pycsw: an OGC CSW server implementation written in Python

The pycsw Development Team

Overview

pycsw is an OGC CSW server implementation written in Python.

pycsw implements clause 10 (HTTP protocol binding (Catalogue Services for the Web, CSW)) of the OpenGIS Catalogue Service Implementation Specification, version 2.0.2. pycsw allows for metadata publishing either from its built-in data model, or through configuration. In the configuration mode the user can bind to an existing metadata model.

pycsw is Open Source, released under an MIT license, and runs on all major platforms (Windows, Linux, Mac OS X).

History

pycsw is a young project. Initial development began in 2010, with the main focus being to provide a very lightweight Python CSW server solution (in comparison to many Java-based CSW servers).

Another goal was to provide a standalone CSW server implementation. This means that metadata is created and managed and updated elsewhere, and pycsw thus acts as the publishing component for geospatial resource discovery.

Another focus was extensibility: OGC Catalogue Service, by design, allows for the definition of application profiles to support additional metadata formats (the core metadata model is Dublin Core + ows;BoundingBox), so this was an important design decision prior to initial implementation. Features (such as metadata formats, encodings, and harvesting) are implemented such that they can be extended to provide additional variations of a given feature.

The OGC CITE tests were extensively used to establish a benchmark of compliance to the specification (pycsw is not an OGC approved compliant product, but does pass all the CITE tests).

Version 1.0.0 was released in 2011, providing initial support for basic CSW operations, ISO Application Profile and INSPIRE Discovery Services support. SQLite and PostgreSQL were the initially supported databases.

In 2012, version 1.2.0 was released, adding additional search interfaces (SRU implementation, OpenSearch). MySQL database support was also added in this version, providing further flexibility for deployment. WMS harvesting and GeoNode support were also implemented, as well as the ability to provide JSON output of search results.

In September 2012, version 1.4.0 was released. This release brought many important features, including WSGI support. The WSGI development enabled pycsw to be integrated into existing Python frameworks, such as Django or Flask.

The first formal deployment of pycsw was by INSIDE Idaho. INSIDE Idaho is the official geospatial data clearinghouse for the State of Idaho. INSIDE Idaho serves as a comprehensive geospatial data digital library, providing access to, and a context within which to use, geospatial data and information by, for, and about Idaho.

What makes this deployment interesting is that pycsw, developed in a Linux / Apache environment, was deployed in a Windows / IIS environment.

Design

From its inception, pycsw strived to be lightweight, flexible, easy to install and deploy. A typical install takes less than 10 minutes.

Simple Configuration

Configuration is governed by a simple configuration file using the familiar Windows INI syntax. Users simply edit this file as required and changes are reflected in server behavior. Python supports this format natively with its ConfigParser standard library. In addition, those with customized applications can generate their own ConfigParser object (e.g. from an external configuration, Python dictionary or database) and send to pycsw in the same fashion. XML configuration files were considered early on in development, but it was decided for performance reasons to stick with a simple, plain text format.

Repository

Metadata is handled by way of a "Repository", which is defined as a physical database instance along with advertised queryables. CSW requires the advertisement of specific queryables, and advertises these via mappings to their underlying columns and properties in the database.

Dispatcher

All requests are performed via HTTP GET or HTTP POST. The dispatcher is vital in deciphering what the interface of the request is (CSW, SRU, OpenSearch, etc.), and responds with the appropriate headers and payload.

Plugin Architecture

A plugin architecture is provided for developers to extend the codebase in order to support additional application profiles, formats, and repositories.

No XSLT

To avoid the overhead of XSLT processing, pycsw was designed to handle metadata by processing XML elements into a relational model (database). Thus there is almost no transformation of XML within the codebase. Metadata XML is generated from the database.

Features

As of current writing (October 2012), pycsw implements the following features:

- Fully implements OGC CSW 2.0.2
- Fully passes the OGC CITE CSW test suite (103/103)
- Implements INSPIRE Discovery Services 3.0
- Implements ISO Metadata Application Profile 1.0.0
- Implements FGDC CSDGM Application Profile for CSW 2.0
- Implements the Search/Retrieval via URL (SRU) search protocol
- Implements OpenSearch
- Supports ISO, Dublin Core, DIF, FGDC and Atom metadata models
- CGI or WSGI deployment
- Simple configuration
- Transactional capabilities (CSW-T)
- Flexible repository configuration
- GeoNode connectivity
- Open Data Catalog connectivity
- Federated catalogue distributed searching
- Realtime XML Schema validation
- Extensible profile plugin architecture

Technology

pycsw is written in Python and leverages the following technologies:

lxml is used for XML request parsing and validation, as well as serializing XML responses. lxml is a cornerstone technology used in the codebase. lxml is the Python binding to the libxml2 C library.

Shapely is used for handling spatial predicates in an independent manner. It was decided to use Shapely to be able to deal with geometries agnostic to a given database environment. This allows pycsw to bind to any database. OGC Well Known Text (WKT) is used as the internal geometry format. EPSG:4326 is used as the internal coordinate reference system. Shapely is the Python binding to the GEOS C library.

SQLAlchemy is used as the database abstraction layer and provides a Pythonic approach to working with databases (as opposed to raw SQL scripting).

OWSLib handles the heavy lifting of parsing XML formats and interacting with OGC Web Services for harvesting.

pyproj is used to handle coordinate transformations, e.g. for CSW requests which provide non-geographic coordinates. pyproj provides Python bindings to the proj.4 C library.

INSPIRE Support

Early versions of pycsw supported only core CSW 2.0.2 and were able to complete OGC CITE tests with a 100% success rate. There was an initial goal of the project to be able to work with plugins, in a way that would make it very extendable and easy to configure. For example, the end user should be able to install or uninstall a plugin by adding or removing a folder within the plugins directory.

The first metadata profile selected to be implemented was the APISO profile of CSW 2.0.2 since it was widely used in current CSW implementations. This profile was ready and stable for version 1.0.0 of pycsw.

Around that time, there was a growing interest in Europe about Catalogue Service implementations that would support the draft guidelines of the INSPIRE Discovery Service specification. This specification was not final at that point. However, through the INSPIRE Directive there was a final decision about the specification of metadata. This specification supported ISO 19115 and 19136 with some additions and profile modifications.

Just before pycsw 1.0.0 was released, Version 3.0 of the INSPIRE Discovery Service specification was the first to leave beta and become official. At this point pycsw supported multilingual metadata, as well as additional service metadata needed to comply with the directive. This was implemented within the APISO profile and not in a separate profile (INSPIRE is heavily based on APISO and would make a new profile highly redundant). The APISO plugin was configured to have INSPIRE support by setting a configuration switch. The user would then provide the additional metadata needed by the INSPIRE specification. The final specification for Discovery Service brought some changes and a new XML namespace to follow. This was added and fully supported in pycsw 1.0.0.

The pycsw database model supported the mandatory fields of INSPIRE for APISO and search/read/write/update procedures. This support was enabled by enhancements made to the underlying OWSLib ISO 19139 implementation. Those patches were contributed back to OWSLib.

Integration with Python Frameworks

pycsw has the ability to operate in 'library' mode by external applications. It is also possible to use pycsw within your application through pure Python request and response (no HTTP) mechanisms. This allows for easy integration and inclusion of a CSW into an existing website or application and information model.

GeoNode

GeoNode is a platform for the management and publication of geospatial data. It brings together mature and stable open-source software projects under a consistent and easy-to-use interface allowing users, with little training, to quickly and easily share data and create interactive maps. GeoNode provides a cost-effective and scalable tool for developing information management systems.

As part of the GeoNode 2.0 roadmap, development was undertaken to abstract CSW functionality to any CSW (pycsw, GeoNetwork OpenSource, deegree), which allows flexibility in cataloguing in GeoNode. In addition, GeoNode was updated to deploy pycsw inline as a WSGI application and working directly off the GeoNode database (Django models), thus reducing redundancy in metadata management and software deployment footprint. pycsw is the default CSW server for GeoNode 2.0.

Open Data Catalog

Open Data Catalog is an open data catalog based on Django, Python and PostgreSQL. It was originally developed for http://OpenDataPhilly.org,a portal that provides access to open data sets, applications, and APIs related to the Philadelphia region. The Open Data Catalog is a generalized version of the original source code with a simple skin. It is intended to display information and links to publicly available data in an easily searchable format. The code also includes options for data owners to submit data for consideration and for registered public users to nominate a type of data they would like to see openly available to the public.

In the same spirit as GeoNode, pycsw was chosen as the CSW server for Open Data Catalogue. Given the ODC's desire to have an interoperable CSW as part of the project, and pycsw's ability to integrate with Django, pycsw was a natural fit for the CSW requirements of the project. As in the GeoNode scenario, pycsw uses ODC's database directly via Django models.

OSGeo Involvement

OSGeo Live DVD

pycsw is available on the OSGeo-Live project. The overview and quickstart provides further information on using pycsw in OSGeo-Live.

Labs/Incubation

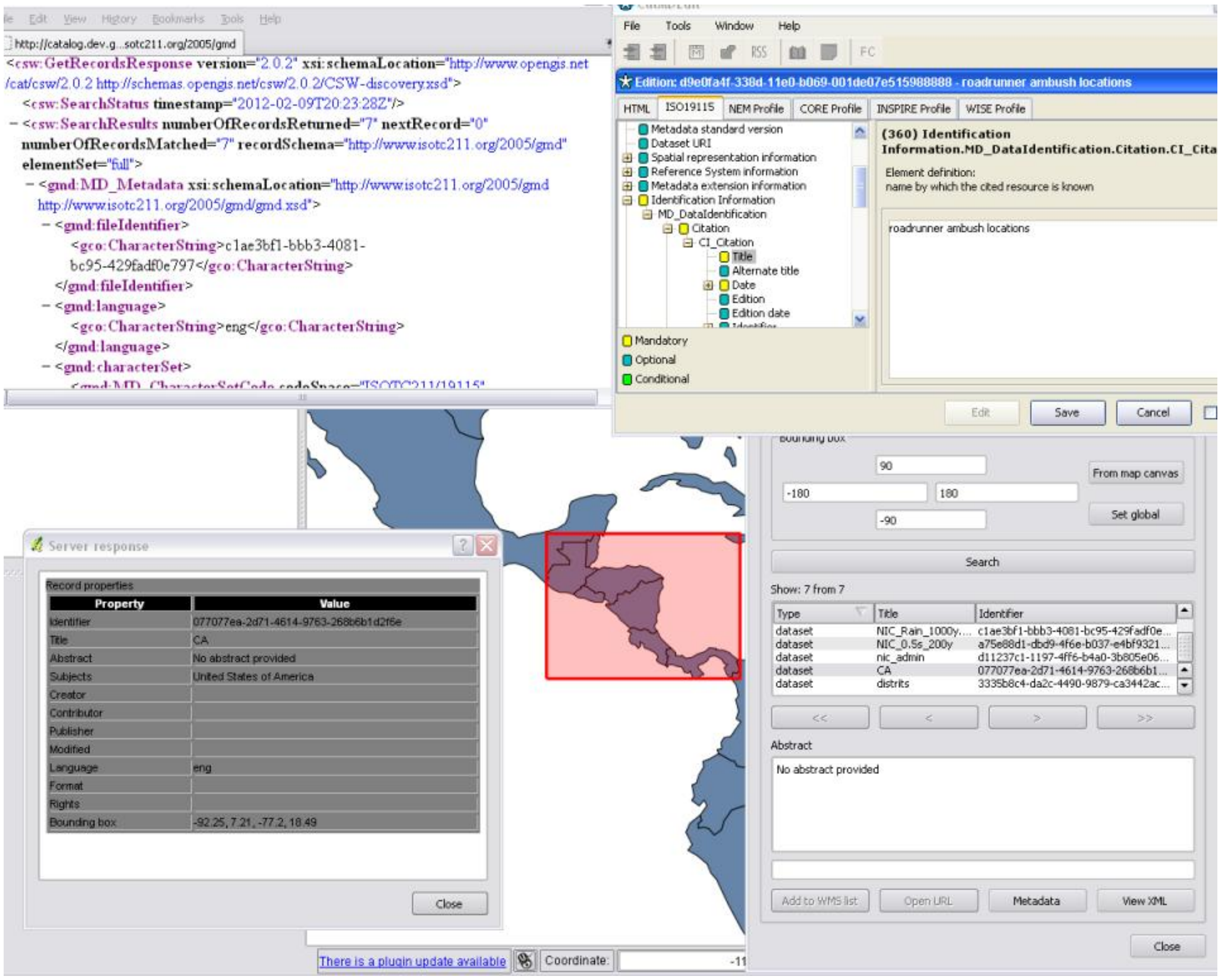
The project is currently in OSGeo Labs and leverages various OSGeo infrastructure, with the goal of reaching incubation status and becoming an approved OSGeo project.

Future Development

A few areas of interest for future development include:

- Spatial database and geometry object support (PostGIS, MySQL spatial): currently pycsw works from a WKT string column and Shapely for spatial operations. Adding support for true geometric objects would enable using direct spatial support. In addition, for non-spatial databases, WKB is being considered as a means to improve performance.

- Search engine support: Currently pycsw works in a similar manner as an OGC Web Feature Service (WFS).



File Edit View History Bookmarks Tools Help
http://catalog.dev.g...sotc211.org/2005/gmd
File Tools Window Help
Edition: d9e0fa4f-338d-11e0-b069-001de07e515988888 - roadrunner ambush locations
HTML ISO19115 NEM Profile CORE Profile INSPIRE Profile WISE Profile
Metadata standard version
Dataset URI
Spatial representation information
Reference System information
Metadata extension information
Identification Information
MD_DataIdentification
Citation
CI_Citation
Title
Alternate title
Date
Edition
Edition date
Mandatory
Optional
Conditional
Edit Save Cancel
From map canvas
Set global
Search
<< < > >>
Add to WMS list Open URL Metadata View XML
Close
There is a plugin update available
Coordinate:

```
<csw:GetRecordsResponse version="2.0.2" xsi:schemaLocation="http://www.opengis.net
/cat/csw/2.0.2 http://schemas.opengis.net/csw/2.0.2/CSW-discovery.xsd">
  <csw:SearchStatus timestamp="2012-02-09T20:23:28Z"/>
 - <csw:SearchResults numberOfRecordsReturned="7" nextRecord="0"
   numberOfRecordsMatched="7" recordSchema="http://www.isotc211.org/2005/gmd"
   elementSet="full">
   - <gmd:MD_Metadata xsi:schemaLocation="http://www.isotc211.org/2005/gmd
     http://www.isotc211.org/2005/gmd/gmd.xsd">
     - <gmd:fileIdentifier>
         <gco:CharacterString>c1ae3bf1-bbb3-4081-
         bc95-429fadf0e797</gco:CharacterString>
       </gmd:fileIdentifier>
     - <gmd:language>
         <gco:CharacterString>eng</gco:CharacterString>
       </gmd:language>
     - <gmd:characterSet>
```

(360) Identification Information.MD_DataIdentification.Citation.CI_Cita

Element definition:
name by which the cited resource is known

roadrunner ambush locations

Bounding box

90

-180 180

-90

Server response

Record properties

Property	Value
Identifier	077077ea-2d71-4614-9763-268b6b1d2f6e
Title	CA
Abstract	No abstract provided
Subjects	United States of America
Creator	
Contributor	
Publisher	
Modified	
Language	eng
Format	
Rights	
Bounding box	-92.25, 7.21, -77.2, 18.49

Show: 7 from 7

Type	Title	Identifier
dataset	NIC_Rain_1000y....	c1ae3bf1-bbb3-4081-bc95-429fadf0e...
dataset	NIC_0.5s_200y	a75e88d1-dbd9-4f6e-b037-e4bf9321...
dataset	nic_admin	d11237c1-1197-4ff6-b4a0-3b805e06...
dataset	CA	077077ea-2d71-4614-9763-268b6b1...
dataset	distrits	3335b8c4-da2c-4490-9879-ca3442ac...

Abstract

No abstract provided

pycsw Screenshot

Adding support for true search engine libraries would enable full text indexing and search result relevance.

- Web administration/metadata editing: Providing a front end administration tool would allow for more user-friendly interaction to load metadata, edit server settings, etc.

- OGC Catalogue 3.0: Future versions of the OGC Catalogue Service Specification will require pycsw to support multiple versions (the codebase is currently bound to 2.0.2).

Some of the abovementioned features will come at the cost of ease of installment and deployment, and will be developed as optional, configurable components.

Conclusion

pycsw is a young, lightweight, flexible and fast CSW server implementation. Already the project is being used as a standalone service and in various applications and will hopefully expand in usage over time.

Community involvement is welcome from users and developers. Please feel free to visit http://pycsw.org.

References

D. Nebert, A. Whiteside, P. Vretanos (2007)
Open Geospatial Consortium - Catalogue Service for Web.
OGC 07-006r1 pp.204.

U. Voges, K. Senkler (2007)
Open Geospatial Consortium Inc. ISO Metadata Application Profile
OGC 07-045 pp.125.

Directive 2007/2/EC of the European Parliament and of the Council of 14 March 2007
http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=CELEX:32007L0002:EN:NOT

Commission Regulation (EC) No 1205/2008
http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=CELEX:32008R1205:EN:NOT

Technical Guidance for INSPIRE Discovery Services
http://inspire.jrc.ec.europa.eu/documents/Network_Services/TechnicalGuidance_DiscoveryServices_v3.1.pdf

Tom Kralidis
http://kralidis.ca/
tomkralidis AT hotmail.com

Angelos Tzotsos
National Technical University of Athens
http://users.ntua.gr/tzotsos/
tzotsos AT gmail.com

Adam Hinz
Azavea
https://github.com/ahinz
hinz.adam AT gmail.com