OGSA Logger and GMA

1. Introduction

We see the Information and Monitoring services as having a fundamental position in OGSA and in any Grid. To demonstrate this we need to show how other services can be implemented as a thin layer on top of an Information and Monitoring service. Here we show how a logger system can be layered on top of GMA where we make use of the OGSA Logger System (Version 9) Document https://forge.gridforum.org/projects/ogsa-wg/document/OGSA_Logger_System_V9/en/1.

We have taken the basic functionality from section 2 and 2.1 of this Logger document and show how GMA can provide this. We have chosen to work from the basic functionality on the assumption that the detailed services listed in section 2.2 and described in that document can be adjusted a little to provide equivalent functionality, if this brings overall economies of a simpler OGSA.

The GMA services are described at rather a high level in the GMA Service Description: https://forge.gridforum.org/projects/ogsa-wg/document/GMA_service_description/en/1. The level is high because this is meant to be a contribution to the OGSA document, which is not meant to go into details of service descriptions. In addition we suggest you look at http://www.r-gma.org/ for a description of R-GMA and at a recent paper presented at COOPIS 2003 that will be appearing in an extended form in the new "Journal on Semantics of Data" (Springer LNCS series, http://lbdwww.epfl.ch/e/Springer/). Meanwhile a version exists at: http://www.cs.tcd.ie/coghlan/pubs/HeriotWatt-coopis-paper-01072003.pdf. This COOPIS paper is especially relevant in the case of the OGSA Logger and GMA because it concentrates on the point of view of *streams of data*.

Field Code Changed

This document relates to R-GMA, which is a relational implementation of GMA. When producing R-GMA we made one significant addition to GMA. We chose to provide a mediator/broker to find suitable producers/suppliers/publishers of information. This has allowed us to hide the registry/directory mechanism, which is why it is not discussed in the OGSA Information and Monitoring Service Description document. This choice makes the system much easier to use as it only offers very simple interfaces.

2. Nomenclature

The GMA document (http://forge.gridforum.org/projects/ggf-editor/document/GFD-I.7/en/1) talks about producers, consumers and *"compound producer/consumers"* while R-GMA introduces archivers. To avoid this last confusing term, we will refer to R-GMA archivers as republishers. (However our R-GMA documentation is still inconsistent, as we cannot change APIs overnight.)

Deleted: complex

Deleted: , which does not roll off the tongue very well

Deleted: and therefore prefer to use the term publisher instead of producer.

3. R-GMA in a nutshell

1. **Producer Service**. This is currently implemented as 3 distinct APIs though we plan to move it to a single service with properties. The 3 APIs offer a way to handle streams of data with different persistency characteristics. The StreamProducer uses an in-memory data structure while the DataBaseProducer also supports historical queries and the LatestProducer offers just the latest information for any primary key. All Producers support streaming.

2. **Consumer Service**. This takes a query and locates the best Producers to use in order to answer the query and then carries out that query. Queries are of 3 different types matching the three kinds of producer. For example, if you pose a Latest query, it will be answered by one or more LatestProducers.
3. **Republisher Service**. This is called an Archiver in our current documentation – The Republisher is a joint Consumer/Producer. Republishers can be used to produce a more efficient system with a better response by reducing the volume of data being transferred. As all Producers now support streaming, it is possible to join the components into different topologies according to the need.

Deleted: but as the term is wrong, now is a good time to get it right

4. What are the special requirements of a logger system?

R-GMA appears to meet all the requirements listed in the OGSA Logger document though the correspondence between R-GMA components and OGSA Logger ones is not as simple as one might have hoped. An R-GMA Consumer corresponds roughly to a LogBrowseSession and a Producer corresponds to a LogStreamA with a LogStreamConnection. 2.1.2 requests different persistency mechanisms. R-GMA offers 3 kinds of Producer as indicated above. For the StreamProducer you can specify a minimum retention period. Data will be kept for at least this period of time. Data are also kept until they are read by existing Consumers. The DataBaseProducer and LatestProducer both offer a clean-up mechanism to delete all data matching a certain pattern. In fact the only use that is made for this is to clean up old data (i.e. only using information in the timestamp).

Deleted: with a LogStreamConnection

Deleted: Connection.

Deleted: seems to correspond to a record type

Deleted: ¶

2.1.3 asks for support for sequential reads and writes. The R-GMA Consumer offers this.

2.1.4 asks for data to be read in sequence. R-GMA Consumers of streams do this. If the Consumer accesses information from a DataBaseProducer or LatestProducer the time ordering is by the time stamp associated with the data when it was first written.

2.1.6 asks for a stateful read cursor. R-GMA allows traversal over a set of records. It is unaffected by Producers or other Consumers.

2.1.7 asks for decoupling. This is fundamental to GMA.

2.1.8 asks for properties of a broker. R-GMA provides this. It is not clear how this requirement differs from 2.1.7.

2.1.9 asks for filtering near the source. The R-GMA producer can specify a predicate defining just what data are being published there. This is not a filter, as any attempt to publish data that is not consistent with the predicate is an error. This is more efficient.

2.1.10 asks for merged logs. It is not clear exactly what this means, however the R-GMA Consumers can merge information. Typically the re-publishers are merging information from multiple streams which they then re-publish.

2.1.11 asks for synchronous and asynchronous writes. The requirement seems to be interested in the tradeoff between speed and reliability. The R-GMA StreamProducer offers the speed – but the data are only held in memory, whereas the DataBaseProducer and LatestProducer are slower but once data are published, they are safe.

2.1.13 asks for deletion of log records. R-GMA does this as explained in the text relating to the Logger requirement 2.1.2.

2.1.15 asks for specialized LogStreams. R-GMA offers different types of Producer. R-GMA used to offer a CircularBufferProducer but this was withdrawn, as it is hard to reason about what data exists with such a data structure. Globally ordered logs are no problem. All R-GMA records carry a timestamp. This is computed automatically if not specified, but the user may specify his own timestamp for high precision measurements.

5. Discussion

The main problem relating the OGSA Logger to GMA and R-GMA is the R in R-GMA. It assumes that a LogStream is logically a set of tuples. The mediator inside R-GMA relies upon the simplicity of the relational model to be able to reason about which set of producers to contact to answer a query. The OGSA Logger document assumes Xpath filters for reading and writing to a log stream. The R-GMA team are very aware of the need to be able to cope with XML data. XML implementations of GMA (X-GMA) that parallel R-GMA, by using XML databases instead of RDBMS and Xpath/XQuery instead of SQL are indeed desirable. In the absence of such implementations today, R-GMA offers a good starting point for building the functionality of a Logging System on top of GMA. One way would be to restrict the XML such that a relational mapping is feasible. Another approach would be to pick out the time stamp and maybe a few more fields (any which occur exactly once, and store the rest as an XML string). This would probably give the worst of both worlds

Deleted: ~~One way would be to restrict the XML such that a relational mapping is feasible. Another approach would be to pick out the time stamp and maybe a few more fields (any which occur exactly once, and store the rest as an XML string). This would probably give the worst of both worlds.~~

At this stage a question remains: what changes would be required to the OGSA Logger description to make it conform to (R-)GMA?

This short note has just addressed the Logger System, however it can be seen that basic services derived from our Information and Monitoring services (and GMA) can have a big impact upon OGSA, if other services make use of those basic information and monitoring services.