Simulating Star Clusters with AMUSE

Alfred J. Whitehead
Drexel University

Collaborators

Stephen L. W. McMillan
Enrico Vesperini
Simon Portegies Zwart

Outline

1. The State of Simulations of Star Clusters
2. Introduction to AMUSE
3. Sample Simulations
4. Comparison of Stellar Evolution Models
5. Conclusion & Questions

State of the Art

- NBody6++: N-body + Stellar Evolution
  MPI-parallel, GPU-enabled, Collisional
  (Aarseth/Spurzem)
- GADGET-2: N-body + SPH
  MPI-parallel, GPU possible, Collisionless
  (Springel)
- Some custom “plug ins” to the above
- Many other codes, but similar in domains addressed

The AMUSE Idea

The AMUSE Idea

Python
C++
C
Fortran

The AMUSE Idea

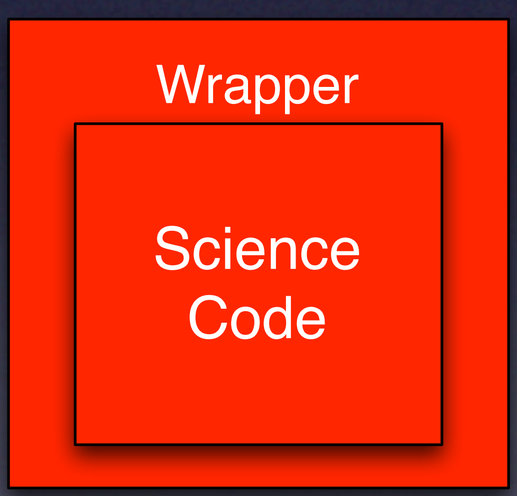
Python
C++
C
Fortran

Science Code

Existing Code
Fortran, C or C++
Solves for a single domain

The AMUSE Idea

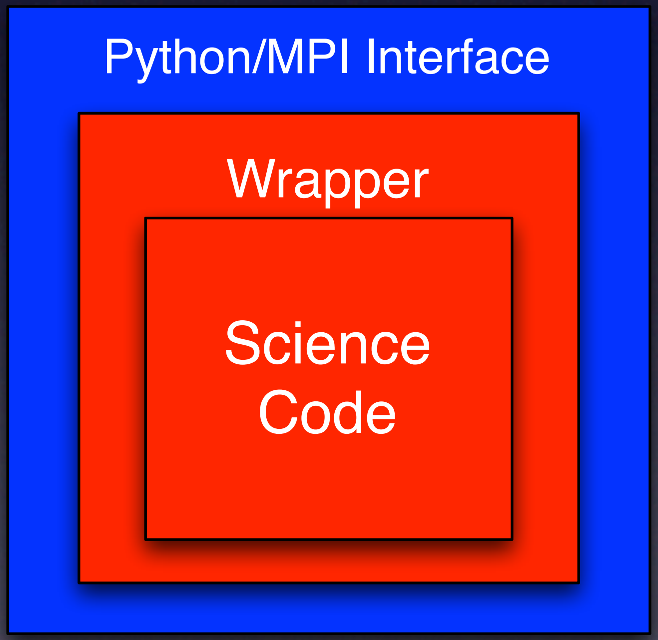
Python
C++
C
Fortran



Custom-written
Exposes standardized functions

The AMUSE Idea

Python
C++
C
Fortran

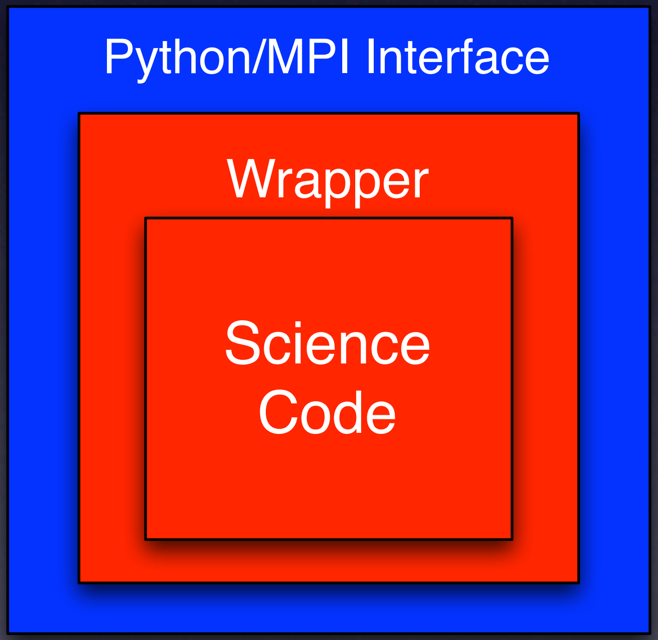


Automatically generated by AMUSE
Provides standardized interface
Isolates science code

The AMUSE Idea

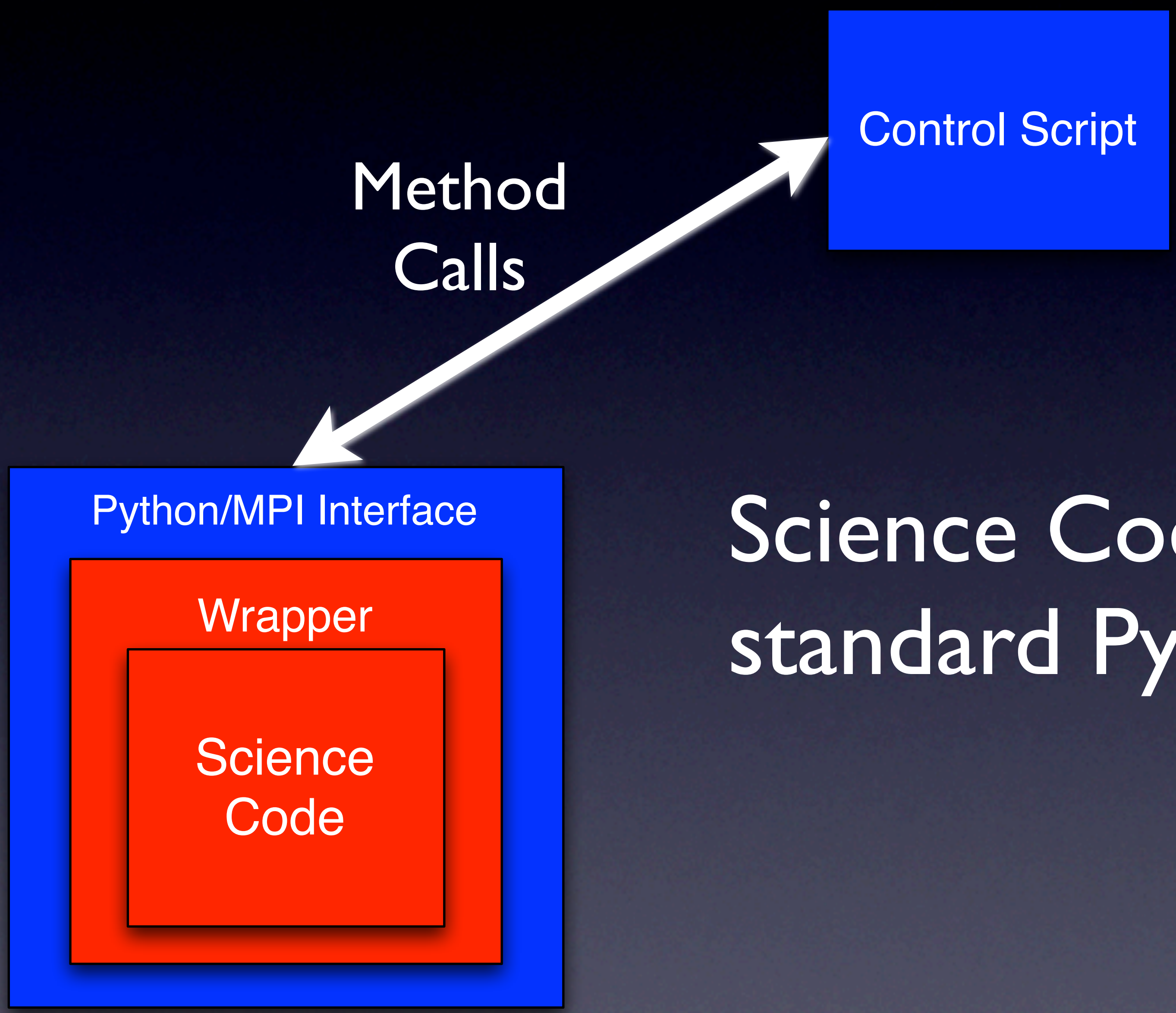
Python
C++
C
Fortran

User-written for a specific sim
Not locked in to a specific module
Contains main "evolve loop"



The AMUSE Idea

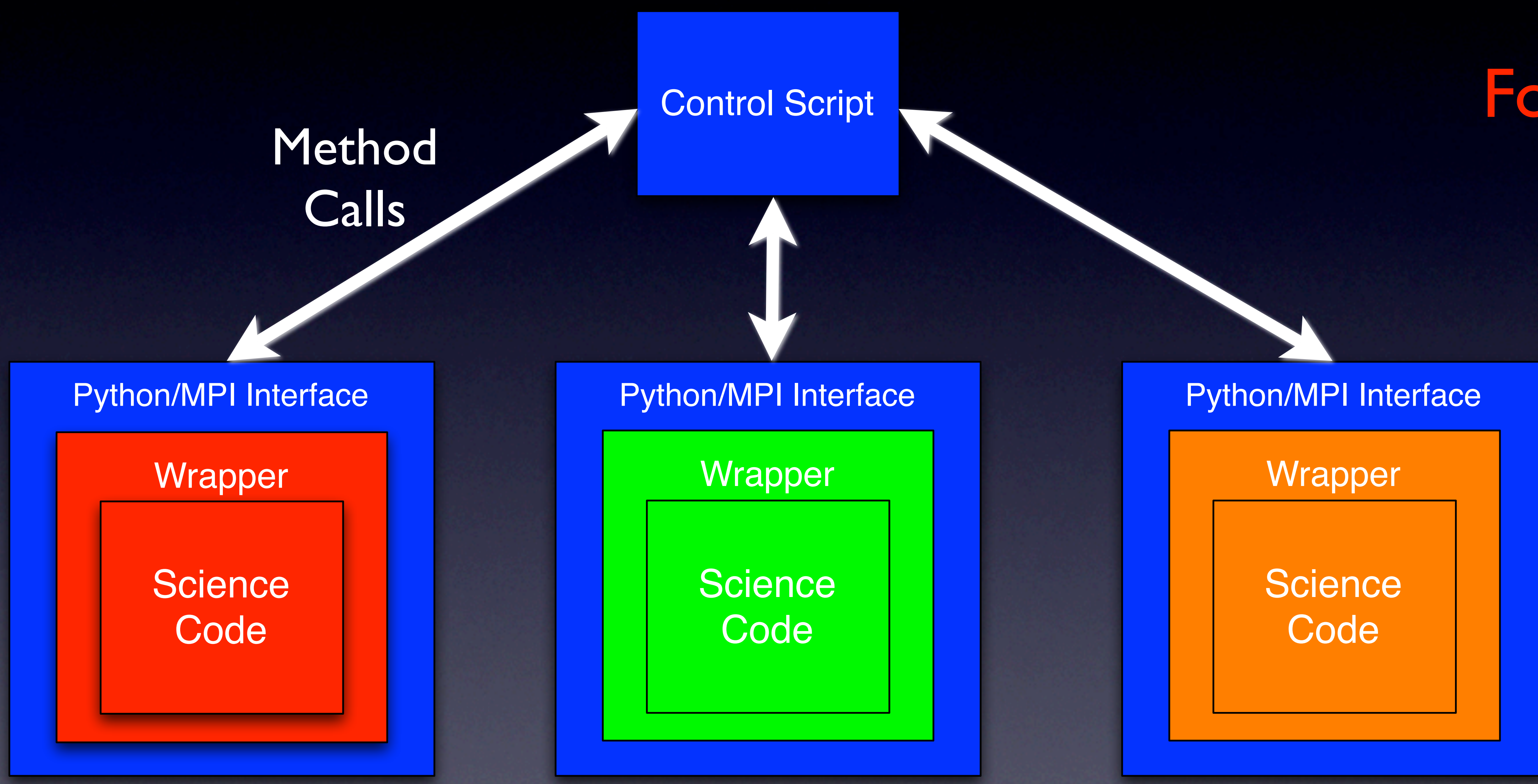
Python
C++
C
Fortran



Science Code appears as a standard Python object

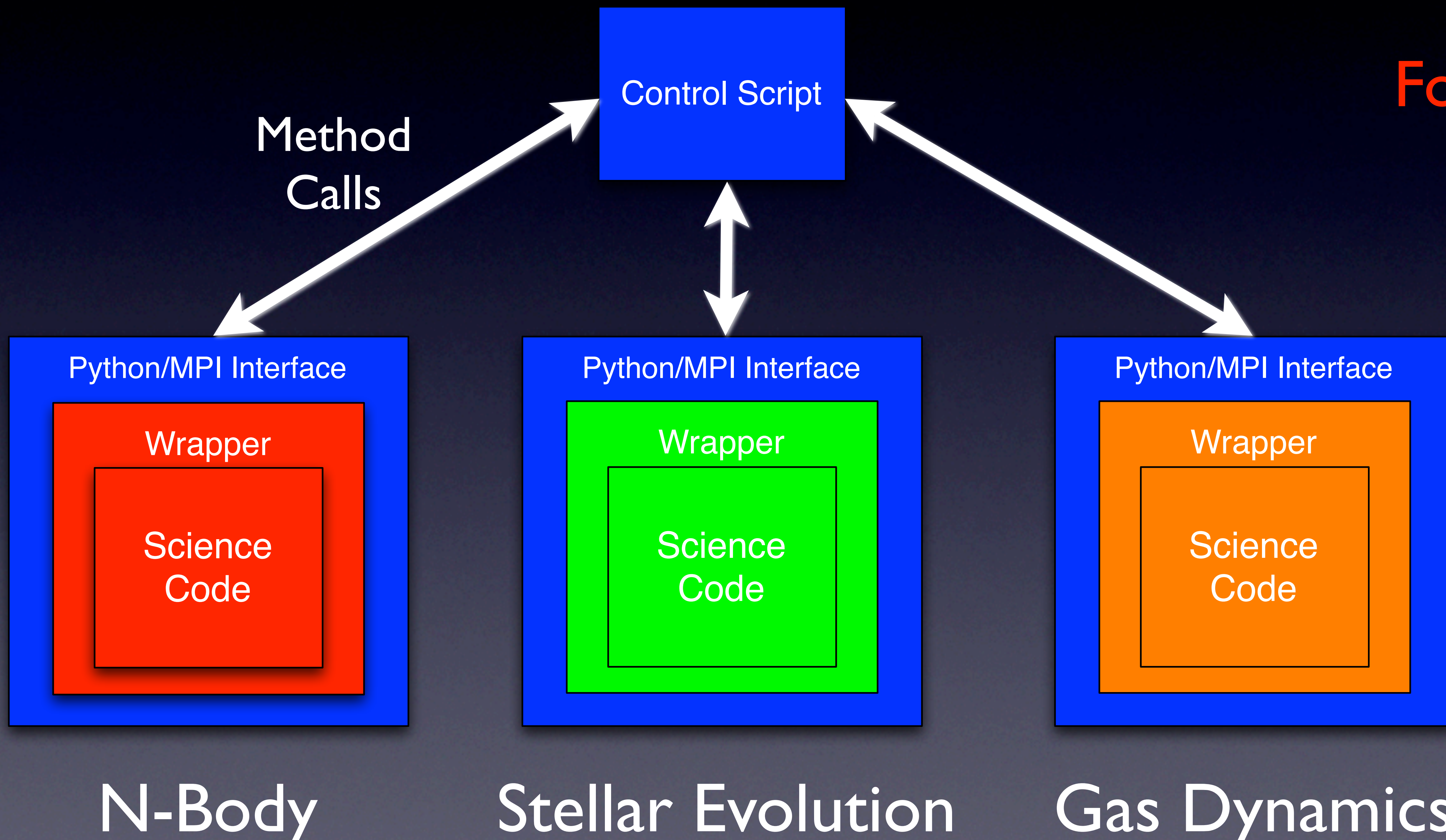
The AMUSE Idea

Python
C++
C
Fortran



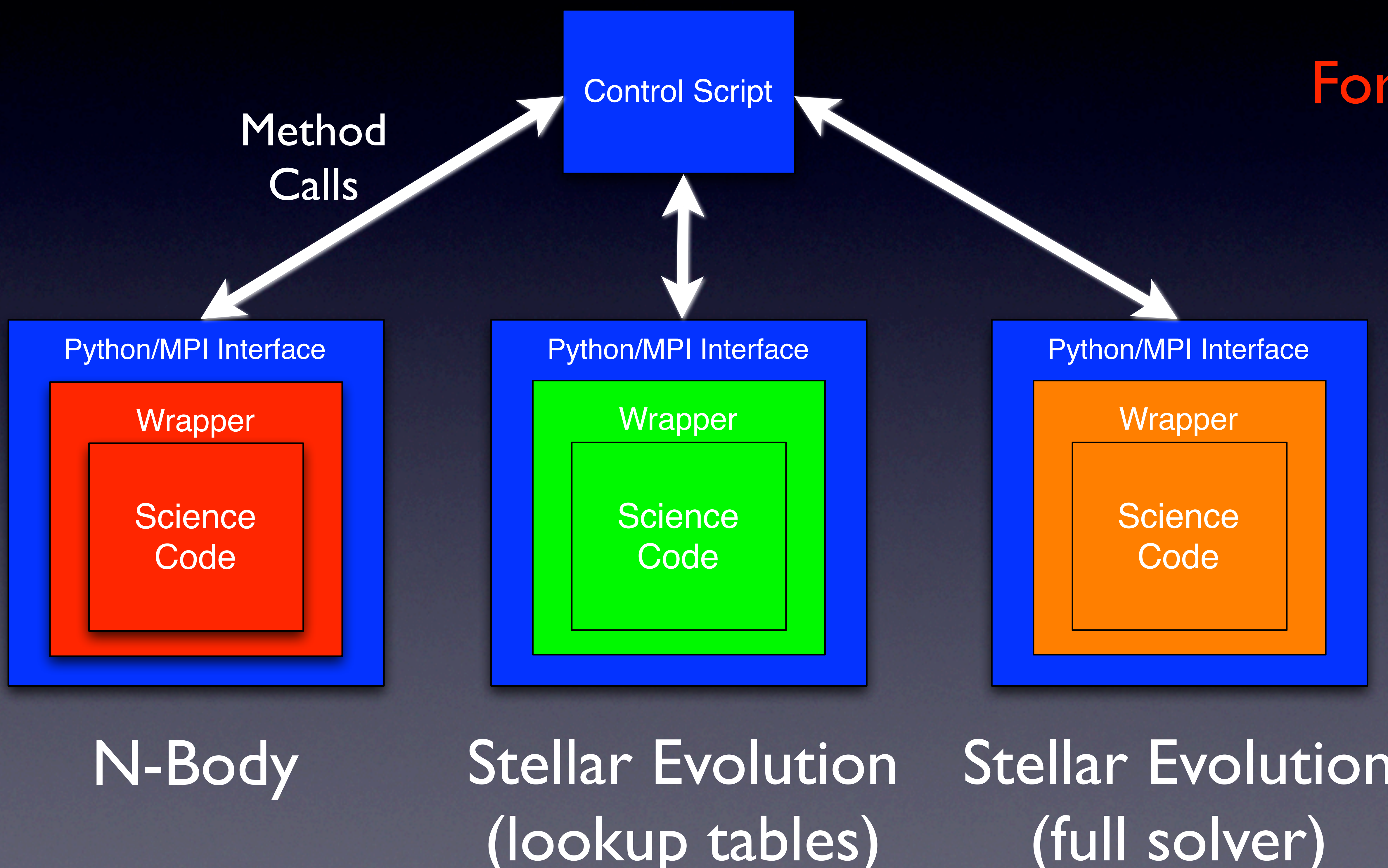
The AMUSE Idea

Python
C++
C
Fortran

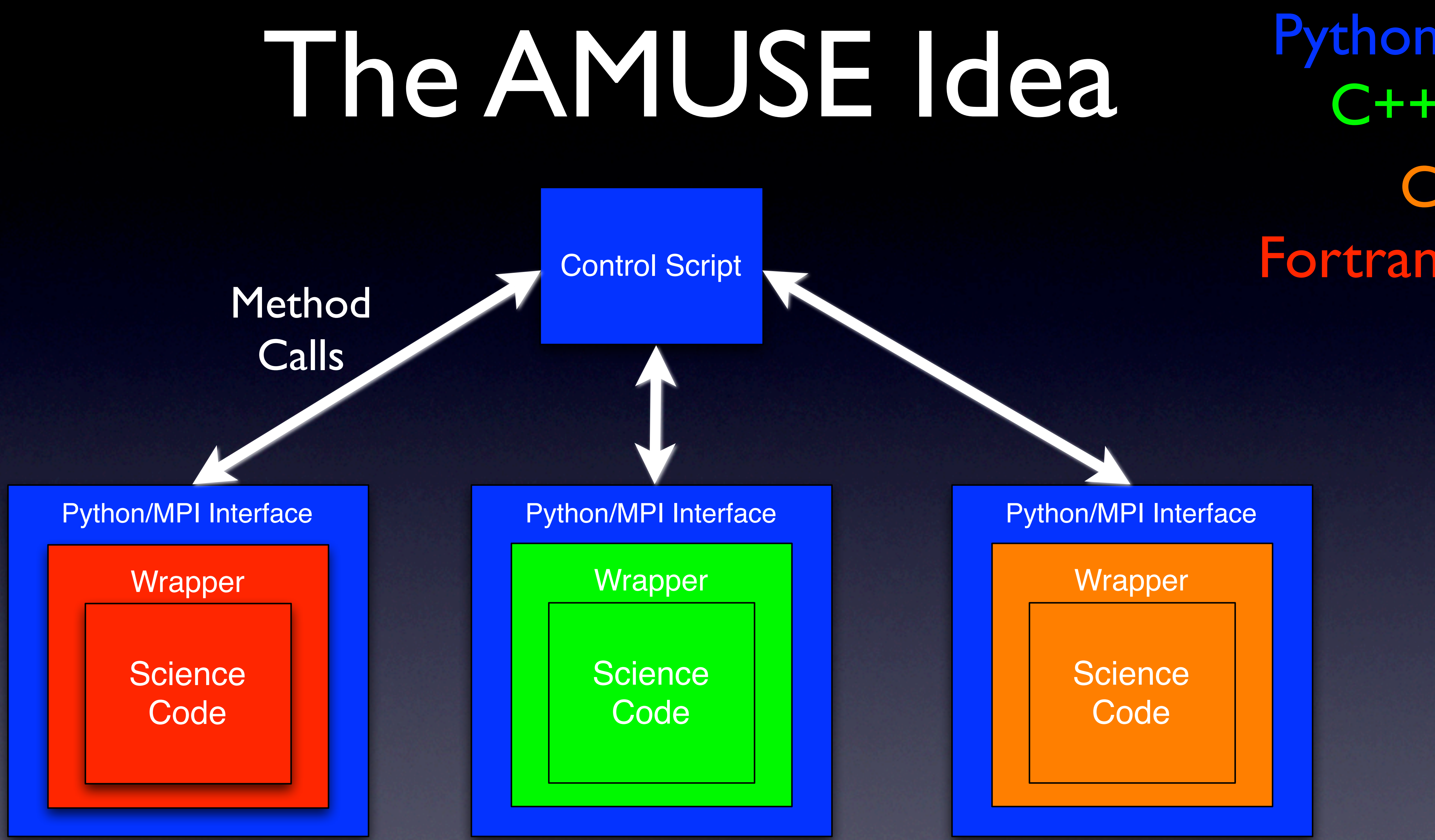


The AMUSE Idea

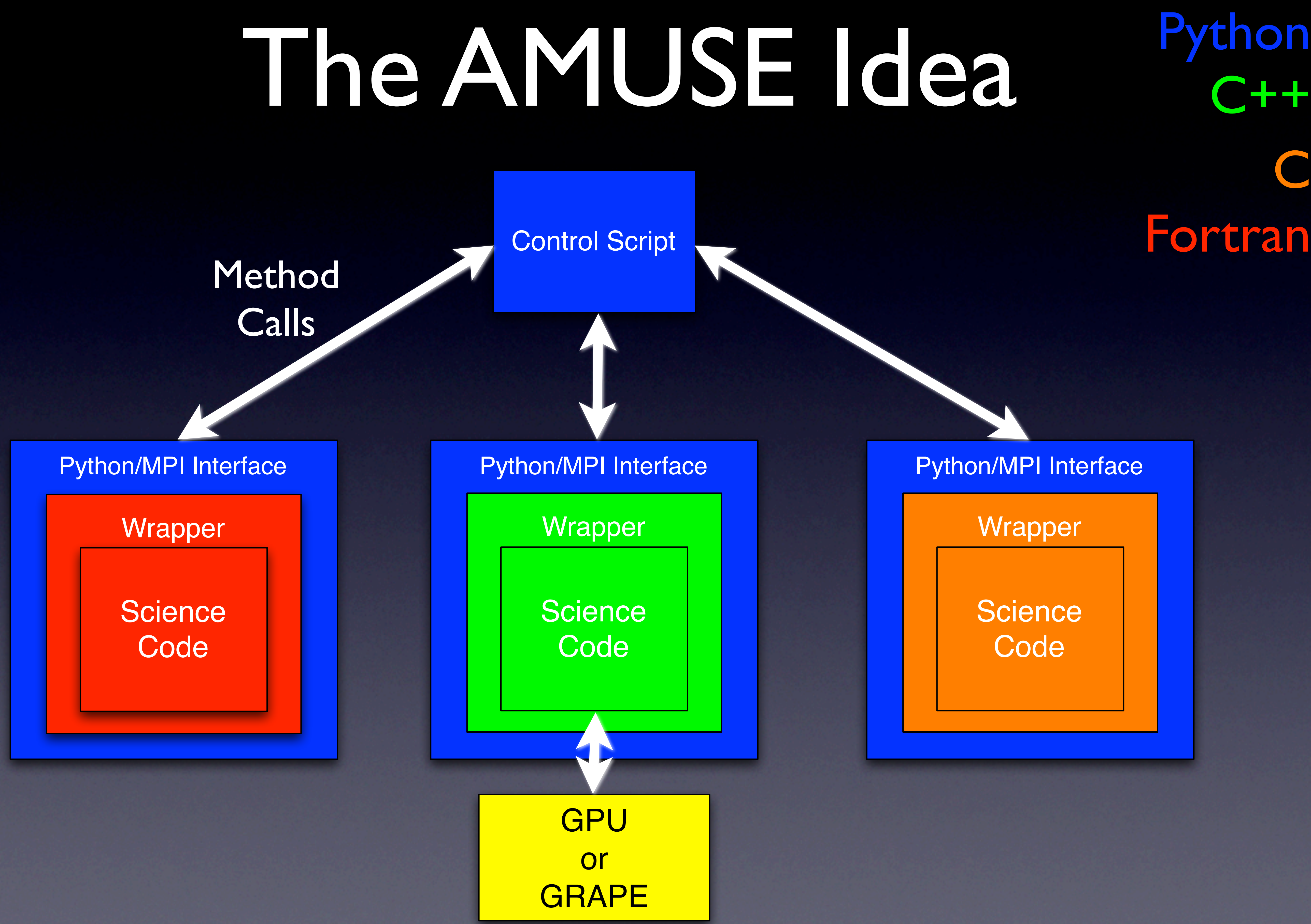
Python
C++
C
Fortran



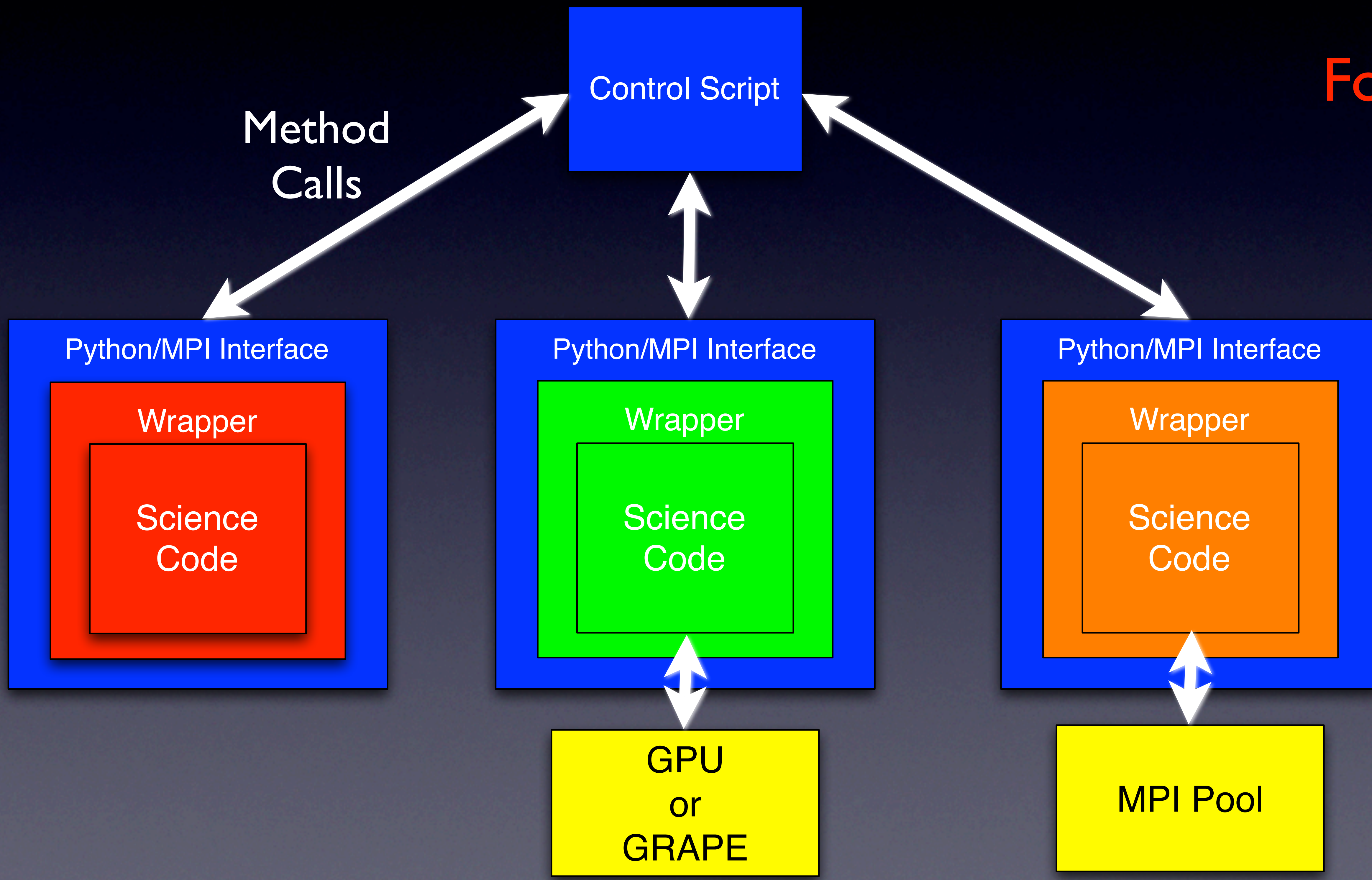
The AMUSE Idea



The AMUSE Idea



The AMUSE Idea



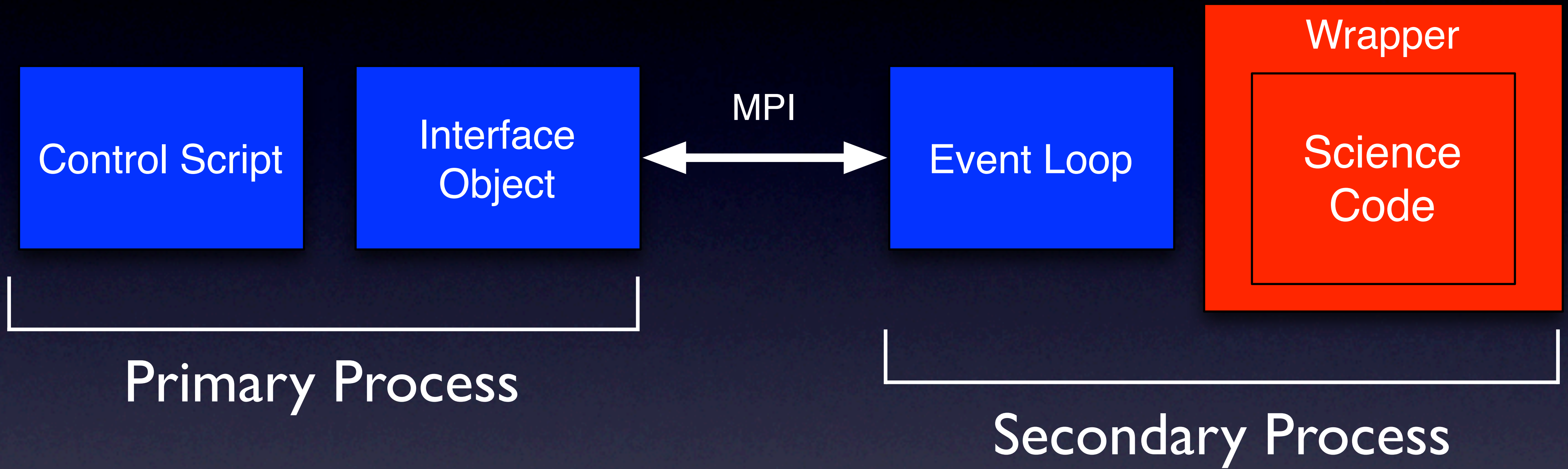
What is AMUSE?

- A way to compose new simulations using existing codes in a modular fashion.
- Standardized interface to different science codes
- “AMUSE” = Astrophysical Multipurpose Software Environment
- ... under construction!

What does it do?

- Gravitational Dynamics
  - BHTree, hermite0, phiGRAPE, twobody, octgrav
- Stellar Evolution
  - SSE, BSE, EVTwin, MESA
- Hydrodynamics
  - Athena, Capreole, FI, GADGET-2
- Next release (April 2011): Radiative Transfer
- Anyone can add a new module
- Also provides: Unit conversion, I/O, other "housekeeping"

Process Isolation



- Control script runs in a different process from calculations
- Each module exists in its own process
- MPI is used for interprocess communication
- MPI is NOT used for parallelism here (although it can be elsewhere)
- Event loop, Interface object are auto-generated by AMUSE at compile time

AMUSE Control Script Example

```
initial_mass_function = SalpeterIMF()
total_mass, salpeter_masses \
    = initial_mass_function.next_set(number_of_stars)

convert_nbody = nbody_system.nbody_to_si(total_mass, 1.0 | units.parsec)

particles = MakePlummerModel(number_of_stars, convert_nbody).result;

gravity = BHTree(convert_nbody)
gravity.initialize_code()
#gravity.parameters.set_defaults()
#print gravity.parameters.timestep.as_quantity_in(units.Myr)
gravity.parameters.timestep = 0.0001 | units.Myr    # tiny!
gravity.parameters.epsilon_squared \
    = (float(number_of_stars)**(-0.333333) | units.parsec) ** 2

stellar_evolution = SSE()
stellar_evolution.initialize_module_with_default_parameters()

print "setting masses of the stars"
particles.radius = 0.0 | units.RSun
particles.mass = salpeter_masses

print "initializing the particles"
stellar_evolution.particles.add_particles(particles)
from_stellar_evolution_to_model \
    = stellar_evolution.particles.new_channel_to(particles)
from_stellar_evolution_to_model.copy_attributes(["mass"])

print "centering the particles"
particles.move_to_center()
print "scaling particles to viridial equilibrium"
particles.scale_to_standard(convert_nbody)

gravity.particles.add_particles(particles)
from_model_to_gravity = particles.new_channel_to(gravity.particles)
from_gravity_to_model = gravity.particles.new_channel_to(particles)

gravity.commit_particles()

time = 0.0 | units.Myr
particles.savepoint(time)


total_energy_at_t0 = gravity.kinetic_energy + gravity.potential_energy
```

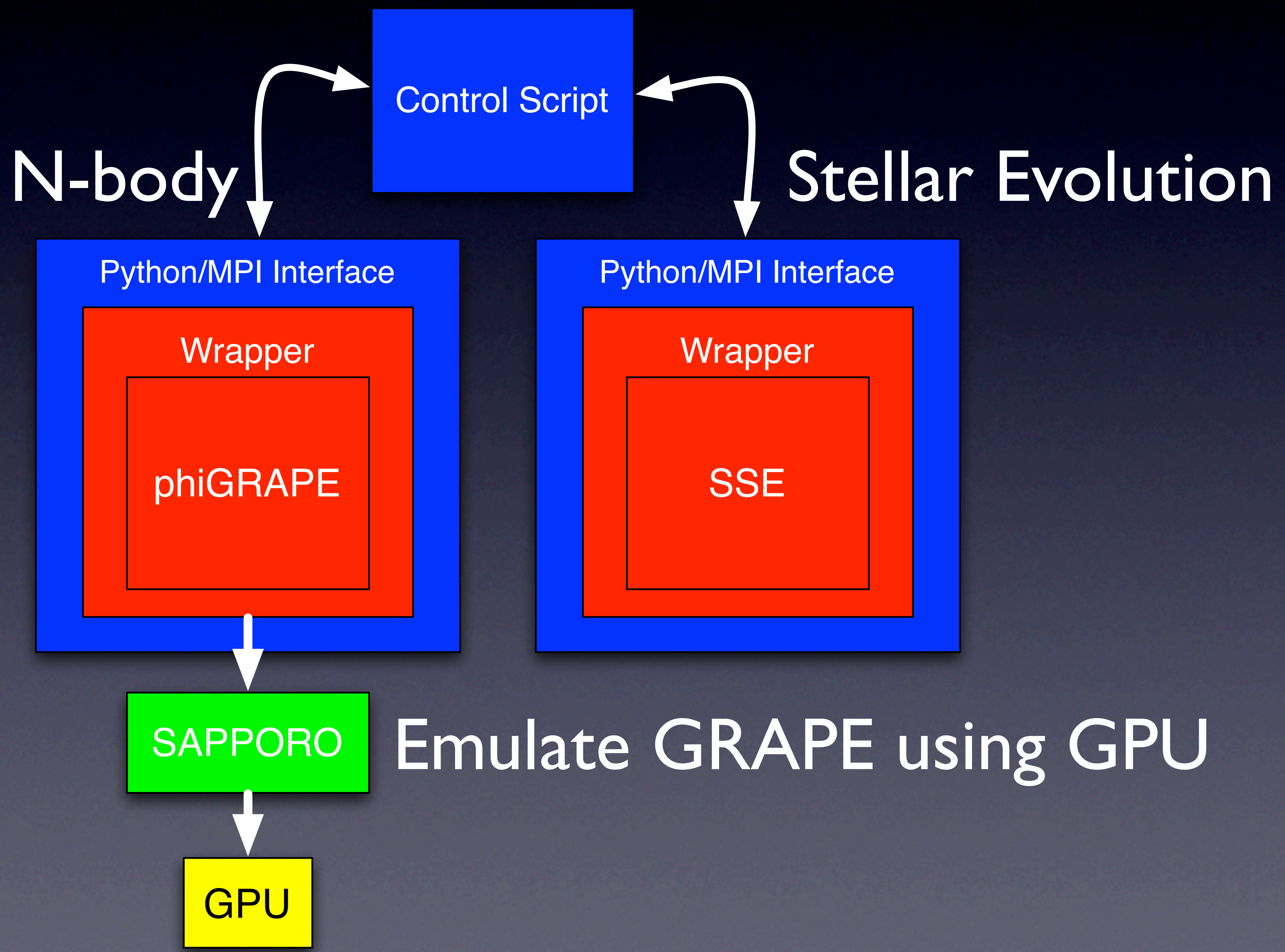
```
print "evolving the model until t = " + str(end_time)
while time < end_time:
    time += 0.25 | units.Myr

    print "gravity evolve step starting"
    gravity.evolve_model(time)
    print "gravity evolve step done"

    print "stellar evolution step starting"
    stellar_evolution.evolve_model(time)
    print "stellar evolution step done"

    from_gravity_to_model.copy()
    from_stellar_evolution_to_model.copy_attributes(["mass", "radius"]

    particles.savepoint(time)

    from_model_to_gravity.copy_attributes(["mass"])

    total_energy_at_this_time \
        = gravity.kinetic_energy + gravity.potential_energy
    print_log(time, gravity, particles,
              total_energy_at_t0, total_energy_at_this_time)


test_results_path = get_path_to_results()
output_file = os.path.join(test_results_path, "small.hdf5")
if os.path.exists(output_file):
    os.remove(output_file)
storage = store.StoreHDF(output_file)
storage.store(particles)

gravity.stop()
stellar_evolution.stop()

plot_particles(particles, name_of_the_figure)
```

Star Clusters in AMUSE

- Goal: N-body + Stellar Evolution, with simple tidal cut-off
- Collisionless (using softening)
- Compare to previous work:
  - Chernoff & Weinberg (1990)
  - Takahashi & Portegies Zwart (2000)

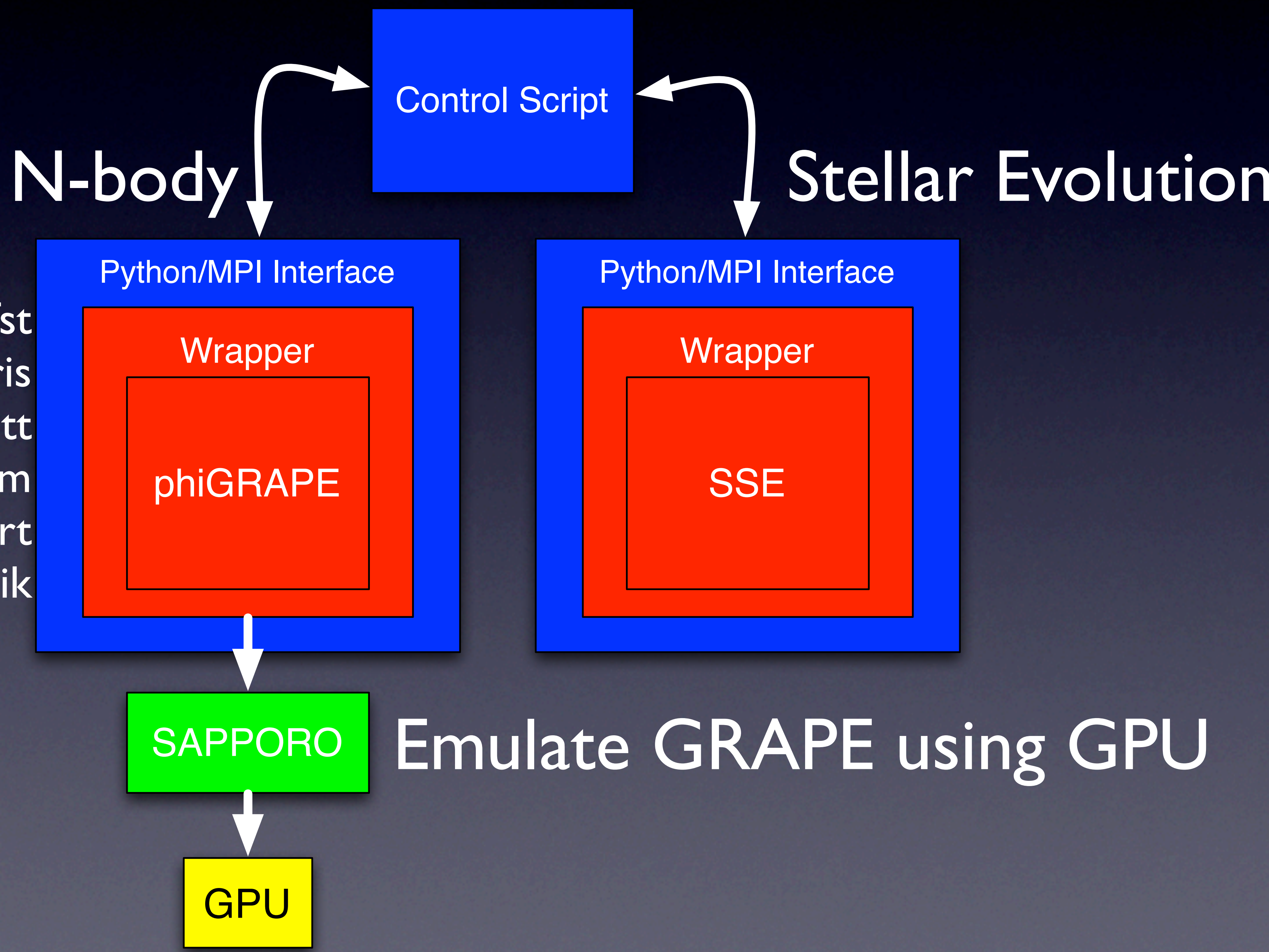
Code Structure

Python
C++
C
Fortran



Code Structure

Python
C++
C
Fortran

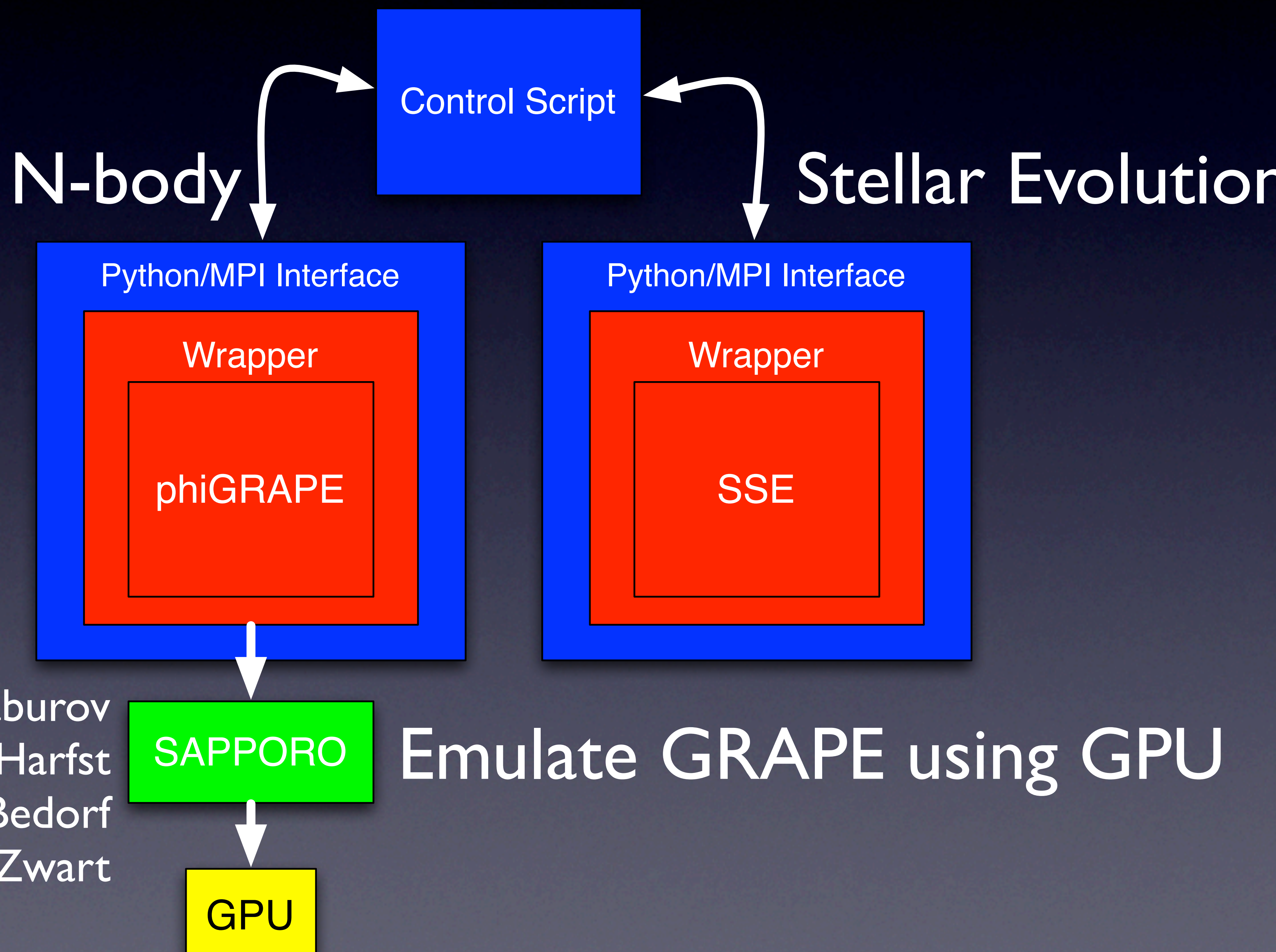


Stefan Harfst
Alessia Gualandris
David Merritt
Rainer Spurzem
Simon Portegies Zwart
Peter Berczik

Emulate GRAPE using GPU

Python
C++
C
Fortran

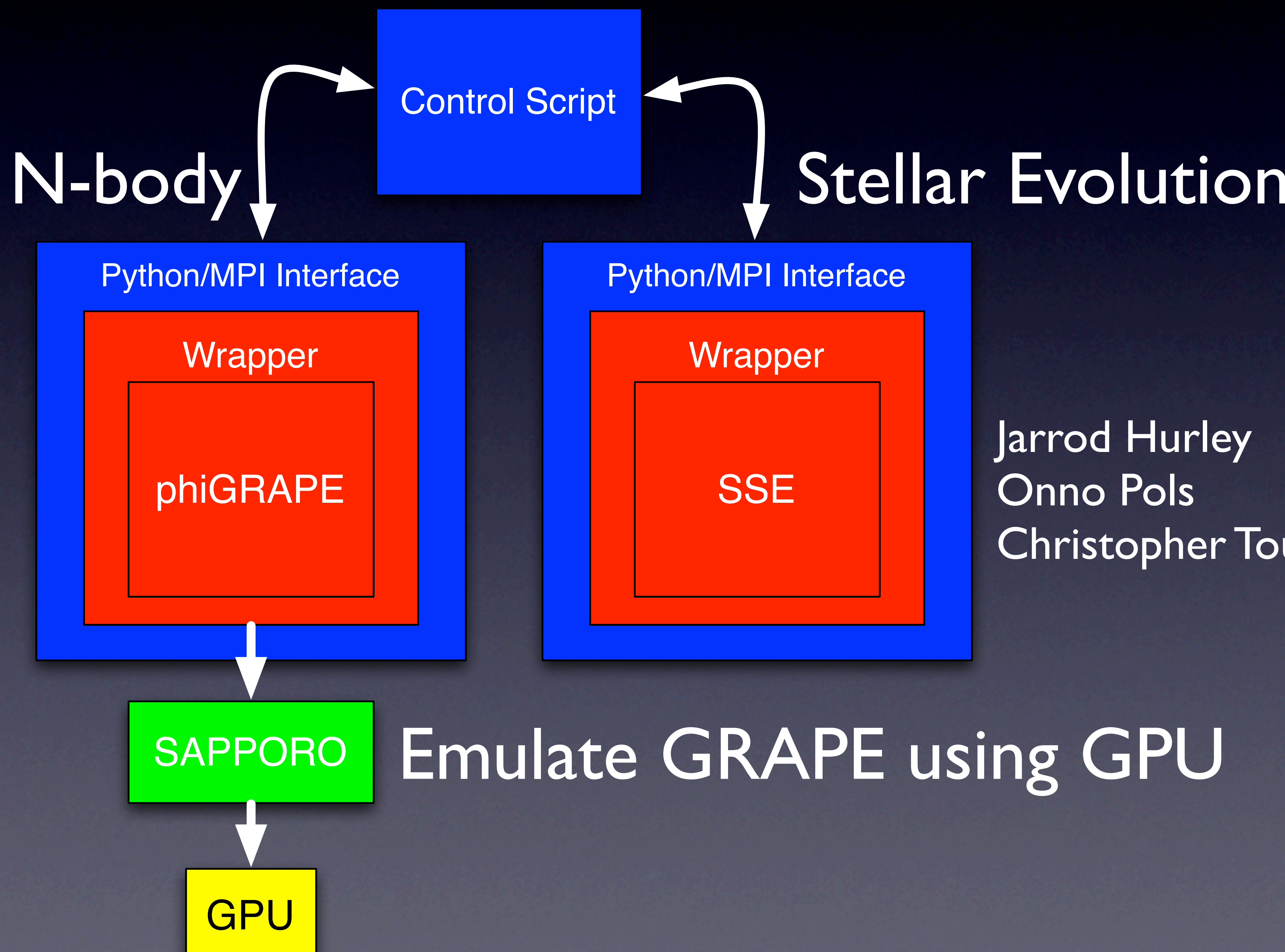
Code Structure



Evghenii Gaburov
Stefan Harfst
Jeroen Bedorf
Simon Portegies Zwart

Python
C++
C
Fortran

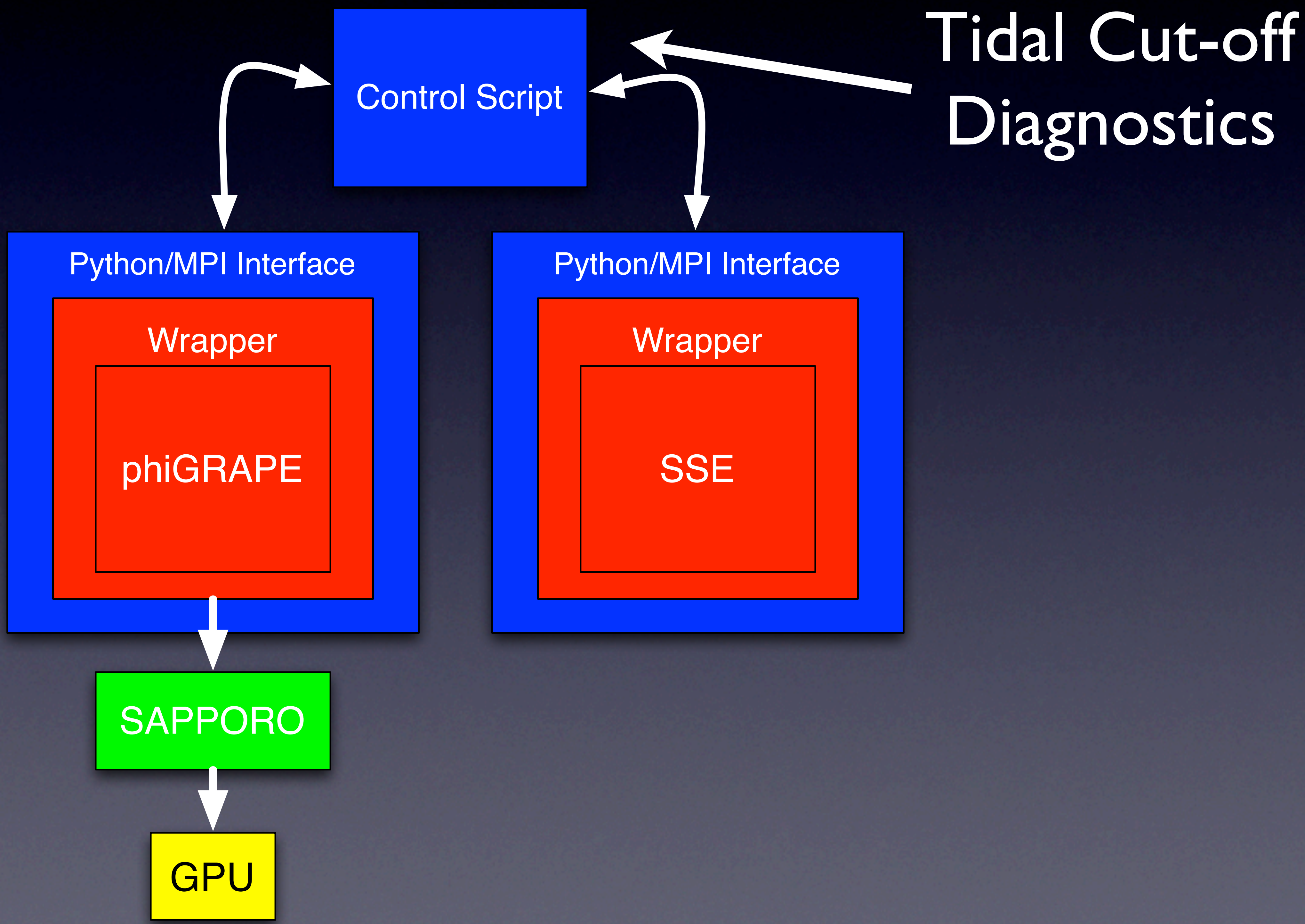
Code Structure



Jarrod Hurley
Onno Pols
Christopher Tout

Python
C++
C
Fortran

Code Structure



Parameter Space

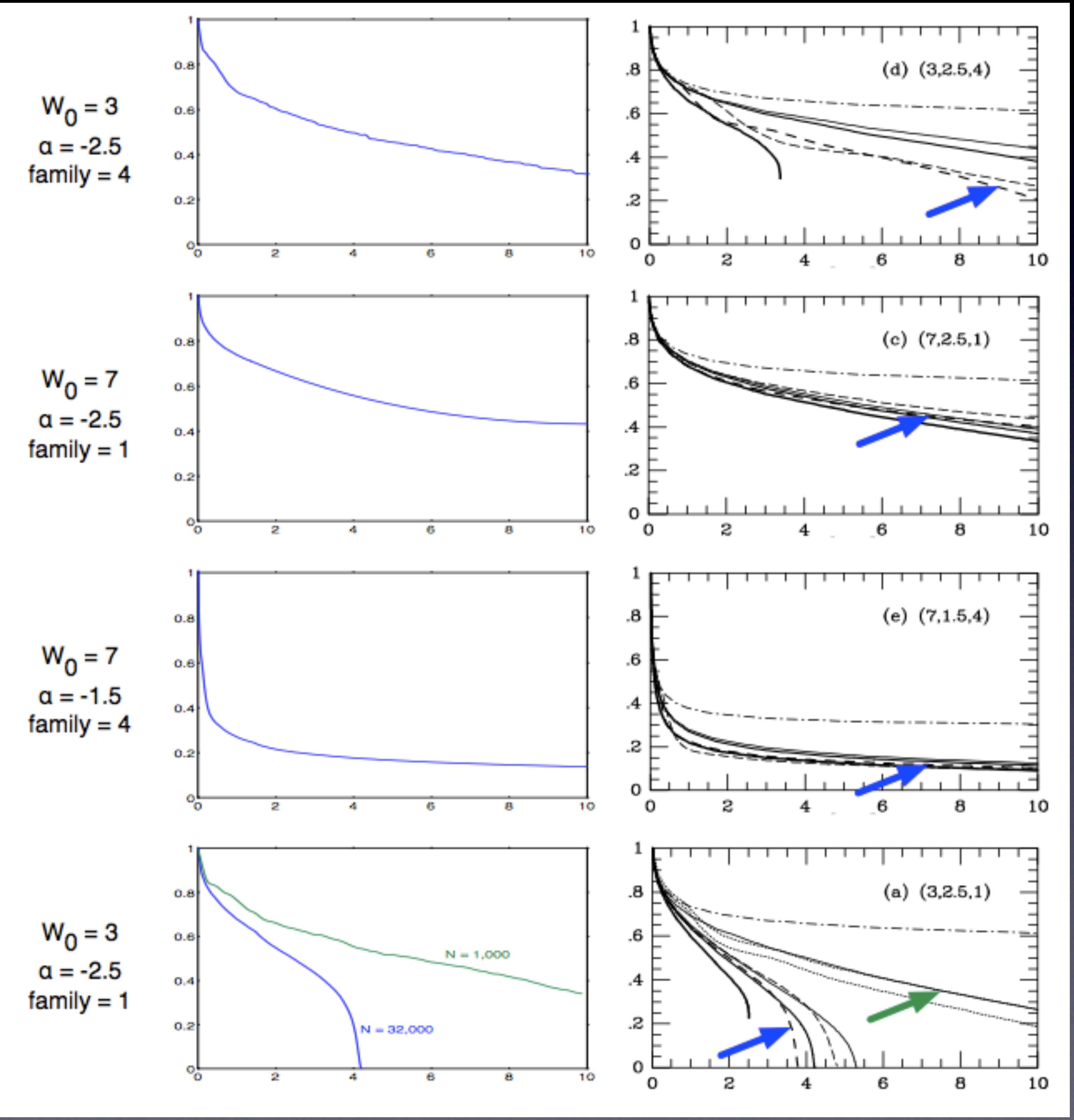
- Subset of Chernoff & Weinberg
- N = 32k
- King Model, W0 = 3 or 7
- Mass Function Slope (in log-log space) is -1.5 or -2.5.
- "Family" parameter takes values 1 through 4. Defines the relative length of the tidal timescale to the stellar evolution timescale

Results

Our Work

All plots are Mass (vertical) vs. Time (horizontal)

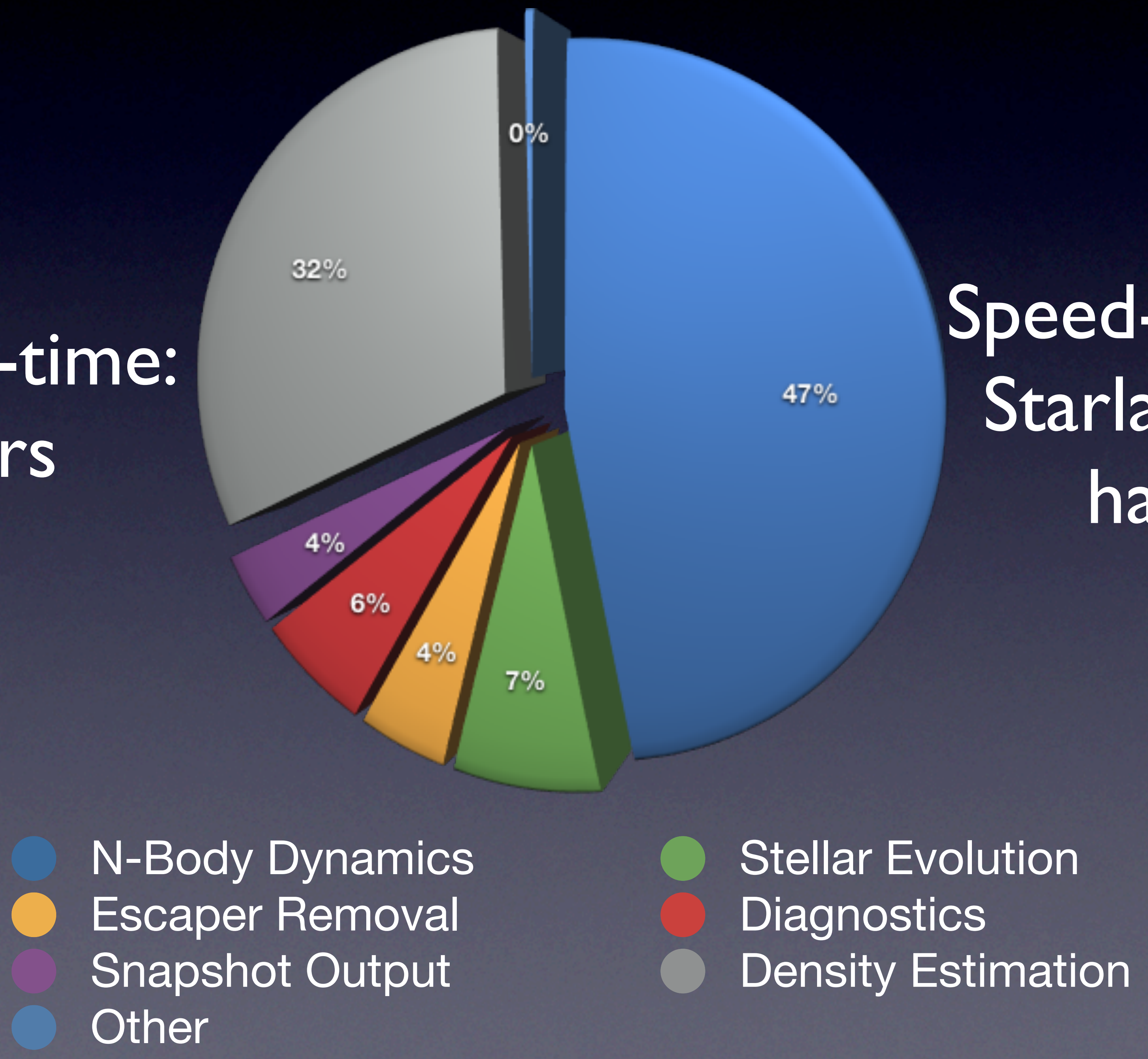
Scales:
Mass normalized to 1 at start
Time in Gyr



Takahashi & Portegies Zwart

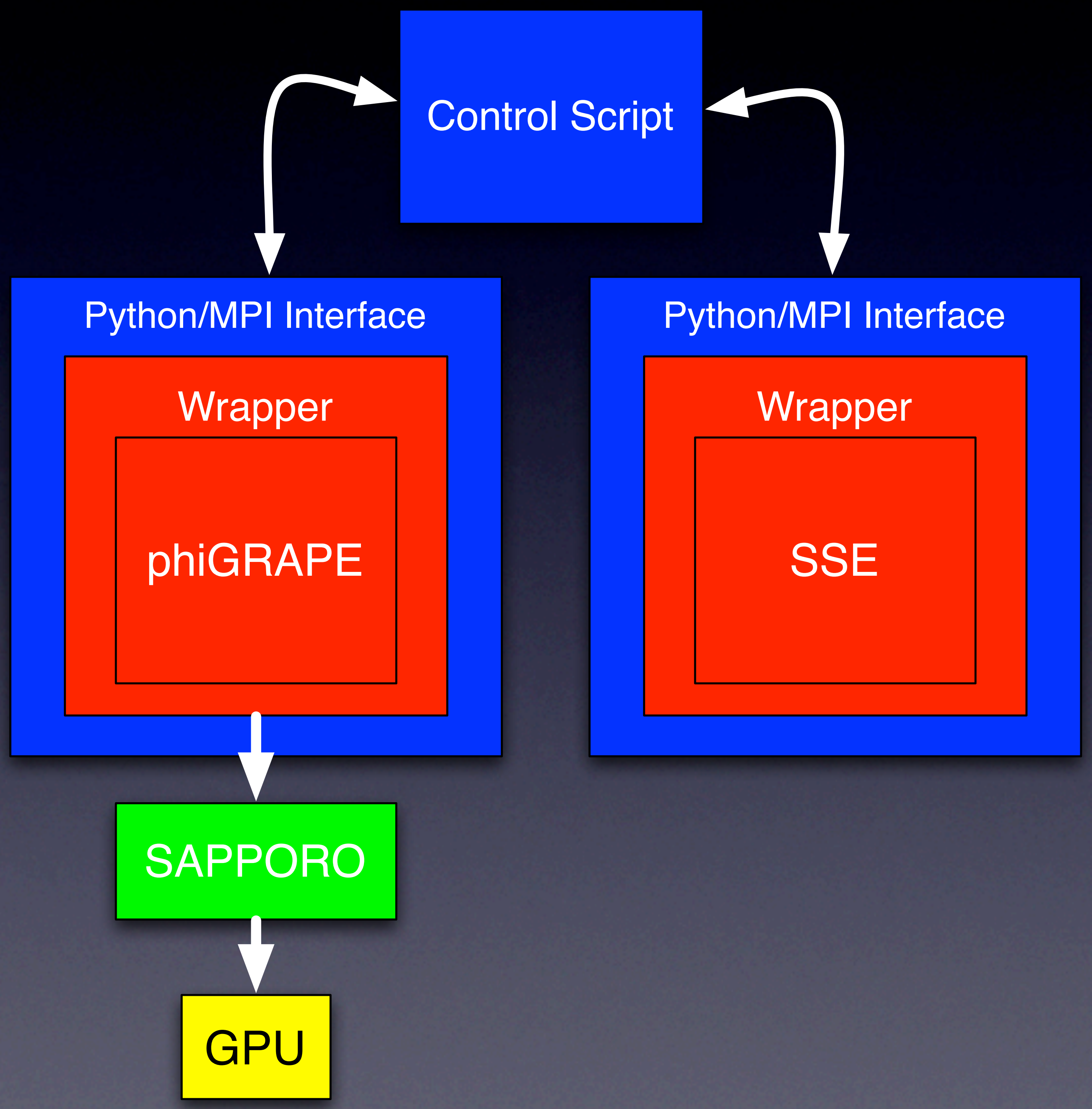
Performance

Typical Run-time: ~ 4 hours



Speed-up: ~3 over Starlab on same hardware

Stellar Evolution Model Comparison



What if we try different stellar evolution models?

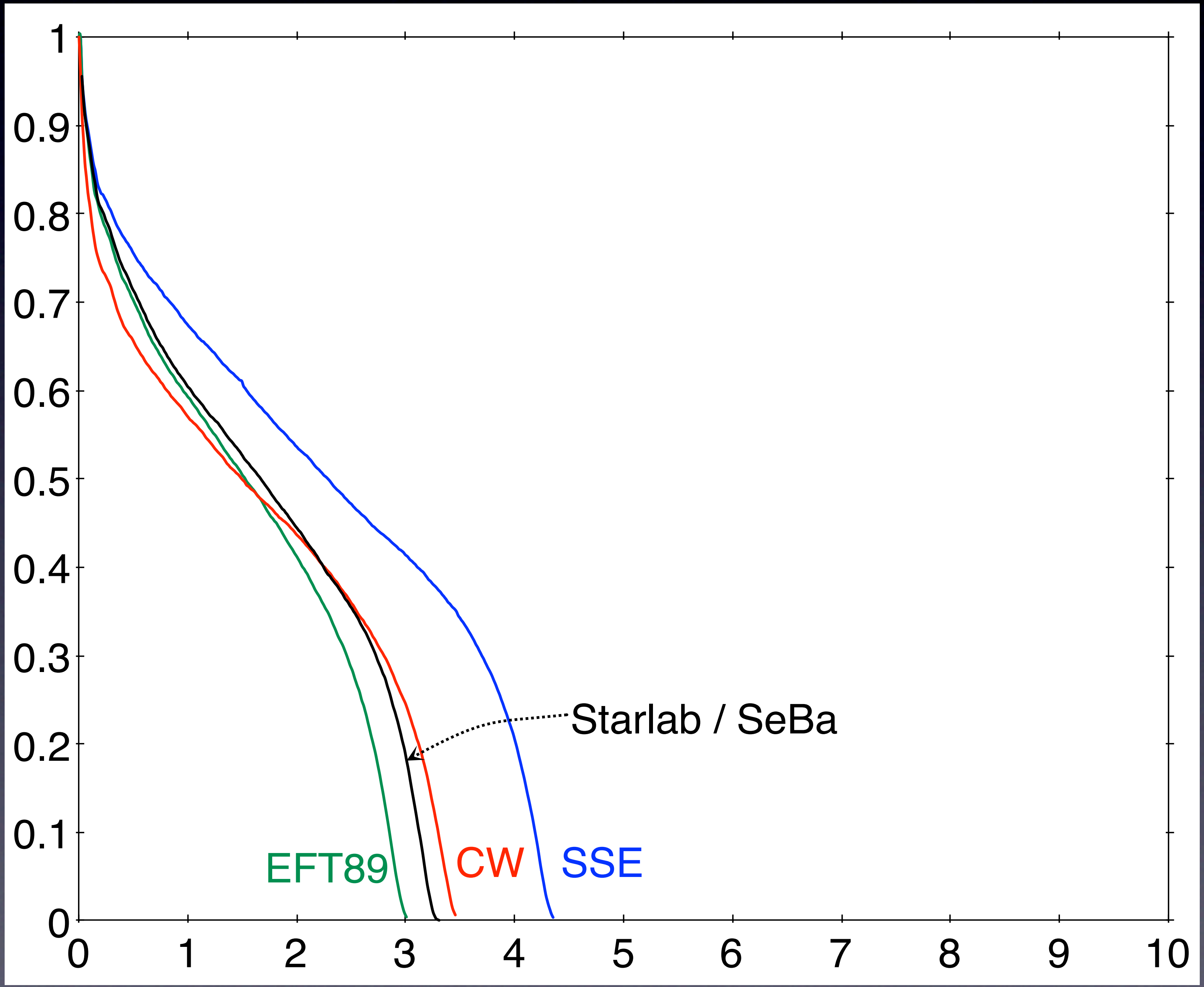
We should be able to directly measure the effect of model choice on the evolution of the system

Alternative S.E. Models

- SSE was used already. It fits analytic formulae to the results of detailed simulations
- EFT89 (Eggleton, Fitchett & Tout) is similar in approach
- CW (Chernoff & Weinberg) use a very simple analytic approximation for stellar evolution
- SeBa is used by the Starlab package, which was used by Takahashi & Portegies Zwart

S.E. Model Comparison

Mass
1 at start



Time (Gyr)

Parameters
W0 = 3
M.F. Slope = -2.5
Family = 1

Conclusions

- Our AMUSE runs are in good agreement with existing work, apart from small differences due the different stellar evolution models used, validating the use of AMUSE as a research tool.
- The modular structure of AMUSE facilitates comparison of physics modules and enables exploration of assumptions and approximations that is difficult or impossible with other simulation codes.

Conclusions

- Specifically, AMUSE allows direct comparison of the effect of differing stellar evolution models. The choice of model can change the computed lifetime of a cluster near disruption by up to ~25%.
- For the adopted parameters, AMUSE outperforms Starlab's kira by a factor of ~2.

Future Work

- Collisional N-body dynamics within AMUSE
- Simulations including gas dynamics
- Large N (1 million) star simulation of a globular cluster

The Core AMUSE Team

- Simon Portegies Zwart (PI)
- Inti Pelupessy (Post-doc)
- Arjen van Elteren (Software Architect)
- Marcell Marosvolgyi (Software Engineer)
- Nathan de Vries (Software Engineer)

Based at Leiden Observatory
Funded by NOVA

Thanks & Questions

Thanks to:

- Collaborators Steve McMillan, Enrico Vesperini, Simon Portegies Zwart
- Jun & Piet for the idea of working in Japan
- CPS for your generous hospitality