Optimizing bulk data transfers using network measurements: a practical case

A. Ciuffoletti[c], L. Merola[a], F. Palmieri[a], S. Pardi[b], G. Russo[a]

[a]University of Napoli Federico II – Napoli, Italy
[b]INFN-Napoli Section - Italy
[c]University of Pisa – Largo B. Pontecorvo – Pisa, Italy

Abstract

In modern Data Grid infrastructures, we increasingly face the problem of providing to the running applications fast and reliable access to large data volumes, often geographically distributed across the network. As a direct consequence, the concept of replication has been adopted by the grid community to increase data availability and maximize job throughput. To be really effective, such process has to be driven by specific optimization strategies that define when and where replicas should be created or deleted on a per-site basis, and which replicas should be used by each job. These strategies have to take into account the available network bandwidth as a primary resource, prior to any consideration about storage or processing power. We present a novel replica management service, integrated within the GlueDomains active network monitoring architecture, designed and implemented within the centralized collective middleware framework of the SCoPE project to provide network-aware transfer services for data intensive Grid applications.

Keywords: *Data Grid, Replica Optimization, Network-aware Grid.*

1 Introduction

An increasing number of high-performance scientific and industrial applications, ranging from tera-scale data mining to complex computational genomics, weather forecast and financial modeling, are starting to take great advantages from the mesh of geographically distributed computing, network and data management resources, commonly referred to as “Grid”. Most of these applications face the problem of sharing large amount of data between geographically distributed organizations and their storage resources. Moving such data may be expensive and can lead to incoherence problems when it is updated and hence data access optimization becomes one of the key features of the modern Grid systems, as the

transfer and access to huge amounts of stored information are often the bottleneck of the data intensive tasks. A slow network limits the efficiency of data transfer regardless of client and server implementation. Unfortunately, in current computational Grid deployments, the underlying communication network infrastructure is considered as a pure facility and the middleware components act agnostically with respect to network performance parameters, in particular during data transfer. In this situation the users are obliged to select by themselves the best storage resources where to put or get their data files they need to analyze. The total absence of any aid or support to the data transfer activity from the Grid infrastructure brings the overall system to work globally below the best effort threshold. Thus, the main challenge is to assist the users and Grid application in getting the right data from the right location, at the right time. Also, we need the system to be efficient not only from the perspective of the performance, but also from user's point of view. Such strategy can significantly reduce data access latencies and improve system robustness and scalability by working according to a locality principle. In this paper we present a simple and effective network optimization framework implemented within the Grid framework realized for the SCoPE project and fully integrated within the GlueDomains [1] active network monitoring architecture, in order to determine the best location of a data to be used in file transfers among the storage elements available on the Grid. The decisions are obtained by a cost-estimation model, based on a set of very simple and not invasive network measurements, used to introduce network aware features in a legacy applications. Such framework works off the shelf with minimal administrative effort, is reliable, and has a negligible impact on system operation. The service's architecture, has been designed to be highly scalable to support the needs and requirements of diverse user communities participating to the SCoPE Grid, concerned with a large number of data intensive eScience application like HEP. Our prototype implementation, compliant with recent technological trends in grid community, and hence based on the web service paradigm and open source technologies, has been evaluated through a set of real world experiments on the SCoPE metropolitan Grid infrastructure and provided promising results.

2 The SCoPE project

SCoPE (Italian acronym for high Performance, Cooperative and distributed System for scientific Elaboration) is a high performance computing project that aims at developing several applications in the field of fundamental research. It has been co-funded by the Italian Ministry for Education and Research (MIUR), under the EU PON program, on a competitive base, according to the 1575/2004 tender. The SCoPE architecture has been conceived to provide the unification of all the main computational and storage resources already available in the participating sites, located in the Naples urban area, by creating an open and multidisciplinary distributed computing infrastructure based on the state-of-the-art Grid paradigm.

The Grid's connectivity is supported by a metropolitan multi-ring shaped optical fibre network that offers high performance communication facilities to the four main university research sites. The SCoPE infrastructure supports many research groups involved into different application field. Two excellent examples of applications that require high performance data transfer are: the area of particle physics and Experiment for Gravitational Waves Detection.

3 The need for network measurements

The LHC experiments of high energy physics, as ATLAS, or CMS are dominated by an enormous data acquisition rate. The CMS, for instance, plans to manage an event rate of 100Hz, corresponding to a data rate of 100MB/s that must be analyzed. Another example is the Virgo Experiment for gravitational waves detection that is characterized by a data acquisition rate of 10MB/s 24h/day, that must be analysed and replicated on geographical scale from experiment Tier 0 to Tier 1, and to Tier 2 upon request. The peculiar requirements that characterize this kind of applications are challenging the Grid development community to improve the middleware component in order to be network aware. Such feature is often discussed in literature with reference to meta-scheduling algorithms. The related works propose new resource brokering strategies that consider availability of computational, storage and network resources, whose target is the optimization of appropriate cost estimators. Another interesting topic is related with replica optimization. File catalogue services like LFC represent with a logical name a set of physical file locations, in order to select the more convenient replica when needed. Many strategies are proposed that aim to improve the use of the file catalogue services by quantifying network costs. All the above experiences confirm that the successful application of such paradigm relies on the availability of network measurements. Hence a specialized infrastructure is required, that provides information about network performance.

4 The GlueDomains network monitoring architecture

On-line Network monitoring is essential for the correct functionality of our optimization model. We need to understand the usage patterns of our network infrastructure and its performance both in real-time and historically to enable the network as a managed, effective and robust component of our Grid infrastructure. The *GlueDomains* prototype service [1], supporting the domain-based INFN Grid network monitoring activity, is used for this sake. The above service is supported by a dedicated central server node, placed in the main MSA Campus Grid site. The network monitoring layout consists of a partitioning of the network monitoring end-points (hereafter the *Edge Services*) into *Domains*: this significantly contributes to its scalability (point above). Such partitioning takes into account the fact that monitoring will in fact aggregate and report domain-to-domain

measurements: therefore domains should be designed so to include edge points that have a uniform connectivity with the rest of the network. Such assumption significantly simplifies the design of the network monitoring layout, while matching a wide range of use cases simply considering a DNS based domain partitioning. Network monitoring is carried out through a number of network monitoring tools, whose location ensures that collected network measurements are significant; the hosts that support the activity of network monitoring tools are called *Theodolites*. One simplifying hypotheses is that they are co-located with one of the Edge Services in each Domain. Such option is valid since the network monitoring activity has a low footprint, and reduces the number of hardware devices dedicated to network monitoring. The activity of the theodolites is coordinated by a centralized database: each theodolite periodically fetches the description of its activity from the database, possibly modifying its activity. The overall activity of the network monitoring infrastructure is represented by a number of network monitoring sessions. Each Theodolite is composed of several monitoring Sessions that can be classified into Periodic and On Demand. The attributes of Periodic and On demand sessions allow the control of the monitoring activity: tool specific configurations are an attribute of a Session. Database updates are considered infrequent events, typically related to a change in the membership of theodolites. The database server offers a web service interface to the theodolites, in order to make the access to the database more flexible and is therefore transparent to the technology used to implement the database. Fault tolerance features of the theodolite cope with transient failures of the server. The software component that manages the data transfer to the publication engine is another pluggable entity: data are presented to that plugin as Perl packed data on a pipe. The plugin is in charge to flush the pipe periodically and forward the data to the publication engine of choice. Thus the theodolite, per se, has a very low footprint, since it does not support any complex operation or storage on collected data. The plugin should operate according to the same principles. In fig. 1 we sketch a functional view according to the above description:

- topology database, that stores and makes available to users the description of the Grid partitioning (e.g. which Domain contains a computing service);
- monitoring database, that describes the planned monitoring activity in the Grid;
- production engine, that makes available the statistics about Grid communication and operation services (e.g., packet loss between computing and storage services);
- producers, the Theodolites, which query the monitoring database and the topology database to configure their network monitoring activity. Observations are published through the production engine
- consumers, that find the domain they belong to, as well as those of other services of interest, querying the topology database. Observations are retrieved using the production engine.

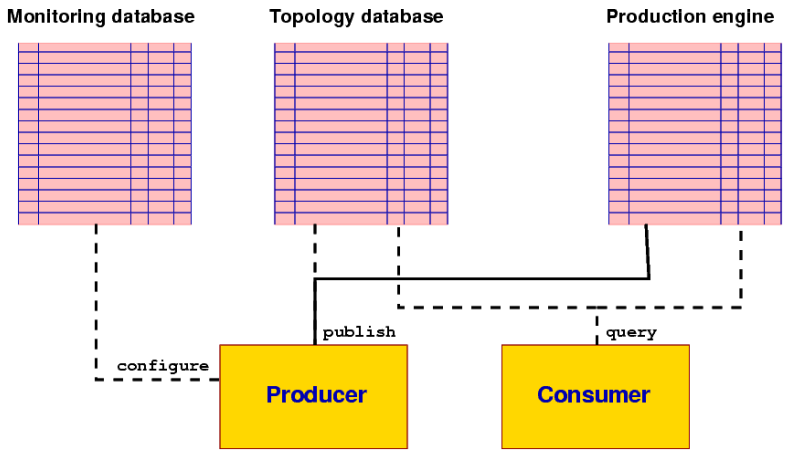

Fig. 1. Modular architecture of a GIS: dashed lines represent read-only access

As explained above, GlueDomains architecture is somewhat open: only the interface of some of its components is in fact specified. In order to adapt such architecture to our purposes, we customized the three pluggable components: monitoring database, tools, and publication engine. As for the monitoring database we reused an existing implementation based on a MySQL database. Although such database might be replicated, mainly for performance reasons, the scale of our experiment did not justify database replication. The domain database is implemented using a set of rules that extract the domain from the DNS. Therefore there is not a real database, but a set of agreed syntactical rules. We used GridICE for data publication, an MDS based tool extensively used in the INFN network. The architecture of the network monitoring host is divided into a GlueDomains and an MDS specific part. The first one is composed of a hierarchy of processes:

- **[GlueDomains]** is a daemon process that controls the whole monitoring activity of the host. It spawns the processes that implement the theodolite services. The description of the theodolite services is obtained querying the monitoring database hosted by the GlueDomains server, each time a theodolite service is spawned. The query returns the list of all theodolite services that are associated with any of the IP addresses of the host.
- **[Theodolite]** is a process that implements a theodolite service. It spawns --- and re-spawns when needed --- all monitoring sessions associated with a theodolite service. The description of all sessions associated with the theodolite service is retrieved from the monitoring database. The identifier of the monitored Network Service is retrieved from the Topology Database, given the identifier of the theodolite and of the target associated with the session. It may initialize the publication of the network services measurements.

- **[Session]** is a process that implements a monitoring session. All parameters that configure a specific session are passed from the theodolite process, so that the session should not need to access the monitoring or topology databases.

The right part of the network monitoring host in fig. 2 is MDS-specific. The design of the R-GMA specific part is simpler, and is omitted. A flow of LDIF entries is generated by the functions provided by the MDS adaptor. Such flow is re-ordered and buffered by an *MDSproxy* process, which runs as a daemon. Periodically, the buffer is flushed by the GRIS host by invoking through ssh the GDIP command.

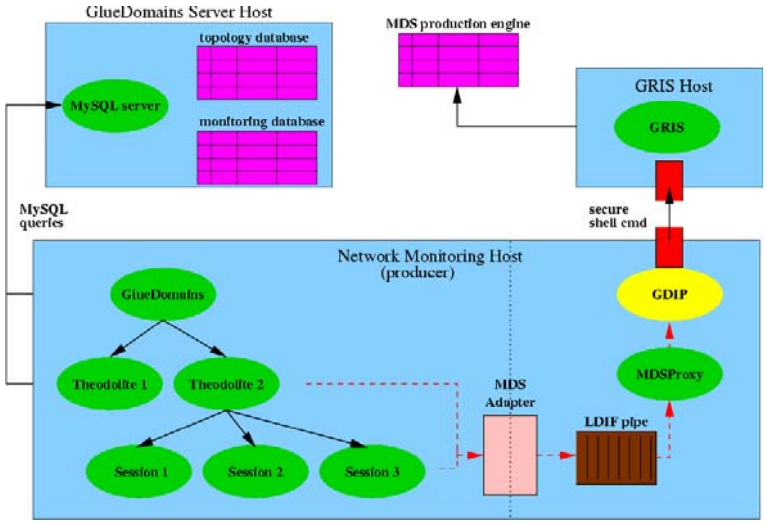

Fig. 2. Prototype architecture (with MDS adapter)

5 Service objectives and architecture and Implementation details

When network connection facilities are considered as resources to be managed on the Grid it becomes necessary to specify exactly what is meant for these resources, how to encapsulate them into Grid services and how to manage these services. A Grid service is a self-contained, self described application that can be published, located and invoked over a network through properly designed interfaces. The replica optimization service is implemented as a light-weighted Web service that is used to select the best replica respect to a specific Computing or Storage Element, identified from its Site URL, of a dataset which will be needed as input to one or more jobs. Such dataset, identified by its logical file name, is considered to be an atomic unit of data, defined within a Virtual Organization (VO). The implemented network-aware replica management logic gathers network status information from the GlueDomains monitoring service and performs access optimization calculations based on this information. Specifically, it provides

optimal replica information on the basis of both faster access and better network performance characteristics to minimize the transfer time and network costs.

5.1 Network-aware replica management

The replication decision process retrieves available replica's information from the Grid catalogs. It is able to locate datasets as well as individual physical files (depending on the information available on the underlying catalogue). Furthermore, a dataset itself can consist of several physical files but the end-user normally only knows the dataset concept. Due to the distributed nature of the Grid, a slow network limits the efficiency of data transfer regardless of client and server implementation. One optimization technique to get the best replica from different physical locations is by examining the available bandwidth between the requesting computing element and various storage elements that hold replicas. The best site will be the one that has the minimum transfer time to transport the replica to the requested site. The LFC catalogue is searched in order to find all the locations where the requested dataset is available, obtaining a list of all the already available replicas. The replica decision process then evaluates the network costs for accessing directly each physical replica location in the above list. For this it uses complex estimations of dataset transfer times based on network monitoring statistics and if the resulting network cost is greater than a properly defined tolerability threshold the available copy having the minimum access and transfer cost is selected and a new replica is created on a specified Storage Element. To perform the best replica selection, the previously obtained list of dataset locations is sorted by the datasets "accessibility" which is provided by the network cost and network features such as bandwidth, packet loss etc, and by the "closeness" determined by some network round-trip time or other ICMP-based network measurements. More precisely, the above accessibility and closeness criteria are defined by using a simple model that estimates the network cost using very basic network measurements according to the DIANA approach [2]. The Mathis's formula [3] allows the estimation of maximum TCP throughput by using the historical measurements of the packet loss and RTT with the following equation:

$$MaxBand < \frac{MSS}{RTT} \times \frac{1}{\sqrt{loss}}. \qquad (1)$$

Where MSS is the maximum segment size, RTT is the observed Round Trip Time and loss is the packet loss rate during the period of observation. Starting form this approximation we can calculate the network Cost with the following equations:

$$netCost = \frac{Losses}{MaxBand} \text{ and } Losses = w_1 RTT + w_2 Loss + w_3 \mid Jitter \mid \qquad (2)$$

The weights w_1, w_2 and w_3 can be chosen based on the different characteristics of the involved data transfer operations. An higher weight indicates the importance of that parameter in the cost calculation so that we can manipulate

these weights to prioritize particular parameters in the formula. A higher RTT indicates that a computation site is distant from the storage site where the data resides and therefore the cost to fetch the data would increase. Moreover, jitter is of less importance for data intensive applications and typically there is no significant impact on the cost due to a higher or lower jitter. The same is the true for the packet loss: a higher packet loss implies a less reliable network, and we should give less importance to such a connected site when making replica selection. In any case, the value obtained enables the user to estimate the overall performance provided by the different network links in best effort regime. Once the best candidate to replication, in terms of minimum network cost, has been selected from the list of available replicas, a copy operation is issued from the involved storage elements and the new replica is registered on the Grid catalogue.

5.2 Service interface

Our replica optimization service contains all the logic to coordinate the underlying services, providing the user with a unified interface to all replica management functionalities. A Web Services-based abstract interface has been deployed within the higher level middleware to offer significant flexibility and ease o use to application developers. Accordingly, the top-level service interface of the proposed replica optimization service has been deployed as a centralized web service within the SCoPE collective services framework. Communication between the GRID applications and the top level service interface takes place via SOAP/HTTP (eventually secured by SSL) using well-defined extended WSDL Grid Web service interface. Requests and responses conform to Web Services specifications, i.e., they are XML/SOAP messages, carried in HTTP envelopes and transported over TCP/IP connections. The web service interface has been developed by using the *nusoap library* [4], providing a set of PHP classes that allows developers to create web services based on SOAP, WSDL and HTTP. Each replication request takes a logical file name (LFN) and the physical file name or SURL of a destination SE as input parameters and chooses the best replica as described above. For the given LFN, all the available physical replicas are identified and the best one in terms of minimal data transfer cost is used as the source file for further replication.

6 Performance and results analysis

In this section, we report the results of the tests ran for performance analysis sake on five major sites of the SCoPE DataGrid testbed by performing optimized replication of sample files. All the involved sites are connected each other through the dark fiber metropolitan area ring, where the involved link capacities vary from 2.5 Gbps POS STM-16 to the single or multiple Gigabit Ethernet. Thus, the goal of the above performance tests was twofold: we verified the consistence of the network cost calculation model and demonstrated that the replica optimization

strategy is really able to improve significantly the data transfer performances. In order to analyze the behaviour of the cost calculation model, we evaluated the *netcost* function between different sites both during a period of low traffic (normal network condition), and under an heavy network load by gathering the information available from their respective theodolites. During an entire night of sampling we observed an almost constant behaviour in the normal traffic scenario (network unloaded or slightly loaded) and, as we expected, a noticeable growth of the netcost value during the heavy network activity. More specifically, regarding the network cost trend between the UNINA-SCOPE-ASTRO and INFN-NAPOLI-VIRGO sites, in 12 hours of observation we found an average netcost value of 6.3 with 0.8 standard deviation, during the normal network activity. On the other side, during the different phases of the heavy network workload we measured a significant growth of the *netcost* value with its maximum in the order of 10^7. The growth of the netcost value when network congestion increases is also strongly related with the measured jitter value. In fig. 3 below we showed the netcost behaviour varying in time under normal and heavy load, calculated between the UNINA-SCOPE-ASTRO and INFN-NAPOLI-VIRGO sites' theodolites. As it can be seen from the above trends the network cost function behaviour, allows us to assert the full validity and consistency of the DIANA model in our environment. The second experiment consists in downloading in the storage element of the INFN-NAPOLI-VIRGO site, a set of 100 1.2 GB sized files that are replicated in the UNINA-SCOPE-ASTRO, UNINA-SCOPE-GSC, UNINA-SCOPE-CEINGE and UNINA-SCOPE-CENTRO sites. The files are registered in the SCoPE logical file catalogue: this allows us to have an single namespace for all the replicas distributed along the different storage elements. The tests have been deliberately run during an intensive synchronization activity among the above sites, and have been split in two phases:

1. Replication of 100 files on the INFN-NAPOLI-VIRGO site by using the lcg-utils tools and the logical file name.
2. Replication of 100 files on the INFN-NAPOLI-VIRGO site aided by the network-aware replica optimization service

During the first test sequence, the lcg-cp command, after querying the LFC catalogue, starts downloading data randomly from the available sites. In the second sequence of tests, data is downloaded using the SURL address returned by the replica optimization service by selecting the "best" replica location based on the network cost measurements gathered through the GlueDomains theodolite interface. We measured an average transfer time on 100 files of 154 seconds for each file by using the lcg-utils versus 145 seconds by obtained by using the replica optimization services. In fig. 3 below we show the replica optimization services effects in reducing the data transfer time. We can note that an average gap of 9 seconds per file is a very good result on a Metropolitan dedicated fiber network, with 1 Gbps or 2.5 Gbps capacities per-link. This result allows us to expect very

interesting improvements when applying this model in a geographical scale with significantly slower links.

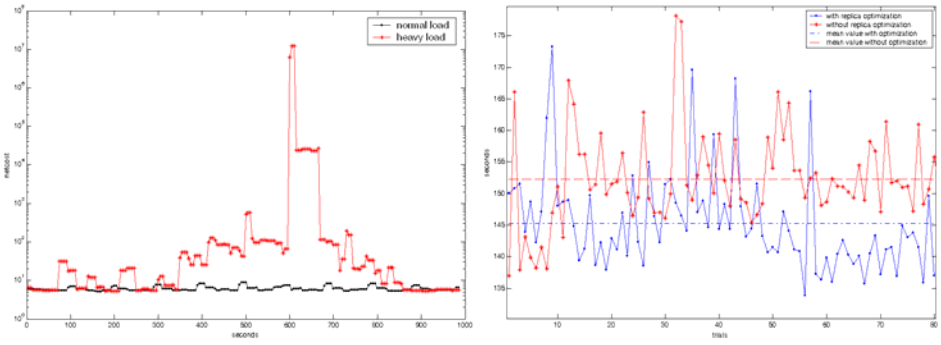

Fig. 3. Netcost and data transfer time (80 files) between UNINA-SCOPE-ASTRO and INFN-NAPOLI-VIRGO

7 Conclusion

We have presented the design and implementation of an high level replica management tool providing the functionality required for efficient data access in a Grid environment. We have discussed in detail our optimized replica selection mechanism that takes into account network monitoring performance data in order to identify the optimal network connection with respect to the required data transfer operation. The experimental results show that our approach significantly reduces wide area transfer times and enhances the overall performance of the involved jobs. The encouraging results that have been obtained suggest us to investigate the possibility of acquiring more accurate bandwidth and Quality of Service measurements in order to improve decision accuracy

8 References

[1] Ciuffoletti, A. et al.: Architecture of monitoring elements for the network element modeling in a grid infrastructure. In Proc. of Workskop on Computing in High Energy and Nuclear Physics, 2003.

[2] McClatchey, R. et al: Data Intensive and Network Aware (DIANA) Grid Scheduling, Jouunal of Grid Computing, DOI 10.1007/s10723-006-9059-z.

[3] Mathis, S. Mahdavi, O.: The macroscopic behavior of the TCP congestion avoidance algorithm. Computer Communications Rev. 27(3), pp. 62–82, 1997.

[4] NuSoap - SOAP Toolkit for PHP http://sourceforge.net/projects/nusoap.

[5] Cameron D., Casey et al.: Replica Management in the EU DataGrid Project, International Journal of Grid Computing, 2(4):341-351, 2004.