Software Development in Academic, Research, and Industrial Environments

Jim Teresco

Department of Computer Science
The College of Saint Rose
Albany, New York, USA

CSC 507, Software Engineering

March 18, 2013

Yet another Powerpoint-free presentation!

Outline

- Software development in several contexts
  - interdisciplinary academic research laboratory
  - national laboratory
  - small Internet company
  - individual/small group academic projects
- Issues for discussion
  - working within each environment
  - naming and style conventions
  - version control and release engineering
  - testing

Interdisciplinary Academic Research Laboratory

- Faculty, postdocs, research staff, graduate and undergraduate students
- Housed across several academic departments
- Competing and sometimes conflicting goals
  - solve problems of interest
  - current and future funding agencies
  - publications
  - complete master's and Ph.D. theses
  - undergraduate research projects
- Challenges
  - highly dynamic group membership
  - variety of backgrounds, especially relating to software development
  - interoperability of components

National Research Laboratory

- Primarily full-time staff
- Work often highly collaborative and interdisciplinary
- Goals
  - overriding: solve problems of interest
  - publications
  - involve academic partners (the “pipeline”)
- Challenges
  - higher verification and validation standards
  - meet needs of government and external customers
  - interoperability of components

Small Internet Company

- Small full-time team of skilled programmers
- Client-focused development
- Goals
  - meet short- and long-term requirements of clients
  - leverage existing open source and proprietary systems
  - reuse!
- Challenges
  - large code base to maintain
  - manage client-specific *vs*. general-usage code
  - highly dynamic code but need high reliability and near 100% uptime

Smaller-scale Academic Software Development

- One faculty member, occasional graduate/undergraduate student involvement
- Often collaborative
- Goals
  - construct useful standalone or support software
  - publications
  - undergraduate research projects
- Challenges
  - highly dynamic group membership
  - variety of backgrounds, especially relating to software development
  - interoperability of components
  - release engineering
  - finding users

Naming and Style Conventions

- Source code organization
- Namespace management (especially important in C)
- Naming conventions
  - prefixes beyond language standard conventions
- Style
  - spacing, indentation, bracket positions
- Safety
  - memory management macros
- Documentation requirements

Version Control and Release Engineering

- All but the most trivial systems require source code control
- Periodic “public” release versions
  - maintain bug fix branches for released versions
  - end-of-life for support?
  - backward compatibility
- Managing development versions
- Development/staging/production servers

Testing

- Testing is obviously important in all contexts
- Unit tests
  - many test cases
  - different target platforms
  - nightly builds and tests?
- Back to development/staging/production server idea
- Major complication with parallelism: different interleavings of order of execution