Building Polargrid Portal using Gadgets and OpenSocial

Gerald Guo, Raminder Singh, Marlon Pierce
Pervasive Technology Institute at Indiana University
{zhguo, ramifnu, marpierc}@indiana.edu
Introduction to Polargrid

- An NSF-funded MRI project that provides computing support for the Center for the Remote Sensing of Ice Sheets (CReSIS, https://www.cresis.ku.edu/)
- CReSIS is primarily concerned with using Synthetic Aperture Radar (SAR) techniques to obtain information on the depth of the Greenland and Antarctic ice sheets and their underlying rock beds.
- Provides both in-the-field computing clusters for initial image processing and larger clusters at Indiana University for full-scale image processing.
- Image processing is needed to produce data products of multiple levels

Partners & Collaborators
- Jeff Woods at ECSU
- University of Kansas
- Ohio State University
- Pennsylvania State University
Portal Requirements

• View CReSIS data sets, run filters, and view results through Web map interfaces.
• See/Share user’s events in a Calendar.
• Update results to a common repository with appropriate access controls.
• Post the status of computational experiments.
• Support collaboration and information exchange by interfacing to blogs and discussion area.
Approaches

• Existing frameworks like Sakai, Moodle, Liferay, Exo and Drupal

• Building open system using OpenSocial Gadget and using Google services or social network services like Facebook, Twitter etc. for collaboration
## Technologies

<table>
<thead>
<tr>
<th>Tech/Design choices</th>
<th>Reason Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>Web 2.0</td>
<td>Improves usability and responsiveness</td>
</tr>
<tr>
<td>Gadget</td>
<td>Makes developers possible to write reusable web components that can be deployed to any Gadget container.</td>
</tr>
<tr>
<td>OpenSocial</td>
<td>Makes portal possible to interact with existing large social networks instead of building our own.</td>
</tr>
<tr>
<td>REST</td>
<td>Makes applications able to access PolarGrid services using simple HTTP requests.</td>
</tr>
<tr>
<td>OpenID</td>
<td>Makes portal able to interact with external OpenID-compliant identity management systems.</td>
</tr>
<tr>
<td>OAuth</td>
<td>Makes portal able to interact with external OAuth-protected services.</td>
</tr>
<tr>
<td>MyProxy</td>
<td>Makes portal able to interact with security infrastructure of Grid systems.</td>
</tr>
</tbody>
</table>
Polargrid Architecture
Process Flow

1. A user visits his/her gadget home page, which is served by OGCE gadget layout manager.

2. The gadget layout manager constructs the user’s custom gadget layout in browser and makes use of a gadget renderer (Shindig in our case) to render each gadget XML to HTML/JavaScript. Then the generated HTML/JavaScript code is displayed in browser.

3. Different gadgets may interact with different backend RESTful services to generate output. A JSON response is sent back to the gadget to display the results.

4. Gadgets and RESTful services also query social data using OpenSocial API’s by sending requests to Shindig server.
# Gadgets

<table>
<thead>
<tr>
<th>Gadgets</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Filter Gadget</td>
<td>User can select different parameters to run a filter to create image. Result image will be displayed on Google map.</td>
</tr>
<tr>
<td>Blog Gadget</td>
<td>To display the feeds from PolarGrid blog site.</td>
</tr>
<tr>
<td>Discussion Board</td>
<td>Google Friend Connect (GFC) gadget to discuss on certain topic.</td>
</tr>
<tr>
<td>Filter Images</td>
<td>Picasa gadget to display all the filter images with filter description.</td>
</tr>
<tr>
<td>FAQ Gadget</td>
<td>GFC gadget for Question/Answer. Moderator can always control the topics and can block people from the list.</td>
</tr>
<tr>
<td>Google Calendar</td>
<td>Calendar gadget to display public PolarGrid-specific activities and tasks.</td>
</tr>
<tr>
<td>Twitter Gadget</td>
<td>To read filter execution updates from twitter related to PolarGrid.</td>
</tr>
<tr>
<td>Facebook Gadget</td>
<td>User can update status of task directly to Facebook from here.</td>
</tr>
</tbody>
</table>
Backend services

• REST service
  a. To integrate gadget with GFAC service.
  b. Call GFAC Webservice and read the SOAP response to read result.
  c. Upload resultant image to Picasa with parameters information.
  d. Add this activity to the calendar under particulate calendar name.
  e. Publish this activity along with Picasa URL and Calendar event to Twitter
  f. Create JSON response for Gadget

• GFAC Service is to wrap the service request and establish communication with Teragrid resources to run the Matlab job.

• Matlab filter: Binary takes all the parameters and process the filters for required data set.
OGCE Gadget Portal

• Goal
  – Build an open, lightweight, flexible, easy-to-build, general portal

• Approach
  – Agile development: make use of existing and widely-accepted technologies and services.
    • Web 2.0, Gadget, OpenSocial, OpenID, OAuth
    • Google Calendar, Picasa, Blogspot, Twitter
    • TeraGrid
PolarGrid Architecture

- PolarGrid specific gadget and services
- Generic services

Diagram: PolarGrid Portal powered by OG CE Gadget Layout Manager
- Map Filter
- Google Calendar
- Facebook
- Twitter
- Picasa
- Google Friend Connect Discussion board

RESTful Services
- Secure communication
- OpenID/GFC/GSI

Handlers
- GFAC
- Picasa
- Google Calendar
- Twitter

Social Data (Shindig)/GFAC

Gadget Renderer (Container)/Shindig

Convert Gadget XML to HTML

PolarGrid specific gadget and services

Generic services
Web 2.0

“Second generation of web development and web design”

<table>
<thead>
<tr>
<th>Enterprise Approach</th>
<th>Web 2.0 Approach</th>
</tr>
</thead>
<tbody>
<tr>
<td>Portlets</td>
<td>Gadgets, Widgets</td>
</tr>
<tr>
<td>SOAP</td>
<td>RSS, Atom, JSON</td>
</tr>
<tr>
<td>WSDL</td>
<td>REST(GET, PUT, POST, DELETE)</td>
</tr>
<tr>
<td>Server side integration</td>
<td>Client-side integration (AJAX)</td>
</tr>
<tr>
<td>Monolithic Workflow managers</td>
<td>Mash-ups (e.g. Yahoo Pipes)</td>
</tr>
<tr>
<td>Gateways</td>
<td>User-centric social network portals</td>
</tr>
</tbody>
</table>
REST

• Uniform interfaces based on HTTP
• Resource-oriented (resource can be anything)
• Each resource is identified by a unique URL
• State transition (Link resources together)
• Resources have multiple representations (JSON,XML)
• Good for both browser-to-server and server-to-server interactions
• Alternative - SOAP-based WS
  – About 60 core ws-* protocols
  – Designed for server-server interactions
    • SOAP and WSDL are really complicated
    • Browser-based apps are second-class citizens.
OpenSocial

• A coherent open architecture designed for social network services and applications.

Components

• Interface: REST, Javascript APIs
• Client: Ajax, Gadget
• Message Format: JSON, XML
• Security: OAuth
• Data Model
OpenID

• Separation of Identity Provider and Relying Parties.

Identity Providers: Blogspot, Flickr, Yahoo!, AOL

Use the same OpenID to log in multiple sites.
Bind OpenID to local accounts.
After the binding, use OpenID for login.
Google Friend Connect

- Novel way to integrate Social Features
- Allow users to login with existing Google, Yahoo, AOL, OpenID accounts
OAuth

- Users’ data is served at service providers.
- Third party apps want to access users’ data.
- Users don’t need to relinquish username and password to third party apps.
• **Drawbacks**
  
  – V1.0 is vulnerable to session fixation attack ([http://oauth.net/advisories/2009-1](http://oauth.net/advisories/2009-1)). Fixed in v1.0a.
  
  – Delegation granularity (Service provider-specific)
    • Operations
    • Data
  
  – Access token management
    • Non-standard
    • Expiration (implicit timeout)
    • Revocation (explicitly revoke assigned privileges)
Tab layout
Tree Layout
OGCE Gadget Portal Features

- Two layouts
  - Tab
  - Tree
- Gadget group manipulation – add/remove
- Four built-in themes
- Gadget manipulation
  - Add/remove Gadget
  - Drag and Drop
  - Two gadget views: home and canvas
  - Gadget setting
- Session persistence
- Layout data
  - View, modify layout data
  - Easy to migrate when you do a new installation
  - Inspect
- On demand rendering
- Customization
Gadget Resource

• Reusability
• Google gadget directory contains about 180,000 gadgets.
• They can be deployed to OGCE gadget portal.
• Common gadgets
  – RSS Feed Reader, Calendar, Email, Task list
• Social gadgets
  – Twitter, Google Talk, Facebook, Youtube
Other Science Use Cases

• TeraGrid
  – MiniGpir, load monitoring, resource usage

• Open Life Science Gateway
  – Use OAuth to submit jobs and access resources

• Cyberaide
  – Interact with Grid using web interface
    • MyProxy authentication
    • Globus Job Submission
    • GridFTP file transfer
OGCE Software’s and Related links

**OGCE Layout Manager:** This project is to provide Open Social-compatible gadget layout container and gadget host server.
- Actively Developed by Gerald Guo, PHD student in Indiana University
- Gadget Develops need not to depend on igoogle or orkut
- Lot of new feature are added lately and several new in a queue.

**GFAC:** Wrap any command-line application as a Webservice.
- Developed for Lead Project and now generalized for any gateway.
- Rearchitected to work as Axis 2 service to leverage handler architecture and REST support.

Publications and related research
- [http://www.collab-ogce.org](http://www.collab-ogce.org)
- [http://collab-ogce.blogspot.com/](http://collab-ogce.blogspot.com/)
- [http://cglreport.zhenhua.info/](http://cglreport.zhenhua.info/)