

Bridging Network Monitoring and the Grid

Tommaso Coviello^{*}, Tiziana Ferrari^{*}, Kostas Kavoussanakis^{**}, Loukik Kudarimoti[†], Mark Leese[§],
Alistair Phipps^{††}, Martin Swany^{§§}, Arthur S. Trew^{**}

^{*}Istituto Nazionale di Fisica Nucleare, Italy

[†]DANTE, United Kingdom

[§]CCLRC, Daresbury Laboratory, United Kingdom

^{§§}University of Delaware, USA

^{††}National e-Science Centre, The University of Edinburgh, United Kingdom

^{**}EPCC, The University of Edinburgh, United Kingdom

Abstract. Several network monitoring frameworks have been successfully deployed in various network domains, but they typically expose proprietary interfaces and feature poor interoperability. In addition, frameworks either focus on end-to-end or on domain edge-to-edge measurements. Consequently, they can only address a partial spectrum of Grid performance characteristics. As Grids are typically large-scale infrastructures spanning multiple network domains, the seamless view that is paramount to Grid middleware and operators, can be severely hindered. We analyse the requirements of important Grid middleware components: job scheduling and data transport. We propose a set of services for the publishing, discovery and gathering of information about network performance and framework capabilities. These expose interfaces that follow the Global Grid Forum recommendations. Finally, we review the implementation details of a prototype of the proposed architecture.

1. Introduction

Network performance monitoring has traditionally been important to the operation of networks of any significant size as an aid to fault detection and for determining expected performance.

The need for such monitoring is enhanced in the Grid computing arena. It is required for:

- Grid middleware and applications to optimise their performance by selecting the *best* resources for computing jobs, and adapting to changing network conditions;
- debugging networks for efficiency, an essential step in supporting data intensive applications because simple over-provisioning is not sufficient;
- the Grid to be *self healing*, and apply the Service Level Agreements associated with the *utility computing* model.

It is the first of these requirements which this paper addresses. Over recent years many excellent network monitoring toolkits and frameworks have been developed, several specifically targeted at Grid environments. While unquestionably useful, these frameworks have often been geared to operating within single administrative domains or focused on end-to-end measurements which require regular tests to be run between all sites involved in a Grid collaboration. This presents a problem to the operation of the flexible Virtual Organisations (VOs) which Grids facilitate. This is well highlighted if we consider a project such as Enabling Grids for E-Science (EGEE) [1], which spans more than thirty countries and over 150 sites.

In addition, developers have focused on development of the frameworks and toolkits themselves, leaving less time for defining mechanisms by which network performance data can be published to the Grid applications and middleware which can make use of it.

This paper focuses on the work of EGEE and its partners in bringing network performance monitoring to the Grid. We first describe the background of how Grid middleware can make use of network performance data, before analysing requirements in more detail. We then propose a network performance monitoring architecture which provides Grid middleware with a single interface from which to obtain network performance data. Although expressed in the context of the EGEE project, the principles can be readily

applied elsewhere. Section 2 outlines Grid Middleware use cases, using the real life examples of the EGEE Workload Management System (WMS) [2] and Data Scheduler (DS) components of gLite [3]. We also briefly consider the needs of Grid operations. Section 3 sees us analyse the use cases to identify a broad set of requirements, both in terms of the WMS and DS, and of Grid operations. Section 4 contains a brief overview of the high-level architecture of the proposed solution, while Section 5 outlines the EGEE Network Performance Monitoring (NPM) prototype implementation, consisting of Mediator, Publisher and Diagnostic Tool components. The Mediator provides an interface to Grid Global Forum (GGF) Network Measurements Working Group (NM-WG) compliant monitoring frameworks, which is in turn used by the Publisher to provide performance data to Grid middleware (Publisher) and by the Diagnostic Tool to visualise performance data for Grid operations. Section 6 looks to further work. The paper concludes in Section 7.

2. Grid Middleware Use Cases

Grid middleware can enhance its performance by collecting and using information on the status and behaviour of the underlying infrastructure. Various network metrics can be made available to the middleware so that a comprehensive and up-to-date picture of the Grid network status can be constructed. Network metrics can be used for Grid middleware optimization and dynamic adaptation in a number of scenarios. In the following sub-sections, focus is on the usage of network performance information for job scheduling and data transport.

2.1. Job scheduling

Grid scheduling allows the execution of applications to distributed resources. This functionality, supported for example by the gLite service named Workload Management System (WMS) [2], comprises of a set of middleware components responsible for the distribution and management of tasks across Grid resources and services. Scheduling policies are defined according to internal strategies, which aim at maximizing the individual task execution efficiency, while at the same time improving the overall Grid resource usage. By means of appropriate scheduling policies, the system can use resources that offer sufficient reliability.

WMS offers several functionalities. It simplifies the Grid – user interaction. For example, resources are discovered by the WMS to meet the user requirements, and application resubmission is attempted in case of execution errors. Logging and Bookkeeping facilities, which are provided to allow job debugging and tracing, add to the robustness of the Grid infrastructure.

Research work on scheduling applied to distributed systems is extensive. Grid scheduling approaches can be classified according to their strategies. For example, *compile-time* scheduling produces schedules at application compile-time, and generally ignores dynamic status information about the Grid infrastructure. Conversely, *run-time* scheduling can rely on dynamic system information. Run-time scheduling can be further classified into either static or dynamic run-time scheduling. A *static* run-time schedule is defined prior to task execution and, later on, the scheduler does not have the opportunity to migrate running tasks to different resources. On the other hand, a *dynamic* run-time scheduler makes decisions that can be dynamically adapted in case of significant state changes, but is much harder to implement. Scheduling in the framework of the EGEE project adopts the static run-time approach. The focus of this paper will hence be on run-time schedulers.

The efficiency of a scheduling policy strictly depends on the type of tasks considered, such as the number of dependent tasks a given application encompasses. Many scheduling research efforts have been dealing with simplified scenarios such as individual independent tasks and/or loosely-dependent tasks. These approaches typically use either analytical methods to calculate optimal schedules, or empirical data analysis and heuristic searches to find near-optimal solutions [4], [5].

In this paper we consider computing tasks scheduling policies which aim at minimizing the tasks' time to complete execution, i.e. at calculating an optimal mapping to resources. The metric used is the estimated *completion time* of a job (also called the turnaround time), which is the time interval from job submission to

the instant when staging of the output results completes. The calculation of the best mapping is a NP-hard combinatorial optimization problem. For this reason, faster heuristic search procedures are often adopted to calculate near-optimal solutions.

Scheduling functionality and policies give rise to algorithms which can be *off-line* or *on-line*. Off-line scheduling algorithms require the *a priori* full knowledge of the input task flow. Conversely, on-line scheduling is only based on current resource status and past events, with a consequent potential loss in accuracy, as the remaining part of the flow is not taken into account. Networks are fundamental components of a distributed system, and as such, network performance information can be effectively applied to on-line scheduling. Besides the network, other Grid resources of interest are *Computing Elements* (CEs) and *Storage Elements* (SEs). The CE refers to a set of computational resources, managed by a local resource management system, where the cluster can encompass resources that are heterogeneous in their hardware and software configuration. On the other hand, the SE is the Grid service responsible for saving/retrieving files to/from some data stores, which can be any device from a hard disk to a mass storage system.

One possible approach to on-line scheduling of executables (jobs) is the preferred selection of CEs that are *logically close* to the input data they need to access. Closeness is a qualitative property. CEs and SEs are considered to be logically close if the network infrastructure offers low communication latency and high bandwidth (such as in a LAN environment). Closeness needs to be complemented by taking into consideration additional attributes such as load, as the use of logical closeness alone, can cause the submission of the job to an overloaded CE. In addition, the support of an increasing number of users, and the adoption of network-agnostic policies, can introduce network traffic bottlenecks. The availability of network performance information can consequently improve scheduling solutions.

The *Minimum Completion Time* (MCT) algorithm assigns each job to the machine which guarantees minimum completion time. MCT facilitates load balancing as resources with large available time are preferred. However, completion time is also influenced by the location of input data as transfer is needed to make data available to a worker node prior to job execution, as shown below:

$$\text{CompletionTime(CE, SE)} = \text{JobExecutionTime} + \max \{ \text{InputDataTransferTime(CE, SE)}, \text{QueueTime} \} ,$$

where *JobExecutionTime* is the job execution time foreseen on a given CE; *InputDataTransferTime* is the predicted time for the transfer of input file from SE to CE; and *QueueTime* is the expected waiting time of the job in the local batch system queue. The maximum is considered as we assume the input file transfer to take place while the job is waiting to be executed. The computation of *InputDataTransferTime* requires knowledge about the *TCP achievable bandwidth* [6] and the amount of data to be transferred.

The *InputDataTransferTime* definition involves achievable bandwidth, however more complex expressions can rely on additional metrics. For example, packet loss frequency is an additional valuable parameter, as network paths affected by non-zero transient or long-term packet-drop rates are typically unstable and can hinder the transport performance. Consequently, the WMS can neglect resources connected by paths affected by non-zero packet loss, a capability which is particularly effective when application-level data transfer re-routing is not possible. In addition, the WMS can use Round Trip Time (RTT) and IP Packet Delay Variation (IPDV) [7],[8], as indicators of standing queues and congestion, as well as of *distance* between two network points. The availability of RTT historical information can be used for trend analysis and correlation against historical data. For example, the connectivity can be evaluated by comparing the most recent RTT sample against the historical data. Paths with a RTT nearer to the historical minimum RTT are preferable. Dramatic changes in RTT can be the consequence of network topology variations, while IPDV fluctuations can be caused by incipient congestion. Consequently, the WMS can use RTT and IPDV to rank end-to-end paths, by giving higher preference to those paths with low average RTT and IPDV.

2.2. Data Transport

Network performance plays a significant role in Data Grids or Grid applications that depend heavily on data movement. For such applications, network performance is clearly a key factor in the total time to perform. The key metric for these situations is achievable bandwidth as it is inversely related to the time to move a given amount of data over the network. As with the WMS discussion above, auxiliary metrics such

as packet loss or IPDV may be used to improve or complement the achievable bandwidth metric, but the most basic consideration is the time that it takes to move data. The Data Scheduler (DS) component of gLite [3] provides for data movement with scheduling agents that initiate data transfers while considering channel policy, etc. The user can specify logical file names (LFNs) to be copied over the network. These LFNs are translated into Storage URLs (SURLs). The DS then schedules the movement of the files via some underlying mechanism such as GridFTP [9]. Achievable bandwidth can be used by the DS in a number of ways. In this section, we will consider three key scenarios.

The first is a data fetch operation when a given dataset is fully or partially replicated in multiple locations. Typically, the best location to fetch data is the place from which the achievable bandwidth is the highest, thus the time to transfer the data will be the lowest. It may also be possible to download part of the file in parallel from different sources. In this case, the transfer is bound by the maximum time to complete. Given equally-sized file chunks, this is equivalent to the time to transfer from/to the Storage Element with the lowest achievable bandwidth to the client. Consider a graph $G = \{V, E\}$, with edges in E consisting of the *inverse* of the reported achievable bandwidth and vertices in V being suitable SEs with the replicated file. This metric for the value of an edge corresponds to the *cost* of an edge, which lends itself to many traditional graph algorithms. For the simple case, the scheduling algorithm should simply choose $\min\{E\}$. In the case of multiple segments of a file (or dataset) being downloaded simultaneously, the time to transfer will be dominated by $\max\{E'\}$ where E' is equal to the set of *chosen links*.

The second scenario for data transport is that of point-to-point data transfer. In this case, a data source (for example, a known SE) must transfer data to a sink, regardless of the connection-end that initiates the transfer. Often it is the case that a transfer via an intermediate point, or via a path other than the default one, can improve the observed throughput. Again, if we consider a graph G as described above, the scheduling algorithm should choose the path from source to destination that minimizes the maximum-cost arc. The solution for this problem is a well-known algorithm known as Minimax Path [10].

The third scenario occurs when a single file or dataset is to be distributed to a set of nodes. This one-to-many data transfer is often referred to as *reliable multicast*, which may or may not use IP multicast. Given again the graph G from above, appropriate algorithm for this case is a Minimum Spanning Tree (MST) and its use for this purpose has been demonstrated in [11]. Note that the time to transfer the data through an intermediate SE can be modelled as an edge without loss of generality, although that is beyond the scope of pure network metrics.

We have considered three scenarios for data movement that can be improved via use of the achievable bandwidth metric. While metrics such as IPDV and RTT may be useful in inferring the achievable bandwidth, they are not directly of interest to the application. The benefit of this formulation is that the algorithms can be solved optimally with reasonable complexity – $O(n \log n)$ – for straightforward implementations.

2.3. Operations

As its name suggests, a Grid Operations Centre is the Grid equivalent of a Network Operations Centre, responsible for overseeing the operation of a particular Grid infrastructure. Their tasks involve monitoring the availability and performance of Grid resources and services, and investigating problems reported by users. As Grids are heavily dependant on their underlying networks, the GOCs work spans the Grid and network domains, and it could be said that GOCs fulfil a liaison role between Grid users and network providers.

Whether a problem is detected during routing monitoring, or reported by a user, the GOC is tasked with deciding if the problem is network or Grid related, as this decides the ownership of the problem. This is best illustrated with a very simple use case. If the GOC's current systems raise an alert that all Grid jobs at a given Grid site are failing, an NPM tool is used, specifying the time range and sites of concern. If the data show any obvious changes in network performance (e.g., available bandwidth has dropped), the NOC or other appropriate networking bodies are involved for some investigation. Conversely, if there are no obvious changes in network characteristics, the fault can be assumed to be Grid related, and logging information from the Grid middleware and applications is considered.

GOC's requirements concern the NPM metrics which are available to them, and how those metrics are presented. These requirements are briefly discussed in the next section, while the requirements of one such GOC are analyzed in more detail in [13].

3. Use Case Analysis

The Grid middleware use cases in Section 2 detail the applicability of network performance data to resource scheduling, transport optimization and operational monitoring. This section analyzes the related requirements and issues in terms of metrics, security enforcement, data exchange and analysis, and communication performance.

3.1. Metrics

Data transfer from SE to CE in the WMS and SE to SE in the DS takes place using GridFTP [9]. As GridFTP is a TCP-based protocol, TCP achievable bandwidth is the primary network characteristic of interest. An important parameter for data transport is the number of parallel streams that maximize the GridFTP transfer rates. Another controllable parameter is the *duration* of the data transfer, which affects the value (particularly for short transfers) due to the dynamics of TCP's slow start mechanism.

If it were possible to get an accurate value for TCP achievable bandwidth between a source and destination at a particular time given a specific number of measurement streams and the total transfer time, then TCP achievable bandwidth would be the most important parameter required by the middleware. Unfortunately, TCP achievable bandwidth is costly to measure and saturation of network links with active probe data is a concern. Therefore, there is interest in estimating this metric through a combination of other, less costly to measure parameters that can be collected more frequently. Lower level metrics such as RTT and Round Trip packet loss come into the picture here.

The duration of a file transfer is not the only relevant quantity to be determined. Other metrics, such as loss, offer complementary information. For example, they provide information about the reliability and stability of the network path connecting the two end-points at a given time. Similarly, RTT allows identification of short-distance paths. This is relevant as performance is difficult to maximise over high-bandwidth, long-distance paths; specific TCP implementations may be required to effectively use available network resources and flows tend to be penalized as congestion window adaptation depends on the timely delivery of acknowledgements. Short-distance paths are therefore often preferable. Available bandwidth can be determined with minimal (or null) load on the network, and in many scenarios it provides a good indicator of TCP achievable bandwidth. Finally, IPDV can be used as an indication of network load and consequently, low IPDV paths are preferred.

Since GOCs support Grid middleware they share an interest in TCP achievable bandwidth, RTT and IPDV, and add to that other metrics which can affect those three, namely:

- Route information, as typically measured by *traceroute*: changes of route, particularly those involving a change in the number of hops, can have a profound affect on performance.
- Metrics affecting general connection quality, including One Way Delay (OWD) and packet loss.
- Basic connectivity metrics: the middleware's interest in bandwidth, RTT and IPDV makes the assumption that the resources and services which they wish to access are actually available for use. GOC's often adopt responsibility for monitoring that this is the case by additionally requesting basic connectivity metrics.
- Network topology: GOCs wish to be able to associate the above metrics with a network topology.

It should be noted that GOCs will likely wish to obtain up-to-the-second metrics via on-demand tests, yet this is a current area of network monitoring research and no suitable implementations yet exist.

3.2. Middleware Functional Requirements

Grid middleware requires retrieval of network performance monitoring data for a specified set of paths defined using source and destination CE or SE IP addresses. However, the retrieval of data for a specified set of path types (e.g. SE to CE paths) without providing specific IP addresses, should be possible. In particular, data should be available for bidirectional measurements on both paths between all SEs and all CEs, and on paths between all SEs.

Historical data is relevant, but the most recent 24 hours are typically the most interesting time window. Raw data should be provided, while custom statistical functions are not required in general. On the other hand, with regards to measurement methodologies, the possibility to request a specific approach is important, but if a specific measurement methodology is not defined, results with the most appropriate one need to be provided.

Performance and query latency are paramount due to the large number of query sessions that can be generated by a middleware client. For WMS the goal is to provide a response within 0.2 s of receiving a request which includes 100 paths. For this reason, data should be accessible via a Web Service for use of access, but only if this does not compromise performance. For Web Services, the Simple Object Access Protocol (SOAP) standard [14] is used to specify how the messages are being passed between services, and the Web Service Definition Language (WSDL) is used to specify the interface a service exposes. From a WSDL document a client or application will know how to bind to a service.

Different access methods are acceptable, such as direct database access, if deemed adequate. Alternatively, a notification mechanism combined with information caching within the client can be adopted. This allows the client to subscribe to measurement data for a set of paths and have new matching data automatically sent, and also to unsubscribe from receiving data when required. A more comprehensive list of Grid middleware requirements can be found in [15].

3.3. Security and Policies

Network measurement data can be sensitive. Security and policy issues are less serious when data is being measured by applications which are part of the Grid community. However, if the providers of measurement data are external to the Grid community (for example, backbone networks), disclosure policies have to be defined and agreed upon by the relevant partners. Security measures need to ensure that data remains safe; this requires the usage of security mechanisms both by Grid middleware and providers.

Since measurement data providers and the Grid middleware may endorse different authentication and authorization approaches, interoperability needs to be assured in terms of security implementation and representation of user credentials. This situation is further complicated due to the heterogeneous security mechanisms being used by different measurement data providers. Grid middleware will need to be aware of the mapping procedure to be used for each of the measurement data providers contacted.

3.4. Communication

Communication protocols are required not only to retrieve measurement data from all sources but also to ask for measurements to be made and to dynamically discover capabilities of measurement providers, among others. The *Global Grid Forum Network Measurements Working Group* (GGF NM-WG) [16] has been at the forefront of defining XML schemas for such protocols. However, due to the very nature of such work, multiple versions of such protocols are present.

Measurement data providers tend to choose the version that is most suitable to them. This leads to a situation where different types of protocols and different versions within these protocols are used. Grid middleware is required to communicate using as many protocols and related versions as possible. As with security and policies, communications require the Grid middleware to learn about the protocols supported, and to translate between them.

Fortunately, this will be a short-term problem. There are currently two broad versions of the NM-WG schemas, and while Grid middleware and operations software should currently understand both versions, organisations are working to stabilise on the latest version.

3.5. Data Analysis, Processing and Presentation

The Grid community defines the semantics of measurement data required for its applications. This format might not be the same as the one that is collected from measurement systems and data providers. Data analysis and processing is needed. Such processing can range from simple derivations to complex aggregations using statistic models and concatenation of measurements. Some of these are listed below:

1. Grid middleware may have an interest in derived metrics such as the "closeness" discussed in section 2.1. And while Grid network monitoring systems can provide the latency and bandwidth data upon which closeness is based, they do not provide closeness itself as a network metric. Processing of NPM data, whether by the NPM system or more likely the middleware, would therefore be required to generate closeness data.
2. Backbone networks may provide data for each hop in the path. GOCs will likely require a complete picture for a domain. This involves concatenating measurements for different hops based on time.
3. GOC operatives are Grid-aware but typically non-network experts, and therefore require information to be presented in a high-level manner, such as aggregated statistics.
4. In terms of basic analysis, GOCs would also like to set defined tolerances for monitored data which raise alarms when exceeded.

While the middleware will process performance data internally, GOCs require the data to be presented to them in a variety of formats. Matrices, tables and graphs of performance data are all popular, as are weather-maps (displays of network status plotted against geographical maps). Presentation is however, largely an issue that can be dealt with within GOC's monitoring applications, such as the Diagnostic Tool (Section 5). The crucial step in this work is to make the requested metrics available to GOCs.

3.6. Discovery of Data Providers

In a distributed system, such as the one being discussed, where services can change their state of availability, a discovery mechanism is required to help the Grid middleware get information about available measurement data providers and their capabilities. This kind of discovery mechanism can either be maintained by the Grid middleware alone or in conjunction with the measurement data providers. Both options require the discovery mechanism to keep up-to-date information about measurement data providers and their capabilities.

In the case that Grid middleware only uses its own discovery mechanism, several approaches are possible. The simplest approach is to be informed of data providers and their capabilities in a static way. The list of capabilities can be updated dynamically through periodic queries. Alternatively, providers can expose information about themselves (*advertisement*) and their capabilities to the discovery mechanism and then periodically provide updates, if any, about such information.

While the static option in the above list is the simplest, in an environment where the list of available measurement data providers can vary dynamically, this solution is inadequate. Conversely, the dynamic option requires interoperability between the middleware discovery mechanisms and the measurement data providers. Also, measurement data providers need to support both the capability data query interface and the advertisement interface.

In the case that Grid middleware exploits the discovery mechanisms available on the measurement providers' side (for example, the PerfSONAR Lookup Service [17]), the Grid middleware's discovery mechanism will only need to contact the measurement data provider's discovery mechanisms. All other complexities are handled by the measurement data provider's discovery mechanisms. The protocols that need to be used for gathering such information would, in comparison, be fewer.

Discovery mechanisms can be implemented with or without caching. While both solutions require protocols as mentioned earlier, a 'without-caching' solution can simplify the process of collecting data, but at the cost of efficiency. Conversely, a 'with-caching' solution, where information about state and

capabilities of data providers are cached in the discovery mechanism, would require the use of reliable data caching mechanisms.

4. Architecture

Given the use cases and related requirements exposed above, we propose a multistage client-server architecture consisting of four major blocks, as shown in Figure 1. The architecture is founded on the premise of standardising the interfaces between the entities involved.

The *Publisher* provides network measurement data as required by the Grid middleware components described in Section 3.2 (the Data Scheduler and Workload Management System). The Publisher is aware of the format and type of data expected by the Grid middleware. The Publisher is also aware of the protocol to be used for querying Network Performance Monitoring (NPM) services.

Diagnostic Tools (DT) support users in diagnosis of network related problems. Different tools can provide different views of network information, targeted at different types of user. For example, a tool can provide the information about network status and measurements required by Grid Personnel, to meet their requirements discussed in Section 2.3. The DTs are at the same level as the Publisher, and like the Publisher, they are aware of the protocol to be used for querying NPM Services.

The *NPM Services* provide the capability to retrieve network measurement data from various data providers. Such services include discovery, security, translation, aggregation and data caching. NPM Services contact the network measurement data providers in order to retrieve the measurement data required by its clients (the Diagnostic Tools and the Publishers).

A *Measurement Data Provider* consists of a set of applications providing network measurement data. The NPM Services are aware of these data providers and also of the data that these providers offer. NPM Services are also aware of the protocols that need to be used in order to communicate with these providers, including the security mechanisms required.

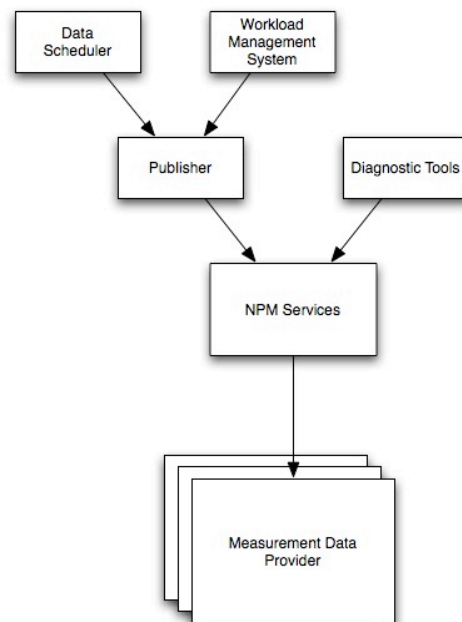


Figure 1: High-Level Architecture

This multistage approach means each stage need only interact with the stages immediately above and below. For example, the clients of NPM Services – the Publisher and Diagnostic Tools – do not need to be

aware of the specific application set used in a Measurement Data Provider, or even the specific Measurement Data Providers that are available, as the NPM Services expose the measurement data to clients in a uniform way.

5. Prototype

In the sections which follow, we detail implementation solutions we propose for the various architecture components in Figure 1.

5.1. Prototype NPM Services

Taking the above requirements and issues into account, the EGEE JRA4 activity has defined the high level architecture and developed prototypes in its efforts to produce components for NPM Services. Many networks such as GÉANT, Internet2, ESNNet and others provide Web Service-based access to some network measurement data. The objective of developing these components is to be able to retrieve data from these providers as well as other applications such as EGEE’s e2emonit [18] and make them available to Grid Middleware and GOC users.

While standardisation of the interfaces has been the main focus of these prototypes, usability, some security aspects and discovery methods have been targeted as well. The internal structure of the prototypes developed and the interaction with target users is depicted in Figure 2.

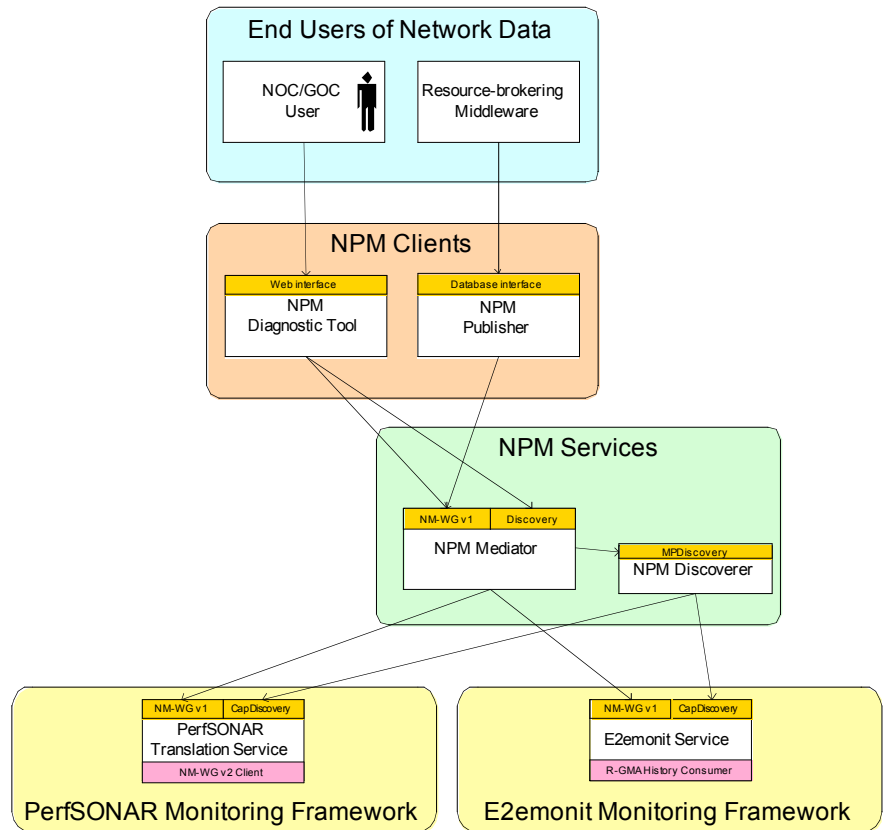


Figure 2: Prototype Components and Users

The Prototype architecture consists of data providers (also called monitoring frameworks), core NPM services (the Mediator and Discoverer) and clients of the NPM services for both NOC/GOC users and resource-brokering middleware (including the WMS and DS). NM-WG-based interfaces are used for

communication between these components. These interfaces are draft standards defined for XML-based communications (e.g., SOAP [14]). Since NM-WG efforts began in 2002, two main versions of these schemas have been released. Data providers use different versions and sub-versions of these schemas. The current NPM Services are standardized on version 1 of this schema, but aim to communicate with as many such data providers as possible.

The *Mediator* provides a NM-WG-based interface for measurement data retrieval and a second Discovery interface to retrieve measurement framework data. The Mediator communicates with the data providers directly to retrieve data and it communicates with the Discoverer to learn about the existence and capabilities of measurement data providers. The internal Mediator structure is shown in Figure 3.

Grid applications require a short response time. Taking into consideration the number of dependencies involved in getting measurement data, caching is required at many levels. From discovery data to measurement data response, caching components have been added into the architecture to cater for these needs. *Data Manipulators* and *Aggregators* take care of any data-processing requirements.

An object model has been defined and used in the Mediator to allow business logic to be represented in a schema-independent form. *Mappers* have been engineered, which convert between the object model data containers and different schema versions (such as NM-WG v1). If there is a need for translation to a new protocol or a different version, only an additional Mapper component must be created. This concept of Mappers has been used in Translation services (discussed later) as well.

Not all the security features required are currently part of the Mediator architecture. However, some of these features, such as the Secure Sockets Layer for secure communication and use of X.509v3 certificates for authentication – have been integrated into the prototype.

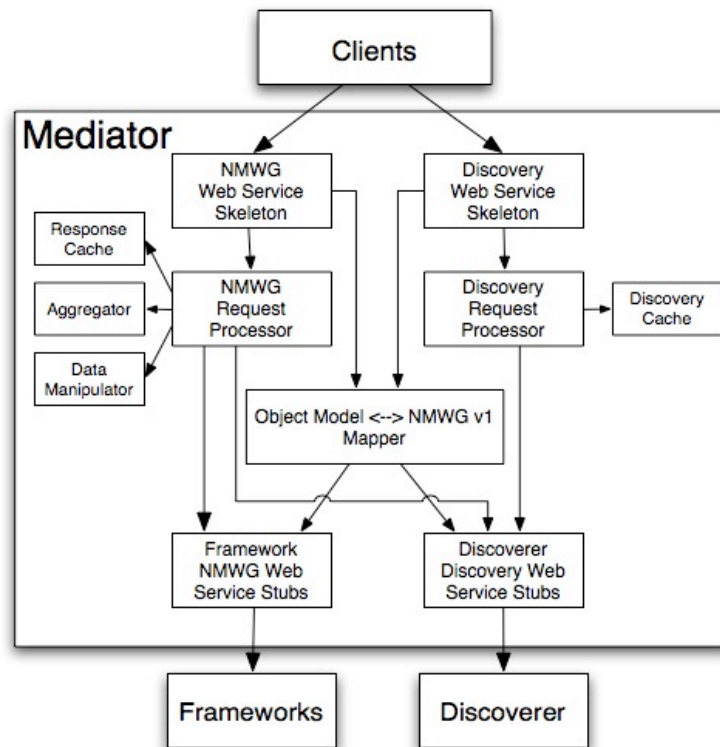


Figure 3: Mediator Design

The *Discoverer* collates information about existing frameworks and provides an interface for the Mediator to access this information. Currently, the protocols in use for the discovery method are designed in-house, using XML schemas to meet specific requirements. We propose the use of GGF NM-WG interfaces in the future (supported by the schema's version 2 onwards) here as well. The discovery information is currently configured statically, but we propose the use of a dynamic discovery mechanism.

The *Translation services* act as proxies and translate protocols to retrieve capability and measurement data from frameworks that do not natively support the NM-WG schema version upon which the NPM Services are standardized. Depending on translation needs, new translators can be created, allowing transparent use of a monitoring framework. Translators make use of the standard data containers and the Mapper concept previously discussed. New types of Translators can be easily created using this Mapper concept. Figure 2 shows a Translation Service for the PerfSONAR [17] monitoring framework, which translates from NM-WG v1 to the incompatible NM-WG v2 schema used by PerfSONAR. It also shows a Web Service for EGEE's e2emonit monitoring framework, natively supporting the NM-WG v1 schema.

5.2. Publisher

Although a matchmaking service such as the WMS could retrieve network performance monitoring data directly from the Mediator's web-service interface, it would not be possible to satisfy the time constraints required for matchmaking operations (the WMS requires a maximum latency of 200 ms) due to the overheads involved in a Web Service call, such as network latency, data transfer, marshalling and security. If the data were not immediately available in the Mediator response cache – a likely situation for very recent data – a Web Service invocation by the Mediator to a measurement framework would be required, delaying the response still further.

The Publisher acts as a client of the Mediator and bridges the gap between network data sources and matchmaking services by pre-caching data for the required set of network paths and metrics in local database storage, as shown in Figure 4. The Publisher is deployed locally to matchmaking services, allowing rapid retrieval of data by the matchmaking services through shared database access. To avoid excessive storage requirements on a site, the local database storage is limited in size by removing monitoring data older than a predefined age, typically 24 hours. Although multiple Publishers will frequently be polling a single Mediator for the same data, the Mediator's response cache minimises overhead on the frameworks.

As network performance monitoring data is gathered in terms of paths between Network Monitoring Points (NMPs) but matchmaking operates in terms of CE and SE addresses, a mapping is required from a CE/SE address to a NMP address. The Publisher provides information on this mapping in another table in the same database, allowing for rapid joins between mapping and network performance monitoring data, hence providing network performance monitoring information between CE/SE addresses.

A WMS prototype client that makes use of network performance metrics is available. In this prototype, we require that input files be staged to the selected cluster prior to job execution time.

The prototype is based on the GlueDomains framework [19]. Performance data is centrally collected and retrieved in XML format. Information is refreshed at roughly ten minute intervals. The WMS obtains network data asynchronously and stores it in a local file, which is used as a temporary WMS cache for matchmaking operations. This cache is important in order to meet the 200 ms time constraint on matchmaking operations.

The WMS normally uses a different repository, the Information Supermarket, in order to collect and cache other types of Grid information. A different cache is used for networking data for safety reasons, as the amount of information needed grows quadratically with the number of network end-points to be considered. Due to such overheads for storage, it is currently under investigation whether a common framework for caching could be used by both the WMS and DS.

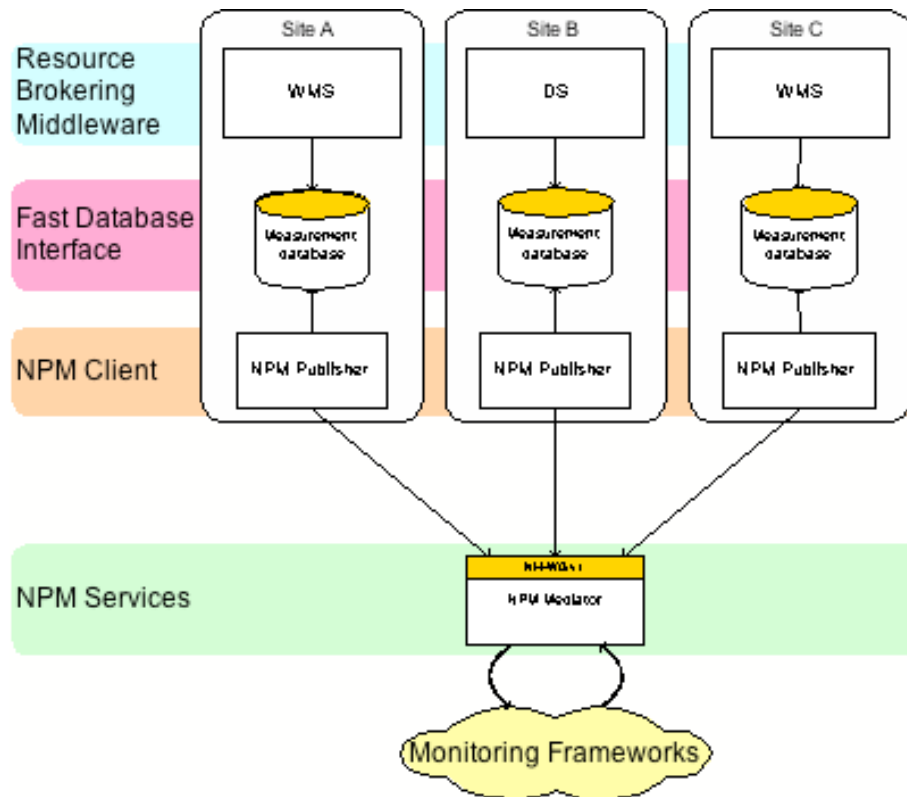


Figure 4: Publisher Architecture

5.3. Diagnostic Tool

The NPM Diagnostic tool (DT) [13], shown in figure 5, can be considered as a graphical interface for the Mediator, although it is in reality a separate product using the Mediator interface to obtain data from a diverse set of (NM-WG compliant) infrastructures for presentation to human users. It is aimed at assisting Grid and Network Operations Centres in diagnosing performance problems by presenting performance data in a range of formats, including tables and histograms.

The DT is accessed via a Web browser and so requires no specialised software to be installed locally, while access is protected via X.509v3 certificate authentication and white-list authorization. Users enter data in the following order, before data is displayed:

1. Select time range, either as a fixed start and end time, or as a single time around which +/- time tolerances preferentially select data
2. Optionally define the maximum number of results to return
3. Select the source of a test path. Based on information from the Discoverer, the DT will reveal a list of valid endpoints defining test paths for which it can retrieve data. It is possible to select multiple test paths.
4. Select the metric to retrieve from a list, which is again retrieved from the Discoverer. The available metrics are dependent on the monitoring infrastructure which is known to test that path. For example, typical metrics for a backbone path are link utilisation and capacity, and available bandwidth, while it is RTT, one way loss and achievable bandwidth (amongst others) for end-to-end paths.
5. Select data type, either raw data or a statistical type such as minimum, maximum or mean.
6. Select the presentation type: table, matrix or graph.

Note that the flexibility of the queries is made possible by the use of the NM-WG schemas, and this same scope is therefore open to queries from the Grid middleware.

Both the Mediator and Diagnostic Tool were demonstrated to good effect in October 2005 at GGF15 and at the 4th EGEE conference, where the DT was used to graph performance data retrieved via the Mediator from the Abilene and ESNet US research and academic backbone networks, their pan-European companion (GÉANT2), and the EGEE e2emonit end-to-end monitoring infrastructure. This was an important step, demonstrating the retrieval of data from multiple administrative domains using a unified interface.

The Diagnostic Tool has also been a useful demonstrator for GÉANT2 operator DANTE at RIPE 51 and other meetings, showing live data from PerfSONAR being used by the EGEE project.

The work contrasts with, yet complements that of network operators such as DANTE and Internet2 due to its focus on end-to-end performance, and presentation of data to end-users, GOCs and Grid middleware.

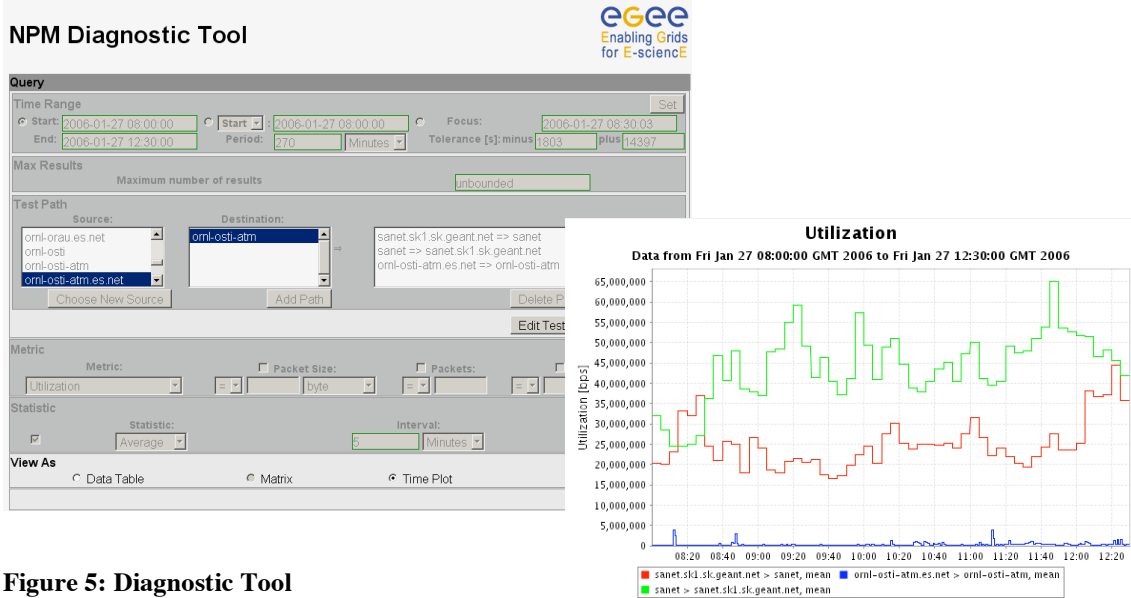


Figure 5: Diagnostic Tool

6. Future work

The work described in this paper is bounded by the current EGEE project, which runs until March 2006. It is hoped however that such efforts will continue in EGEE-II.

We have previously described the Data Scheduler (DS) component of gLite, which facilitates data movement within the Grid. In scenarios where there are a number of potential data sources (replicas of the required dataset) and/or data sinks (suitable SEs to be used for data storage) the DS requires a means of evaluating the suitability of each associated network path. There is consequently the clear need for a service to carry out network cost function evaluation for a set of paths, with the evaluation based on historical network performance data and knowledge of other transfers known to be scheduled within the required time period or our own transfer. This service should subsequently recommend a path.

There is also a requirement that the service should in large be network metric agnostic, or rather, independent of which network metrics a measurement framework can provide. A range of cost functions should instead be available, so arbitrary input can be accepted and a relevant cost function selected based on the supplied metrics. Initially, this would be an experimental service, with basic interfaces defined, yet allowing experimentation with different cost functions.

Further, the cost to move data is all that matters to Resource Brokering middleware, which is almost always time-based. QoS services such as Premium IP [20] may introduce a monetary value in the future, but this remains to be seen.

In summary for the DS, a network cost function evaluation service is required which will facilitate evaluation of several cost functions, whether time or funds-based.

We have also outlined the Grid scheduling service of gLite, the Workload Management System (WMS), which controls the distribution and management of tasks across Grid resources and services. The future here is less clear, as the Workload Management group, though sharing the need for cost function evaluation, reason that the cost function has such a strong impact on scheduling policies and the efficiency of the resource brokering function of the WMS, that cost function evaluation should remain within the WMS. This would allow the Workload Management group to change the cost functions as needed, and reduce the dependency between the NPM and WMS systems. This is a matter for further investigation and discussion, yet the differing views could be very beneficial if they encourage the various data transfer, Grid scheduling and network monitoring parties to work more closely, something which until recently had not happened as often as required to best utilise the available skills and technologies.

For Grid operations, the next big goals are the support of on-demand tests and the mapping of network topology. Mapping network topology is currently restricted by a lack of both reliable sources of topology information and a standardised and widely acceptable textual means of representing that information. We must investigate how these problems can be overcome.

Access to measurement data from as many network providers (backbone, regional, campus, etc) as possible would arm the GOC with all the necessary information in order to be able to contact the appropriate NOC when problems arise. Therefore, there is a need to get as many network providers on board this effort as possible.

7. Conclusion

The benefits of the Grid have attracted e-Scientists from various disciplines to use large-scale, distributed infrastructures. As an example, EGEE, the biggest Grid in Europe, hosts applications from areas as diverse as Earth Sciences, High Energy Physics, Bioinformatics and Astrophysics, currently offering over 17,000 processors in 179 sites across the world.

Access to these resources requires access to well-managed, well-performing networks. Equally important to network health is a constant, consistent view of its status. This is important to the Grid middleware, and by inference to the applications, as it allows efficient scheduling and efficient execution of computing jobs, especially when data movement is required.

Research in Network Performance Monitoring initially focused on development of new monitoring frameworks and infrastructures, and sharing data was often driven by a desire to share data between network operators. Leveraging the work of the GGF Network Measurements Working Group, EGEE instead focused on a common, standard interface, which will allow the networks to operate their desired tools, while making their data available to the middleware and the Grid network operators irrespective of the deployed NPM frameworks.

Having worked closely with EGEE's WMS team as well as network operators, we arrive at an architecture which can exploit our standard interfaces so as to provide NPM data to the middleware and to the Grid operators. We have produced prototypes which allow access through a single, standard interface to heterogeneous NPM data, emanating from heterogeneous NPM frameworks. At GGF13 (March 2005) our prototypes demonstrated the first-ever such access, retrieving data from EGEE's e2emonit framework and GÉANT's perfmonit [21] framework.

8. Acknowledgements

Francesco Prelz and many colleagues from the EGEE JRA1 Workload Management team assisted us during the requirement gathering phase and provided valuable input about enhancements of existing WMS scheduling policies.

Mathieu Goutelle has provided valuable insight to the EGEE JRA4 prototypes. Robert Stoy has solicited Diagnostic Tool requirements from the DFN NOC. The EGEE-JRA1-UK team has accommodated our R-

GMA requests and participated in the e2emonit testbed. Our colleagues in CNRS have led GARR and the University of Edinburgh, and also the EGEE-JRA1-UK team in the set-up of the e2emonit testbed.

This work is supported by the EU project EGEE, sponsored by the European Union under contract number INFSO 508833; by the INFN project INFN-GRID; and by the UK Joint Information Systems Committee (JISC).

References

- [1] *Enabling Grid for E-science* (<http://egee-intranet.web.cern.ch/egee%2Dintranet/>).
- [2] Andreetto, P.; Borgia, S. et al.; *Practical Approaches to Grid Workload and Resource Management in the EGEE Project*; in Proc. of the Conference on Computing in High Energy and Nuclear Physics (CHEOP 2004), Interlaken (CH), Sep 2004, Vol. 2, pp. 899-902.
- [3] *EGEE Middleware Architecture and planning*, EGEE Project Deliverable EGEE-DJRA1.1-594698-v1.0, Jul 2005 (<https://edms.cern.ch/document/594698/>).
- [4] Krauter, K.; Buyya, R.; Maheswaran, M.; *A taxonomy and survey of Grid resource management systems*; Software Practice and Experience; Vol. 32 (2), pp. 135-164, Feb 2002.
- [5] Casavant, T., L.; Khul, G.; *A Taxonomy of scheduling in General-Purpose Distributed Computing Systems*; IEEE Transactions on Software Engineering, Vol. 12 (2), pp. 141-154, Feb 1988.
- [6] Lowekamp, B.; Tierney, B.; Cottrell, R.; Hughes-Jones, R.; Kielmann, T.; Swany, M.; *Enabling Network Measurement Portability Through a Hierarchy of Characteristics*, 4th International Workshop on Grid Computing (Grid2003), Nov 2003. (also appears as GGF GFD-R-P.023).
- [7] Almes, G.; Kalidindi, S.; Zekauskas, M.; *A Round-trip Delay Metric for IPPM*; RFC 2681, Sep 1999.
- [8] Demichelis, C.; Chimento, P.; *IP Packet Delay Variation Metric for IP Performance Metrics (IPPM)*; RFC 3393, Nov 2002.
- [9] Mandrichenko, I.; Allcock, W.; Perelmutov, T.; *GridFTP v.2 Protocol Description*; GFD.47, May 2005.
- [10] Swany, M.; *Improving Throughput for Grid Applications with Network Logistics*, Proceedings of IEEE/ACM Conference on High-Performance Computing and Networking (SC2004), Pittsburgh, PA, Nov 2004.
- [11] Swany, M. and Wolski, R.; *Network Scheduling for Computational Grid Environments*; in Proc. of the workshop on Adaptive Grid middleware (AGridM) 2003 (in association with IEEE/ACM Parallel Applications and Compilation Techniques 2003), Sep 2003.
- [12] Leese, M.; Stoy, R.; Phipps, A.; *Diagnostic Tool Requirements*; EGEE Technical Report (<https://edms.cern.ch/document/593620/>).
- [13] *NPM Diagnostic Tool User Guide* EGEE Technical Report (<https://edms.cern.ch/document/653967/>)
- [14] *SOAP Version 1.2 specification* (<http://www.w3.org/TR/soap12>).
- [15] Phipps, A.; *NPM Use Cases and Requirements of Resource-Brokering Middleware*; EGEE Technical Report (<https://edms.cern.ch/document/672906/>).
- [16] *The Network Measurements Working Group*, GGF (<https://forge.gridforum.org/projects/nm-wg>).
- [17] *The PerfSONAR project* (http://monstera.man.poznan.pl/jra1-wiki/index.php/Main_Page
<http://www.geant2.net/server/show/nav.754>).
- [18] *E2emonit* (<http://marianne.in2p3.fr/egee/network/download.shtml>) and
<https://edms.cern.ch/document/672382/>
R-GMA: Relational Grid Monitoring Architecture (<http://www.r-gma.org/>).
- [19] Andreozzi, S.; Ciuffoletti, A.; Ghiselli, A.; Vistoli, C.; *Monitoring the Connectivity of a Grid*; in Proc. of the 2nd International Workshop on Middleware for Grid Computing (MGC 2004) in conjunction with the 5th ACM/IFIP/USENIX International Middleware Conference, Toronto (CA), Oct 2004.
- [20] Campanella, M.; Ferrari, T.; Leinen, S.; Sabatino, R.; Reijs, V.; *Specification and Implementation Plan for a Premium IP Service*; GÉANT project (IST-2000-26417), Deliverable D9.1, Apr 2001.
- [21] Ubik, S.; Smotlacha, V.; Simar, N.; *Performance monitoring of high-speed networks from the NREN perspective*; Terena Networking Conference.
<http://staff.cesnet.cz/~ubik/publications/2004/terena2004.pdf>