The Sandia Analysis Workbench
Leveraging a COTS Framework To Provide Integrated Engineering Analysis Workflows On HPC Systems

Ernest J. Friedman-Hill
Edward L. Hoffman
Robert L. Clay
Outline

- Problem Space
- The Eclipse Framework
- Tool Case Studies
Support the Design-To-Analysis process, capturing data in context

- **Design Model Development**
- **Analysis Model Development**
- **Simulation**
- **Results Processing**

Uncertainty Quantification/Design Improvement

**Model Database**
**Analysis Database**
**Technical Basis Database**

Other databases that contain technical basis data (e.g., ROA, product acceptance, flight test, lab test, etc.)
Integrating Analysis Systems

Model Building

Workbench Client

Job Submission & Remote Viz

HPC Computing Platforms

Archive

Team Projects

Workbench Central Repository

- Traceability
- Provenance
- Knowledge Retention
- Disaster Recovery
Sandia Laboratories

- Scientists
- Engineers
- Developers

} All working together

- Wide variety of software in use
  - Commercial
  - In-house
- Corporate services available
- Change is a given

- How to make everything work together?
Eclipse

OSGi Architecture
Bundles

Tool A

Bundle A

Bundle B

Bundle C

Tool C

Eclipse Framework
Tool Interactions

- A is an input deck editor
- How can A use C to plot functions in input decks?
- C is a plotting package

Eclipse Framework

- Bundle A
- Bundle B
- Bundle C
Direct Dependencies

- Tool A
- Bundle A
- Bundle B
- Bundle C
- Tool C

Eclipse Framework

- Hard-wired dependency
- No user choice
Extension Points

- “Inversion of Control”
- Framework gives A a list of extensions at runtime

Bundle A

Extension point
(Interface plus property list)

Bundle B

Bundle C

Extension
(Implementation plus property values)
Extension Points

- Compile-time dependencies
- User choice enabled
Bridge Plugins

- Tools are independent
- User choice enabled
Object Adapters

- Adapt one object type to another
- IAdaptable interface
-AdapterManager
  - Any bundle can provide adapters
- No need to use common interfaces!
Eclipse platform selection service

Cubit VTK Viewer

Eclipse Selection Service

Sierra Editor

Common Model Navigator

Properties View
Object Action Contributions

- Eclipse Command/Handler/Menu framework
- Action enablement based on selected object type
- Implemented once, available in many places

Model Tree
Model Viewer
Sierra Editor Tree (via Object Adapter)
Cubit Model Tree
Eclipse “Perspectives”

- An arrangement of views, buttons, menus
- One button push separates these two screens
Sierra Analysis Codes

- Single framework
- Many different physics codes (Thermal, Structural)
- Used separately or coupled
- Available commands described in XML
Sierra Editor

- Reads XML and provides
- Syntax highlighting
- Validation
- Completion
- Content assist
- Hyperlinking

- Supporting other codes
  - Same XML format for commands
  - Syntax implemented with code module
Sierra Builder

- Builds on editor to provide
  - Fully-graphical model building
  - Generated dialogs
  - Custom dialogs (via extension point)
  - Tree-based navigation
CUBIT Mesh Generator

- C++ library
- Interactive application
- Existing Qt GUI
  - Imperative command panels
  - Properties view
  - VTK-based viewer
  - Console
- Our approach: keep mesh viewer, recreate the rest of the GUI
CUBIT: Command Panel Builder

- Novel XML format
- Describe command, not GUI
  - “Hints” for GUI implementation
- Testing for panel generator
CUBIT: Native Code Integration

- Generate glue code with SWIG
- Less than 500 lines of handwritten C++
- Platform-specific fragments
Data Management

- Teams
- NTK
- Metagroups
- Web services
Job Submission

- Modular architecture
  - Machine templates
  - Code templates
  - Defaults
  - Custom templates
- Remote access
  - Heterogeneous machines
- Local access
Multiple Configurations

- Workbench
  - Everything
- CompSimUI
  - Model Building
  - Meshing
  - Job Submission
- Sierra Editor
  - Model Building
  - Job Submission

Eclipse allows us to build multiple application distributions by choosing from among our set of components.
Conclusions

- Eclipse and the OSGi architecture let us
  - ... reduce dependencies between integrated software projects
  - ... integrate diverse components smoothly and robustly
  - ... create and deploy customized solutions easily
Acknowledgements

- The DART Team
  - Douglas Clay, Marcus Gibson, John Greenfield, Stephen Mueller, Kevin Olson, Edward Walsh

- The CUBIT Team
  - Randy Morris, Matt Staten

- The SIERRA Team
  - Mike Glass, Martin Heinstein, Garth Reese, Brett Robertson

This work is supported by the Advanced Strategic Computing program of the NNSA.
Questions

Technical contact:
Ernest Friedman-Hill
925-294-2154
ejfried@sandia.gov

Programmatic contact:
Robert Clay
925-294-3114
rlclay@sandia.gov