ZOO Project : Open WPS platform

www.zoo-project.org

Mr. Gérald FENOY, GeoLabs
Dr. Nicolas BOZON, 3LIZ
Pr. Venkatesh RAGHAVAN, OCU

Presentation outline

ZOO Project general presentation:

- General goals of the ZOO Project (WPS & FOSS4G)
- Story and status of the ZOO Project (Dev & release)
- ZOO Project community (ZOO Tribe & PSC)

ZOO Project technical presentation:

- ZOO Kernel (WPS compliant C Kernel)
- ZOO Services (Sample FOSS4G-based webservices)
- ZOO API (Server-side Javascript API)

ZOO Development plans, demos and discussion

www.zoo-project.org
What is ZOO?

ZOO is a **WPS (Web Processing Service)** compliant server-side platform based on OGC's WPS 1.0.0

ZOO is **open source software** released under MIT/X-11 licence

ZOO is designed to create and chain web processing services easily, using **FOSS4G libs or existing code**.

ZOO is based on a C Kernel (**ZOO Kernel**) able to load dynamic libraries and to orchestrate Webservices coded in several programming languages.

www.zoo-project.org
ZOO Project goals

- Make open source libs communicate in a standardized way using WPS
- Make Webservices deployment and chaining easier and faster
- Create a ready-to-use Webservices suite based on stable open source libs
- Enable advanced Web GIS functionalities such as:
  - Conversion, reprojection and processing of data
  - Advanced GIS algorithms (spatial and raster operations)
  - External spatial-based scientific models

www.zoo-project.org
ZOO Project story

- ZOO Project was founded by Gerald FENOY (GeoLabs), Venkatesh RAGHAVAN (Osaka City University) and Nicolas BOZON (3LIZ) at FOSS4G 2008 in Cape Town, South Africa.

- Conception and development by GeoLabs and 3LIZ since nov. 2008

- ZOO presentation at FOSS4G 2009 in Sydney, Australia.

- ZOO 1.0 released under MIT/X11 on April 13th 2010

The story will continue in Barcelona, Spain...

You are invited to the ZOO Workshop at FOSS4G2010
ZOO Project status (1)

- ZOO 1.0 release under MIT/X-11 license:
  
  ZOO Kernel ( C )
  ZOO Services ( C and Python)
  ZOO API ( Javascript )

svn checkout http://svn.zoo-project.org/svn/trunk zoo

- ZOO social media

http://www.zoo-project.org
zoo-discuss@gisws.media.osaka-cu.ac.jp
#zoo_project@irc.freenode.net
http://www.twitter.com/ZOO_Project
http://www.linkedin.com/groups?home=&gid=2532284
ZOO Project status (2)

- **ZOO Kernel**: zcfg/yaml support, error management, win32 support

- **ZOO Services**: GDAL (*translate*, *grid*), OGR (*ogr2ogr*), OO, QR, … Tests on GRASS, R, CGAL...

- **ZOO API**: WPS orchestration and chaining with server-side JS API (*ZOO.Request*, *ZOO.Process* …)

- Effort needed on ZOO packaging and documentation
  http://www.zoo-project.org/trac

- Effort needed on ZOO tutorials, HelloWorlds and examples
ZOO Project community (aka ZOO tribe)

Sponsors

Knowledge partners

Welcome to the ZOO Tribe!
ZOO Project PSC  (Aka ZOO Tribal Council)

Géraud FENOY (GeoLabs), FR
Venkatesh RAGAHAVAN (OCU), JP
Nicolas BOZON (3LIZ), FR
Jeff McKenna (Gateway geomatics), CA
Hirofumi HAYASHI (AppTech), JP
Markus Neteler (Centro di Ecologia Alpina), IT
Massimiliano Cannata (Supsi), CH
Daniel Kastl (GeoRepublik), DE & JP
Satoshi SEKIGUCHI (AIST), JP
ZOO Kernel handles and chains ZOO Services

A **ZOO service** is composed of:

A metadata file **.zcfg** (Title, Metadata, Inputs, Output...)

A Service Provider: **« Service Shared Object »** (SSO)
(Dynamic library, Python modules, JAVA Class, PHP script ...)

WPS **GetCapabilities** et **DescribeProcess** requests are solved by parsing .zcfg file using Flex and Bison

ZOO Kernel is able to load SSO dynamically, to extract specific functions and to execute them, answering the **WPS Execute** requests

How does ZOO work? (1)
How does ZOO work? (2)

- **Execute request** parsing (XML / KVP) to fill internal data structure

- **Xlink:href checking**, conditional data download to provide input data value

- **Dynamic loading** of the Service Provider Shared Objects

- **Specific service function call**, passing internal data structure by reference

- **ResponseDocument / Raw data output**, using the previous data structure (modified by the service itself)
ZOO Kernel supports several programming languages

- C / C++ Native support
- Python Native support (Python interpreter)
- Fortran Optionnal support (F77, F90)
- PHP Optionnal support (PHP embedded)
- Java Optionnal support (Java SDK)
- Javascript Optionnal support (SpiderMonkey)
ZOO Service examples

GDAL ZOO Service code and .zcfg

http://zoo-project.org/trac/browser/trunk/zoo-services/ogr/ogr2ogr/service.c
http://zoo-project.org/trac/browser/trunk/zoo-services/ogr/ogr2ogr/cgi-env/Ogr2Ogr.zcfg

OGR ZOO Service code and zcfg

http://zoo-project.org/trac/browser/trunk/zoo-services/gdal/translate/service.c
http://zoo-project.org/trac/browser/trunk/zoo-services/gdal/translate/cgi-env/Gdal_Translate.zcfg
**ZOO API**

**ZOO API** is a server-side Javascript library designed to make the WPS Process creation and chaining easier.

**ZOO API** is based on ZOO Kernel Javascript support and uses SpiderMonkey the Mozilla foundation JavaScript engine.

**Proj4js** adaptation for server-side reprojection.

Easy conversion to vector formats (GML, KML, GeoJSON, etc).

**ZOO API** allows to orchestrate WPS services simply and offers the ability to add logic and controls in the WPS chaining.

[www.zoo-project.org](http://www.zoo-project.org)
ZOO Development Plans

ZOO Kernel:
- .zcfg/yaml support for configuration file
- Error Management system
- Win 32 support

ZOO Services:
- Other GDAL/OGR functions implementation
- GRASS, R and external models research and development
- Other OpenOffice functions implementation

ZOO API:
- More demos and examples

www.zoo-project.org
DRAFT PROGRAM:

- Install **ZOO Kernel** on Linux

- Use ZOO Services (C/Python)

- Create a new ZOO Service

- Use ZOO API simply

- Link with client side
Thanks for your time

nbozon@3liz.com
gerald.fenoy@geolabs.fr
info@zoo-project.org
zoo-discuss@gisws.media.osaka-cu.ac.jp