

# Open Grid Manager - Documentation 1.1

---

## Table Of Contents

---

1. Introduction.....	3
2. Installation .....	8
2.1. OGM Server.....	8
2.2. OGM Agent (Windows) .....	10
2.3. OGM Agent (Unix) .....	11
2.4. OGM Browser .....	12
2.5. OGM Portlet.....	13
3. Usage .....	14
4. Limitations .....	15
5. Troubleshooting.....	16
6. Change History.....	17
7. License .....	18
8. IPR Registry .....	19

# 1. Introduction

Open Grid Manager (OGM) is a lightweight open source grid management framework that provides a cohesive solution for monitoring and managing arrays of heterogeneous grid resources deployed within live production grids.

The system consists of a light weight agent (OGM-Agent) deployed on the grid resources to be managed that communicates with a suite of web services (OGM-Server) exposing persistent registry capabilities.

As an open and extensible solution, OGM allows you to easily integrates Grid management information into a range of Service Oriented Architecture (SOA) based infrastructures including existing enterprise management and operations support system (OSS) solutions.

## Background

Currently, explosive growth has been experienced in the area of Grid Technology with various global efforts underway to develop production grids, middleware technologies and application development tools to support the uptake and development of grids within academic and commercial environments.

However, as the grid evolves as a heterogeneous array of hybrid grid resources, the management of such grids becomes more and more prevalent. With the rush to build live production Grids, strategies to manage such collections of resources across distributed, heterogeneous dynamic virtual organizations have been overlooked and if not addressed soon, will inhibit the rapid adoption of Grid technology to the wider communities.

While the existing Relational Grid Monitoring Architecture (R-GMA) implementation from EGEE (Enabling Grids for EScienceE) provides information and monitoring services deployed as an RPM on top of the LCG stack, OGM provides the advantage of an open, lightweight and cross platform framework with a minimal footprint on both the resource and the server.

## Architecture

OGM comprises of a number of components including

- **OGM-Agent** for deployment in the OMII container or alongside it on the resources to be managed

- ⌘ **A standards based schema** to communicate management information with the middleware registry
- ⌘ **OGM-Server** to expose persistent registry capabilities (using GRIMOIRES) via a suite of standards based Web Services
- ⌘ **OGM-Browser** to provide a web based management interface to provide a graphical representation of the real-time state of resources in the Grid

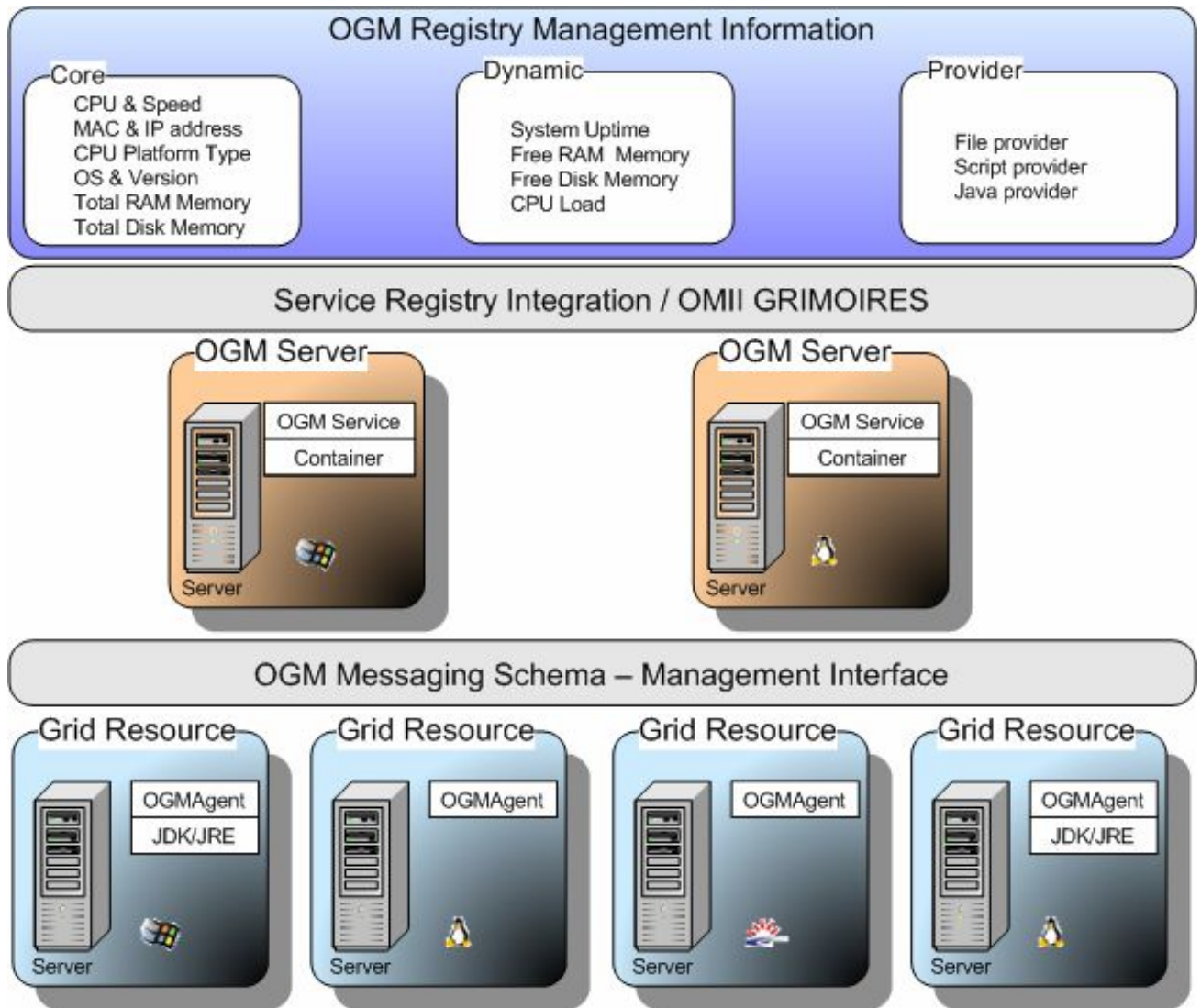


Figure 1: OGM Architecture

OGM is based on a software component called **Grid Manager** developed at the **Belfast e-Science Centre (BeSC)**. Grid Manager monitors the state of resources and nodes in a Grid defined here as web services running within distributed hosting environments.

The OGM-Agent is deployed onto each node and is responsible for monitoring its state and reporting this to a centralised monitoring service (i.e. the OGM-Server). The OGM-Server collects management information from nodes in the deployed Grid and persists this in a registry. Analysis of the individual and collective state of nodes can determine the performance of a Grid and enable management activities to respond in an efficient manner. For example, if the Grid is performing poorly, OGM should identify the nodes which are contributing to poor performance and enable corrective action.

The OGM Browser presents a graphical representation of the state of the Grid. Potential exists to enhance service discovery and brokering mechanisms through the real-time analysis of the persistent information.

Figure 1 illustrates the OGM architecture. The OGM Server is deployed into a hosting environment such as a servlet container or the OMII Server and is responsible for reporting the management information to a registry such as UDDI or GRIMOIRES.

## Management Information

Live Production Grids consist of a collection of heterogeneous Grid resources with varying dynamic states. The state of a resource depends on its hardware configuration and software deployed. This includes

1. Number of CPUs
2. Size of Physical Disk
3. Operating system installed (name and version)
4. Total and current memory available (incl. Swap)
5. current CPU load
6. average load over 1, 5, and 15 minutes
7. hostname, ip, mac address, uptime, architecture

## Registry Integration

Monitoring the state of the component nodes will increase the efficiency of a Grid by enabling the identification of undesirable states and presenting the opportunity to address the cause. In addition, the collation of state information enables enhanced strategies for service deployment, job submission and service brokering when integrated with a service registry. Within the Belfast eScience centre and the Grid community there are a number of different service registries with which integration is possible to provide the enhanced functionality of service discovery and brokering. However, Grid Registry with Metadata Oriented Interface: Robustness Efficiency, Security (GRIMORIES) augments a UDDI

Registry with meta-data and is an OMII project through the managed programme. Therefore, it is proposed to supplement the initial integration with the OMII middleware, to report state data from the OGM to GRIMORIES. Integration of OGM with GRIMORIES will enhance service and workflow discovery mechanisms by providing additional meta-data regarding the state of nodes in the Grid to GRIMORIES via interoperability with the GMS. In order to provide integration between the OGM and GRIMORIES, the definition of a messaging infrastructure is required, that is the web service message detailing the state data that will pass from OGM to GRIMORIES and stored in the GRIMORIES registry, that is the RDF store. An example of the benefits to the end user entail, discovering a service or workflow which is currently available in a hosting environment offering the best performance, or the most available memory etc.

In a live production Grid, analysis of the collected OGM management information will provide tangible benefits and facilitate:

- Identification of performance problems and early corrective action, thereby enhancing the robustness and reliability of a Grid
- Enhanced resource selection for service deployment, job submission etc
- Enhanced service brokering capabilities



## 2. Installation

- 2.1. OGM Server
- 2.2. OGM Agent (Windows)
- 2.3. OGM Agent (Unix)
- 2.4. OGM Browser
- 2.5. OGM Portlet

### 2.1. Installation - OGM Server

1. Install a Registry (such as GRIMOIRES or [Apache jUDDI](#) )
2. **Download** the latest release for OGM from <http://sourceforge.net/projects/ogm>
3. Extract with

```
$tar -xvf ogm-src.tar.gz
```
4. Inside the ogm directory, use the command `ant build-server` to build the OGM-Server

```
$ant build-server
```
5. Drop the generated **ogm-server.war** inside your Servlet Container
6. Configure the OGM Server to point to the registry by editing the file **webapps/ogm-server/WEB-INF/classes/samples.prop**. For example:

```
inquiryURL=http://c21.besc.ac.uk:18080/grimoires/services/inquire
publishURL=http://c21.besc.ac.uk:18080/grimoires/services/publish
userid=
password=
```

7. Start your Servlet Container

## Installation - OGM Agent

## 2.2. Installation - OGM Agent - Windows

1. To build Java Agent open windows command prompt.
2. To build C Agent open the Visual Studio command prompt by going to Start-->Program Files-->Visual Studio Tools-->Visual Studio Command Prompt.
3. Go to ogm source directory. Inside ogm source directory use the command ant build-client to build the OGM-Agent

```
$ant build-client
```

4. edit the configuration config.xml file inside dist/client/config/, which looks like

```
<config>
<Server
url="http://193.61.123.73:8080/ogm-server/services/ogm">
<Frequency>30</Frequency>
<FileProvider>
<name>Uptime</name>
<source>/proc/upime</source>
</FileProvider>
<ScriptProvider>
<name>Date</name>
<source>date</source>
</ScriptProvider>
</Server>
<Server
url="http://193.61.123.71:8080/ogm-server/services/ogm">
<Frequency>20</Frequency>
</Server>
<NetworkCard>eth0</NetworkCard>
</config>
```

5. This config.xml is used by the ogm Agent to report to two different servers. Start the Agent

## 2.3. Installation - OGM Agent - Unix

1. Go to ogm source directory. Inside ogm source directory use the command `ant build-client` to build the both Java and C OGM-Agent

```
$ant build-client
```

2. edit the configuration `config.xml` file inside `dist/client/config/`, which looks like

```
<config>
<Server
url="http://193.61.123.73:8080/ogm-server/services/ogm">
<Frequency>30</Frequency>
<FileProvider>
<name>Uptime</name>
<source>/proc/upime</source>
</FileProvider>
<ScriptProvider>
<name>Date</name>
<source>date</source>
</ScriptProvider>
</Server>
<Server
url="http://193.61.123.71:8080/ogm-server/services/ogm">
<Frequency>20</Frequency>
</Server>
<NetworkCard>eth0</NetworkCard>
</config>
```

3. This `config.xml` is used by the ogm Agent to report to two different servers. Start the Agent

## 2.4. Installation - OGM Browser

1. **Download** the latest release for OGM from <http://sourceforge.net/projects/ogm>
2. Extract with  

```
$tar -xvf ogm-src.tar.gz
```
3. Inside the ogmbrowser directory, use the command `ant build-browser` to build the OGMBrowser  

```
$ant build-browser
```
4. Drop the generated **ogm-browser.war** inside your Servlet Container
5. Start your Servlet Container
6. Open a browser and go to `http://localhost:8080/ogmbrowser`.
7. Read the Help Section of the OGM Browser to configure the OGM Browser

## 2.5. Installation - OGM Portlet

1. Download **GridSphere** and install by following the instruction on the gridsphere web site.
2. Open a browser and go to `http://host:port/gridsphere`. Set the portal administrator name and password.
3. Inside the `ogm-src/portal/ogmportlet` directory, edit the `build.properties` to set Gridsphere home and build directory.
4. To install the OGMPortlet, use the command `ant install`
5. Start your Servlet Container
6. Login as the administrator inside the Portal and add the ogmportlet to the group.
7. Go to the config section to add registry details.

### 3. Usage

As shown in Fig1. Information sent by OGM Agent is categorized as core , dynamic and providers. The core information is sent only once when the agent starts. The dynamic information is regularly updated. <Frequency> tag in config.xml determines how frequently the dynamic information is to be updated.

Apart from this Providers can be configured to add more information. There are two types of provider FileProvider and ScriptProvider. FileProvider reads the content of the file and adds the content of the information to be sent. ScriptProvider executes the script and adds the output of the results to the information to be sent.

In config.xml you can specify this provider by using

```
<Frequency>30</Frequency>
<FileProvider>
<name>Uptime</name>
<source>/proc/upime</source>
</FileProvider>
<ScriptProvider>
<name>Date</name>
<source>date</source>
</ScriptProvider>
```

where name tag is used for specifying the keyname for the identifier inside registry. Source tag is used to specify the path of the file or the script to be used.

OGM Agent can be configured to send soap messages to multiple OGM Server using the server tag as shown in config.xml file

## 4. Limitations

1. OGM Agent can send soap messages to multiple servers. The number of servers is limited to 10.
2. As per the UDDI specification registry can only store 256 characters inside the identifier value.

## 5. Troubleshooting

1. Segmentation Fault  
OGM C Agent needs to be compiled for the architecture.
2. No uuid returned  
Check the registry details in samples.prop file of the ogm-server.
3. Could not understand must userstand headers  
Check the security configuration in crypto.properties.
4. Unable to open crypto.properties  
Make sure that crypto.properties is in classpath

## 6. Change History

### 1) ogm1.1

- New Portlet Interface
- Enhanced OGM Browser
- Support for UDDI and Grimoires

## 7. License

Copyright (C) [Belfast e-Science Centre \(BeSC\)](#), 2007

Copyright in this software belongs to the Belfast e-Science Centre (BeSC), Queen's University Belfast.

OGM is distributed under the following licenses

- OGM is available under the [GNU Lesser General Public License \(LGPL\)](#) recognized by [The Open Source Initiative](#).

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2.1 of the License, or (at your option) any later version.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public License along with this library; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA.

- For alternative licensing, please contact us at [BeSC](#)

For more info on OGM usage, see the docs directory in the distribution.

## 8. IPR Registry

Software Libraries used and/or distributed with OpenGridManager.

Software	Distribution	Owner	License	Download	Comments
<b>ezXML</b>	aggregation	Aaron Voisine	<b>MIT License</b>	<b>Source</b>	License allows binary to be redistributed
<b>gSOAP</b>	aggregation	Robert A. van Engelen, Genivia inc.	<b>gSOAP Public License 1.3a</b>	<b>Source</b>	Based on Mozilla Public License 1.1.  MPL1.1-like license which allows binary to be redistributed