GPUnit

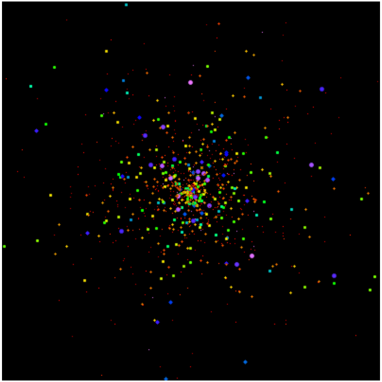
Gabriel Schwartz
Tim McJilton
Andrew Sherman
Rajkumar Jayachandran
Daniel Bagnell
Jason Economou

Advisor: Prof. Jeremy Johnson
Stakeholders: Prof. Steve McMillan
Alfred Whitehead
The Leiden Observatory

May 24, 2011

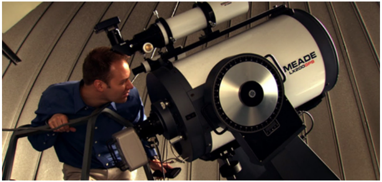
Motivation

- Why simulate astrophysics?
- Why make tools to help run simulations?
- Simulations are large and complex.
  - 1K - 1M particles
  - Many steps.
- Complex software (AMUSE) exists to perform these computations efficiently.
  - Hardware-specific algorithms
  - More/less accurate algorithms



Target Audiences

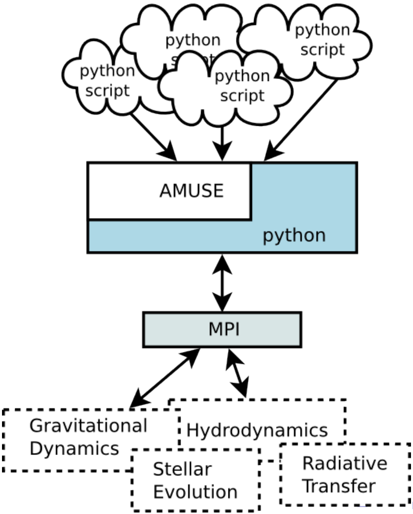
- Physics Students
- Observational Astrophysicists
- Theoretical Astrophysicists



$$
\begin{aligned}
&\ddot{X} + \frac{C}{M}\dot{X} + \omega_n^2\left[1 - \frac{f(\Phi)}{2}(\Delta k_1 + \Delta k_2 \cos 2\Phi)\right]X \\
&\quad - \frac{\omega_n^2 f(\Phi)\Delta k_2 \sin 2\Phi}{2}\left(Y_m - \frac{P}{K}\right) \\
&\quad = \varepsilon(\Omega + \dot{\varphi})^2 \cos(\Phi + \delta) + \varepsilon\ddot{\varphi}\sin(\Phi + \delta), \\
&\ddot{Y}_m + \frac{C}{M}\dot{Y}_m - \frac{\omega_n^2 f(\Phi)\Delta k_2 \sin 2\Phi}{2}X \\
&\quad + \omega_n^2\left[1 - \frac{f(\Phi)}{2}(\Delta k_1 - \Delta k_2 \cos 2\Phi)\right]Y_m \\
&\quad = \varepsilon(\Omega + \dot{\varphi})^2 \sin(\Phi + \delta) - \varepsilon\ddot{\varphi}\cos(\Phi + \delta) \\
&\quad - \frac{Pf(\Phi)}{2M}(\Delta k_1 - \Delta k_2 \cos 2\Phi), \\
&\ddot{\theta} + \frac{K_t + K_c}{I_0}\theta - \frac{K_t}{I_0}\varphi = -\frac{C_t + C_c}{I_0}\dot{\theta} + \frac{C_t}{I_0}\dot{\varphi}, \\
&\ddot{\varphi} + \frac{C_t}{I}\dot{\varphi} - \frac{C_t}{I}\dot{\theta} + \frac{K_t}{I}\varphi - \frac{K_t}{I}\theta \\
&\quad = \frac{P\,\varepsilon f(\Phi)}{2I}(\Delta k_1 \cos(\Phi + \delta) - \Delta k_2 \cos(\Phi - \delta)) \\
&\quad + \frac{P^2}{2KI}\left[\frac{1}{2}\frac{\partial f(\Phi)}{\partial \Phi}(\Delta k_1 - \Delta k_2 \cos 2\Phi) \right. \\
&\quad \left. + f(\Phi)\Delta k_2 \sin 2\Phi\right] + \Gamma_c,
\end{aligned}
$$

Overview

Astrophysical Multipurpose Software Environment (AMUSE)

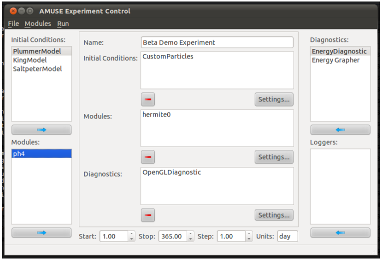


State of AMUSE

- Currently used by researchers to run large-scale simulations.
- Scripts, diagnostics, logging are all written by hand.
- AMUSE API/programming knowledge is required to create experiments.
- Still better than separated and opaque FORTRAN codes.

Purpose of GPUnit

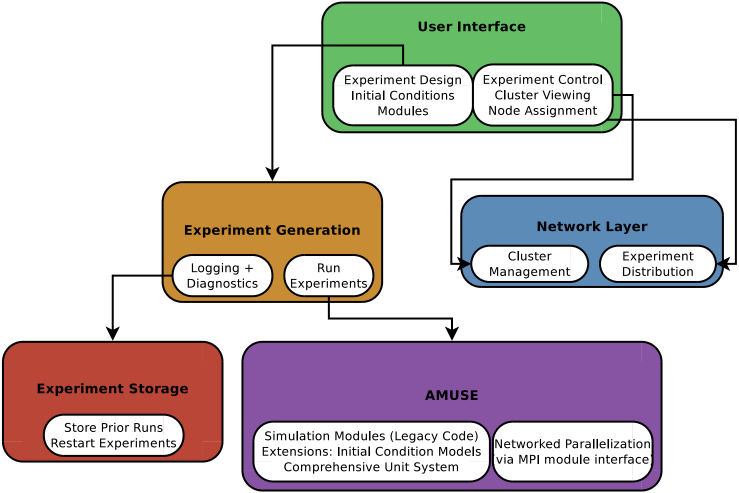
- Ease the use of AMUSE
- Create/Design/Modify experiments
- Select, configure, swap out modules and initial conditions
- Store and restore progress of running experiments.



Features

- Configurable experiments that can be saved and shared.
- Diagnostic tools that compute and display useful measurements.
- Storage of experiment state in case of crashes.
- Custom diagnostics and code generation.
- Provides a display of cluster usage to aid in scheduling.

Architecture



Design

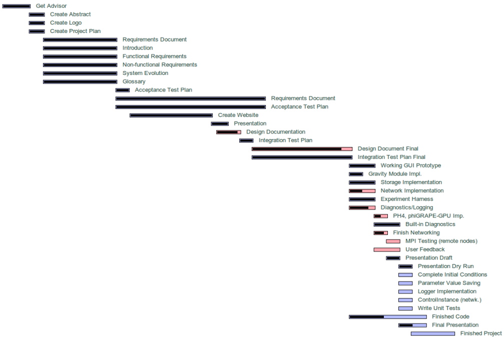
- AMUSE is the only integrated simulation environment available.
- AMUSE is written in Python, streamlines interaction.
- C++ was considered as it supports Qt as well.
  - Communication w/AMUSE would be cumbersome.
  - AMUSE would be in a separate process.
- Designed APIs for diagnostics, logging and experiment persistence.
  - Users can create new diagnostics easily.
  - Experiments can be stored in a file structure, a remote DB etc...

Tests

- GUI / Integration tests were performed manually.
  - Created and ran a simple experiment from scratch to ensure functionality.
- Unit testing performed using Python's unittest module (PyUnit).
- Tests:
  - networking
  - built-in diagnostics
  - object serialization
  - experiment storage
  - experiment running

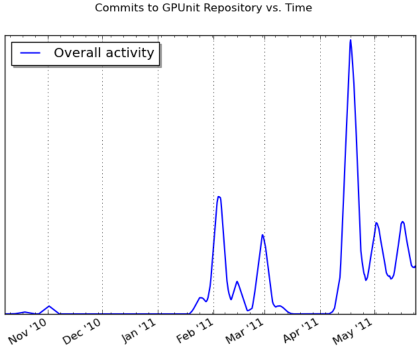
Project Plan

- Mostly waterfall design process.
- Initial phases were spent learning the domain (Physics/AMUSE).
- Roles
  - Tim: Physics reference, test subject
  - Andrew/Jason: Experiment and Module design.
  - Dan: Diagnostics
  - Raj: Logging
  - Gabe: Network, GUI.



Team Management

- Used Mercurial as our version control system.
  - Distributed, allows off-line commits.
- Team met weekly.
  - Once to plan work, once to code.
- Bi-weekly advisor meetings.



Project Impact

- Gives students and physicists easy access to state-of-the-art tools.
- Simple experiment creation $\rightarrow$ faster turnaround on experiments.
- Faster experiments $\rightarrow$ more time to study them.
- Current state:
  - Software is usable to create simple experiments.
  - Comes with useful diagnostics, from real experimental setups.
  - Ready to get feedback from more advanced users.
  - Capability/APIs already exist to provide more advanced features.

Demo

- Demonstration of a simulation.

Questions

- Questions?