Building Web Applications with Protégé

Csongor Nyulas, Tania Tudorache
Stanford University

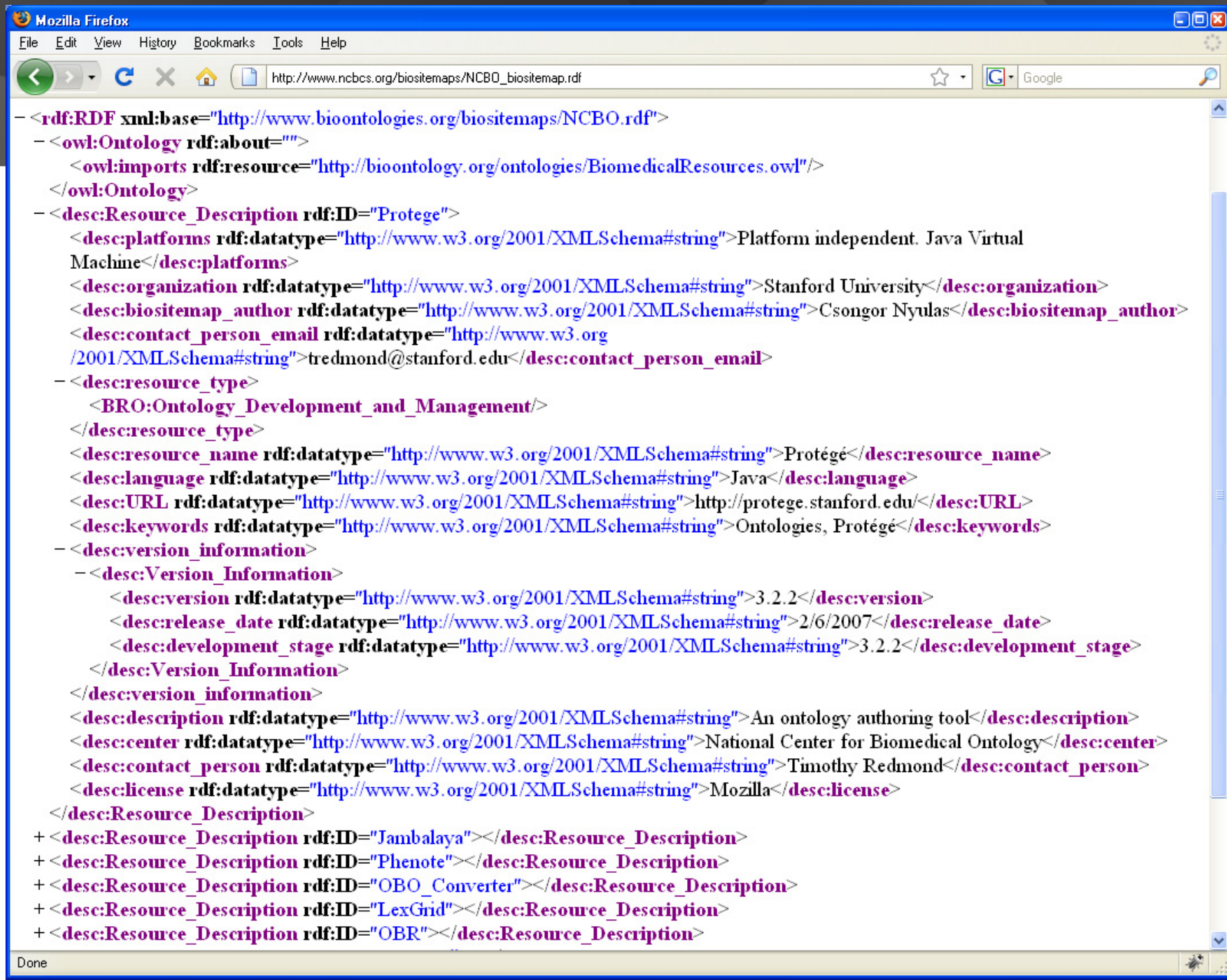
11th Protégé Conference, Amsterdam, The Netherlands
July 23-27, 2009

Web Applications Are Flourishing

- Everything is going web!
- Software applications as well; especially when:
  - they solve relatively simple tasks,
  - involve access to multiple resources from the web,
  - require easy accessibility, or
  - involve user collaboration
- Web apps are a great way to bring the power of Protégé to the every-day computer user

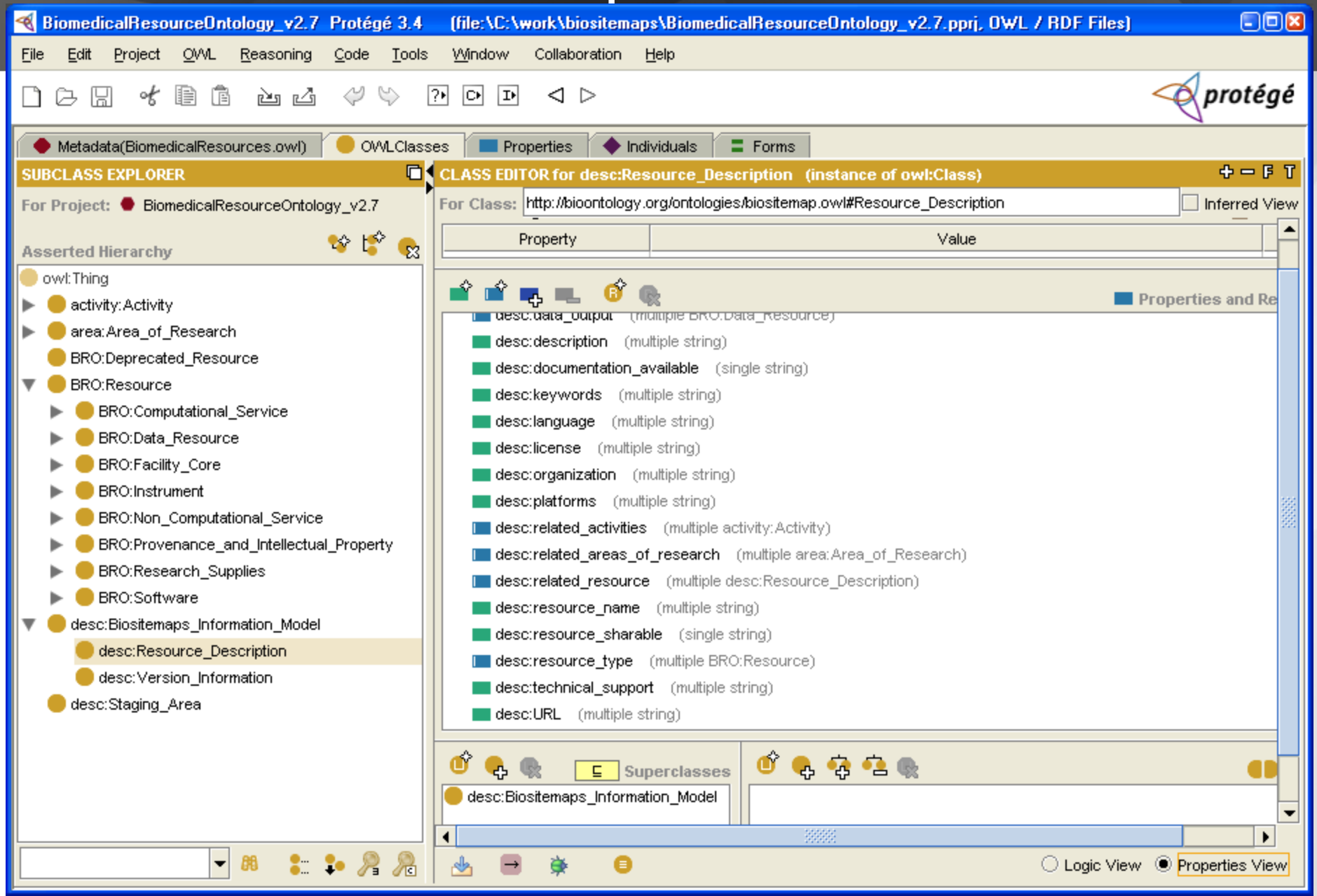
Biositemaps

- An NIH Roadmap Initiative for National Centers of Biomedical Computing (NCBCs)
- Goal of the project: To publish, locate, query, compose/combine, and mine biomedical resources on the Internet
- Inspired by Google's "sitemap protocol"
- Biositemap = RDF file conforming to the Biositemap Information Model (IM) and Biomedical Resource Ontology (BRO)

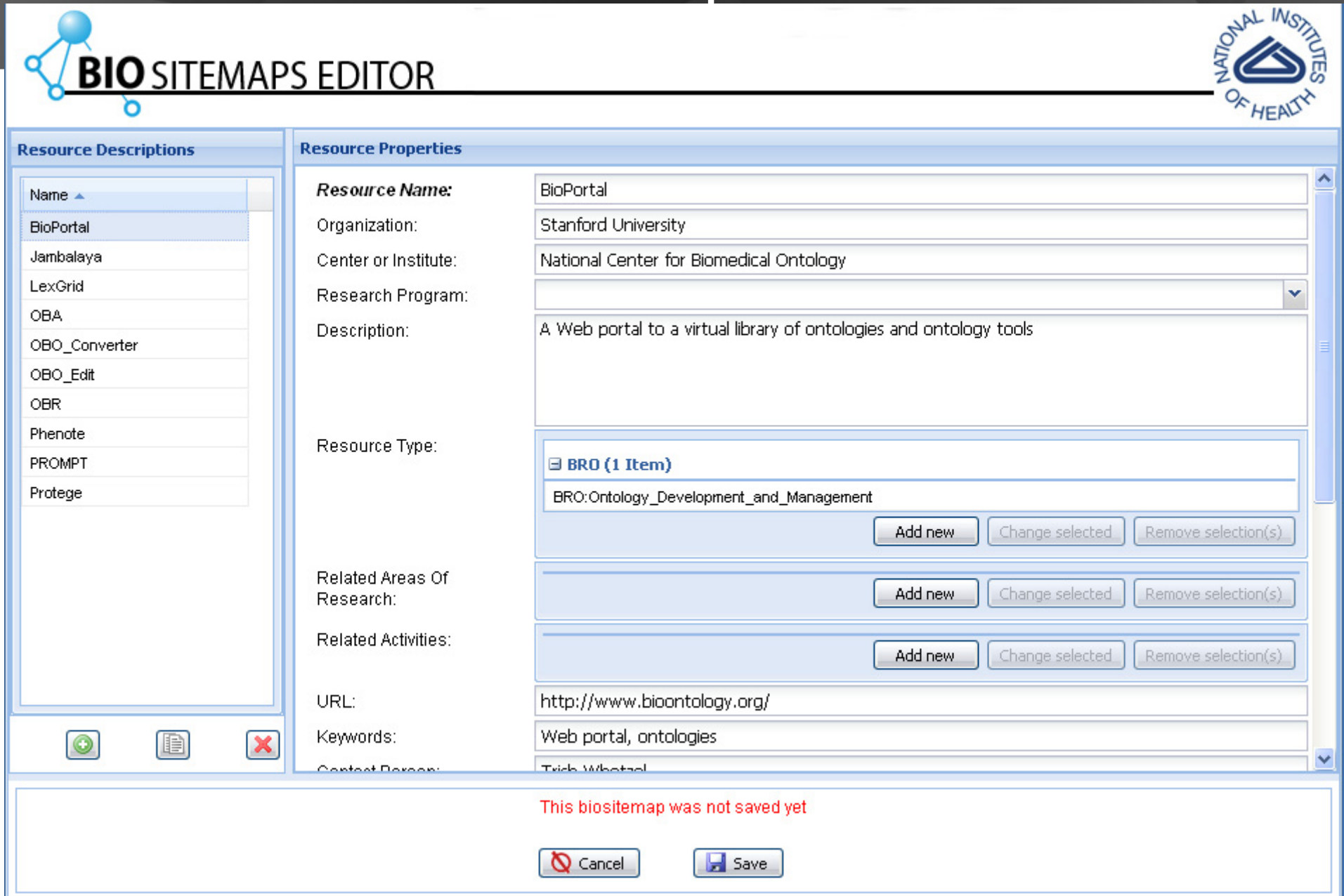



```
- <rdf:RDF xml:base="http://www.bioontologies.org/biositemaps/NCBO.rdf">
  - <owl:Ontology rdf:about="">
      <owl:imports rdf:resource="http://bioontology.org/ontologies/BiomedicalResources.owl"/>
    </owl:Ontology>
  - <desc:Resource_Description rdf:ID="Protege">
      <desc:platforms rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Platform independent. Java Virtual
      Machine</desc:platforms>
      <desc:organization rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Stanford University</desc:organization>
      <desc:biositemap_author rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Csongor Nyulas</desc:biositemap_author>
      <desc:contact_person_email rdf:datatype="http://www.w3.org
      /2001/XMLSchema#string">tredmond@stanford.edu</desc:contact_person_email>
    - <desc:resource_type>
        <BRO:Ontology_Development_and_Management/>
      </desc:resource_type>
      <desc:resource_name rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Protégé</desc:resource_name>
      <desc:language rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Java</desc:language>
      <desc:URL rdf:datatype="http://www.w3.org/2001/XMLSchema#string">http://protege.stanford.edu/</desc:URL>
      <desc:keywords rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Ontologies, Protégé</desc:keywords>
    - <desc:version_information>
      - <desc:Version_Information>
          <desc:version rdf:datatype="http://www.w3.org/2001/XMLSchema#string">3.2.2</desc:version>
          <desc:release_date rdf:datatype="http://www.w3.org/2001/XMLSchema#string">2/6/2007</desc:release_date>
          <desc:development_stage rdf:datatype="http://www.w3.org/2001/XMLSchema#string">3.2.2</desc:development_stage>
        </desc:Version_Information>
      </desc:version_information>
      <desc:description rdf:datatype="http://www.w3.org/2001/XMLSchema#string">An ontology authoring tool</desc:description>
      <desc:center rdf:datatype="http://www.w3.org/2001/XMLSchema#string">National Center for Biomedical Ontology</desc:center>
      <desc:contact_person rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Timothy Redmond</desc:contact_person>
      <desc:license rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Mozilla</desc:license>
    </desc:Resource_Description>
  + <desc:Resource_Description rdf:ID="Jambalaya"></desc:Resource_Description>
  + <desc:Resource_Description rdf:ID="Phenote"></desc:Resource_Description>
  + <desc:Resource_Description rdf:ID="OBO_Converter"></desc:Resource_Description>
  + <desc:Resource_Description rdf:ID="LexGrid"></desc:Resource_Description>
  + <desc:Resource_Description rdf:ID="OBR"></desc:Resource_Description>
```

The Biositemap IM and the BRO



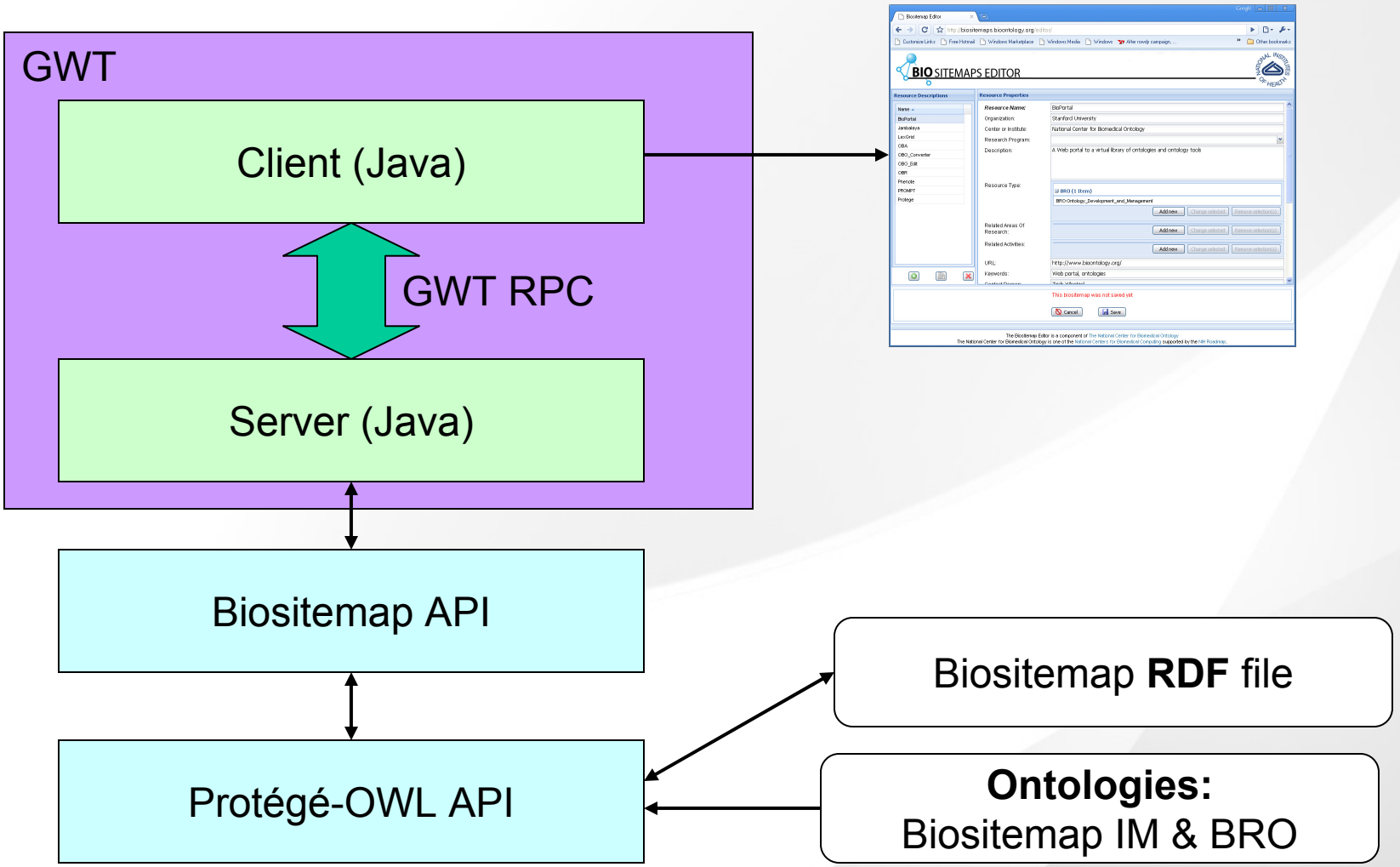
Biositemap Editor



The Google Web Toolkit (GWT)

- http://code.google.com/webtoolkit/
- Write AJAX front-end in Java that GWT compiles into optimized, cross-browser compatible JavaScript
  - The changes made in the Java files can be seen immediately in the web browser, without recompiling.
  - Developers can step through live AJAX code with the Java debugger.
  - GWT compiles and deploys strongly optimized, cross-browser JavaScript.
  - In addition to supporting an open ended set of transfer protocols, GWT also offers a simple, but efficient and sophisticated client-server communication with GWT RPC.
  - Supports efficient application localization, and optimized JavaScript script downloads based on user profile.
  - UI component can be easily reused across projects.
  - Allows the usage of other JavaScript libraries and native JavaScript code.
  - Easily supports the browser back button and history.
  - Works with the Java development tool of choice, incl. testing with JUnit.

System Architecture: Overview



Client-Server Communication

- Data Structures (encoded in package org.bioontology.biositemaps.editor.client.rpc.data) for:
  - File handling (FileHandlingData)
  - Property-values pair (BiositemapPropertyValuesData)
  - Resource description (BiositemapElementData)
  - Class hierarchy (BROClassHierarchyData)
  - Field layout (FieldLayoutData)
  - Form layout (FormLayoutData)
  - A full form configuration (FormConfigurationData)

Client-Server Communication - continuation -

- Define Service interfaces to be implemented on the server side, and …

```
@RemoteServiceRelativePath("biositemap-service")
public interface BiositemapService extends RemoteService {
    public FileHandlingData createBioSitemap(FileHandlingData data) throws BiositemapFormsException;
    public FileHandlingData openBioSitemap(FileHandlingData data) throws BiositemapFormsException;
    public FileHandlingData saveBioSitemap(FileHandlingData data) throws BiositemapFormsException;
    public FileHandlingData closeBioSitemap(int clientId) throws BiositemapFormsException;

    public FileHandlingData ping(FileHandlingData data) throws BiositemapFormsException;

    public FormConfigurationData getFormConfiguration(String layoutConf, boolean isEditable) throws BiositemapFormsEx

    public BROClassHierarchyData getBROClassHierarchy(String layoutConf, String className) throws BiositemapFormsExce

    public BiositemapElementData[] getAllTopLevelElements(int clientID) throws BiositemapFormsException;
    public BiositemapElementData[] getElementsOfType(int clientID, String className) throws BiositemapFormsException;

    public BiositemapElementData createAnonymousElementOfType(int clientID, String className) throws BiositemapFormsE
    public BiositemapElementData renameElementOfType(int clientID, String className, String oldName, String newName)
    public BiositemapElementData updateElementOfType(int clientID, String className, String name, BiositemapElementDa
    public BiositemapElementData removeElementOfType(int clientID, String className, String name) throws BiositemapFo
}
```

Client-Server Communication - continuation -

- … and asynchronous service handlers to be called on the client.

```
public interface BiositemapServiceAsync {
    public void createBioSitemap(FileHandlingData data, AsyncCallback<FileHandlingData> cb);
    public void openBioSitemap(FileHandlingData data, AsyncCallback<FileHandlingData> cb);
    public void saveBioSitemap(FileHandlingData data, AsyncCallback<FileHandlingData> cb);
    public void closeBioSitemap(int clientId, AsyncCallback<FileHandlingData> cb);

    public void ping(FileHandlingData data, AsyncCallback<FileHandlingData> cb);

    public void getFormConfiguration(String layoutConf, boolean isEditable, AsyncCallback<FormConfigurationData> cb);

    public void getBROClassHierarchy(String layoutConf, String className, AsyncCallback<BROClassHierarchyData> cb);

    public void getAllTopLevelElements(int clientID, AsyncCallback<BiositemapElementData[]> cb);
    public void getElementsOfType(int clientID, String className, AsyncCallback<BiositemapElementData[]> cb);

    public void createAnonymousElementOfType(int clientID, String className, AsyncCallback<BiositemapElementData> cb)
    public void renameElementOfType(int clientID, String className, String oldName, String newName, AsyncCallback<Bio
    public void updateElementOfType(int clientID, String className, String name, BiositemapElementData newValue, Asyn
    public void removeElementOfType(int clientID, String className, String name, AsyncCallback<BiositemapElementData>
}
```

The Server Side

- Use API calls to operate on ontology content
- Implement any complex operation on the server
- Maintain Pool of OWL models to serve multiple clients
- Implement server-side caching
- Implement time-out mechanism

The Client Side

- Use appropriate client-side widget libraries: “standard” GWT widget library, 3rd party widget libraries (GWT-Ext, getting replaced by SmartGWT; Ext GWT; etc.) or even native JavaScript
- Implement client-side caching if appropriate
- Make client configurable if possible

Lessons learned

- Keep it simple!
- Send to the client only what it really needs
- Use as much as possible the GWT features:
  - Modules, internationalization, RPC, internal viewer, debugger, etc.
- Split your services in logical modules
- Theoretically, it is only Java, but…some JS knowledge never hurts

Summary

- GWT offers a convenient way to quickly build web application to create lightweight, easily accessible Protégé “interfaces”
- The Biositemap Editor is accessible at: http://biositemaps.bioontology.org/editor/
- Biositemap Browser: http://biositemaps.bioontology.org/browser/
- Open source project. Source code available at: https://bmirgforge.stanford.edu/gf/project/biositemaps/scmsvn/?path=Biositemap Editor

Thank You!