GPUnit

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

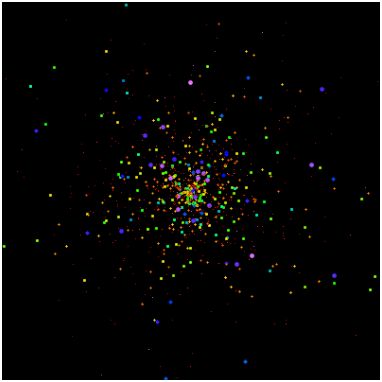
Advisor: Prof. Jeremy Johnson
Stakeholders: Prof. Steve McMillan
Alfred Whitehead
The Leiden Observatory

May 9, 2011

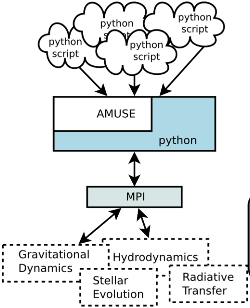
Overview

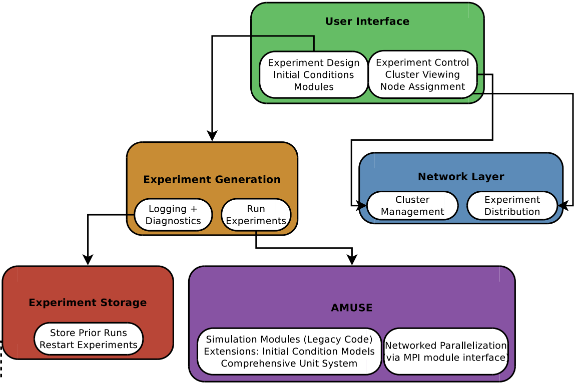
Motivation

- Astrophysics researchers need to simulate movement and evolution of star clusters and galaxies.
- Every star pulls on all of the others: $O\left(n^2\right)$ for the simplest algorithm.
- Stars evolve over time, mass and size changes.



Astrophysical Multipurpose Software Environment (AMUSE)





http://www.amusecode.org

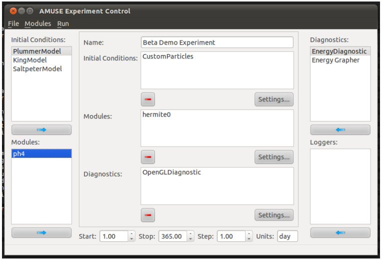
State of AMUSE

- 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.

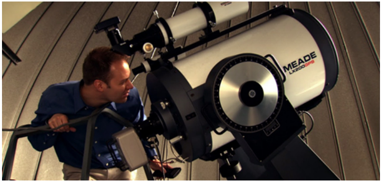
```
case 219539042:
  for (int i = 0 ; i < request_header.len; i++){
    ints_out[i] = get_number_of_stopping_conditions_set(
      &ints_out[( 1 * request_header.len) + i]
    );
  }
  reply_header.number_of_ints = 2;
  break;

case 243588422:
  doubles_out[0] = par_getd(
    characters + ( 0- 1 < 0 ? 0 :strings_in[0 - 1] + 1) ,
    characters + ( 1- 1 < 0 ? 0 :strings_in[1 - 1] + 1)
  );
  reply_header.number_of_doubles = 1;
  break;

case 271440754:
  for (int i = 0 ; i < request_header.len; i++){
    ints_out[i] = is_stopping_condition_set(
      ints_in[i] ,
      &ints_out[( 1 * request_header.len) + i]
    );
  }
  reply_header.number_of_ints = 2;
  break;

case 280123374:
  for (int i = 0 ; i < request_header.len; i++){
    ints_out[i] = esys_roe_adb_hydro(
      &ints_out[( 1 * request_header.len) + i] ,
      &doubles_out[i] ,
```

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.



Target Audiences



$$\ddot{X} + \frac{C}{M}\dot{X} + \omega_n^2\Big[1 - \frac{f(\Phi)}{2}(\Delta k_1 + \Delta k_2 \cos 2\Phi)\Big]X - \frac{\omega_n^2 f(\Phi)\Delta k_2 \sin 2\Phi}{2}\Big(Y_m - \frac{P}{K}\Big) = \varepsilon(\Omega + \dot{\varphi})^2 \cos(\Phi + \delta) + \varepsilon\ddot{\varphi}\sin(\Phi + \delta),$$

$$\ddot{Y}_m + \frac{C}{M}\dot{Y}_m - \frac{\omega_n^2 f(\Phi)\Delta k_2 \sin 2\Phi}{2}X + \omega_n^2\Big[1 - \frac{f(\Phi)}{2}(\Delta k_1 - \Delta k_2 \cos 2\Phi)\Big]Y_m = \varepsilon(\Omega + \dot{\varphi})^2 \sin(\Phi + \delta) - \varepsilon\ddot{\varphi}\cos(\Phi + \delta) - \frac{Pf(\Phi)}{2M}(\Delta k_1 - \Delta k_2 \cos 2\Phi),$$

$$\ddot{\theta} + \frac{K_t + K_c}{I_0}\theta - \frac{K_t}{I_0}\varphi = -\frac{C_t + C_c}{I_0}\dot{\theta} + \frac{C_t}{I_0}\dot{\varphi},$$

$$\ddot{\varphi} + \frac{C_t}{I}\dot{\varphi} - \frac{C_t}{I}\dot{\theta} + \frac{K_t}{I}\varphi - \frac{K_t}{I}\theta = \frac{P\varepsilon f(\Phi)}{2I}(\Delta k_1 \cos(\Phi + \delta) - \Delta k_2 \cos(\Phi - \delta)) + \frac{P^2}{2KI}\Big[\frac{1}{2}\frac{\partial f(\Phi)}{\partial \Phi}(\Delta k_1 - \Delta k_2 \cos 2\Phi) + f(\Phi)\Delta k_2 \sin 2\Phi\Big] + \Gamma_c,$$

- Physics Students
- Observational Astrophysicists
- Theoretical Astrophysicists

Features

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

Design

- Written in Python using the PyQt4 GUI toolkit.
- 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

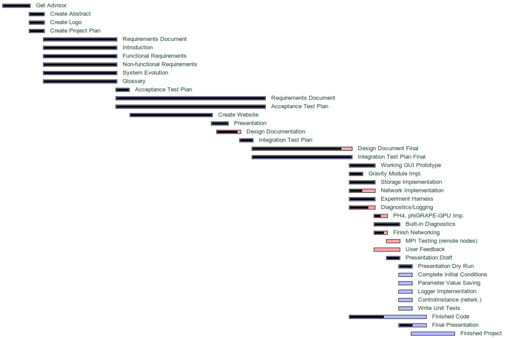
- Table of tests that pass.

User Testing

- Tested with customers (Steve/Tim)

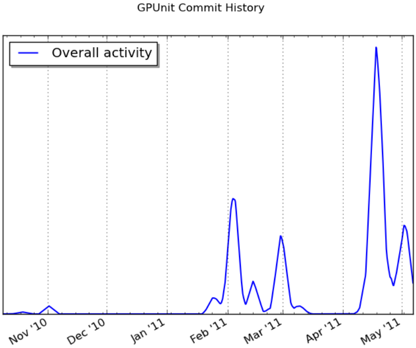
Project Plan

- Mostly waterfall design process.
- Initial phases were spent learning the domain (Physics/AMUSE).
- Needed more knowledge before making a plan.



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.



Potential Benefits

Demo

- Demonstration of a simulation.

Questions

- Questions?