Cyber Dumpster Diving – creating new software systems for less

Ian Gorton,
R&D Manager,
Data Intensive Scientific Computing,
Computational Sciences and Math Division
Pacific Northwest National Lab
Pacific Northwest National Lab

- Department of Energy Science Lab
  - Fundamental sciences
  - National security
- 4500+ people
- Business volume of over $1bn per annum
- Large scale experimental facilities, e.g.
  - Environmental Molecular Sciences Lab (EMSL)
  - 161 Tflop supercomputer
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

High Performance Computing
The middle is a hard place …

► Requirements
- Need to understand science domain
- Need to understand HPC
- Difficult to define, constant refinement, negotiations, communications
- “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?
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
Carbon Sequestration (Storage)
Geological Sequestration Software Suite (GS3)

- Large-scale, complex data
  - Experimental
  - HPC Simulation inputs/outputs
  - Multiple realizations for uncertainty quantification

- Long-lived projects
  - Modeling
  - Analysis
  - Monitoring (100+ years)
A powerful, usually legal, source of information that isn't seriously defended because of social taboos.
‘Write-as-little-code-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
Velo –
Knowledge Management for Modeling and Simulation
Supporting Carbon Sequestration Modeling

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

- Small project, team ~1.75 people, 3 years
Cyber Dumpster Diving Process ;)
# 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 3rd party tools
GS3 Examples – Tool Plugins
GS3 Examples – Black box Tool Plugins
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 …
Velo - Flexible, Rigorous Scientific Knowledge Management

- 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

GS3
SimSeQ
ASCEM
FutureGen
Velo
Tools
Site Data
Model Data
Simulators
Visualization
Plume Calcs
Velo Architecture

Velo Knowledge Base

Velo synchronization process

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

Data Ingest Pipeline

Convert Markup Store

Semantic Wiki

Core Wiki

MediaWiki

CMS Integration

Tool Integration

Wiki Database
Core Database
Semantic Database

CMS
(Simulations, Models, Projects)
Some reflections

► Science is a complex domain
  ■ Requirements, funding models
  ■ 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 ;)

Proudly Operated by Battelle Since 1965
Questions?