uDig
User-friendly Desktop Internet GIS

Paul Ramsey
Jody Garnett
Jesse Eichar
• Victoria, BC, Canada
• Spatial Systems Consulting
• Open Source Software
• PostGIS
  uDig / GeoTools
Geospatial Architecture

- Spatial Database
  - Concurrency
  - Transactions
  - Seamlessness

- Internet Publishing
  - Feature Access
  - Map Access

- Data Manipulation
  - Direct Access
  - Editing
  - Cartography
ESRI Architecture

- Web Pages
- ArcMap
- ArcIMS
  - ImageServer
  - FeatureServer
- ArcSDE

MUM/EOGEO 2005
Open Source Architecture

- Web Pages
- uDig
- GeoServer
- MapServer
- PostGIS
• “Integrated Client”
• Ability to directly view WMS
• Ability to directly edit WFS
• Ability to search catalogues
• Ability to integrate standard GIS data
• Hides complexity of network access
Missing Link for Open Source

• “Standard GIS Functionality”
• Ability to directly edit GIS data
• Ability to connect to PostGIS, Mapserver, GeoServer
• Ability to create paper cartography
• Ability to integrate with proprietary infrastructures
• “User Friendly”
  - Sensible Defaults
    • Use Preferences for Complexity
  - Automatic Integration
    • Coordinate Reference Systems
    • Formats, Services
  - Drag and Drop Everywhere
  - Hide Differences
• “Desktop”
  – Not a Web Application
  – Desktop Look and Feel
    • Windows, Linux, Apple OS/X
  – Desktop Integration
    • Cut and Paste, External Drag and Drop
  – Desktop Installation
    • One Click Installers
“Internet”
- Consume Remote Data and Services
- View OGC Web Map Server
- View/Edit OGC Web Feature Server
- View ESRI ArcIMS Server *
- Multi-catalogue Search
- Treat Local and Internet Layers Equally

* Coming Soon
• “GIS”
  – Platform Extensibility
  – Platform “De-stensibility”
  – Hooks for Analytical Plugins
  – Data Editing
  – Standard GIS Data Sources
  – Paper Cartography
Whaddaya Got ?!? 

**uDig**

DEMO...

MUM/EOGEO 2005
Jody Garnett

- uDig Development Team Leader
- GeoTools Project Management Committee (PMC) Member
- GeoServer contributor
- Anachronistic pugilist
- Technology Platforms and Decisions We Made
Java 1.5

• Risks
  – Compatibility with other Java libraries that are 1.4 only
  – Danger of the bleeding edge

• Rewards
  – New language features
    • type narrowing
    • enumerations
  – More explicit API for uDig
  – Better Linux support for JAI under SWT
• Risks
  – Needed to add a **GML** parser
  – Had to coordinate a scheduled project with an unscheduled community

• Rewards
  – Larger base of developers
  – Rich API
  – Standards (OGC, ISO) compliant by design
• We wrote generic WMS client
  – WMS has 4 versions!
• We wrote generic WFS client
  – GML handling is hard!
• We wrote generic WFS client
  – GML handling is hard!

Our XML Schema Processing System

Parser

  Schema

  XML Data Objects (XDO)

JVM

MUM/EOGEO 2005
Eclipse RCP

• Risks
  – Bleeding edge when we started
  – Not Swing
  – SWT integration with core Java

• Rewards
  – Plug-in environment pre-defined
  – Documented platform
  – Attractive and extensible by design
Eclipse RCP is more than an application framework system, it is an **platform** that not only allows extension via plug-ins, but also organizes the plug-ins into a **rational structure**.
Eclipse EMF

• Risks
  – Eclipse Modelling Framework (EMF)
  – New technology
  – More complex API

• Rewards
  – Integrates modelling into programming environment
  – Allows rapid changes in application model
    • Weeks versus Hours
  – Drag-n-drop, events, persistence all built-in
Eclipse EMF

Edit -> Compile -> Debug

Model
Eclipse GEF

• Risks
  – Graphical Editing Framework (EMF)
  – New technology

• Rewards
  – Pre-built graphical manipulation toolkit
  – Future use in GIS process model builder
<table>
<thead>
<tr>
<th>Component</th>
<th>Product</th>
</tr>
</thead>
<tbody>
<tr>
<td>IDE</td>
<td>Eclipse</td>
</tr>
<tr>
<td>Build System</td>
<td>Eclipse / PDEBuild</td>
</tr>
<tr>
<td>Version Control</td>
<td>Subversion</td>
</tr>
<tr>
<td>Real Time Build</td>
<td>Cruise Control</td>
</tr>
<tr>
<td>Wiki / Documentation</td>
<td>Confluence</td>
</tr>
<tr>
<td>Bug Tracking</td>
<td>JIRA</td>
</tr>
<tr>
<td>Ooops!</td>
<td>Aaaahh!</td>
</tr>
<tr>
<td>----------------</td>
<td>---------------------------------</td>
</tr>
<tr>
<td>Catalog 2.0</td>
<td>CGDI / Google XML</td>
</tr>
<tr>
<td>Drupal</td>
<td>Confluence</td>
</tr>
<tr>
<td>Scope</td>
<td>Release early, release less</td>
</tr>
<tr>
<td>Java 2D Render</td>
<td>OpenGL Draw2D</td>
</tr>
</tbody>
</table>
How to Contribute

• Everyone
  – Join the udig-devel list

• Developers
  – Quickstart
  – Source Code

• Users (Future Developers)
  – Download
  – Test / Bug Report
  – Translate
  – Supply Crazy Ideas (Toolkits, 3D, GML3!)
Architecture of uDig

GIS Platform
- Platform
- Runtime
- Workbench
- Help
- Search

GIS Application
- Template
- Catalog
- Page
- Project
- Map
- Discovery
- Render
- Visualization

GIS Toolkits
- Geotools
- GeoAPI

Oracle Spatial
ArcSDE
Postgis
MySQL

Geographic Information
- shp
tiff
gml
-wms
-wfs
-wrs

-vpf
tiger
arc
grid
• Developed uDig Core Model
• Developed uDig Renderer
• GeoTools Committer
• Solver of Hard Problems
• Brazilian pugilist
• Example of Extending uDig
• A tool that returns the ground distance between the point the mouse is clicked and the point the mouse is released.
Plug-ins

• Basic building block of Eclipse and uDig
• Have dependencies on and provide facilities for other plug-ins
Extension Points

- Every plug-in uses extension points
- Some plug-ins **provide** extension points
- Extension points provide organization in a potentially chaotic situation:
  - Applications built entirely of different plug-in components
  - Each plug-in must know its role, and advertise it, so that others know too
  - Each plug-in must know what other plug-ins it depends on
  - Stay in your sandbox!
Extension Points

[Image of Eclipse Extension Point editor]

## Extension Points

### Extensions

**All Extensions**

- net.refractions.udig.project.ui.tool
  - Measurements (category)
    - distance (modalTool)
  - org.eclipse.ui.commands
    - Measurement Tools (command)
- org.eclipse.ui.bindings

### Extension Element Details

Set the properties of "modalTool"

- **id**: net.refractions.udig.distanceTool.distanceTool
- **tooltip**: Measure the surface distance between two points
- **class**: net.refractions.udig.distanceTool.Dist
- **icon**: Browse...
- **name**: distance
- **inMenu**: true
- **onToolbar**: true
- **categoryIds**: net.refractions.udig.measurements
- **commandIds**: Browse...
- **commandHandler**: Browse...
uDig Extension Points

- Resources
- Renderers / Styles
- Operations
- MapGraphics
- Tools
- Drag’n’Drop
- FeatureEditor
Distance Tool

- Will use the uDig **Tool** extension point.
- **Tool** extension point provides:
  - A Context (simple API to Map)
  - A Panel (the Map Editor UI)
- **Tools** must extend an AbstractTool:
  - Modal Tool ("info")
  - Action Tool ("zoom in")
  - Background Tool ("current coords")
Distance Tool

SimpleDistanceTool

Context
Map

Panel

Tool Extension Point
SimpleDistanceTool

• Small tool, so just one class with three methods:
  • mouseDown
    – Mark the button click
  • mouseUp
    – Mark the button release and calculate the distance
  • displayOnStatusBar
    – Get the calculated distance into the workbench status bar
package net.refractions.udig.distanceTool;

import org.eclipse.jface.action.IStatusLineManager;
import org.geotools.referencing.CRS;
import com.vividsolutions.jts.geom.Coordinate;
import net.refractions.udig.project.ui.tool.AbstractModalTool;
import net.refractions.udig.project.ui.tool.ModalTool;

public class DistanceTool extends AbstractModalTool implements ModalTool {
    public DistanceTool() {
        // Register for mouse events
        // Options are: MOUSE, MOTION, WHEEL
        super(MOUSE);
    }

    Coordinate start;
    . . .
}
public void mousePressed(MapMouseMouseEvent e) {
    start=getContext().pixelToWorld(e.x, e.y);
}
public void mouseReleased(MapMouseEvent e) {
    Coordinate end=getContext().pixelToWorld(e.x, e.y);
    try {
        double distance=CRS.distance(
            start, end, getContext().getCRS());
        displayOnStatusBar(distance);
    }
    catch (Exception e1) {
        displayError();
    }
}
private void displayError() {
    final IStatusLineManager statusBar = getContext().getStatusBar();
    if( statusBar==null )
        return; // shouldn't happen if the tool is being used.
    getContext().updateUI(new Runnable() {
        public void run() {
            statusBar.setErrorErrorMessage("Unable to calculate the distance");
        }
    });
}
private void displayOnStatusBar(double distance) {
    final IStatusBarManager statusBar =
        getContext().getStatusBar();
    if (statusBar == null) return;
    int totalmeters = (int) distance;
    final int km = totalmeters / 1000;
    final int meters = totalmeters - (km * 1000);
    float cm = (float) (distance - totalmeters) * 10000;
    cm = Math.round(cm);
    final float finalcm = cm / 100;
    getContext().updateUI(new Runnable() {
        public void run() {
            statusBar.setMessage("Distance = "+km+","+meters+"m "+finalcm+"cm");
        }
    });
}
Demonstration

Questions?