# A Scalable Registry Service for jGMA

Distributed Systems Group, University of Portsmouth
**Mark Baker and Matthew Grove**
**[mark.baker@computer.org, matthew.grove@port.ac.uk]**

*Abstract*

jGMA is a Java-based wide-area message passing framework designed to be a reference implementation of the GGF's Grid Monitoring Architecture. jGMA clients require a scalable, fault tolerant mechanism for joining and querying the distributed network; the so called jGMA Virtual Registry. In the first half of this paper we briefly outline the current jGMA architecture, and then move on to describe the requirements and design of the Virtual Registry; including here a description of how we intend to use P2P technologies to provide the required functionality.

## 1 Introduction

jGMA [1] is a pure Java reference implementation of the GGF's GMA [2], which represents the characteristics required for a scalable grid-based monitoring infrastructure. In this model, producers or consumers that accept connections publish their existence in a directory service (registry). Producers and consumers can then both use the registry to locate parties, which will act as a source or destination for the data they are interested in.

Our motivation for developing jGMA, is described elsewhere [3][4], but can be summarised by saying existing systems where either embedded into larger software packages, such as the MDS in the Globus Toolkit [5], and Network Weather Service [6], and were deemed difficult to breakdown into a standalone version needed for our purpose. R-GMA [7] and pyGMA [8], two standalone GMA implementations were considered, but both were less than ideal for our purpose; the drawbacks are discussed elsewhere [9].

## 2 The jGMA Architecture

Our reference implementation of GMA a has number of idealised features, including complying with the GMA specification, being capable of scaling from LAN to WAN proportions, and across thousands of sites with millions of producers/consumers, supporting both non-blocking and blocking events and having the capability to take advantage of TLS [10] and/or the GSI [11]. In addition, we wanted the actual implementation to have a small and well-defined API, a minimal number of other installation dependencies, be easy to install and configure, have a minimal impact on its hosts, good performance and be capable of working through firewalls.

jGMA consists of three entities:

1. Mediators that permit producers and consumers discover each other and allow remote communications,

2. Consumers,
3. Producers.

The focus of this paper is on the development of the registry services within the mediator component, which allow consumers and producers to discover and communicate with each other over the wide-area. Figure 1 shows the registry components within the jGMA architecture [4].

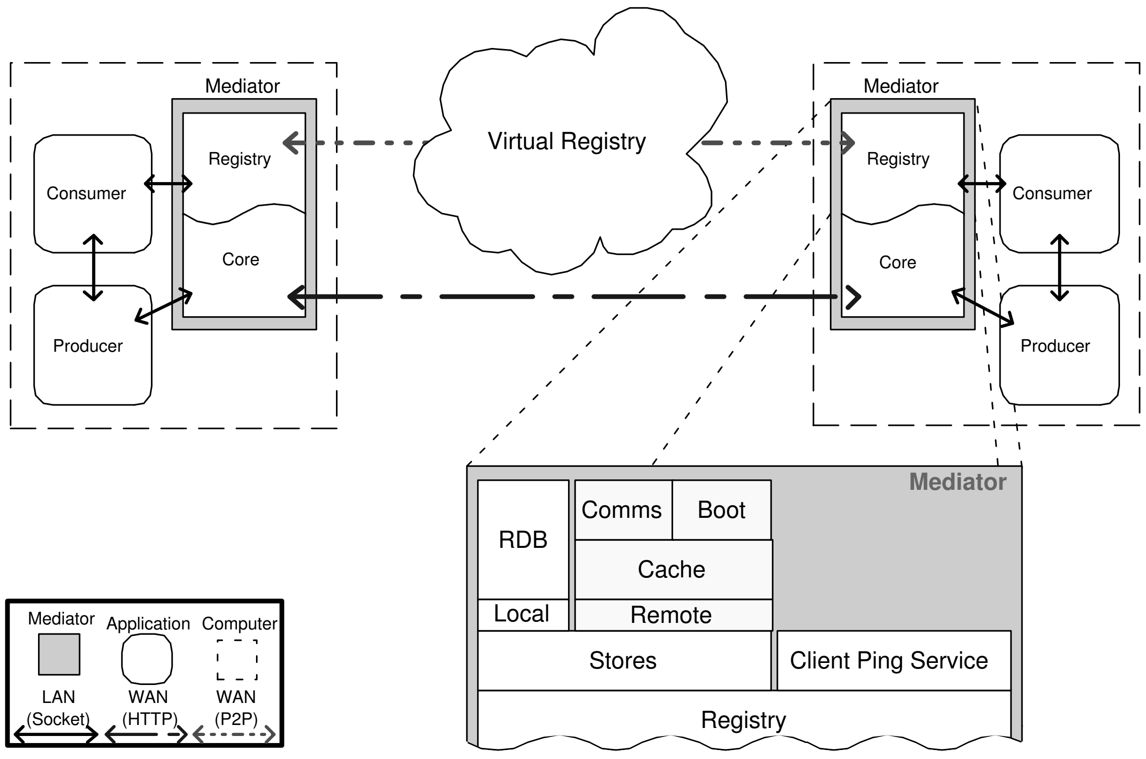

Figure 1: The jGMA architecture - highlighting the Virtual Registry components.

## 3 The jGMA Virtual Registry

A simple volatile data store was initially implemented in jGMA; this only stores information about clients that are connected directly to the mediator. The new Virtual Registry (VR) component within the mediator needs to provide discovery, naming, and querying services for jGMA clients. For the VR we need mechanisms to allow the registry components to locate and query, over the wide-area, the localised data stores within other mediators to form a virtualised registry.

Virtual Registry requirements:

- Be scalable,
- Store sufficient information to be GMA compliant,
- Be secure, and prevent unauthorised access to the data,
- Need minimal configuration,

- Ideally have no single point of failure,
- Be robust and tolerant of network failure,
- Efficient query routing between VRs.

The VR architecture provides two pluggable layers; one allows the use of different implementations of local data storage, for example using text files, a relational database such as MySQL [12] or an XML database such as Xindice [13]. The second layer allows different protocols to be used for communications between components in the VR; we are exploring the use of Peer-to-Peer (P2P) technologies to provide a distributed database.

### 3.1 Virtual Registry Design

The VR provides three core services: service discovery, querying, and caching. It should be noted that not all P2P systems provide all three services, so some generic jGMA services are needed to provide the missing functionality. Implementing these services as separate modules makes it possible to try different implementations of each service without having to redesign the other services. A feature of this approach is interoperability with other systems, should a standard protocol be adopted for monitoring information on the Grid, it would be possible to connect jGMA without redesign or significant refactoring.

1. Service discovery: This is the process by which the registry component within a mediator locates other registries to form the VR. The more this mechanism relies on centralised services, the greater the likelihood of a failure in the discovery process. For example, one solution is to contact a well-known server to get a list of existing VRs; this is not a scalable or fault tolerant approach. We require a more dynamic discovery process, which does not rely on so called hardwired addresses to work. We are currently assessing the suitability of existing P2P communications protocols for providing mediator discovery and inter-VR communications.
2. Distributed querying: Once the VR has been joined, queries can be despatched and routed through the VR infrastructure in an efficient manner. The VR network topology must be self-healing, by this we mean that should one or more of the registries fail; it should have a minimal affect on the overall performance of the VR. The VR network topology should aim to reduce the number of hops required to traverse the network in order to minimise the time required to propagate a search query. The design of the VR infrastructure will allow us to experiment with different network topologies and P2P protocols, such as Gnutella [14].
3. Caching: Using cached registry queries reduces the number of queries sent over the wide-area to other registries. A cache hit will allow the VR to respond immediately to the query rather than having to wait for a remote query to complete. Caching also provides some fault tolerance during transient faults with either the WAN connectivity or remote registries. Issues of data consistency will have to be dealt with at this level too.

## 4 Summary and Conclusions

In this paper we have described the requirements for a distributed registry service for jGMA. We have outlined the design for a pluggable registry framework for jGMA, which will allow us to explore how best to leverage existing P2P technologies to create a scalable, robust Virtual Registry. We currently support SQL as a query language and LDIF [15] as a response mark up. The VR has a layer of abstraction, which translates queries and responses into an intermediate format internally, this allows us to add support for different query languages, such as XPATH [16], to permit interoperability with other GMA implementations in the future.

In order to demonstrate the capabilities of the jGMA framework we are intend to develop a library for online distributed gaming, which has become increasingly popular with the widespread uptake of broadband. Games publishers have each tried to provide an infrastructure to support their online games. We Believe there is an opportunity to develop a standard services, based on a jGMA, to support these games [17].

An early binary version of jGMA [18] is currently available for developers interested in investigating and further enhancing its capabilities. jGMA has been repeatedly benchmarked throughout its development, the latest results are presented in [3].

## References

[1] jGMA, http://dsg.port.ac.uk/projects/jGMA/

[2] GMA, http://www.ggf.org/documents/GFD/GFD-I.7.pdf

[3] Wide-Area Resource Monitoring using GridRM and jGMA, M.A. Baker, G. Smith and M. Grove, submitted to a special issue of the International journal Concurrency and Computation: Practice and Experience, Malcolm Atkinson, Simon Cox, Ian Sommerville, and David Walker (eds), Wiley and Sons ltd., January 2005, ISSN 1040-3108

[4] jGMA: A lightweight implementation of the Grid Monitoring Architecture, M.A. Baker and Matthew Grove, UK e-Science All Hands Meeting, September 01-03 2004

[5] Globus MDS, http://www-unix.globus.org/toolkit/mds/

[6] Network Weather Service, http://nws.npaci.edu/NWS/

[7] R-GMA, http://www.r-gma.org/

[8] pyGMA, http://www-didc.lbl.gov/pyGMA/

[9] jGMA: A lightweight implementation of the Grid Monitoring Architecture , UKUUG LISA/Winter Conference, February 2004

[10] TLS, http://www.ietf.org/html.charters/tls-charter.html

[11] GSI working Group, https://forge.gridforum.org/projects/gsi-wg

[12] MySQL, http://www.mysql.com/

[13] Xindice, http://xml.apache.org/xindice/

[14] Gnutella, http://www.gnutella.com/

[15] RFC 2849 - The LDAP Data Interchange Format (LDIF) - Technical Specification, Gordon Good, http://www.faqs.org/rfcs/rfc2849.html, June 2000

[16] XPATH, http://www.w3.org/TR/xpath

[17] jGMA: A Reference Implementation of the Grid Monitoring Architecture, Progression from MPhil to PhD Document, http://dsg.port.ac.uk/˜mjeg/docs/progression_report2004.pdf, October 2004

[18] jGMA Download, http://dsg.port.ac.uk/projects/jGMA/software/