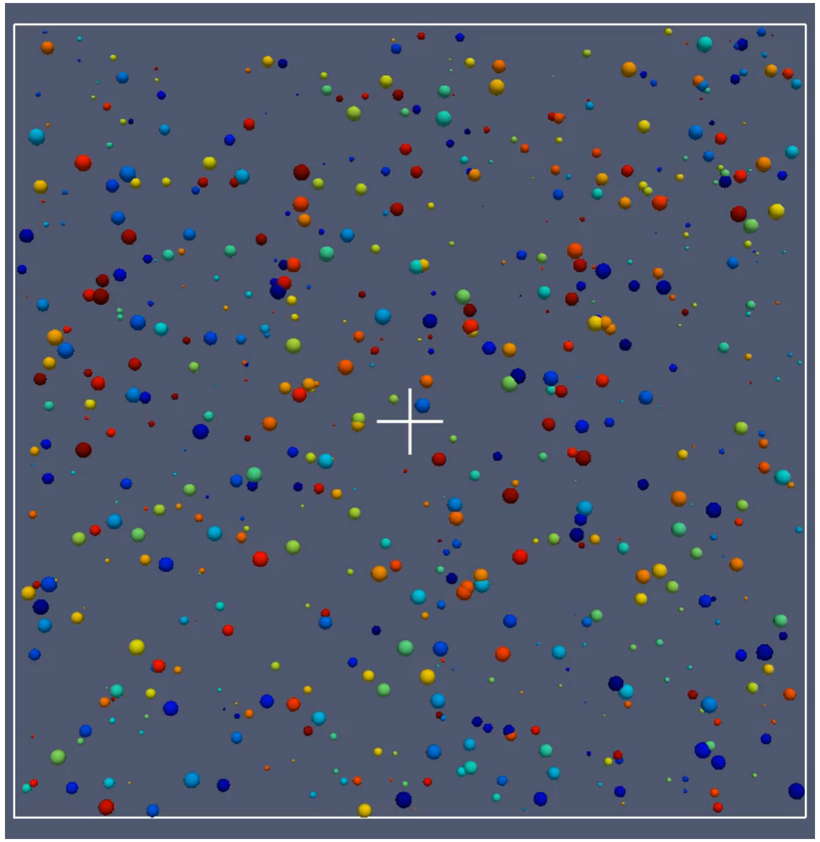
Velocity Verlet method

```
Fi = Fj = G*mi*mj/rij**2

xi' = vi

vi' = Fi/mi
```



Newton mini-app

- direct solver, O(N**2)
  - Velocity Verlet
    - » second order, symplectic, conserves momentum exactly, time reversible
- the simplest possible code
  - a single file, <400 lines, to better focus on use of SENSEI interface
  - a production quality code could easily be thousands of lines (see NBODY6 ~6K lines)



Instrumenting the simulation

```
# set up the initial condition
n_bodies = args.n_bodies*n_ranks
ic = uniform_random_ic(n_bodies, -5906.4e9, \
    5906.4e9, -5906.4e9, 5906.4e9, 10.0e24, \
    100.0e24, 1.0e3, 10.0e3)
ids,x,y,z,m,vx,vy,vz,fx,fy,fz = ic.allocate()
h = args.dt if args.dt else ic.get_time_step()

# create an analysis adaptor(bridge code)
bridge = newton_bridge()
bridge.initialize(args.analysis, args.analysis_opts)

# run the sim and analysis
bridge.update(0,0,ids,x,y,z,m,vx,vy,vz,fx,fy,fz)
i = 1
while i <= args.n_its:
    velocity_verlet(x,y,z,m,vx,vy,vz,fx,fy,fz,h)
    bridge.update(i,i*h,ids,x,y,z,m,vx,vy,vz,fx,fy,fz)
    i += 1

# finish up
bridge.finalize()
```

Bridge

```
class newton_bridge:
    def __init__(self):
        self.DataAdaptor = sensei.VTKDataAdaptor.New()
        self.AnalysisAdaptor = sensei.ConfigurableAnalysisAdaptor.New()

    def initialize(self, analysis, args=''):
        # select and configure SENSEI analysis adaptor
        …

    def finalize(self):
        self.AnalysisAdaptor.Finalize()

    def update(self, i,t,ids,x,y,z,m,vx,vy,vz,fx,fy,fz):
        # convert simulation data to VTK
        # invoke the analysis
        …
```

Invoking in situ analysis

```
def update(self, i,t,ids,x,y,z,m,vx,vy,vz,fx,fy,fz):

    # construct VTK a dataset
    node = points_to_polydata(ids,x,y,z,m,vx,vy,vz,fx,fy,fz)
    mb = vtk.vtkMultiBlockDataSet()
    mb.SetNumberOfBlocks(n_ranks)
    mb.SetBlock(rank, node)

    # pass it to the data adaptor
    self.DataAdaptor.SetDataTime(t)
    self.DataAdaptor.SetDataTimeStep(i)
    self.DataAdaptor.SetDataObject(mb)

    # execute the in situ analysis
    self.AnalysisAdaptor.Execute(self.DataAdaptor)

    # free up memory
    self.DataAdaptor.ReleaseData()
```

In situ demo

- Run the the simulation 2 times
- Use XML to switch back end between Libsim and Catalyst

Catalyst

```
<sensei>
   <analysis type="catalyst" pipeline="pythonscript" filename="catalyst_config.py" enabled="1" />
</sensei>
```

Libsim

```
<sensei>
  <analysis type="libsim" plots="Pseudocolor" plotvars="ids" image-filename="image_%ts"
    image-width="800" image-height="800" slice-project="1" image-format="png" frequency="1" enabled="1"/>
</sensei>
```

In transit demo

In transit demo

Simulation: XML configures ADIOS analysis with FLEXPATH

ADIOS

```xml
<sensei>
  <analysis type="adios" filename="newton.bp" method="FLEXPATH" enabled="1" />
  <analysis type="adios" filename="newton.bp" method="DATASPACES" enabled="0" />
  <analysis type="adios" filename="newton.bp" method="MPI" enabled="0" />
</sensei>
```

End-point: XML configures either Catalyst or Libsim

Catalyst

```xml
<sensei>
   <analysis type="catalyst" pipeline="pythonscript" filename="catalyst_config.py" enabled="1" />
</sensei>
```

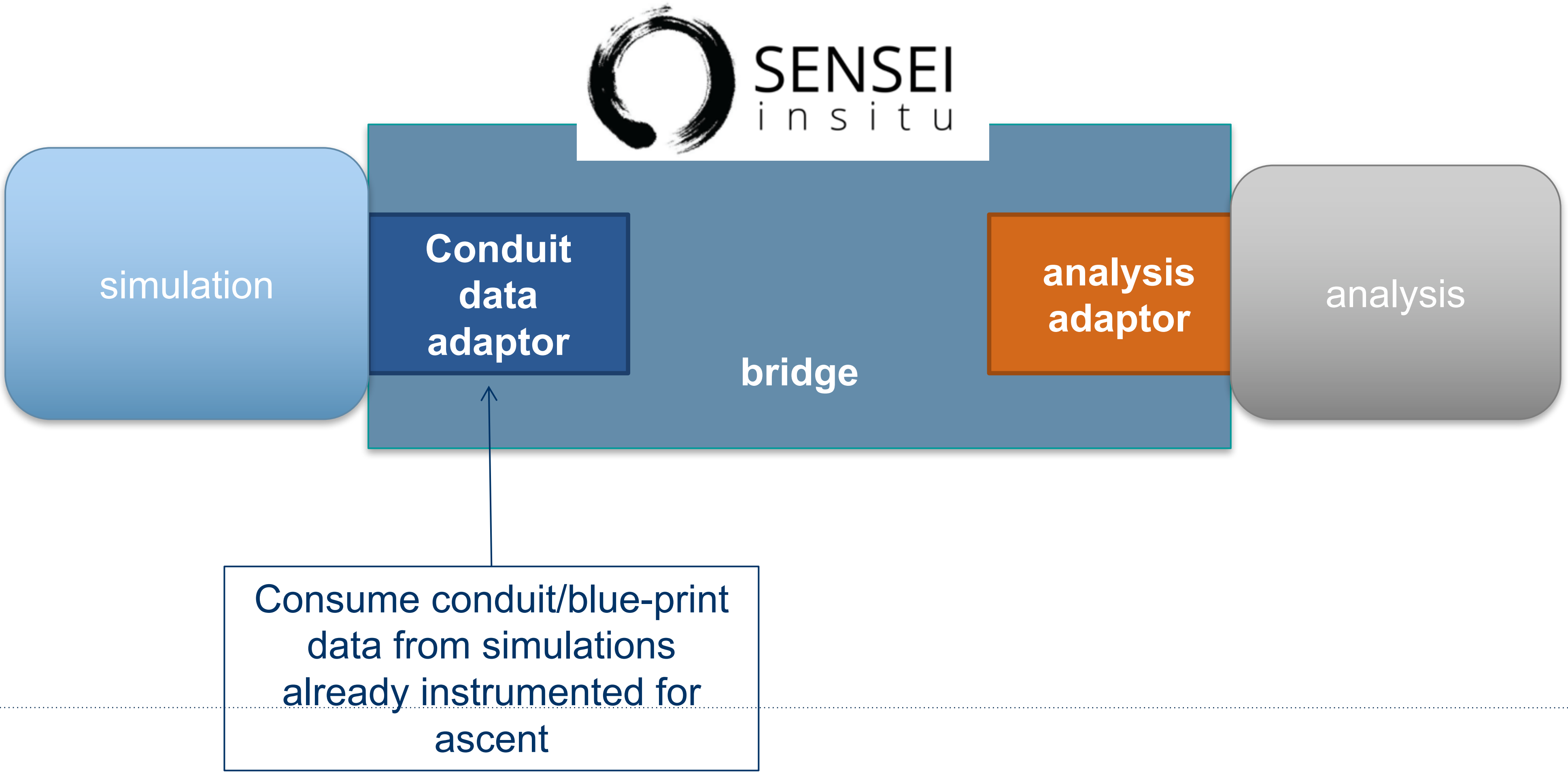
Libsim

```xml
<sensei>
  <analysis type="libsim" plots="Pseudocolor" plotvars="ids" image-filename="image_%ts"
    image-width="800" image-height="800" slice-project="1" image-format="png" frequency="1" enabled="1"/>
</sensei>
```

Links

- Main page – http://www.sensei-insitu.org/
- Software repo – https://gitlab.kitware.com/sensei/sensei
- ADIOS – https://www.olcf.ornl.gov/center-projects/adios/
- VisIt/Libsim – https://www.visitusers.org/index.php?title=Category:Libsim
- ParaView Catalyst – http://www.paraview.org/in-situ/

Conduit data adaptor



VTK-m as an analysis back-end