Cyber Dumpster Diving – creating new software systems for less

Ian Gorton, Lab Fellow, Group Lead, Data Intensive Scientific Computing, Computational Sciences and Math Division, Pacific Northwest National Lab
PNNL

► Department of Energy Science Lab
  ■ Physical and fundamental sciences
  ■ National security
► 4500+ people
► Business volume of over $1b per annum
► Large scale experimental facilities, e.g.
  ■ Environmental Molecular Sciences Lab (EMSL)
  ■ 161 Tflop supercomputer
The most radical possible solution for constructing software is not to construct it at all.

Fred Brooks: No Silver Bullet: Essence and Accidents of Software Engineering
welcome to my world
DISC@PNNL

- Data Intensive Scientific Computing
  - User platforms
  - Data management
  - Tool integration
  - Workflows
  - Provenance
- Applications in e.g.
  - Bioinformatics
  - Climate modeling
  - Carbon sequestration
  - Subsurface modeling
The middle is a hard place …

**Requirements**
- Need to understand science domain
- Need to understand HPC
- Difficult to define, constant refinement, negotiate, negotiate
- “The hardest single part of building a software system is deciding precisely what to build.”

**Design**
- Conflicting quality requirements
- Complex, heterogeneous technologies
- Large data
- Proliferation of tools, variable quality
Project Funding Profiles

- Typically fixed amounts
  - What can we build with X dollars?
  - Fixed amounts per year, 1-3 year lifecycle

- Limited funding
  - From .25 to 10 team size per year
  - 1-2 people per year most common

- High expectations
  - Scientists think ‘software is easy’
  - it’s just coding, right?
Some Examples
Carbon Sequestration (Storage)
Geological Sequestration Software Suite (GS3)

- Large-scale, complex data
  - Experimental
  - HPC Simulation inputs/outputs
  - Multiple realizations for UQ
- Long-lived projects
  - Modeling
  - Analysis
  - Monitoring (100+ years)
Advanced Simulation Capability for Environmental Management (ASCEM)

- A State-of-the-art tool for predicting contaminant fate and transport through natural and engineered systems
  - Open source
  - Modular and extensible
  - ‘born’ parallel for execution on emerging architectures.

- A User Platform to manage data, create models and analyze results
Integrated Regional Earth System Model

Key Attributes:
- Modularity
- Portability
- Open-Source
- Integrated Regional Analyses
A powerful, usually legal, source of information that isn't seriously defended because of social taboos.
‘Write-as-little-as-possible’ Reuse

Approach:
- Leverage open source frameworks and tools
- Extend to support science applications
- Generalize to support multiple science domains

Requires:
- Careful technology selection
- Creative design
- Robust architectures
An simple example - bioinformatics
Bioinformatics Workflows

- Biologists want to create computational workflows:
  - Integrate disparate tools and data
  - Span multiple execution platforms
  - Execute reliably and efficiently
  - Capture provenance

- Workflow tools for biology
  - Galaxy
  - Taverna

- Limitations in handling large scale data and computations
Computational Pipeline Infrastructure

Components in the Pipeline

Ian Gorton, Adam Wyman, Yan Liu, and Jian Yin,
Pacific Northwest National Lab

When developers use lightweight approaches such as scripting languages to construct pipelines, restructuring and optimization involves changing the scripts themselves. Because no direct correspondence exists between the major abstractions in the pipeline architecture and the script code, the implementations often drift from the original designs, no longer adhering to the pipeline pattern. At the other extreme, pipelines built using general-purpose graphical workflow environments lack the descriptive scalability to clearly express complex applications. The visual pipeline’s descriptions become littered with low-level data formatting and plumbing that obscure the logical application design and create resistance to change.

http://www.computer.org/portal/web/csdl/doi/10.1109/MS.2011.23
Velo –
Knowledge Management for Modeling and Simulation
Supporting Carbon Sequestration Modeling

- Requirements
  - Collaboration
  - Sharing data
  - Metadata management
  - User-driven customization
  - Extensibility
  - Model versioning
  - Provenance
  - Robust, scalable

- Small project, team ~1.75 people, 3 years
Cyber Dumpster Diving

- Open source
- Candidate technology assessments:
  - Quality of docs
  - Release schedule
  - Community scope
  - APIs
  - Code/architecture
  - Install and workout, simple tests
## Feature-Reuse Matrix

<table>
<thead>
<tr>
<th>Feature</th>
<th>Solution</th>
<th>Notes</th>
<th>Reuse</th>
</tr>
</thead>
<tbody>
<tr>
<td>Collaboration</td>
<td>Mediawiki</td>
<td>Core wiki features support this</td>
<td>100%</td>
</tr>
<tr>
<td>Sharing data</td>
<td>Mediawiki</td>
<td>Requires integration of MW and Alfresco</td>
<td>60%</td>
</tr>
<tr>
<td></td>
<td>Alfresco</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Metadata management</td>
<td>Mediawiki</td>
<td>Requires customization of MW and Alfresco basic features</td>
<td>80%</td>
</tr>
<tr>
<td></td>
<td>Alfresco</td>
<td></td>
<td></td>
</tr>
<tr>
<td>User-driven customization</td>
<td>Mediawiki</td>
<td>Core wiki features support this</td>
<td>100%</td>
</tr>
<tr>
<td>Extensibility</td>
<td>Mediawiki</td>
<td>APIs support extension, but requires design of exact integration mechanisms</td>
<td>20%</td>
</tr>
<tr>
<td></td>
<td>Alfresco</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Model versioning</td>
<td>Mediawiki</td>
<td>Minor extensions for MW/Alfresco capabilities</td>
<td>75%</td>
</tr>
<tr>
<td></td>
<td>Alfresco</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Provenance</td>
<td>Mediawiki</td>
<td>Some for free in MW, but advanced features need developing</td>
<td>20%</td>
</tr>
<tr>
<td>Role-based Security</td>
<td>Halo ACL</td>
<td>Mediawiki extension</td>
<td>100%</td>
</tr>
</tbody>
</table>
GS3 Examples - Semantic Capabilities - Metadata Extraction

- Metadata:
  - Generic information e.g. file size, owner, preview/thumbnails
  - Specific to the file type, e.g. keywords, geographic location

- Metadata is searchable

- Extensible architecture for custom data types ingest pipelines, e.g.
  - Simulation outputs
  - Spreadsheets
  - Input files
GS3 Examples - Tool Integration

- Mediawiki plugins
- ‘Black box’ tools
- External 3\textsuperscript{rd} party tools
GS3 Examples – Tool Plugins
GS3 Examples – Black box Tool Plugins

Number of Target Formations within Reservoir: 1

Target Formation Name: Mt Simon
Geologic Age: Cambrian

Target Formation Rock Types (check all that apply):
- Sandstone
- Limestone
- Dolomite
- Shale
- Coal Seam
- Basalt

Depositional Environment (check all that apply):
- Continental: |
- Alluvial
- Aeolian
- Fluvial
- Lacustrine
- Transitional: |
- Deltaic
- Tidal
- Lagoonal
- Beach
- Marine: |
- Shallow Water
- Deep Water
- Reef
- Others: |
- Evaporite
- Glacial

Sequestration Trapping Mechanisms (check all that apply):
- Dissolution and Diffusion
- Physical Containment
- Mineralization
- Residual Saturation

Target Reservoir Depth and Thickness:
- Top Depth: Min: __________ Max: __________ Mean: 6705 ft
- Bottom Depth: Min: __________ Max: __________ Mean: 9241 ft
- Thickness: Min: __________ Max: __________ Mean: 2536 ft

Estimated Fracture Gradient: 0.8 psi/ft
Estimated Fracture Opening Pressure: 200 psi
What Happened?

- Iterative development process
  - Design, build and demo, repeat
- Interest from user community was strong
  - Power of mock-ups and prototypes
- New funding obtained
- Initial sites deployed
- And along the way …
Flexible, Rigorous Scientific Knowledge Management

<table>
<thead>
<tr>
<th>GS3</th>
<th>ASCEM</th>
<th>FutureGen</th>
</tr>
</thead>
<tbody>
<tr>
<td>SimSeQ</td>
<td>Velo</td>
<td></td>
</tr>
<tr>
<td>Site Data</td>
<td>Model Data</td>
<td>Simulators</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Visualization</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Plume Calcs</td>
</tr>
</tbody>
</table>

User customizable ‘skins’
Web-based
Extensible

Raw data and metadata storage
Versioning
Provenance
Tool registry
Many deployment options

Extensible data types
Extensible tool repository
Programming interfaces
Velo Architecture

Velo Knowledge Base

Velo synchronization process

CMS
(Simulations, Models, Projects)

Data Ingest Pipeline

Convert
Markup
Store

Core Database
Semantic Database

Wiki Database

Core Wiki

Semantic Wiki

Tool Integration

CMS Integration

External Tools
(3D Visualization, Job Execution, Rich GUI)

MediaWiki
Basic Velo ‘Skin’

1. Tool Navigation
2. File Browser
3. Wiki Functions
4. Content Viewer
When requirements collide

Serendipity is not an accident
Bringing it all Together – iRESM Platform

Data Processing
To BEND Model
1. Transforming CASCADE climate to VIC inputs
2. Spatially interpolating CASCADE climate data into 16 locations and get every location's daily precipitation, maximum temperature, minimum temperature from 2001 to 2100 for each scenario (A2 and B1)
3. Linear temporal interpolation to generate 20Y data for each leap year
4. Converting CASCADE data into the format that the VIC model requires
5. Running VIC model on a Meteorological Focusing Disaggregator to generate hourly precipitation, air temperature, shortwave, longwave, air pressure, vapor pressure for each of 16 locations
6. Partitioning shortwave radiation to diffuse and direct radiation, and calculating relative humidity...

Data Outputs
To BEND Model
The BEND Model requires MEUDRY variables of:
- 14 locations in continental US
- 2001 - 2100 100 years with leap years
## iRESM Feature – Reuse Matrix

<table>
<thead>
<tr>
<th>Feature</th>
<th>Solution</th>
<th>Notes</th>
<th>Reuse</th>
</tr>
</thead>
<tbody>
<tr>
<td>Collaboration</td>
<td>Velo</td>
<td>Core wiki features support this, includes support for discussions</td>
<td>100%</td>
</tr>
<tr>
<td>Sharing data</td>
<td>Velo</td>
<td>Requires integration of MW and Alfresco</td>
<td>100%</td>
</tr>
<tr>
<td>Metadata extraction</td>
<td>Velo</td>
<td>Requires creation of data-type specific extraction pipelines</td>
<td>50%</td>
</tr>
<tr>
<td>User-driven customization</td>
<td>Velo</td>
<td>Core wiki features support this</td>
<td>100%</td>
</tr>
<tr>
<td>Linking to model execution</td>
<td>Velo, MeDICi</td>
<td>Need to construct MeDICi pipelines and link to Velo</td>
<td>70%</td>
</tr>
<tr>
<td>Model versioning</td>
<td>Velo</td>
<td>Supported</td>
<td>100%</td>
</tr>
<tr>
<td>Data set versioning</td>
<td>Velo</td>
<td>Supported</td>
<td>100%</td>
</tr>
<tr>
<td>Role-based Security</td>
<td>Velo</td>
<td>Supported</td>
<td>100%</td>
</tr>
</tbody>
</table>
Some reflections

► Science is a complex domain
  ■ Understanding requirements
  ■ Diversity of software/data
  ■ Users who are pushing the boundaries
  ■ Scientists don’t (in general) understand complexity of software systems
  ■ Architectures, integration, testing
  ■ Different to implementing a set of equations

► Through deliberate, creative reuse and a strong focus on architecture, we’ve:
  ■ Built generically useful technologies at low cost)
  ■ They work ;)

33
Questions?