European Research Network on Foundations, Software Infrastructures and Applications for large scale distributed, GRID and Peer-to-Peer Technologies
A Network of Excellence funded by the European Commission

Network Monitoring Session Description

Augusto Ciuffoletti
augusto@di.unipi.it
INFN-CNAF
Via B. Pichat - Bologna - ITALY

Antonis Papadogiannakis, Michalis Polychronakis,
{papadog,mikepo}@forth.ics.gr
FORTH
Heraklion - Crete - Greece

CoreGRID Technical Report
Number TR-0087
June 27, 2007

Institute on Grid Information, Resource and Workflow Management Services

CoreGRID - Network of Excellence
URL: http://www.coregrid.net

CoreGRID is a Network of Excellence funded by the European Commission under the Sixth Framework Programme

Project no. FP6-004265

Network Monitoring Session Description

Augusto Ciuffoletti
augusto@di.unipi.it
INFN-CNAF
Via B. Pichat - Bologna - ITALY

Antonis Papadogiannakis, Michalis Polychronakis,
{papadog,mikepo}@forth.ics.gr
FORTH
Heraklion - Crete - Greece

CoreGRID TR-0087

June 27, 2007

Abstract

Network Monitoring is a complex activity that is based on the coordination among distinct local activities: those that use monitoring results, those that implement monitoring activity, and other that act as intermediaries.

We illustrate a comprehensive view of a Network Monitoring Architecture, starting from the demands of security and scalability, and define, to give a practical perspective to our investigation, the XML Schemas of the information exchanged between actors.

We address scalability by introducing monitoring sessions activated on demand, with a declared preference for passive tools, and security by enforcing authenticated communications at every step. A scalable protocol for public key diffusion is introduced in a companion paper.

1 Introduction

We focus on the information exchange related to the Network Monitoring activity. The actors involved are the producers of monitoring data, and the consumers. In a workflow monitoring activity, consumers are parts of a complex architecture that manages the tasks submitted by users: we call such distributed activity *Workflow Management* (here including also the monitoring activity successive to task allocation) and Workflow Management Agent (also WMA) the local agents that cooperate in its implementation.

While *allocating the resources* for user tasks, the interest of such agents is for snapshots of recent performances as well as static capabilities of resources (in this case, Network Elements[1] candidate for supporting user tasks). In a reservation oriented system, resource allocation can be virtually completed without any information coming from monitoring activity. While *running* a user task the behavior of the resource must be permanently monitored, in order to guarantee an appropriate quality of service and for accounting purposes.

Such considerations narrow our interest to a subset of what is often considered as Network Monitoring: we exclude the maintenance and transfer of pointwise historical traces, and consider that the monitoring activity is anticipated by a request from a WMA. Therefore we do not consider the existence of a *repository* for network observations, and we are

This research work is carried out under the FP6 Network of Excellence CoreGRID funded by the European Commission (Contract IST-2002-004265).

[1] the term Network Element in this paper is more restrictive than than in QoS literature, for instance RFC2216

marginally interested to the availability of generic aggregated statistics of dynamic behaviors and of static properties of network elements. Instead, we concentrate on the transfer of streams of observations from producers to WMA.

On the side of the distributed functionality in charge of producing Network Monitoring data, we introduce specialized agents (the Network Monitoring Agents, NMAs) in charge of implementing local functionalities.

Such agents are located according to a domain partitioning of the whole Grid: each partition, a *Domain* in our terminology, is a set of Grid components characterized by a uniform connectivity with the rest of the system. Such abstraction is often used in the Internet architecture, therefore we have opted for an overloaded term to indicate it. However, it is worth stating that a Network Monitoring *Domain* does not necessarily correspond to a DNS domain, or to a routing AS or area. Equivalence with such existing entities can be stipulated whenever non contradicting the principle of uniform connectivity.

Such principle justifies the collection of aggregate statistics and of static capabilities (whenever needed) for network elements between domains, thus limiting monitoring activity. As anticipated, such information is mainly directed to task allocation, and we consider that such activity should be preferably addressed using a reservation protocol.

The rationale behind the introduction of NMAs is the localization of the capabilities and of the workload related to network monitoring. NMAs act as proxies for addressing monitoring requests, and manage the streaming of monitoring data for the whole domain.

Each domain may contain one or more NMA, which may be responsible for the observation of distinct Network Elements. They are responsible for the control of the Network Monitoring Elements located inside the *Domain*. Network Monitoring Elements (NMEs) represent resources provided for monitoring the network using appropriate techniques. Figure 1 illustrates such concepts in a simple system consisting of three domains (large ovals labelled with the domain ID), each with a NMA (a small circle on the border of each oval). Two NME are included in domains “FORTH” and “INFN-CNAF”, while the other domain “INFN-NA” contains a WMA.

We distinguish two basic techniques for Network Monitoring: active and passive. For the sake of scalability we prefer the latter, although the former should be provided as a fallback solution. For instance, in case of a simple request of connectivity monitoring between two sites the option of a slow ping should be provided in case passive monitoring is not available. Other scenarios, and especially systematic monitoring, should address passive techniques.

Network Monitoring Elements should accept controls only from local Network Monitoring Agents, thus enforcing a secure control over the expensive activity of traffic monitoring. In their turn, Network Monitoring Agents should accept requests only from peer Network Monitoring Agents, as well as from local Workflow Analysis Agents: requests from external WMA should be authenticated by peer Network Monitoring Agents.

Such complex architecture is based on a piece of data describing a single instance of Network Monitoring activity, that we call Network Monitoring Session. This paper is devoted to the exhaustive description of such piece of data. The next section outlines the Network Monitoring operation, and describes the data needed for its operation.

2 The operation inside the Network Monitoring Agent

The purpose of a Network Monitoring Agent is to monitor the performance of the communication resources used to carry out the computational tasks coordinated by the Workflow Analyzer module. More precisely, we distinguish four distinct activities:

- accept (*proxying*) network monitoring requests coming from Workflow Analyzers providing the description of the monitoring activity. Such request may come either from a WMA inside the same *Domain*, or from another NMA. In either case the request must be authenticated.
- route the request to another Network Monitoring Agent which is able to control an appropriate Network Monitoring Element;
- coordinate the monitoring activity carried out by Network Monitoring Elements;
- support the streaming of Network Monitoring data to the requesting Workflow Monitoring agent, possibly through other NMAs.

We outline such activities, paying special attention to the data needed to perform them, which will be the subject of the next section.

In the case of *proxying*, a Workflow Management Agent that coordinates a given computational activity will produce a number of *Network Monitoring Session Descriptions*. Each of them is in charge of monitoring the activity induced by the computation itself.

As for the *request routing* activity, the Workflow Management Agent will forward such requests to the local Network Monitoring Agent, which will authenticate the request, and forward it to the appropriate Network Monitoring Agent.

Here we do not detail how such request is routed, but consider that this operation is based on the accessibility of a database containing *Network Monitoring Agents Descriptions*. Such descriptions should locate an agent inside a domain, thus defining its monitoring capabilities, as well as its connectivity towards other NMAs.

The *control of Network Elements* requires a local knowledge of the Network Monitoring capabilities available on local Network Monitoring Elements, and of their interfaces.

In order to support the *streaming* of Network Monitoring results, a data channel is built between the Network Monitoring Agent in charge of coordinating the monitoring session and the NMA hosting the Workflow Analyzer. In principle such path may traverse several Network Monitoring Agents, and should consider the possibility to optimize the path in case the same information is requested by many different Workflow Analysis tasks.

In conclusion, we have identified 3 data structures:

- a *local directory* that supports authentication of requests from WMAs in the local *domain*, as well as the description of local Network Monitoring Elements interfaces;
- a *global directory* that supports authentication of NMA from other *domains*;
- a *network monitoring session description* which contains the description of a single session.

While the design of the *local directory* does not address any challenging aspect, the other two have distinct reasons of interest from a research point of view. The implementation of a *global directory* implies the solution of a number of problems concerning distributed processing, while the *description of a monitoring session* should flexibly cope with the diversity of network monitoring requests.

Here we focus on the latter problem, addressing the reader interested in the former to specific articles [2].

In order to exhaustively describe such data structure, and to provide a practical hint for implementation, we opt to describe such data structure as an instance of an *XML Schema Description* document.

3 The XML schema of a Network Monitoring Session

The complex type `NetworkMonitoringSessionType` is the frame for a monitoring request: it describes a specific monitoring activity related to a given aspect of a workflow, and inherits some of its characteristics from the workflow itself: namely, its scheduling and the involved network resources. Attributes are the following, and are a sort of header for the Session Description:

SessionId it is a way to identify and refer to a session. Its syntax can be constrained into a URI-like form using an appropriate pattern, which is not considered here;

Elements are a more composite description of the monitoring activity, which is described as a sequence of elements with a complex type.

RequestFrom The workflow management activity that requests the activates the monitoring session. We consider that several agents contributing to the management of a given workflow may be all interested in receiving the same monitoring data, and therefore admit a broadcast of limited span. This information is used to generate or extend the multicast tree, as well as to check privileges. The complex type that describes such element contains two attributes that represent identifiers: one globally unique for the NMA (**NetworkMonitoringAgentId**), and another locally unique for the specific NMA task (**TaskId**);

```
<schema
 xmlns="http://www.w3.org/2001/XMLSchema"
 xmlns:sched="http://www.di.unipi.it/~augusto/schema/Schedule-0.1.xsd"
 targetNamespace="http://www.di.unipi.it/~augusto/schema/Schedule-0.1.xsd">

 <annotation>
  <documentation xml:lang="en">
   Network Monitoring Session Scheduling indication
   Copyright CoreGRID. All rights reserved.
   Version 0.1
  </documentation>
 </annotation>

<complexType name="NetworkMonitoringScheduleType">
  <attribute name="StartAt"
                type="dateTime"
                use="required"/>
  <attribute name="Duration"
                type="duration"
                use="required"/>
  <attribute name="BandwidthLimit"
                type="nonNegativeInteger"
                default="0"/>
  <attribute name="Priority"
                type="sched:priority"
                default="bestEffort"/>
</complexType>


<simpleType name="priority">
  <restriction base="string">
    <enumeration value="guaranteed"/>
    <enumeration value="elastic"/>
    <enumeration value="bestEffort"/>
  </restriction>
</simpleType>

</schema>
```

Table 1: Schedule namespace

Schedule It is a complex type element which in its turn contains two elements: one specifying the starting time of the activity **StartAt**, and another specifying its **Duration**. These two elements are derived from the workflow scheduling, while the others refer to the resources associated to the session. One is represented by the **Bandwidth** element allocated to the stream associated with the return of monitoring data. This may be seen as an implicit indication of the precision with which a certain data should be represented. Another is the **Priority**, which describes to what extent the absence of monitoring activity impacts workflow management. We envision three coarse grain categories: *guaranteed* when part of workflow specifications (e.g., the purpose of the workflow is network monitoring), *elastic* when temporary unavailability is not an issue (e.g., monitoring data is used for buffer sizing), *bestEffort* when the workflow can proceed even if the session is not activated (e.g., monitoring data is used for runtime resource optimization). For the sake of future extensions, we have introduced a separate namespace for its definition (see table 1)

Route The indication of the route the stream is going to follow, represented as a stack of NMAs, indicated by their Id in **Agent** element and ordered by an **Index** element. The case study in section 5 exemplifies its management;

NetworkElement A single session monitors one single domain-to-domain pair. The opportunity of aggregating the monitoring of several Network Elements within the same Session appears as of moderate interest, and complicates session management. The element contains two Domain identifiers, one for the **SourceDomain**, another

for the **DestinationDomain**. Note that the meaning of source and destination depends on Tool Wrappers semantics (see below);

MeasurementStream This element contains the operational definition of the low level network monitoring activity. Such data should be passed to the back-end supported tool, which results in the production of a stream of data of known content and syntax. The next section analyzes the content of such element;

We opted to indicate one single Network Element in accordance to the fact that a given session is implemented by a single Network Monitoring Agent. It is impossible to guarantee such fact if several Network Elements or tool activations are allowed to produce a single stream.

Advanced passive network monitoring tools that are able to observe distinct characteristics of traffic flowing between given endpoints should incorporate such data into the same stream.

We get into the description of the complex types that are instantiated into a Network Measurement.

4 MeasurementStreamType

The MeasurementStreamType is where the monitoring tools is indicated and configured. As a general rule, a single frame in the stream will contain several numerical values produced (quasi) synchronously within the same session. The option that one tool produces several streams is therefore supported.

Requests returning a single data chunk are considered as "singleton" streams, and indicated with a 0 Duration.

A MeasurementStreamType element therefore contains a collection of **CharacteristicStream** elements, each dedicated to the configuration of a specific tool. An attribute **CharacteristicStreamId** attaches an identifier to each element, while a number of elements describes tool configuration:

SamplePeriod this is the granularity of the time axis, in seconds;

Path a complex type that describes the sub-network under test. Here we assume that the sub-network simply consists of a pair of IP addresses. A more detailed description is beyond the scope of the present paper, and a direction of future research. For the sake of expandibility and flexibility we have introduced a separate namespace for its definition (see table 2)

In addition, the **CharacteristicStream** includes one in a set of elements containing either the controls specific for a given network monitoring tool or the description of the requested characteristics, according to OGF recommendations.

When the **CharacteristicStream** includes the configuration of a *monitoring tool*, the element is intended to configure the activation of a given monitoring tool that applies a known methodology in order to assess the performance of the sub-network. In fact, any tool that applies the same **or methodology** can be used to collect the data. In this case we do not consider abstract *characteristics* (for instance *roundtrip time*) but make explicit reference to the operational description of its computation. In other words, a ping is a ping, and not a roundtrip time. The WMA is free to use it as a roundtrip time, but it cannot confuse it with a roundtrip time measured during a TCP connect (which is not simply a *protocol* difference). The use of a trade mark (e.g. linux-ping) is OK, but in many cases a more abstract reference to the methodology used to measure it (e.g., ICMP ping) is preferable. The tool wrapper may accept both a tool specific name and a methodology to indicate the same operation. The WMA may indicate either a methodology or a tool specific name, and the NMA should not interfere with such indication. Descriptive statistics (historical average, stddev etc.) are indicated as distinguished CharacteristicId's.

When the **CharacteristicStream** indicates a set of well known characteristics, the monitoring tool should be activated in order to carry out a measurement that adheres to some form of standardization. We indicate the OGF recommendation as an instance.

The next subsections give simple examples of three elements of such kind: one controlling a trivial pinger, another describing the controls for a sophisticated traffic analyzer, and finally the container of an OGF compliant characteristic descriptor. These XSD documents should be included in separate namespace, in order to ensure the flexibility of the session description schema.

```
<schema
 xmlns="http://www.w3.org/2001/XMLSchema"
 xmlns:path="http://www.di.unipi.it/~augusto/schema/Path-0.1.xsd"
 targetNamespace="http://www.di.unipi.it/~augusto/schema/Path-0.1.xsd">

 <annotation>
  <documentation xml:lang="en">
   Network Path Description
   Copyright CoreGRID. All rights reserved.
   Version 0.1
  </documentation>
 </annotation>

<complexType name="NetworkMonitoringPathType">
 <sequence>
  <element name="SourceIP"
        type="string"/>
  <element name="DestinationIP"
        type="string"/>
 </sequence>
</complexType>

</schema>
```

Table 2: Path namespace

4.1 Ping options schema

The trivial ping (see the XSD in table 3) indicates an option for the length of the ICMP packet. Two distinct characteristics can be requested: the roundtrip time, and the packet loss rate. The source and the destination of the echo packets are indicated in the **Path** element.

4.2 MAPI options schema

The passive network monitoring tools that we use are based on the MAPI monitoring library [11] (see table 4).

Based on the source and destination domains, and optionally on the protocol name and the type of a specific application, which are already indicated in the `NetworkElementType`, we can filter the traffic that we are interested in. The `ProtocolName` element can be any network protocol at the transport layer (such as TCP and UDP) while `ApplicationName` may correspond to any Grid-related application (such as HTTP, GridFTP, and Globus). The identification of a specific application in the Grid network traffic can be as simple as looking for a static port number, or more complex based on deep packet inspection, application-level protocol decoding, or other heuristics. Also, the measurement interval can be defined by the `SamplePeriod` element, that is part of the `CharacteristicStreamType`.

Some additional options for the passive monitoring tools include requests for anonymization of sensitive fields in the results (e.g., IP addresses) and use of a third host, whenever needed, for gathering and correlating the results.

Using these passive monitoring tools, the following characteristics can be requested: round-trip time [7], packet loss rate [8], available bandwidth, and per-application bandwidth usage [1].

4.3 Modularization Issues

One of the relevant aspects of a data description schema is its possibility to evolve in time, in response to technical advances and to new application frameworks. In order to keep our schema as much flexible as possible, we have designed it as a set of namespace: one is considered as the root, and characterizes our approach.

Two other namespaces are considered as somewhat simplified, and are useful in the prototype design: the *Schedule* namespace, which describes the Scheduling requirements of the monitoring activity, the *Path* namespace, that

```
<schema
 xmlns="http://www.w3.org/2001/XMLSchema"
 xmlns:pt="http://www.di.unipi.it/~augusto/schema/PingTool.xsd"
 targetNamespace="http://www.di.unipi.it/~augusto/schema/PingTool.xsd">


 <annotation>
  <documentation xml:lang="en">
   Network Monitoring Tool Ping.
   Copyright CoreGRID. All rights reserved.
   Version 0.0
  </documentation>
 </annotation>

<complexType name="PingOptionsType">
 <sequence>
  <element name="PacketSize"
               type="integer"
               minOccurs="0"/>
 </sequence>
 <attribute name="CharacteristicId"
               type="pt:PingCharacteristicIdType"
               use="required"/>
</complexType>

<simpleType name="PingCharacteristicIdType">
 <restriction base="string">
   <enumeration value="RoundTrip"/>
   <enumeration value="PacketLoss"/>
 </restriction>
</simpleType>

</schema>
```

Table 3: Trivial Ping Options

```
<schema
 xmlns="http://www.w3.org/2001/XMLSchema"
 xmlns:am="http://www.di.unipi.it/~augusto/schema/MAPIMonitoringTools-0.1.xsd"
 targetNamespace="http://www.di.unipi.it/~augusto/schema/MAPIMonitoringTools-0.1.xsd">

 <annotation>
  <documentation xml:lang="en">
   Passive Network Monitoring Tools (FORTH).
   Copyright CoreGRID. All rights reserved.
   Version 0.1
  </documentation>
 </annotation>

 <complexType name="MAPIMonitoringToolOptionsType">
  <sequence>
   <element name="ProtocolName"
                type="string"
                minOccurs="0"/>
   <element name="ApplicationName"
                type="string"
                minOccurs="0"/>
   <element name="Anonymize"
                type="string"
                minOccurs="0"/>
   <element name="ThirdParty"
                type="string"
                minOccurs="0"/>
  </sequence>
  <attribute name="CharacteristicId"
                type="am:MAPIMonitoringToolCharacteristicIdType"
                use="required"/>
 </complexType>

 <simpleType name="MAPIMonitoringToolCharacteristicIdType">
  <restriction base="string">
    <enumeration value="RoundTripTime"/>
    <enumeration value="PacketLossRate"/>
    <enumeration value="AvailableBandwidth"/>
    <enumeration value="UsedBandwidth"/>
  </restriction>
</simpleType>

</schema>
```

Table 4: MAPI options

describes the monitored subnetwork. The task of designing a comprehensive description for such aspects justifies the presence of OGF Working Groups dedicated to their investigation. We therefore prefer to leave at an embrional stage their definition, to what is strictly required in order to have a working prototype.

We leave unlimited possibility of extending our definitions for what is concerning the configuration of network monitoring activity. Aside the namespaces that exemplify the use of active and passive tools, we also introduce one that demonstrates the possibility of including tool independent characteristic descriptions. Such feature can be used in order to incorporate characteristic descriptions like those indicated by the OGF Network Monitoring working group.

5 A case study: monitoring Processor to Storage connectivity

A simple example illustrates the request of an active monitoring session over a Network Element to monitor the connectivity through an ICMP ping (see table 5).

The origin of the Network Monitoring Session descriptor is the WMA represented as a green circle inside the INFN-NA Domain (see figure 1). The WMA has no hints about the Network Monitoring Architecture, so it delivers the bare MeasurementStream to the local Network Monitoring Agent.

At this point the Measurement Stream is encapsulated into a Network Monitoring Session description, and routes the request to the known NMA at one end of the Network Element. The identifier of the forwarding NMA is placed in the route stack.

The NMA in the INFN-CNAF domain discovers that it cannot handle the request: there is no ping wrapper on the Computing element, and therefore the monitoring activity cannot be carried out. It forwards the xsd to the known NMA on the other Network Element endpoint, the FORTH, pushing its own address on the stack.

The next NMA discovers that the storage element is equipped with a ping wrapper: therefore it extracts the MeasurementStream description from the Session description, and delivers it to the Network Monitoring Element co-located with the Storage Element. It also discovers that it is adjacent to the NMA in the INFN-NA domain, and eliminates the intermediate INFN-CNAF agent from the Route stack.

The Network Element activates a ping process, formatting the data coming from such process according to its specifications, and forwarding successive datagrams to the local NMA, which in its turn encapsulates the data by indicating the session they belong to and passing it to the next NMA in the stack.

In our case this is the NMA located at INFN-NA, which decapsulates the data and passes it to the WMA, which is able to unmarshall the data contained in the datagram according with tool specifications, and process the data.

The WMA finally interrupts the monitoring session notifying the local NMA, which propagates the request according to the route stack known to it. When the request reaches FORTH NMA, it stops the monitoring activity on the computing element. Alternatively, FORTH NMA will perform the same activity when the “Duration” timeout expires. Intermediate NMA’s will suspend and remove the registration of the session from their soft state.

6 Related works

The coordination of a network monitoring infrastructure is a matter of active research. The first effort in this sense is probably the Network Weather Service [12], which still offers relevant suggestions. However, such prototype indicates but solves only partially the real challenges of a coordinated network monitoring architecture: scalability and security.

Successive studies mainly focussed towards the publication of network monitoring results in view of retrospective analysis: this option limits the application of such infrastructures to those scenarios where monitoring requests can be anticipated and concentrate on a restricted subset of paths. Without such limit any solution is deemed to unscalability, since the number of paths grows with the square of the number of resource elements in the network.

Such scenario is nonetheless of great practical relevance: administrative monitoring, as well as accounting or diagnosis fall into the category of a monitoring task that concentrates on few routes, known a priori. To cite some of the works on this trail, we cite the Globus MDS [10], EGEE [4].

In this paper we explore another facet of the problem, which is relevant to cope with *unplanned* monitoring requests. The interest for such aspect of network monitoring is that monitoring requests from the agents responsible for the coordination of Grid jobs cannot be anticipated, they extend to a limited lifetime, they have a moderate (if any)

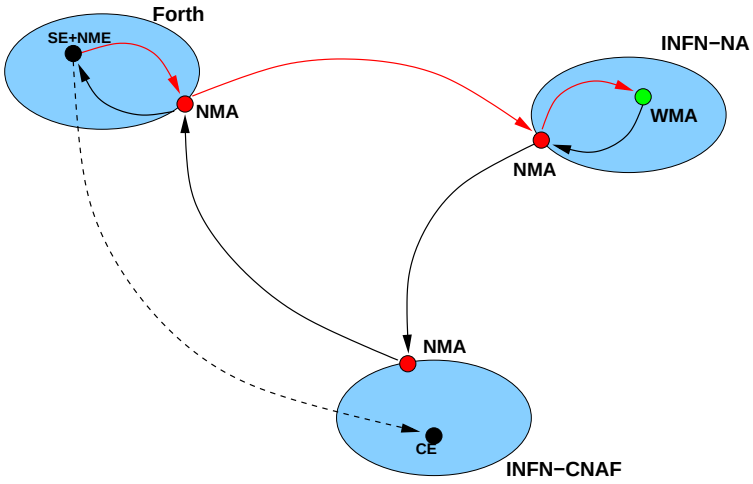

Figure 1: Information flow related to a ping session: the green circle indicates a Workflow Management Agent, black arrows indicate the flow of a Network Monitoring Session description representing a request, red circles represent NMAs, black circles represent monitored sites, and red arrows represent the data stream from the Network Monitoring Element to the Workflow Management Agent.

```
<?xml version="1.0"?>

  <nmsd:NetworkMonitoringSession
    xmlns:nmsd="http://www.di.unipi.it/~augusto/schema/
                NetworkMonitoringSessionDescription-0.5.xsd"
    SessionId="456@this.NMagent.ip">
    <RequestFrom TaskId="WF245" WorkflowMonitoringAgentId="OurBroker@FORTH"/>
    <Schedule StartAt="2007-09-17T12:00:00.000-01:00" Duration="2H"/>
    <Route>
      <NextAgent Agent="NMAgent@FORTH" Index="1"/>
      <NextAgent Agent="Theodolite@CNAF" Index="2"/>
    </Route>
    <NetworkElement SourceDomain="FORTH" DestinationDomain="CNAF"/>
    <MeasurementStream>
      <CharacteristicStream CharacteristicStreamId="1">
        <SamplePeriod>5</SamplePeriod>
        <Path>
          <SourceIP>processor_1.ics.forth.gr</SourceIP>
          <DestinationIP>ftp.cnaf.infn.it</DestinationIP>
        </Path>
        <PingOptions CharacteristicId="RoundTrip">
          <PacketSize>2048</PacketSize>
        </PingOptions>
      </CharacteristicStream>
    </MeasurementStream>
  </nmsd:NetworkMonitoringSession>

```

Table 5: XML instance for the example in figure 1

need of historical data, mainly to improve measurement robustness. Such aspect of network monitoring is far less studied, but exhibits a number of challenges: flexibility, since new requests must be activated dynamically for scalability reasons, and security, since network monitoring is an expensive activity, and requests must be authenticated.

Our approach to this aspect of network monitoring is marginally related to the past experience with *planned* network monitoring. The problems raising in the two cases are too different to justify a common solution: one for all, *unplanned* network monitoring in principle does not need a measurements database, while *planned* network monitoring relies on the availability of a powerful repository for measurements (think for instance to the R-GMA [3] architecture). Therefore we aimed at a different approach.

The architecture we propose has strong relationships with Internet streaming protocols: the basic requirements are those announced in [5], but our embrional solution for the request of a Network Monitoring Session is also inspired to the Internet SIP [6] protocol. We also take into account the RTP [9] protocol as for the components of a network monitoring request. In analogy to the *application profiles* introduced in RTP that characterize the payload in a flexible and expandable way, we opted for a *monitoring tool* oriented description, instead of a *characteristic oriented* approach, that characterizes OGF recommendations. Just like in the case of RTP, the *neutrality* of an approach that leaves to monitoring tool designers the freedom to introduce new measurements that do not exactly match existing characteristics, and to workflow managers designers the ability to use them, leaves space to research and new products in the rapidly evolving field of network monitoring tools.

Although such philosophy is in collision with OGF approach, which tends to define non-ambiguously network characteristics, the data framework we propose does not exclude a *characteristic oriented* description of the request. To clarify such issue, we include a skeleton **OGFCharacteristicsType**, encapsulated in its own external namespace: our intent is to indicate the way to proceed in order to include a *tool independent* description. We do not provide such description, since it is out of the scope of our present work, and already addressed by a OGF Working Group.

7 Conclusions

We introduce a distinction between planned and unplanned network monitoring activities: we claim that each of them exhibits challenging aspects, and requires distinct solutions, although the latter is receiving less attention than the former from the research community.

The fact that unplanned activities are requested by Workflow Management Agents introduces the need of a scalable and flexible authentication scheme. Once they are activated their output should not be stored for future use, but directly delivered to the requester with a lightweight streaming protocol. The request and reply protocol should be flexible and allow the integration of new monitoring tools, leaving to tool designers the task of describing data format.

In this paper we address a fundamental step in the design of a solution for the management of unplanned monitoring activity, which consists in the definition of the information needed to describe a single monitoring session, and the scope of such entity. In order to give an intuitive framework, we outline the architecture of the network monitoring infrastructure, identifying the actors and their inter-play.

A Network Monitoring Session Schema

```
<schema
 xmlns="http://www.w3.org/2001/XMLSchema"
 xmlns:pt=
   "http://www.di.unipi.it/~augusto/schema/PingTool.xsd"
 xmlns:am=
   "http://www.di.unipi.it/~augusto/schema/MAPIMonitoringTools-0.1.xsd"
 xmlns:sched=
   "http://www.di.unipi.it/~augusto/schema/Schedule-0.1.xsd"
 xmlns:path=
   "http://www.di.unipi.it/~augusto/schema/Path-0.1.xsd"
 xmlns:ogf=
   "http://www.di.unipi.it/~augusto/schema/OGFCharacteristics-0.1.xsd"
 xmlns:nmsd=
   "http://www.di.unipi.it/~augusto/schema/NetworkMonitoringSessionDescription-0.5.xsd"
```

```
15
16 targetNamespace=
17   "http://www.di.unipi.it/~augusto/schema/NetworkMonitoringSessionDescription-0.5.xsd"
18 elementFormDefault="unqualified"
19 attributeFormDefault="unqualified">
20
21 <import namespace=
22   "http://www.di.unipi.it/~augusto/schema/PingTool.xsd"/>
23 <import namespace=
24   "http://www.di.unipi.it/~augusto/schema/MAPIMonitoringTools-0.1.xsd"/>
25 <import namespace=
26   "http://www.di.unipi.it/~augusto/schema/Schedule-0.1.xsd"/>
27 <import namespace=
28   "http://www.di.unipi.it/~augusto/schema/Path-0.1.xsd"/>
29 <import namespace=
30   "http://www.di.unipi.it/~augusto/schema/OGFCharacteristics-0.1.xsd"/>
31
32 <annotation>
33  <documentation xml:lang="en">
34   Network Monitoring Session Description.
35   Copyright CoreGRID. All rights reserved.
36   Version 0.1
37  </documentation>
38 </annotation>
39
40 <element name="NetworkMonitoringSession"
41               type="nmsd:NetworkMonitoringSessionType"/>
42
43 <element name="comment" type="string"/>
44
45 <complexType name="NetworkMonitoringSessionType">
46  <sequence>
47   <element name="RequestFrom"
48               type="nmsd:WorkflowMonitoringTaskType"
49               maxOccurs="unbounded"/>
50   <element name="Schedule"
51               type="sched:NetworkMonitoringScheduleType"/>
52   <element name="Route"
53               type="nmsd:RouteStackType"
54               minOccurs="0"/>
55   <element name="NetworkElement"
56               type="nmsd:NetworkElementType"/>
57   <element name="MeasurementStream"
58               type="nmsd:MeasurementStreamType"/>
59  </sequence>
60  <attribute name="SessionId"
61               type="string"
62               use="required"/>
63 </complexType>
64
65 <complexType name="WorkflowMonitoringTaskType">
66  <attribute name="TaskId"
67               type="string"/>
68  <attribute name="WorkflowMonitoringAgentId"
69               type="string"/>
70 </complexType>
71
72 <complexType name="RouteStackType">
73  <sequence>
74  <element name="NextAgent" minOccurs="0" maxOccurs="unbounded">
75   <complexType>
76    <attribute name="Agent"
77               type="string"/>
78    <attribute name="Index"
79               type="nonNegativeInteger"/>
```

```
   </complexType>
  </element>
  </sequence>
 </complexType>

 <complexType name="NetworkElementType">
  <attribute name="SourceDomain"
                type="string"
                use="required"/>
  <attribute name="DestinationDomain"
                type="string"
                use="required"/>
 </complexType>

 <complexType name="MeasurementStreamType">
  <sequence>
   <element name="CharacteristicStream"
             minOccurs="1" maxOccurs="unbounded">
    <complexType>
     <sequence>
      <element name="SamplePeriod"
                type="float"
                minOccurs="0"/>
      <element name="Path"
                type="path:NetworkMonitoringPathType"
                maxOccurs="unbounded"/>
      <choice>
       <element name="PingOptions"
                        type="pt:PingOptionsType"/>
       <element name="MAPIOptions"
                        type="am:MAPIMonitoringToolOptionsType"/>
       <element name="OGFCharacteristics"
                        type="ogf:OGFCharacteristicsType"/>
      </choice>
     </sequence>
     <attribute name="CharacteristicStreamId"
                type="string"/>
    </complexType>
   </element>
  </sequence>
 </complexType>

</schema>
```

References

[1] Demetres Antoniades, Michalis Polychronakis, Spiros Antonatos, Evangelos P. Markatos, Sven Ubik, and Arne Øslebø. Appmon: An application for accurate per application network traffic characterization. In *In IST Broadband Europe 2006 Conference*, 2006.

[2] A. Ciuffoletti. The wandering token: Congestion avoidance of a shared resource. In *Austrian-Hungarian Workshop on Distributed and Parallel Systems*, page 10, Innsbruck (Austria), septemper 2006.

[3] A. Cooke, A. Gray, L. Ma, W. Nutt, J. Magowan, P. Taylor, R. Byrom, L. Field, S. Hicks, and J. et Al. Leak. R-GMA: An information integration system for grid monitoring. In *Proceedings of the Eleventh International Conference on Cooperative Information Systems*, 2003.

[4] EGEE. *Network Performance Monitoring Architecture*, october 2006.

[5] Sally Floyd, Van Jacobson, Ching-Gung Liu, Steven McCanne, and Lixia Zhang. A reliable multicast framenwork for light-weight sessions and application level framing. *IEEE/ACM Transactions on Networking*, November 1996.

[6] H. Handley, H. Schultzrinne, Schooler E., and J. Rosenberg. SIP: Session initiation protocol. Request for Comment 2543, Network Working Group, March 1999.

[7] Hao Jiang and Constantinos Dovrolis. Passive estimation of TCP round-trip times. *SIGCOMM Comput. Commun. Rev.*, 32(3):75–88, 2002.

[8] Antonis Papadogiannakis, Alexandros Kapravelos, Michalis Polychronakis, Evangelos P. Markatos, and Augusto Ciuffoletti. Passive end-to-end packet loss estimation for grid traffic monitoring. In *Proceedings of the CoreGRID Integration Workshop*, 2006.

[9] H. Shultzrinne, S. Casner, R. Frederick, and V. Jacobson. RTP: A transport protocol for real-time applications. Request for Comment 1889, Audio-Video Transport Working Group, January 1996.

[10] The Globus Team. Globus Toolkit 2.2 MDS Technology Brief, Jan 2003. Draft.

[11] Panos Trimintzios, Michalis Polychronakis, Antonis Papadogiannakis, Michalis Foukarakis, Evangelos P. Markatos, and Arne Øslebø. DiMAPI: An application programming interface for distributed network monitoring. In *Proceedings of the 10th IEEE/IFIP Network Operations and Management Symposium (NOMS)*, April 2006.

[12] Rich Wolski. Dinamically forecasting network performance using the Network Weather Service. Technical Report TR-CS96-494, University of California at San Diego, January 1998.