|  |  |  |  |
| --- | --- | --- | --- |
| logofire-standardresolution | jaune | logo_ce-en-rvb-hr | 7plogo |

|  |  |
| --- | --- |
| Project Acronym | **Fed4FIRE** |
| Project Title | **Federation for FIRE** |
| Instrument | **Large scale integrating project (IP)** |
| Call identifier | **FP7-ICT-2011-8** |
| Project number | **318389** |
| Project website | **www.fed4fire.eu** |
| Experiment | **GEO-Cloud** |

**IDV Software Document**

|  |  |
| --- | --- |
| Work package | WP10 |
| Task | T10.1.2 GEO-Cloud Experiment |
| Due date | 28/02/2014 |
| Submission date | 28/02/2014 |
| Report Lead | Félix Pedrera (DEIMOS) |
| Version | 1.0 |
| Authors | Félix Pedrera, José Manuel Latorre (DEIMOS) |
| Reviewers | Jonathan Becedas (DEIMOS) |

|  |  |
| --- | --- |
| **Document ID** | **GEO-Cloud-DMS-TEC-TEC09-E** |
| Abstract | This document provides a description of the software generated and used to design the IDV module |
| Keywords | Report |

|  |  |  |  |
| --- | --- | --- | --- |
| Nature of the document | R | Report | X |
| P | Prototype |  |
| D | Demonstrator |  |
| O | Other |  |
| Dissemination level | PU | Public |  |
| PP | Restricted to other programme participants (including the Commission) |  |
| RE | Restricted to a group specified by the consortium (including the Commision) |  |
| CO | Confidential, only for members of the consortium (including the Commission) | X |

**Disclaimer**

*The information, documentation and figures available in this deliverable, is written by the Fed4FIRE* (Federation for FIRE) *– project consortium under EC co-financing contract FP7-ICT-318389 and does not necessarily reflect the views of the European Commission. The European Commission is not liable for any use that may be made of the information contained herein.*

**Executive Summary**

This document describes the software that is being used to build the Image Distribution and Visualization Software.

It is constituted by a combination of selected open source software and their adaptation for the GEO-Cloud experiment.**Acronyms and Abbreviations**

|  |  |
| --- | --- |
| IDV | Image Distribution and Visualization |
| HTTP | Hypertext Transfer Protocol |
| LGPL | GNU Lesser General Public License |
| OGC | [Open Geospatial Consortium](http://www.opengeospatial.org/) |
| JRE | Java Runtime Environment |
| API | Application Programming Interface |
| WCS | [Web Coverage Service](http://www.opengeospatial.org/standards/wcs) |
| WMS | [Web Map Service](http://www.opengeospatial.org/standards/wcs) |
| JDK | Java Development Kit |
| WFS | [Web Feature Service](http://www.opengeospatial.org/standards/wfs) |
| JAI | Java Advanced Imaging |
| GIST | Generic Index Structure |

**Table of Contents**

[1 Introduction 7](#_Toc381344968)

[2 IDV System 8](#_Toc381344969)

[2.1 Apache Tomcat 9](#_Toc381344970)

[2.2 GeoServer 9](#_Toc381344971)

[2.3 GeoWebCache 9](#_Toc381344972)

[2.4 Marlin Renderer 10](#_Toc381344973)

[2.5 Java Advanced Imaging API 12](#_Toc381344974)

[2.6 PostGIS 12](#_Toc381344975)

[2.6.1 PostgreSQL 13](#_Toc381344976)

[3 Conclusions 15](#_Toc381344977)

# Introduction

The GEO-Cloud Image Distribution and Visualization (IDV) System communicates with the Archive and Catalogue, and it is the interface with the end users. This provides a web service that can be accessed through the Internet.

It is mainly constituted by open source software that has been adapted for this project, and runs on Linux Debian Squeeze 6.0.

The IDV system makes use of the following software:

* Apache Tomcat
* GeoSever
* GeoWebCache
* Marlin Renderer
* Java Advanced Imaging API
* PostGIS

In the following section, the software is described.

To this document, the executable files and the generated code are associated:

1. apache-tomcat-8.0.3.tar.gz
2. geoserver-2.5-RC1-war
3. geowebcache-1.5.1-war
4. jai\_imageio-1\_1-lib-linux-amd64-jdk
5. jai-1\_1\_3-lib-linux-amd64-jdk
6. marlin-0.4
7. postgis-2.1.1.tar.gz
8. postgresql-9.3.3.tar.bz2

# IDV System

The GEO-Cloud Image Distribution and Visualization System will be almost composed by existing open source software components, widely used and maintained by the geospatial community. After selecting, configuring and installing these components in each virtual machine, we will use BonFire´s facilities to communicate with each other.

Based in our previous experience, and once studied the image distribution and visualization systems’ state of art, we have decided that the following software will be the most suitable to optimize the IDV System performance, running on Linux Debian Squeeze (6.0):

* Apache Tomcat
* GeoSever
* GeoWebCache
* Marlin Renderer
* Java Advanced Imaging API
* PostGIS

The Figure 1 shows how some of these software components are distributed along the IDV System Architecture.

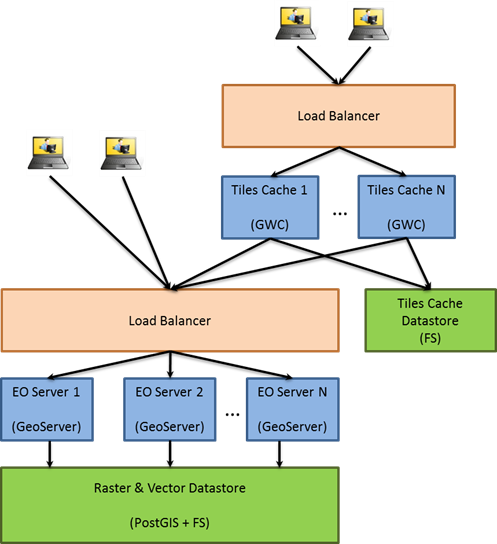


Figure 1. IDV System Architecture

## Apache Tomcat

**Version**: 8.0.3

**Homepage**: http://tomcat.apache.org

**Download URL**: http://ftp.cixug.es/apache/tomcat/tomcat-8/v8.0.3/bin/apache-tomcat-8.0.3.tar.gz

Apache Tomcat is an open source software implementation of the Java Servlet and JavaServer Pages technologies from Sun Microsystems, and provides an HTTP web server environment for Java code to run in, being one of the most popular Java application servers.

Apache Tomcat is developed in an open and participatory environment and released under the [Apache License version 2](http://www.apache.org/licenses), powering numerous large-scale, mission-critical web applications across a diverse range of industries and organizations.

The IDV System mainly needs Apache Tomcat in order to run GeoServer and GeoWebCache.

## GeoServer

**Version**: 2.5 RC1

**Homepage**: http://geoserver.org

**Download URL**: http://sourceforge.net/projects/geoserver/files/GeoServer/2.5-RC1/geoserver-2.5-RC1-war.zip

GeoServer is an open source software server licensed under the GNU General Public License, written in Java that allows us to share and edit geospatial data. Designed for interoperability, it publishes data from any major spatial data source using open standards.

GeoServer is the reference implementation of the [Open Geospatial Consortium](http://www.opengeospatial.org/) (OGC) [Web Feature Service](http://www.opengeospatial.org/standards/wfs) (WFS) and [Web Coverage Service](http://www.opengeospatial.org/standards/wcs) (WCS) standards, as well as a high performance certified compliant [Web Map Service](http://www.opengeospatial.org/standards/wms) (WMS). GeoServer forms a core component of the Geospatial Web.

GeoServer’s speed highly depends on the chosen Java Runtime Environment (JRE). To increase the performance, it is recommended the use of [Oracle JRE 6](http://www.oracle.com/technetwork/java/javase/downloads/index.html) or newer. For our system, we will use Oracle JRE 7. JREs other than those released by Oracle may work correctly, but they are not tested or supported in general; using OpenJDK, for example, produces reductions in the 2D rendering performance.

We configure GeoServer and install plugins, extensions and add-ons, some of them listed below, to obtain the best performance for the IDV system.

## GeoWebCache

**Version**: 1.5.1

**Homepage**: http://geowebcache.org

**Download URL**: http://sourceforge.net/projects/geowebcache/files/geowebcache/1.5.1/geowebcache-1.5.1-war.zip/download

GeoWebCache is a Java web application licensed under the [GNU Lesser General Public License (LGPL](http://www.gnu.org/licenses/lgpl.html)), used to cache map tiles coming from a variety of sources such as OGC Web Map Service (WMS). It implements several service interfaces (such as WMS-C, WMTS, TMS, Google Maps KML, Virtual Earth) in order to accelerate and optimize the map image delivery. It can also recombine tiles to work with regular WMS clients.

As most mapping clients render WMS (Web Map Service) data every time they are queried this can result in unnecessary processing and can increase waiting times. GeoWebCache optimizes this experience by saving (caching) map images, or tiles, as they are requested, actually acting as a proxy between a client (such as [OpenLayers](http://openlayers.org/) or [Google Maps](http://maps.google.com/)) and a server (such as [GeoServer](http://geoserver.org/), or any WMS-compliant server). As new maps and tiles are requested, GeoWebCache intercepts these calls and a return pre-rendered tile if stored, or calls the server to render new tiles as necessary. Thus, once tiles are stored, the speed of map rendering increases, creating a much improved user experience.

GeoWebCache can even be used when maps are not completely static, since it allows for the selective expiration of tiles, so that data remains current. Thus GeoWebCache acts as an all-purpose accelerator for map rendering.

GeoWebCache is automatically installed and configured with GeoServer as a built-in extension by default, but we are going to disable it for the IDV system. For a better performance, we will install and configure GeoWebCache as a standalone web application in order to load layers from multiple sources (allowing cache distribution along different nodes) and to isolate it from any data servers.

## Marlin Renderer

**Version**: 0.4

**Homepage**: https://github.com/bourgesl/marlin-renderer

**Download URL**: https://github.com/bourgesl/marlin-renderer/releases/download/v0.4.0/marlin-0.4.jar

As we said above, [GeoServer](http://www.geo-solutions.it/technologies/geoserver/) is a Java based software, so its performance is heavily dependent on what the Java Virtual Machine and the Java standard library can offer. [GeoServer](http://www.geo-solutions.it/technologies/geoserver/) developers have been pushing the boundaries of what can be done by replacing portions of the Java standard libraries by adopting faster replacements (as we are going to do in the IDV system). But it was difficult to find a better solution for the anti-aliased rasterizer contained in the JDK, that is, the portion of software that turns all vector data into its raster representation while painting maps. There were two choices, depending on the JDK adopted:

* The Oracle JDK comes with the Ductus rasterizer, a closed source, native antialiased rasterizer that has serious scalability issues, since it allows only a single shape at a time to be rasterized. While it provides good performance for the single threaded cases, it is not really thought out for server side usage.
* The OpenJDK comes with the Pisces rasterizer, an open source, pure java rasterizer that has no scalability issues, but it is slower than Ductus. It takes at least 4 concurrent requests for it to become faster than Ductus. Another issue is that OpenJDK is only available for Linux.

But in 2013, Marlin renderer, an open source (GPL2+CP) Java2D rendering engine, optimized the performance (improved memory usage and footprint, better multi-threading). In every test good results and benchmarks were obtained: the harder to draw the map, the more information to have to render, and the greater the benefit in the use of Marvin. Even in a configuration with multiple copies of GeoServer with Oracle JDK 7, each running in its own JVM and load balance them (as the IDV System scenary) to mitigate the Ductus scalability issue is still about 30% slower than what Marlin provides.

In addition, Marlin rasterizer is pluggable, so it is possible to build and use in both OpenJDK and Oracle JDK.

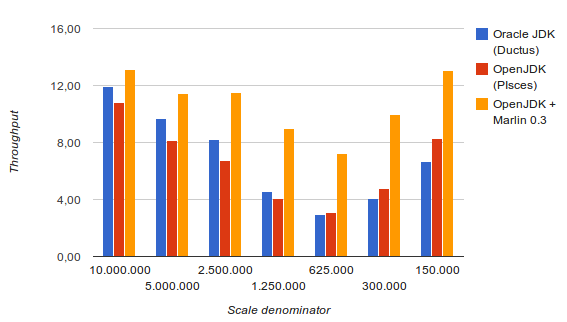


Figure 2. GeoServer 2.5 with different rasterizers

## Java Advanced Imaging API

**Version**: JAI 1.1.3, JAI Image I/O 1.1

**Homepage**: http://www.oracle.com/technetwork/java/index.html

**Download JAI URL**: http://download.java.net/media/jai/builds/release/1\_1\_3/jai-1\_1\_3-lib-linux-amd64-jdk.bin

**Download JAI Image I/O URL**: http://download.java.net/media/jai-imageio/builds/release/1.1/jai\_imageio-1\_1-lib-linux-amd64-jdk.bin

The [Java Advanced Imaging API](http://java.sun.com/javase/technologies/desktop/media/) (JAI) is an advanced image manipulation library built by Oracle. GeoServer requires JAI to work with coverages and leverages for WMS output generation. By default, GeoServer ships with the pure Java version of JAI, but to improve the performance, we will install the native JAI version in the JDK/JRE.

In particular, installing the native JAI is important for all raster processing, which are heavily used in both WMS and WCS services in order to rescale, cut and re-project the rasters. Installing the native JAI is also important for all raster reading and writing, which affects to both WMS and WCS services. Finally, native JAI is very useful even if there is no raster data involved, as the WMS output encoding requires writing PNG/GIF/JPEG images, which are rasters themselves.

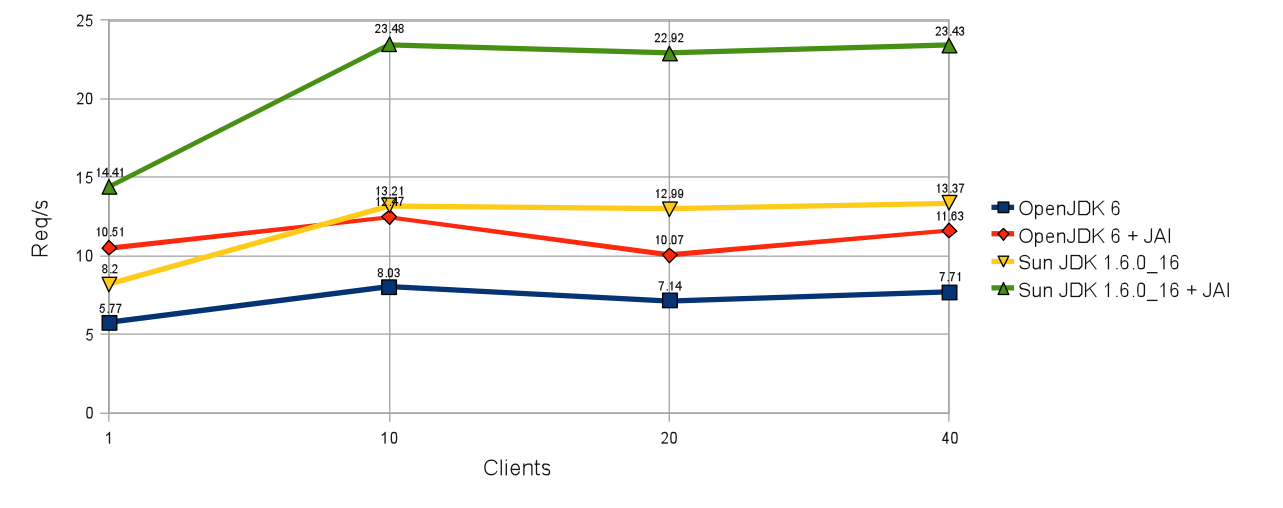


Figure 3. JDK and JAI performance comparison

## PostGIS

**Version**: 2.1.1

**Homepage**: http://postgis.net

**Download URL**: http://download.osgeo.org/postgis/source/postgis-2.1.1.tar.gz

PostGIS is a spatial database extender for [PostgreSQL](http://postgresql.org/) object-relational database, released under the GNU General Public License ([GPLv2](http://opensource.org/licenses/gpl-2.0.php)). It provides support for geographic objects (spatial types, spatial indexes and spatial functions) allowing the location of queries to be run in SQL and, because it is built on PostgreSQL, as said before, automatically inherits important features from it: [ACID](http://en.wikipedia.org/wiki/ACID) transaction guarantees, reliability, crash recovery, hot backup, replication, full SQL92 support, etc.

The main reason to use PostGIS/PostgreSQL instead of other spatial databases such as MyGIS/MySQL is because it has a built-in “type extension” mechanism. It also has a generic index structure (GIST) to build indexes.

PostGIS is widely extended in the geospatial community, for example, it is used by the Institut Geographique National of France to store the high resolution topographic map of the country or GlobeXplorer, a web-based service providing online access to petabytes of global satellite and aerial imagery, using PostGIS to manage the metadata associated with the imagery catalogue.

### PostgreSQL

**Version**: 9.3.3

**Homepage**: http://www.postgresql.org

**Download URL**: http://ftp.postgresql.org/pub/source/v9.3.3/postgresql-9.3.3.tar.bz2

PostgreSQL is a powerful, open source object-relational database system with more than 15 years of active development and a proven architecture that has earned a good reputation for reliability, data integrity, and correctness, running on all major operating systems. It is fully ACID compliant, has full support for foreign keys, joins, views, triggers, and stored procedures (in multiple languages), and supports storage of binary large objects.

An enterprise class database, PostgreSQL boasts sophisticated features such as Multi-Version Concurrency Control (MVCC), point in time recovery, tablespaces, asynchronous replication, nested transactions (savepoints), online/hot backups, a sophisticated query planner/optimizer, and write ahead logging for fault tolerance. It supports international character sets, multibyte character encodings, Unicode, and it is locale-aware for sorting, case-sensitivity, and formatting. It is highly scalable both in the sheer quantity of data that it can manage and in the number of concurrent users it can accommodate. There are active PostgreSQL systems in production environments that manage in excess of 4 terabytes of data

PostgreSQL is highly customizable, featureful, standards compliant (Its SQL implementation strongly conforms to the ANSI-SQL:2008 standard) and has won [praise from its users](http://www.postgresql.org/about/quotesarchive) and [industry recognition](http://www.postgresql.org/about/awards), including the Linux New Media Award for Best Database System and five time winner of the The Linux Journal Editors' Choice Award for best DBMS.

# Conclusions

This document summarizes and describes the software used to build the IDV system for the GEO-Cloud experiment.

The next step will be the implementation of the IDV system in the Fed4FIRE facilities, specifically in the the BonFIRE cloud. Afterwards, the IDV system will communicate with the virtual machines deployed in Virtual Wall, which will emulate the end users’ loads over the IDV system.

Notice that during the integration it is possible that some parts of the code will be adapted to fit in the whole GEO-Cloud architecture, when deployed in Fed4FIRE.