

02.06.05

Deliverable DJ.1.2.1: GEANT2 General Monitoring Framework Design



Deliverable DJ.1.2.1

Contractual Date: 31/03/05
Actual Date: 02/06/05
Contract Number: 511082
Instrument type: Integrated Infrastructure Initiative (I3)
Activity: JRA1 – Performance Monitoring and Measurement
Work Item: 2
Nature of Deliverable: R - Report
Dissemination Level : PU - Public
Lead Partner: RENATER
Document Code : GN2-05-057v5

Authors: Jeff Boote (Internet2), Eric Boyd (Internet2), Mauro Campanella (GARR), Jérôme Durand (RENATER), Susan Evett (Internet2), Maciej Glowiak (PSNC), Andreas Hanemann (DFN-Muenchen), Roland Karch (DFN-Erlangen), Stephan Kraft (DFN-Erlangen), Loukik Kudarimoti (DANTE), Olav Kvitem (NORDUnet), Roman Lapacz (PSNC), Athanassios Liakopoulos (GRNET), Luís Marta (FCCN), Joe Metzger (ESnet), Maurizio Molina (DANTE), Martin Swany (Internet2), Szymon Trocha (PSNC), Sven Ubik (CESNET)

Abstract

This document presents the architecture of a performance measurement framework for the GÉANT2 and Internet2 community. The document aims to provide a framework for integration and evolution of monitoring tools deployed in different domains. This integration effort is aimed at giving an end-to-end (between network edges) approach for network management: this includes performance monitoring, security management and fault tracking.

Table of Contents

0	Executive Summary	vii
1	Introduction	1
1.1	Design Procedure	3
1.2	General Architecture Summary	5
1.2.1	Identification and Definition of Services	6
1.2.2	Definitions of Relationships and Communication Flows between Services	6
1.2.3	Services in an Interdomain Environment	6
1.2.4	Design of services	6
1.2.5	Architecture Validation	7
2	Identification and Definition of Services	8
2.1	Base Services	9
2.1.1	Measurement Point Service	9
2.1.2	Measurement Archive Service	10
2.1.3	Lookup Service	10
2.1.4	Authentication Service	11
2.2	Additional Services	12
2.2.1	Resource Protector Service	12
2.2.2	Transformation Service	12
2.2.3	Topology Service	13
3	Measurements in a Single Domain	14
3.1	Service Registration	14
3.2	Authentication and Authorization	15
3.3	Request Data	17
3.4	On-demand Test	18

4	Inter-domain Infrastructure	20
4.1	Lookup services interaction	21
4.2	Authentication, Authorization and Security	21
4.2.1	Measurement from a Related (Federated) Domain	22
4.2.2	Measurement between Unrelated Domains	25
4.3	Storage of the Measurement Results	25
4.4	Coordination of a Distributed Measurement Setup Process	26
5	Design of Services	29
5.1	Measurement Point Service	30
5.1.1	Specification of Tasks	31
5.1.2	Components	31
5.1.3	Dependencies	32
5.1.4	Data Types	32
5.1.5	Interfaces and Interactions	33
5.2	Measurement Archive Service	37
5.2.1	Specification of Tasks	37
5.2.2	Components	38
5.2.3	Dependencies	38
5.2.4	Data Types	39
5.2.5	Interfaces and Interactions	39
5.3	Lookup Service	41
5.3.1	Specification of Tasks	42
5.3.2	Components	42
5.3.3	Dependencies	43
5.3.4	Data Types	43
5.3.5	External Interactions	43

5.4	Authentication Service	48
5.4.1	Specification of tasks	48
5.4.2	Components of the Authentication Service	49
5.4.3	Dependencies	49
5.4.4	Interfaces and Interactions	50
5.4.5	Authentication service information	50
5.5	Resource Protector Service	50
5.5.1	Specification of Tasks	50
5.5.2	Components of the Resource Protector	51
5.5.3	Dependencies	51
5.5.4	Data Types	52
5.5.5	Interfaces and Interactions	52
5.5.6	Resource Protector Information	52
5.6	Transformation Service	54
5.6.1	Specification of Tasks	55
5.6.2	Components	55
5.6.3	Dependencies	56
5.6.4	Data Types	56
5.6.5	Interfaces and Interactions	56
5.7	Topology Service	59
5.7.1	Specification of Tasks	59
5.7.2	Components	60
5.7.3	Dependencies	60
5.7.4	Data Types	60

5.7.5	Interfaces and Interactions	60
5.7.6	Topology Service Information	61
6	Architecture validation against existing measurement capabilities	63
6.1	Existing Measurement Capabilities and the MP Service	63
6.1.1	Active Measurements	64
6.1.2	Passive Measurements	69
6.2	Measurement Storage and the MA Service	71
6.3	Other Services	71
7	Conclusions	73
8	Acronyms	74
9	References	75
9.1	GN2 project documents	75
9.2	Related technology references	75
9.3	Measurement tools and frameworks references	75

Table of Figures

Figure 1-1 GFD as an abstraction layer between the measurement tools deployed within networks and the User interface	2
Figure 1-2: General architecture summary	5
Figure 3-1: Service registration message sequence	15
Figure 3-2: authentication and authorization message sequence	16
Figure 3-3: Data request message sequence	17
Figure 3-4: On-demand test message sequence (test initiation)	18
Figure 3-5: On-demand test message sequence (test results processing)	19
Figure 4-1: Type of measurement based on the number and location MP(s)	21
Figure 4-2: Measurement from a related domain message sequence	23
Figure 5-1: Introduction to conventions used in the section	30
Figure 5-2: MP Service components	30
Figure 5-3: Requesting a measurement from a MP Service	35
Figure 5-4: Making a measurement and providing data to the subscribers	36
Figure 5-5: Registering, De-registering and Keep-Alives	36
Figure 5-6: MA internal components	37
Figure 5-7: Requesting data from a Measurement Archive service	40
Figure 5-8: Registering, De-registering and Keep-Alives	41
Figure 5-9: Lookup Service internal components	41
Figure 5-10: Registering / De-registering and Keep-Alives with a Lookup Service	45
Figure 5-11: Find other Lookup Services	46
Figure 5-12: Querying for information	47
Figure 5-13: Query other Lookup Services	47
Figure 5-14: Authentication Service internal components	48
Figure 5-15: Resource Protector internal components	50
Figure 5-16: Resource Protector – Requesting resource availability	53
Figure 5-17: Resource Protector – contacting external Resource Protectors	54
Figure 5-18: Transformation Service internal components	54
Figure 5-19: Setup for a Transformation	57
Figure 5-20: Transformation of Data by the Transformation Service	58
Figure 5-21: Registration, De-registration and Keep-Alives	58
Figure 5-22: Topology Service internal components	59
Figure 5-23: Correlating Topology data	62

0 Executive Summary

This document details the General Framework Design (the architecture) of a performance measurement framework. The architecture is designed to allow autonomous network operators to create measurement tool daemons open to the world but governed by locally-defined policies and limits, allow discovery of measurement tool daemons along the end-to-end path, and facilitate, but not require, the use of federated trust models. The design is “services-based,” allowing multiple autonomous systems performing authentication, discovery, data creation, data transformation, and data storage roles to work together in a modular fashion. The design is largely decentralized, maximizing scalability and minimizing administrative overhead.

The main goal of the GN2 JRA1 activity is to provide a multi-domain monitoring framework. Although the main objective of this document, as described in the technical annexe, is to provide the General Framework Design for a single domain, the activity had to keep in mind from the very beginning the multi-domain issues, in order to avoid the complete redesign of the framework at a later stage. The current version of the General Framework Design (GFD) document presents an architecture which will work pretty much the same way in a single domain environment as in a multi-domain environment. The services and their inter-action described in this document are mostly valid in both types of environments.

The Internet2 piPEs (End-to-End Performance Initiative Performance Environment System) has very similar goals to the ones of the GN2 JRA1. They aim at providing a framework which enables end-users and network operators to determine end-to-end (E2E) performance capabilities, locate E2E problems and enable remote initiation of partial path performance tests. They have accomplished this by developing home grown performance measurement tools. The particularity of those tools is that a user located within another domain can remotely initiate some tests. They have gained some very valuable experience over the last few years. However, the Internet2 system is currently limited to the particular tools they have developed. The GN2 JRA1 system intends to build a system, which exchanges monitoring information seamlessly between domains using pre-defined interfaces to provide better visibility of end-to-end path behavior. The JRA1 system must take into consideration the diversity of access policies and equipment deployed within the various network. The JRA1 can rely on the experience gained during the prototype perfmonit trial¹, which was developed by Dante over the last two years.

As the Internet2 piPEs and the GN2-JRA1 activity have found out:

¹ More information about the Perfmonit project can be found at <http://www.geant.net> in section Dante Home > Work > Technical Programme > Multi-domain monitoring

- Their main objectives are very similar (provide network end-to-end performance information)
- Their experiences and work plan are quite complementary (multi-domain and diversity aspects for JRA1 and on-demand tests for Internet2)
- It makes a lot of sense for the two to be inter-operable as they encounter several projects spread over both Europe and the US for which such monitoring information may be beneficial or who have requested such a feature along those lines (eVLBI, GRIDs, NASA)
- They have to develop the same concepts and functionalities

Both activities agreed to work jointly on a single General Framework Design as the work done by both groups is similar.

This choice of collaboration with the Internet2 piPEs group is bringing to both groups advantages and inconveniences. The advantages are, in addition to the above-mentioned ones:

- Both groups can share the development of a module as well as re-use the ones developed by each other.
- By joining our forces for the designing and the development of the framework, the frameworks inter-operability issues between the EU and the US that may arise will be virtually eliminated, and this would thus allow to save some time at a later stage.
- By joining our forces on a common system, it is intended to create a precedent and gain a strong position and it is hoped that other domains will be following our track and either adopt our framework or make sure that they are inter-operable with us. This should allow us a reduction of the time spent on inter-operability issues and to get more benefit out of the developments.
- It is also expected that the joint activity would get the capability of attracting additional resources for those tasks (ESnet has already joined the project and is contributing to the GFD).

Unfortunately, this also brings inconveniences and it mostly brought some additional delays in respect with our respective plans. Both groups (Internet2 PAT² and GN2-JRA1) have different backgrounds and needed a certain amount of work to overcome their differences in understanding one another's work, specifically on terminology, objectives, scope of the projects, past experience and work procedure. The groups also had to learn to work with a time difference of 6-8 hours.

The multi-domain constraints and the collaboration with Internet2 piPEs have had impacts in term of the achievement for this deliverable. The scope widened (multi-domain, inter-operability with Internet2) in respect with what is described in the technical annexe. But, on the other hand we haven't had the opportunity to go as much in detail as we would have initially wished (a certain number of final design decisions haven't yet been

² Performance Architecture & Technologies

Project:	GN2
Deliverable Number:	DJ.1.2.1
Date of Issue:	02/06/05
EC Contract No.:	511082
Document Code:	GN2-05-057v5

taken and the interfaces haven't been specified). But we have strong hopes that the long term benefits and time saved over the duration of the projects will counterbalance and overcome the initial inconveniences.

Project:	GN2
Deliverable Number:	DJ.1.2.1
Date of Issue:	02/06/05
EC Contract No.:	511082
Document Code:	GN2-05-057v5

1 Introduction

The networks make use of a diversity of tools for monitoring a variety of network characteristics, from simple link usage to detailed protocol analysis. The mechanisms used to collect the data, and above all the presentation to users, differ quite significantly between different network domains. The notion of users differs as well, and different policies to access the available data are in place in the various administrative domains.

A Network Operation Center (NOC) that has to troubleshoot an end-to-end performance problem does not have a standard access to network performance data all along the data path to help understand and identify what is the issue. Each network domain along the path (in many cases at least five domains are involved: GEANT, the two NRENs³ and the two campus/university domains) between the two users has its own set of performance data and its own policies to access this data.

This leads to the situation where, due to this lack of co-ordination for network management, the NRENs, the regional networks and the campus NOC staffs do not have a global view of the network and may spend considerable time troubleshooting some problems which are often out of their scope.

The main objective of the GN2-JRA1 activity is to provide the design and the implementation of a middle layer (see **Figure 1-1**) between the visualisation tools from a user and the measurement tools deployed within the networks and expose seamlessly to the first ones, the data collected by the second ones.

³ National Research and Education Network

Project:	GN2
Deliverable Number:	DJ.1.2.1
Date of Issue:	02/06/05
EC Contract No.:	511082
Document Code:	GN2-05-057v5

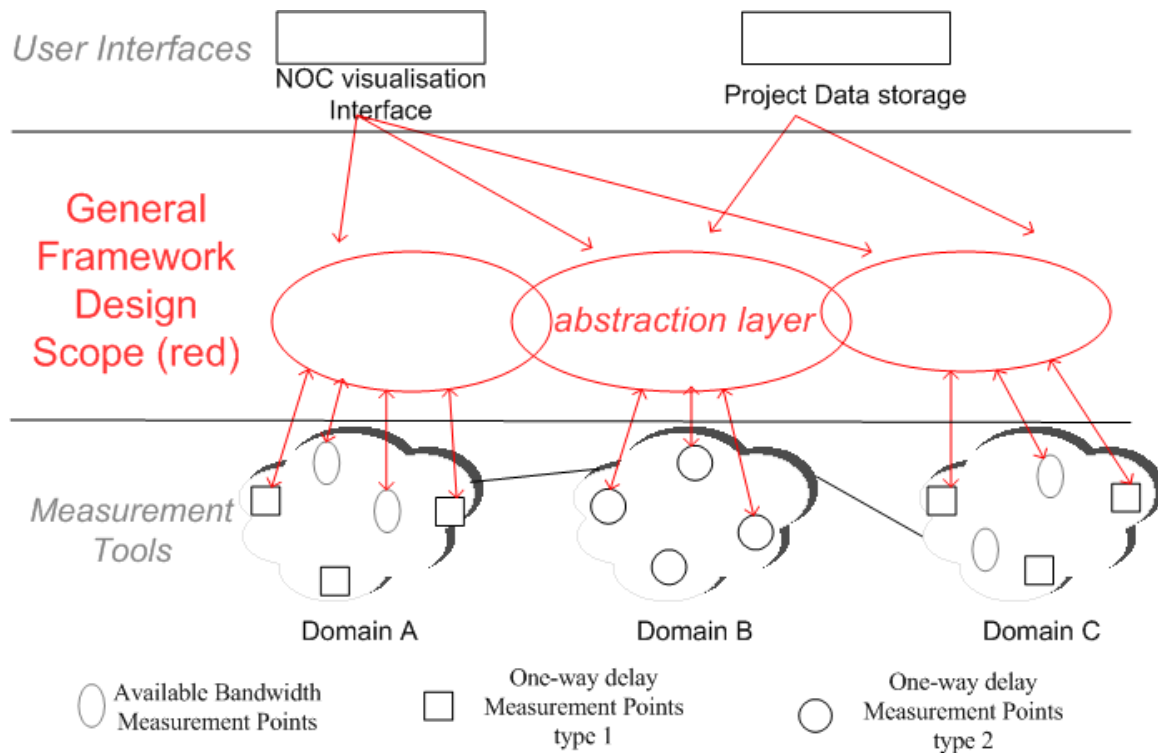


Figure 1-1 GFD as an abstraction layer between the measurement tools deployed within networks and the User interface

The goal of the General Framework Design is to provide the basis for this activity. The system should be general enough to allow remote triggering of measurement tests, collection of measurement results and presentation of network monitoring information to any data consumers. It must also take into account the particularity of each domain (policies, type of measurements tools). A data consumer, is broadly speaking a tool which can make use of monitoring data (e.g. for visualisation, prevision, storage, etc).

The general framework doesn't aim at providing the design of the measurement tools nor of the data consumer tools. However it provides a description of an infrastructure, which allows the information to be exchanged between the first category of tool and the second one and how the data consumer should interact with the system. One should note that this version of the framework design focuses on defining the different building blocks (or services) and the way they interact together. The protocols exact definition (messages format...) and detailed interfaces are not specified in this document but will be described in the next version.

From new deployments being made in some research networks in Europe, it appears that the network is slightly moving from a purely IP based infrastructure to an hybrid one, where IP cohabits with lambda provisioning, and other protocols, at different layers. The JRA1 project is focusing on the exchange of IP measurement information. We recognise that there is an increasing need to covers lower layer monitoring information. However the lower layer services and their monitoring haven't yet been fully specify and defined even for multi-domain schenarios. JRA3 (Bandwidth on Demand services) mention that they would define how to monitor those services and how to retrieve the information. Until then, JRA1 will concentrate its work on IP

Project:	GN2
Deliverable Number:	DJ.1.2.1
Date of Issue:	02/06/05
EC Contract No.:	511082
Document Code:	GN2-05-057v5

services which are already in use in multi-domain environment. Nevertheless the framework designed in this document is not only based for IP only networks, but it can be by extension applied to other types of infrastructures.

1.1 Design Procedure

A top down approach has been followed in the monitoring architecture design procedure. In this approach, the process of design starts with a 'Service' level view of the perceived architecture, i.e. the required services are identified. This view is then detailed to illustrate how the current monitoring infrastructure (bottom layers) fit within the architecture. The main goal of such an architecture is to enable Users to make network measurements in a multi-domain environment and to validate the service-level agreements provided by an NREN.

The main steps in defining a monitoring architecture are:

- Identifying and defining "Services," listing tasks that each service can accomplish, and identifying elementary components required within each service
- Defining relationships among services and the information flow between services in some elementary use cases, including functional interface specifications
- Analyzing the issues related to deploying services in a multi-domain environment
- Detailing service design
- Confirming that the base services have been properly identified by verifying them against simple use cases.

The design of the monitoring architecture is driven by user requirements and the state-of-the-art of the network monitoring which provided directions and suggestions for the development of a common monitoring architecture [DJ1.1.1]. Information on existing tools deployed within the networks and the state-of-the-art ones and their features (on-demand tests, network services based –IP version, IP ToS-, etc) as well as general type requests made by users (data retrieval, on demand measurements, scheduled measurements, administrative requests) identified the main elements of the architecture:

- It must operate inter-domain and end-to-end. There is also a need for intra-network measurements in order to identify the exact point causing the problem. Therefore, metrics should be provided on a hop-by-hop basis.
- It should be flexible enough to accommodate different types of metrics and parameters including new ones in the future. All metrics monitored by the architecture should be available for the network services provided e.g. IPv4 and IPv6, both for unicast and multicast and in different time-scales from short-term for performance debugging to long-term to observe trends.

- Whichever monitoring approach or a combination thereof should be able to be used by the architecture to provide the necessary information. Examples would be active and passive monitoring or using information gathered from the networking infrastructure.
- The architecture must encompass different existing monitoring tools to measure various metrics. It must standardise interfaces between various architecture components and provide a clear flexible system structure for easy extensibility.
- It should carefully cover security issues. Users are generally open to provide monitoring information from their location to other parties but some groups wish to restrict some information to a well identified groups of users such as the network operators, the project partners, etc. The architecture must be able to deal with customised requirements on what metrics and with whom they can be shared.
- It should offer access to visualisation tools representing to the user the behaviour of one or several parameters along a given path and to be able to start on-demand tests.
- The framework designed should be “user friendly”. Software developers willing to make their monitoring tools compliant with the framework need to be able to do it easily. Also the framework should be easily understood for people willing to deploy it.

The following general principles provide directions for the design of the resulting architecture:

- Scalability of the monitoring infrastructure (number of management domains, network elements, measurement points, monitoring tools, and management domains)
- Extensibility of the monitoring infrastructure, end-to-end, to satisfy needs in a multi-domain environment
- Interoperability with other monitoring systems, technologies, or implementations (this requires open interfaces and adherence to available standards)
- Open software solutions and independence from commercial software, unless the commercial software may provide a key benefit to the system
- Resilience of the monitoring infrastructure in common network failures by using distributed technologies and avoiding building too centralised entities
- Monitoring at the IP layer, principally; the possibility to monitor the transport and application layer is considered an important extension. Also, the framework should consider in a longer term extensions to monitor layers below IP

1.2 General Architecture Summary

This section positions the monitoring infrastructure designed in this document in the complete monitoring environment.

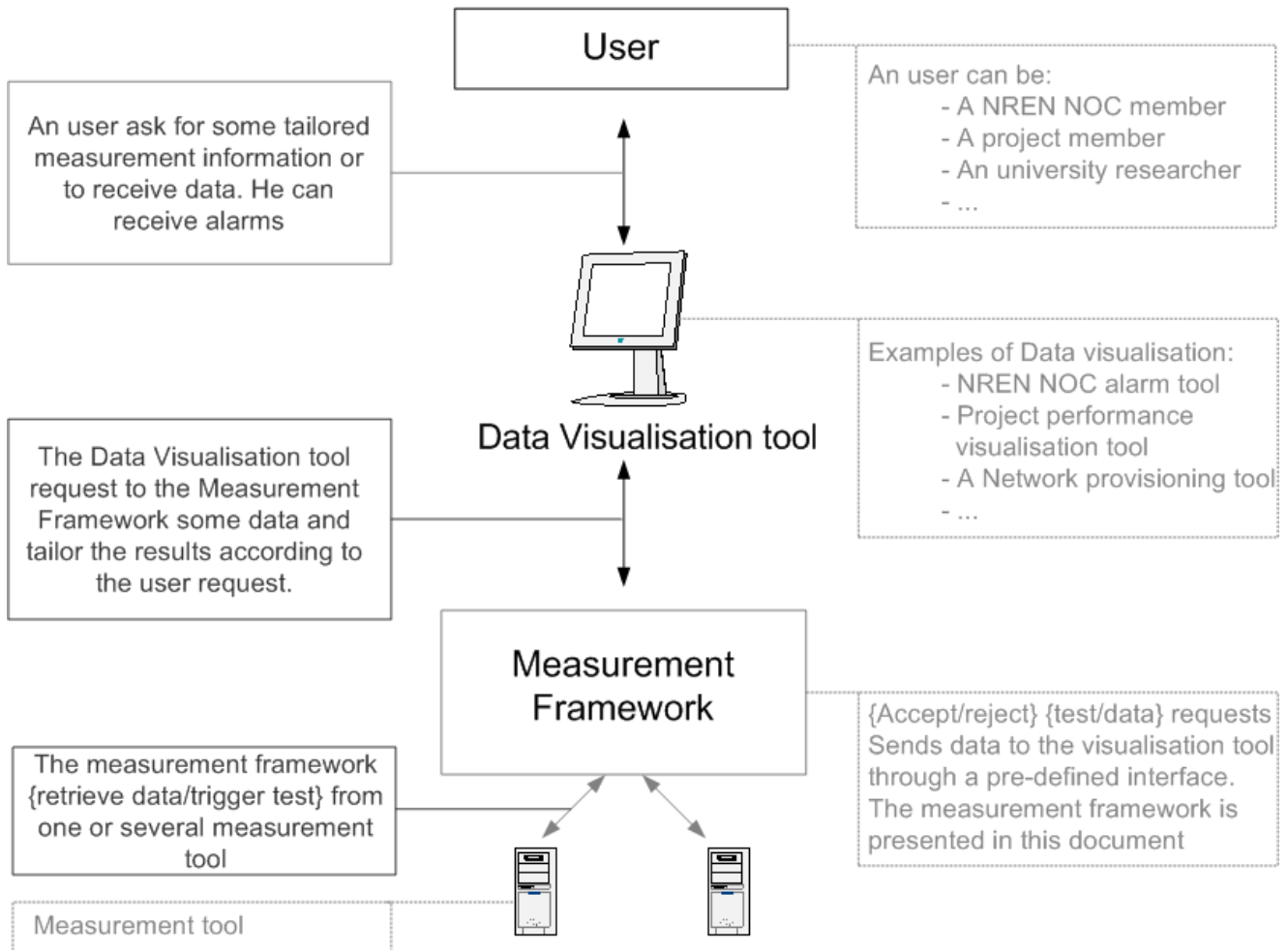


Figure 1-2: General architecture summary

The measurement framework receives monitoring test/data requests from the data visualisation tool (the monitoring application). It then gathers, transforms, stores, and publishes monitoring data to the tool through a standardised interface. This document only specifies the measurement framework and the interface to the tool. The design of the measurement tools, to perform the measurements, and of the visualisation application, which will make use of multi-domain monitoring information (for display or alarm triggering), and how they interact with the user is considered out of the scope of this document. Nevertheless, this document provides key elements so that any person implementing a tool, or porting an existing application knows how it can interact with the framework to get data collected from the system designed by this project (with the exception that the message format which will be provided in the next document).

Project:	GN2
Deliverable Number:	DJ.1.2.1
Date of Issue:	02/06/05
EC Contract No.:	511082
Document Code:	GN2-05-057v5

The project team will design one or two tools to be able to interact with the measurement framework. These tools will help in testing the measurement framework design. Some other tools may be implemented to test the system conceived in this document.

The following sections describe the content of these document different chapters.

1.2.1 Identification and Definition of Services

Network operators must be able to monitor the status and performance of the network. Within a set of policy constraints, users of the network ought to be able to do the same. This possibility is offered through a set of “services,” each performing specific actions, offering a set of functionalities necessary to assist in the monitoring activity and accessible via well defined interfaces. Services can be deployed in multiple different configurations. Examples of obviously needed services are Measurement Points (MP) capable of making measurements upon reception of requests and Measurement Archive (MA) that stores the measurement data. The section 2 aims at identifying and defining all the required services.

1.2.2 Definitions of Relationships and Communication Flows between Services

The section 3 analyses the communication flow between all the services. For simplicity, the communication is shown in a single domain case. Mutual dependencies are reported. Functional interface specifications in all the relevant communication channels flowcharts are specified.

1.2.3 Services in an Interdomain Environment

The main challenges of the service-oriented monitoring architecture used in an interdomain environment in which performance measurements are initiated between domains are described in section 4.

1.2.4 Design of services

The details of how a service is built using a number of “elementary components” are provided in section 5. Some of these components may be similar (if not identical) in multiple services, but, for implementation considerations and simplicity of design, we preferred to replicate them in each service. Most of the interactions between the services components are described, at the level of a message flow sequence.

1.2.5 Architecture Validation

The architecture is validated through its applicability of significant use cases in section 6. The analysis must detail how the use cases can be satisfied by the architecture and the solutions are searched for highlighting problems and possible optimisations.

2 Identification and Definition of Services

The measurement framework will be created using a SOA (services-oriented architecture). SOAs are particularly well suited for this task because they closely match the actual makeup of the network that the framework is being designed to monitor. The network is a dynamic entity with routes, connections, and devices in a constant state of change. The measurement framework must have these same properties. Just as new devices can easily be added and removed from the network, it must be possible to add and remove components of the measurement infrastructure with a minimum effort of re-configuration.

Additionally, most of the interesting network connections cross multiple administrative domains, and, therefore, the measurement framework must also have this capability. Data collected on one network or between two networks must be available to users on other networks (subject to locally-determined policy restrictions). It is unlikely that domain administrators will have predefined measurements setup for all paths of interest. Therefore, for the framework to provide the most benefit to end users, Measurement Points (MP), described in section 2.1 must be able to run measurements across multiple administrative domains.

The framework is made up of data producers and data consumers. An example of a data producer would be a measurement point. Examples of data consumers would be diagnostic tools and Measurement Archives (MA), detailed in section 2.1.2. Between them lies a chain of zero or more data transformers and data archivers. The other portions of the framework are primarily infrastructure used to help the data consumers find a data producer that has the information needed for some operation and gain permission to access that data.

Services in the framework will support both push and pull data flow models. To achieve this all data providers implement a publisher interface and all data consumers implement a subscriber interface. When a data flow is requested a consumer will provide a handle to a subscription interface if it wants a push interaction. If it does not provide a subscription handle, the data producer will create a publisher interface that the consumer can poll.

The remainder of this section describes the services provided by the measurement framework. There are three infrastructure-level services that primarily are used to build the framework itself: Lookup, Topology, and Authentication. The remaining services are dependent upon those services and provide the real functionality of the framework: Measurement Point, Resource Protector, Measurement Archive, and Transformation.

It has to be noted that the set of Measurement Point service, Measurement Archive service, Lookup service and Authentication service have the same functionalities than the domain measurement controller layer presented in the GN2 technical annexe. These are the base services. The remaining three services (Resource

Protector, Transformation and Topology services) have been identified as additional services which can bring capabilities to framework, but which are not mandatory.

2.1 Base Services

Those services are the minimum set of services which need to be defined to allow the framework to cover most of the current needs: retrieving data, triggering tests, authorise and authenticate a user, find out where a measurement point is located.

2.1.1 Measurement Point Service

The Measurement Point (MP) service creates measurement data either by initiating active measurement tests or querying passive measurement devices. Formally, it does not store or transform (i.e. aggregate/correlate/filter) existing data, although, in practice, a particular implementation may do that “under the hood”. Storage is done on the Measurement Archive Service and any transformation is made by the Transformation service. These 2 services are described in the following sections.

Different types of (physical) measurement capabilities may be implemented. Currently, the following six types of measurement data have been identified as required:

- Active delay, loss, and jitter measurements
- Processed information retrieved from network equipment
- Flow-based measurements
- Active stress-type achievable bandwidth measurements
- Active probe-type available bandwidth measurements
- Passive packet traces capturing

Each type of measurement capability has its own unique interface that can be used to acquire monitoring data. A common interface to these capabilities is required for ease of integration into the monitoring system as a whole. In effect, an MP service is a standard interface ‘wrapper’ around one or more measurement capabilities. To make effective use of these measurement capabilities in a larger framework, additional features are needed for coordination.

A measurement point can for example be a piece of hardware located in the core of the networks in order to get precise data and deterministic behaviour of the networks, but can also be a software installed at the edge of the network on the hosts, to perform end-to-end measurements.

Measurement points can also retrieve network topology information to be used then by the framework for data correlation. For example the output of a traceroute command can provide both performance measurement and topology data.

2.1.2 Measurement Archive Service

A Measurement Archive (MA) service is used to store and publish historical monitoring data produced by MP services and/or Transformation services. It does not create (generate new raw data) or transform (i.e. aggregate/correlate/filter) any data. The MA is essential for many reasons, including efficiency, to provide historical data for trend analysis, to reduce the load on MPs, and to ensure a high availability of network measurement data.

To store different types of measurement data, multiple databases within each service might be necessary. The service will be aware of the type of data stored in each of its databases and it uses this information to satisfy requests for storage and retrieval of network measurement data.

This service must be designed for high availability of the data to MA clients. This may be done in many ways, including reliable software and hardware, online (maintain a change log in a different server to guard against failures in the storage media) and offline backups (back up the entire databases to an independent device), redundancy, and fault recovery.

2.1.3 Lookup Service

The Lookup service enables users to discover other services (MP, MA, Authentication service, etc.) that can satisfy their needs. The Lookup service should give the requestor all information required to find and contact those services directly and employ the functionalities provided by them. In essence, the Lookup service acts as a service directory, where services can advertise themselves (provide their lookup information) and requestors are able to find any service they need. Some information about services can be hidden but the general topological positioning of the MP can (should) be exposed, so unauthorized users can determine if it is “interesting” (i.e. worth seeking authenticated access to) or not.

The Lookup service is a key element of the proposed measurement framework because it allows the treatment of every independent service as a visible part of the system. New services may identify themselves to the community and tender their capabilities subject to locally-determined policies or withdraw from the community without disrupting the interaction between other services. As the Internet is a dynamic environment, these issues must be addressed:

- Availability – plain registering and requesting lookup information.
- Knowledge of the lookup service – It may be advertised or statically set up in services.
- Scalability – readiness for increasing numbers of service instances. The implication is that there should be few, if any, centralised components that would act as bottlenecks when the system scaled.
- The deployment of at least one (possibly more) Lookup service per domain may be required. When multi-domain tests are required, Lookup services from neighbour domains could communicate to find needed services.

- Fault-tolerance – easy recovery and distribution of lookup service information.
- By distributing (and caching) lookup information across multiple Lookup services, it is possible to avoid making certain services unavailable when a particular Lookup service is unavailable.
- Security – authenticated access to services and discovery of “trusted” services (the source of authentication information is the Authentication service)
- Expressiveness – support wide range of queries for services

The Lookup service should support the discovery process by type or service characteristics. This gives users the ability to send multiple, exhaustive queries.

2.1.4 Authentication Service

The Authentication service provides the authentication (AuthN) functionality for the framework as well as an attribute authority (attribute authority is a component of the authentication server that decide which attribute can be disclosed to a resource). The detailed design and implementation of Authentication (AuthN) and Authorization (AuthZ) will be provided by GN2 JRA5 activity: Ubiquity (Mobility) and Roaming Access to Services. Because there is a strong need for authentication and authorization for the JRA1 infrastructure, there is a high collaboration between both activities. This section presents the list of JRA1 requirements for the Authentication service:

- Authentication service should support clients with multiple identities. (Some users are affiliated with more than one organization and/or have different roles at different times.)
- It should be possible to use role based authentication using emerging attribute assertion style authorization therefore protecting the privacy of the user. This typically means that a handle is created to provide additional information about the attributes of that user, and that resources can use that handle to make queries about the user (subject to privacy policy). The Handle should be included within the Authentication Token (AuthNToken) that is returned to a client after it has successfully authenticated to the Authentication service.
- It should be possible to federate authentication services to form communities that accept each others authentication.
- Federation details should be hidden from other services within a given administrative domain. The “trust” relationship within an existing domain should be between the services in the domain and the Authentication Service in the domain.
- As a consequence of the previous point, Authenticated requests will provide a way for all the services of making attribute assertion queries back to the authenticating entity. A handle should be included within the Authentication Token that is sent with the request. This will make it possible for each service to determine

what rights a particular resource requestor should have to the given resource without fully divulging the identity of the requestor.

2.2 Additional Services

These are optional services that perform some specific task not required by all the services such as the protection of a network resource, the transformation of data and the gathering of network topology information (e.g. to create network display).

2.2.1 Resource Protector Service

The Resource Protector (RP) service is used to arbitrate the consumption of limited resources, such as network bandwidth. It also has a scheduling component to deal with the consumption of time-dependant resources. When measurement activities are involved, resources may be related to the measurement infrastructure or ordinary network resources. The RP can allocate portions of a resource based upon configuration rules and can schedule the time-dependent resources. Services that consume resources will contact the RP to get them allocated. The primary example of this will be the MP. Because RPs reduce scheduling flexibility, by design, an RP should be deployed only to protect limited resources. In other words, some MP services will not have to contact an RP at all or will only have to contact a limited number of resources.

To allow for flexibility in deployment, it makes sense to allow MPs to request resources from a list of RPs (potentially different ones, depending upon the particular resources needed to perform the given test). Additionally, each RP can be configured with a list of additional RPs to protect resources of other types. This allows for hierarchical organizations of RP services, which can be useful for certain types of resource protection.

One has to note that every MP manages its own resources locally, with its embedded resource manager (described in detail in section 5.1.2). The resource manager can take decisions based on local information (CPU, memory of the MP or authorisation can be given based on the measurement requester). That resource manager (that can be seen as a host-level RP) might be configured to contact an RP that has been configured to protect a local network link if some tests have to be performed along the path. The MP asks RPs until it has validated that there were enough resources for the new test. While talking to RP services across several domains, priority for resource usage issues should be taken into account. These issues are not depicted in this version of the framework design and more details will be provided for the second release of the document.

2.2.2 Transformation Service

The Transformation service is used to pipeline data between the other services within the framework. It can be used to perform any function upon data. It fits between data producers (e.g. MPs, MAs, or other Transformation services) and data consumers (e.g. MAs, other Transformation services, or Clients) and is, itself both a data consumer and producer. Therefore, a Transformation service acts as both a publisher and a subscriber of data. The architecture prevents the transformation service for subscribing its own data to avoid loops that could be a

Project:	GN2
Deliverable Number:	DJ.1.2.1
Date of Issue:	02/06/05
EC Contract No.:	511082
Document Code:	GN2-05-057v5

potential security hole in the architecture. A service can appear only once from the place it is collected and its final destination.

There are a variety of potential functions that a Transformation service might provide including aggregation, correlation, filtering, and translation of data. (Note that a Transformation service would not be required for a simple “fan-out” of data, as that could be achieved by multiple consumers subscribing to a single data producer.) For example, a Transformation service might compress datasets from the more recent high resolution data to less recent lower resolution data and publish that data to an MA service.

A Transformation service also might read from multiple data producers to create a specific correlation. A very simplistic data analysis example would be a threshold detection operation, which then pushes data out for the purposes of triggering a NOC (Network Operational Centre) alarm.

An important example of a Transformation service is the Topology service. This piece is fundamental to the operation of the network measurement framework and is described in the next section.

2.2.3 Topology Service

The Topology service is used to make topological information about the network available to the framework. The Topology service is a specific example of a Transformation service. It collects topological information from a variety of sources (i.e. multiple MP services) and uses algorithms to deduce the network topology. The Topology service also reflects multiple network layers. That is, topology described on the domain level through network to wavelengths and the physical level should be possible. In addition to the network topology, it contains geographic information, like GPS coordinates.

Understanding the network topology is necessary for the measurement system to optimize its operation. For example, the Lookup service relies on the Topology service to determine which MP services are “closest to” interesting network landmarks (e.g. routers). Thus, in the same way that a host may query for a Measurement service instance that has a particular set of properties, a service component can also ask about node proximity. Additionally, the Topology service may be used for overviews/maps that illustrate the network with relevant measurement data.

It is also important to note that the definition of proximity can vary depending on the context of the request. Some services may wish to optimize themselves by minimizing the number of hops, whereas others may depend on high bandwidth or low latency. For this reason, it is desirable to leave open the exact nature of a query for proximity and simply allow a client to make various queries to the Lookup service based on its particular need.

3 Measurements in a Single Domain

Communication flows between all the services are shown for the most building block operations which are expected from the framework: service registration, authentication and authorisation, data request and on-demand test. For simplicity, the communication is shown in a single domain case.

3.1 Service Registration

All services must register themselves with a Lookup service (LS) to participate in the framework. (In other words, if a service does not register itself, no other service will know it exists.) There is an open, well-defined interface on LS that accepts registrations or updates from all the other services available in the infrastructure.

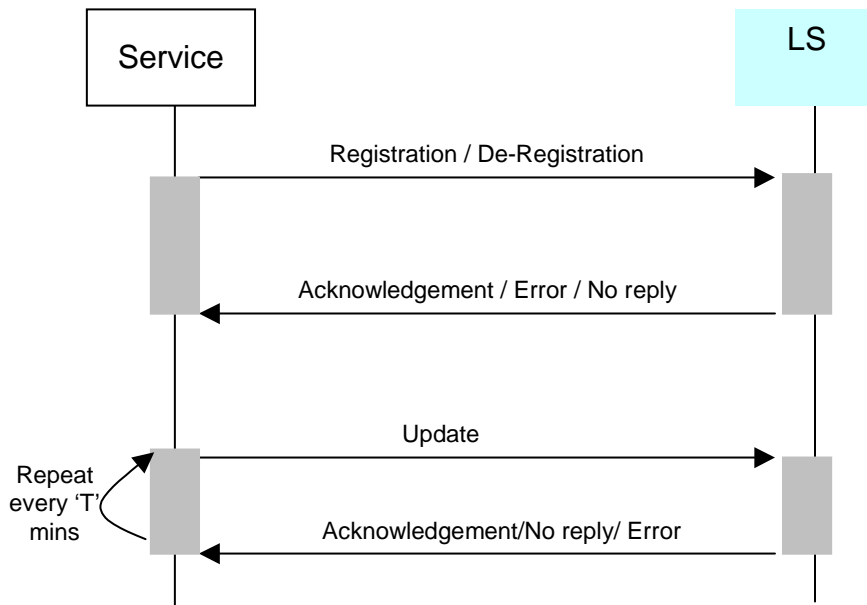


Figure 3-1: Service registration message sequence

If the service does not receive the acknowledgment message from LS it assumes the latter is unavailable. If one LS is unavailable then the service may register with another available LS (even from another domain).

At least one LS must be well-known to any of the services. The initial phase is thus discovering a lookup service. The search for a lookup service can be done either by unicast or by multicast. Initially, simple unicast discovery will be used. It is in a sense not discovery - the client will attempt to connect to a lookup service having known the existence and location of the lookup server. The location will be configured manually. In a later stage, if requirements emerge, more sophisticated methods like multicast discovery could be used. Then the client sends out a multicast request and lookup services that come across the request will respond to it. But using this type of finding of LS implies multicasts are enabled within the network. Another possible discovery methods are based on DNS records, P2P and/or anycast.

3.2 Authentication and Authorization

The first thing a client must do to use resources from the framework is to authenticate. The following descriptions use the term realm when most users might use the term domain. The reason for this is that the

authorization model envisions the need to support virtual organizations, that do not necessarily match the network domains. The following picture shows the control flow for a measurement request where the client has credentials for the same authentication realm as the MP (this example shows a request of service from a MP that needs to contact a RP for the final authorization decision, but it could just as easily be any other kind of service that requires authentication and authorization.):

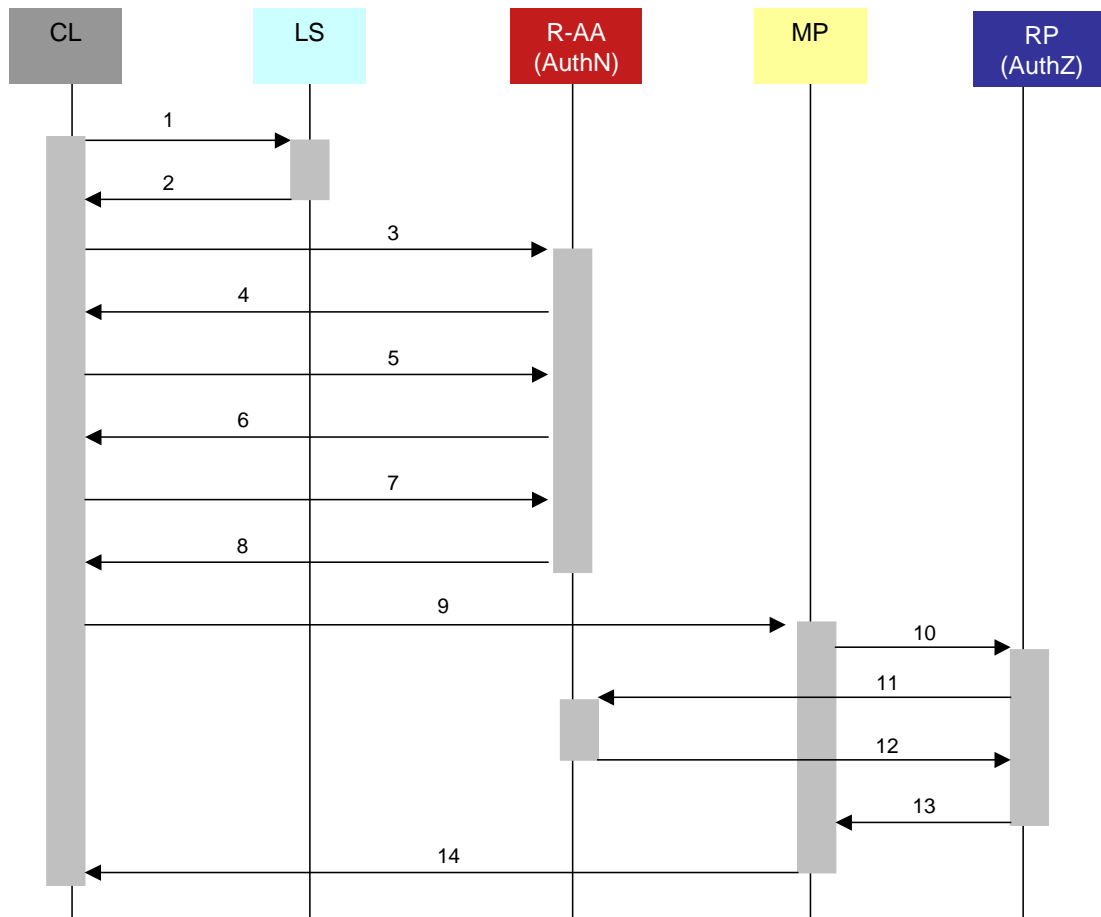


Figure 3-2: authentication and authorization message sequence

1. Client queries Lookup service (LS) for MPs that match a given criteria.
2. LS returns a list of candidate MPs including an indication of the authentication realms that manage authentication for each one. (Each MP could actually be managed by more than one realm.) LS also returns the address of an authentication service that can authenticate for each of the returned authentication realms.
3. Client contacts the authentication service that manages authentication for the resource realm (R-AA-Service) and requests an authentication token approved for use in the resource Realm (R-AuthRealm).
4. R-AA-Service returns a list of known (federated) authentication realms and asks the client to choose one for authenticating.
5. Client specifies @R-AuthRealm
6. R-AA-Service manages identities for R-AuthRealm, so R-AA-Service asks client for identity credentials.

7. Client presents credentials.
8. If credentials are valid, R-AA-Service creates a handle that can be used to request additional attributes about the identity subject to attribute release policies in R-AuthRealm. This handle is returned to the client encoded as an AuthToken approved by R-AuthRealm (R-AuthToken).
9. Client requests a measurement from MP. Request includes the R-AuthToken.
10. MP requests resources from the Resource Protector service (RP), required for the particular measurement. The R-AuthToken is passed along in the request.
11. RP needs more information about the identity requesting the resources and makes an attribute query to R-AA-Service using the R-AuthToken handle.
12. R-AA-Service releases only as much information about the client identity as is allowed.
13. RP returns resource availability. (allowed/disallowed) This portion will include scheduling.
14. MP returns response to measurement request.

3.3 Request Data

Test results are published by data producers to subscribers. The client (CL) asks Lookup service (LS) for the location of measurement data.

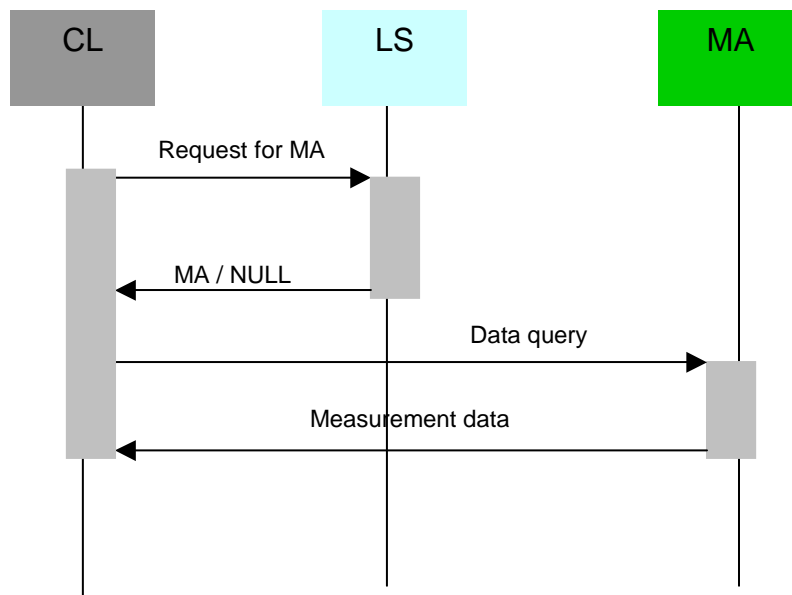


Figure 3-3: Data request message sequence

Project:	GN2
Deliverable Number:	DJ.1.2.1
Date of Issue:	02/06/05
EC Contract No.:	511082
Document Code:	GN2-05-057v5

3.4 On-demand Test

One of the most important goals of the measurement system is the ability to run on-demand tests. Its complexity requires many interactions between different kinds of services.

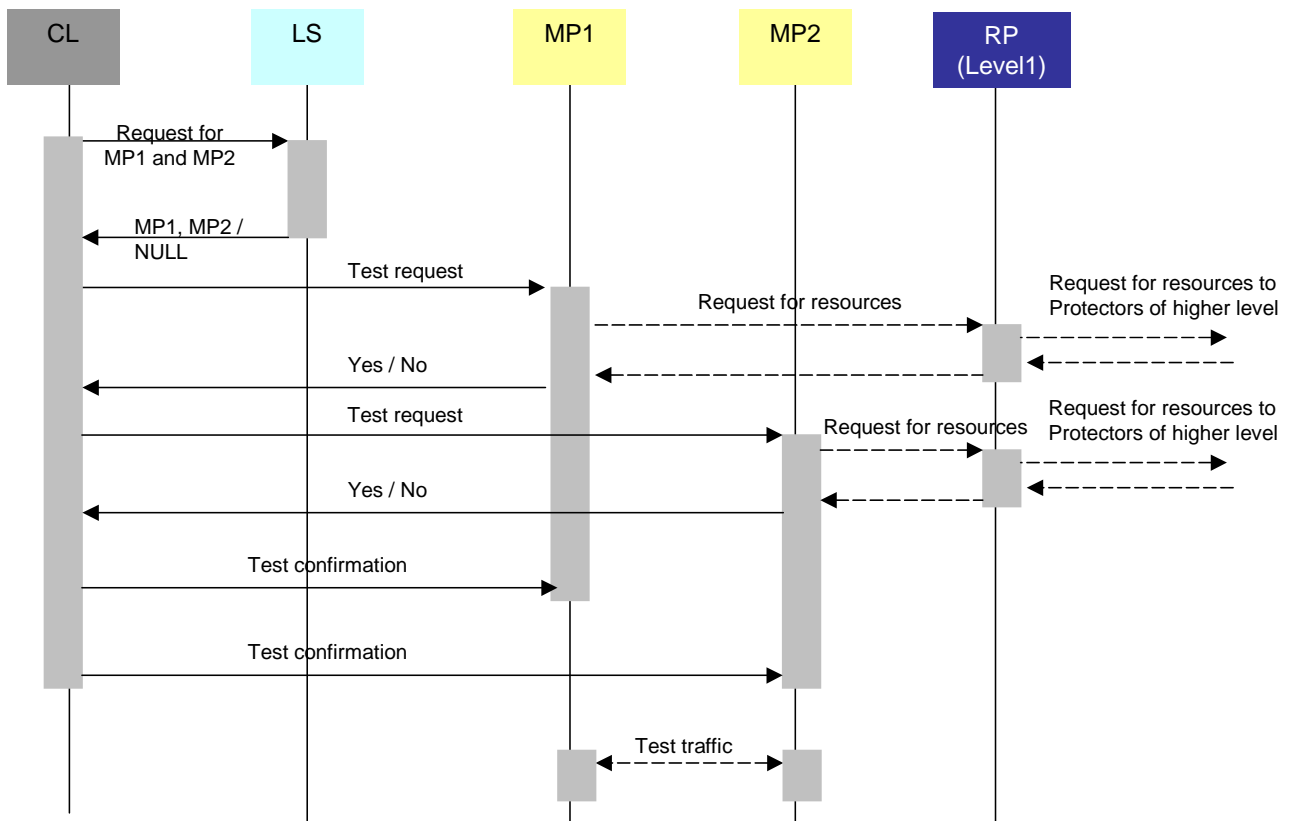


Figure 3-4: On-demand test message sequence (test initiation)

During test setup each MP determines a map of available times for the request and returns that along with a tentative reservation of the first time slot that fits the request. The tentative reservation is held for a time needed to get responses from all MPs that were selected to participate in a test.

After the Client (CL) contacts each MP, it adjusts its request for the next MP to fit within the parameters of the available times it has seen from each MP until it either has a tentative agreement from all MPs for the same time slot, or it determines no common timeslot is available (test cancellation). If CL collected positive responses from all MPs within defined time then sends confirmations of accepted test.

After the test setup CL does not have the permanent connection with MPs. To get data from the test it must do one of two things:

Project:	GN2
Deliverable Number:	DJ.1.2.1
Date of Issue:	02/06/05
EC Contract No.:	511082
Document Code:	GN2-05-057v5

1. subscribe directly to the MP publisher interface
2. or subscribe to a MA publisher interface associated with the requested MP's measurement data.

In the latter case, CL asks LS which MA service(s) subscribe(s) to the MP service. During a test, the MP services send data to one or more MA services (possibly via a chain of Transformation and MA services that process the data (e.g. transformation, correlation, filtering, or translation). CL may simply subscribe to the final MA service in the chain prior to the test commencement and receives the data in that fashion.

In order to get necessary data from MA, CL sends a request with a set of attributes e.g. test end points, duration, type of data, etc. Match checks are then performed against those attributes to find measurement results which are sent back to CL.

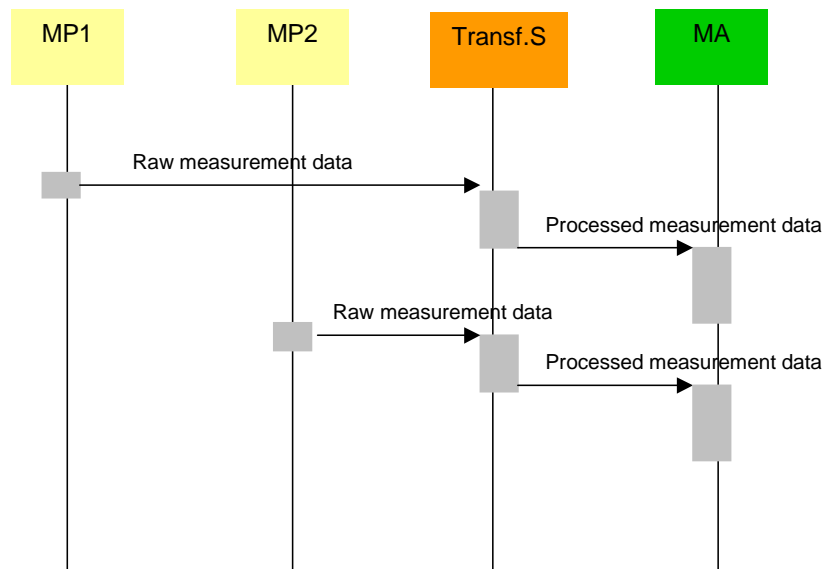


Figure 3-5: On-demand test message sequence (test results processing)

4 Inter-domain Infrastructure

Most interesting end-to-end performance issues span multiple administrative domains. The framework, as described in section 2, should work equally well for a set of connected networks run by autonomous administrative entities as it does for a single domain. Although it is possible that a single administrative domain might employ multiple Lookup services, multiple Authentication services, and multiple Resource Protector services, in practice it is the introduction of additional administrative domains that necessitates additional instances of such services. The ensuing complications are mainly related to three aspects:

- Authentication, authorization and security
- Storage of the measurement results
- Coordination of the distributed measurement set-up processes

Before looking in detail at the three mentioned aspects, let us introduce a categorization of the types of measurements that can be established in a multi-domain environment, according to the number of Measurement Points involved and number of domains involved in measurements.

A first category are measurements that are taken at a single “logical” point. This may be the case of processing information retrieved from network equipment, or Flow-based measurements, or a collection of packet traces. We’ll call them **Single Point** Measurements. These measurements are intrinsically Intra-domain.

A second category are measurements established between two MPs (e.g. packet loss, one-way delay, throughput) that are both located in the same domain. We’ll call them **Double Point, Intra-domain**. (One might also envision a multicast test involving 3 or more MP services, generalizing this category to Multi-point, Intra-domain.)

A third category are measurements that are established between two MP services belonging to different domains (neighbouring or not). We’ll call them **Double Point, Inter-domain**. (One might also envision a multicast test involving 3 or more MP services, generalizing this category to Multi-point, Inter-domain.)

As sub-categories of this last one, when both measurement points are logically peering edge routers of two neighbouring domains we’ll use the term **Double Point, Neighbour Domain**. When the measurement results can be partitioned into portions, and partial results can be assigned to domains, we’ll use the term **Double**

Point, Partition-able. (One might also envision a multicast test involving 3 or more MP services, generalizing these categories to Multi-point, Neighbour Domain, and Multi-point, Partition-able.)

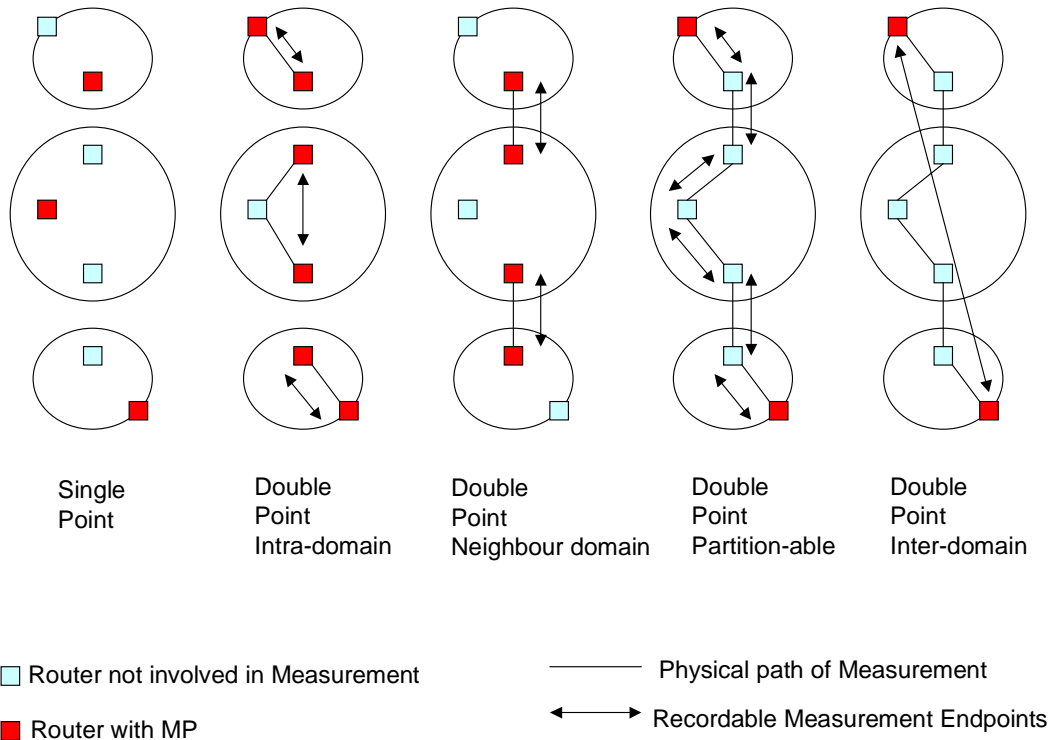


Figure 4-1: Type of measurement based on the number and location MP(s)

4.1 Lookup services interaction

A Lookup Service can peer with other Lookup Services and exchange all or part of the information it contains. This peer-to-peer model used for Lookup Services is suitable for making service information available to other domains, having local Lookup Services as well. It is also suitable for having a resilient architecture in case of failures.

4.2 Authentication, Authorization and Security

Giving access to a network measurement infrastructure to set up measurements, or even simply let users access results of measurements set up by other users exposes domains to potential threats.

We envisage that the installation and configuration of measurement tools will be always restricted to the domain's administrators or entities closely related to it. Other users wanting to use these tools have to access

them through the MP service. Authentication and authorization is necessary to limit the amount of active measurement traffic injected in the network, to guarantee a “fair” access to the measurement infrastructure to all authenticated and authorized users (i.e. do not overload the measurement tools), and to prioritize the measurements according to locally-defined policies. Authentication and authorization is also relevant for the other services not directly related to the establishment of a measurement but necessary for that. For example, the Topology service that derives information about the domain’s topology may be restricted to supplying that data to the locally-configured Lookup service or a networkweathermap. Likewise, an MP service that produces raw Netflow data might only allow a specific Transformation service that implements anonymisation to subscribe to the MP service’s publisher interface.

To control that in an inter-domain environment, we see two main possibilities:

- 1) Place dedicated entities, per each domain, to act as collection points for requests that have to be sent to services outside the domain itself;
- 2) Let the users directly contact, in sequence or in parallel, services in foreign domains.

There are pros and cons with both options, but both allow autonomous administrative entities to set local policies and limits on access to the measurement infrastructure.

The first alternative is more restrictive in terms of what the user can do, and is based on the mutual trust of these peer entities. The services will receive requests from foreign domains only after they have been filtered by the “central foreign collecting request” entity of the domain to which they belong. The single services need not do any further local authorisation for these requests, but simply try to fulfil them (if there are enough resources). The “logic” that the user application has to implement is also rather simple, because the complexity of ensuring that the measurement is feasible (e.g. all the needed MPs acknowledge the measurement establishment) is simple, because the complexity is handed off to the central entity. The main drawback is the additional complexity introduced by another category of services, which, in case of large domains with several MPs, can become a bottleneck, because not only measurement setup, but also measurement data retrieval should flow through the same entity.

The second alternative does not require a centralized control (but still, an authenticated user may have to “visit” in sequence more than one domain). The overall architecture is more lightweight, but the local authorisation component in each of the services becomes fundamental, as the service has the need (and right) to sort the requests in categories and possibly reject, fulfil only partially, or prioritise them.

The framework we are proposing adopts the second alternative.

4.2.1 Measurement from a Related (Federated) Domain

The following picture shows the control flow for a measurement request where the client has credentials for a different authentication realm from a MP. This request could be to any type of service but is illustrated in this example using a MP that needs to contact a RP for the final authorization decision, but it could just as easily be

any other kind of service that requires authentication and authorization. Federation has created a trust relationship between the two authentication realms (F and R):

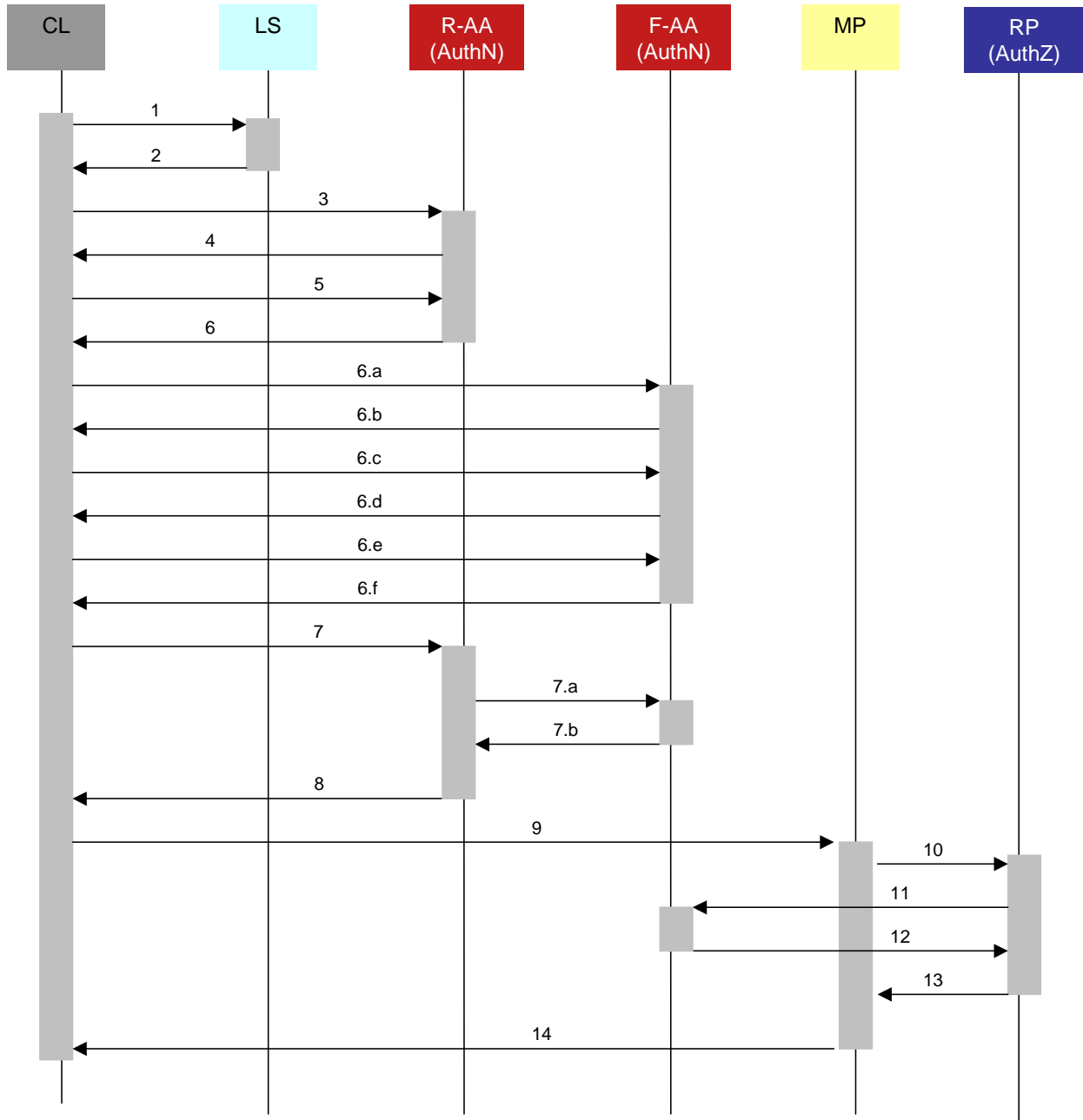


Figure 4-2: Measurement from a related domain message sequence

1. Client queries Lookup service (LS) for MPs that match a given criteria.
2. LS returns a list of candidate MPs including an indication of the authentication realms that manage authentication for each one. LS also returns the address of an Authentication service that can authenticate for each of the returned authentication realms.
3. Client contacts the authentication service that manages authentication for the resource realm (R-AA-Service) and requests an authentication token approved for use in the resource realm (R-AuthRealm).
4. R-AA-Service returns a list of known (federated) authentication realms and asks the client to choose one for authenticating.
5. Client specifies desire to authenticate at F-AuthRealm (F for Foreign). (JRA5 Terminology: Client specifies @F-AuthRealm.)
6. R-AA-Service does not manage F-AuthRealm so it redirects the client to F-AA-System.
 - a. Client contacts the authentication service that manages authentication for the client-selected realm (F-AA-Service) and requests an F-AuthToken authentication token for use in R-AuthRealm (as in step 3.)
 - b. F-AA-Service returns a list of known (federated) authentication realms and asks the client to choose one for authenticating (as in step 4.)
 - c. Client specifies desire to authenticate at F-AuthRealm (as in step 5.)
 - d. F-AA-Service manages identities for F-AuthRealm, so F-AA-Service asks client for identity credentials (where credentials could be password, one-time token, finger print scan, etc.) (As in step 6, non-federated case.)
 - e. Client presents credentials (as in step 7.)
 - f. If credentials are valid, F-AA-Service creates a handle that can be used to request additional attributes about the identity subject to the attribute release policies in F-AuthRealm. This handle is returned to the client encoded as an AuthToken approved by F-AuthRealm (F-AuthToken) (as in step 8.)
7. Client presents credentials to R-AA-Service. In the federated case, the credentials can be the authentication token from a federated authentication realm, in this case F-AuthToken.
 - a. R-AA-Service needs more information about the federated identity requesting access to R-AuthRealm protected resources and makes an attribute query to F-AA-Service using the F-AuthToken handle.
 - b. F-AA-Service releases only as much information about the client identity as is allowed.
8. If credentials are valid, R-AA-Service approves the F-AuthToken for use with R-AuthRealm resources. This effectively creates an R-AuthToken but keeps the handle pointing back to the F-AA-Service for attribute queries.
9. Client requests a measurement from MP. Request includes the R-AuthToken.
10. MP requests resources from the Resource Protector service (RP). The R-AuthToken is passed along in the request.
11. RP needs more information about the identity requesting the resources and makes an attribute query. The query goes to the F-AA-Service.
12. F-AA-Service releases only as much information about the client identity as is allowed.
13. RP returns resource availability (allowed/disallowed.) This portion includes scheduling.
14. MP returns response to measurement request.

It is important to note that the AuthToken's described above have a limited lifetime, but that they are valid for use with all the services within the given authentication realm. Therefore, most of the interactions shown in the above diagrams will be done infrequently.

Some important aspects about the model shown that are not present in many authentication systems is the assumption that users may have more than one identity. This is needed to support measurements that require more than one MP where the MPs involved are not within the same authentication realm and they have no common federated relationships. When a client wants to access a resource, the AS should allow the client to make their request using whichever legitimate identity they choose for accessing this resource. The AS, however, does not have to relate these multiple identities in any way

Also, note that federations are not required to be hierarchical. Simple peer agreements are possible, although hierarchical federations should also be supported and would be beneficial, especially with regard to distributing identities within large organizations.

In the federated case, the AuthToken from the federated realm must be verified and encoded as an AuthToken for the resource realm. This way, services within a given authentication realm do not, themselves, need to have a trust relationship with the federated domains (i.e. they don't need to be able to validate a message from another domain). All inter-realm trust relationships are managed by the authentication services of the realm. In a framework such as this where there are potentially many deployed services, it is important not to force full distribution of the trust relationships.

4.2.2 Measurement between Unrelated Domains

To successfully diagnose real end to end network performance problems, it is often necessary to perform measurements across domains that may not have any direct business relationships with each other. However, the individuals trying to run those applications often do. Therefore, it is desirable to give the user the ability to run tests between MPs on completely unrelated domains. To support this, the requesting entity must have credentials for both authentication realms represented. This does not fundamentally change the authentication and authorization model represented above. The only requirement is that the client application must be able to manage multiple identities and it may need to authenticate more than once to produce a single measurement result.

4.3 Storage of the Measurement Results

As we described in the example reported in section 3, measurement results are sent from the Publisher Interface of some MP service to one (or more) Subscriber Interface(s). The latter can be directly on the client, an MA service, or a Transformation Service. The same holds for measurements spanning more than one domain, as this framework does not put any limitation on the Subscriber Interface. That is, in principle a MP can send its measurement results to a Subscriber Interface on a MA service in a foreign domain. However, authorization policies of the publisher's domain determine whether data can be published to subscribers of other domains and similarly authorization policies of the clients determine whether data from other domains can be accepted (for storage, processing, etc). For example, a measurement archive in a domain can reject to store any data coming from foreign measurement points. Such authorization policies are configured and enforced by Resource Managers and Resource Protectors.

However, there are some considerations to make about where it is more efficient to store such measurement results, in the sake of their accessibility (reusability) by users different from the one having originally requested the measurement. This is the subject of the next section.

4.4 Coordination of a Distributed Measurement Setup Process

We envisage two main use cases: a user wants to setup a measurement (and get the results) or a user wants to access measurement results of measurements initiated by other users.

In the first case, the user can setup an “on demand” measurement along with a specification of one or more Subscriber interfaces to which the results of the measurement should be sent. “On demand” measurements can be of any of the categories listed before (Single Point, Double Point, etc). Further, “on demand” doesn’t necessarily mean a single instance measurement (a measurement made only once for the given parameters). It can also be a multiple instance measurement (measurement with the same parameters made more than once for example, every 5 minutes for an hour) or a permanently scheduled measurement (measurements with same parameters made continuously until changed later on). In all these cases, the Subscriber Interface specified by the user can be on the user/client itself or it can be on a MA Service. This framework doesn’t put any limitation to the relative location of users, MPs and Subscriber Interfaces. If the user specifies an MA as one of the subscriber interfaces (to avoid interruption in the delivery of data due to a network hiccup, or to access the data in the future) the user will need the rights to do so, as it will need the right to initiate the measurement itself. All this is governed by local authorization policies in the involved entities (e.g. MPs and MAs). In the rest of this document, the single instance, multiple instance and permanent measurement requests to a MP Service will be treated in the same way, like an on-demand test.

In the second case a user wants to access measurement results of measurements established by somebody else. The measurements can be of any of the type of the above ones. To enable a wide group of consumers/users to access measurement data, it is necessary to have a set of measurements scheduled on Measurement Point Services and have their measurements stored in a Measurement Archive. The retention and access policies of the MA will determine if these measurements will be available generally or to a specific group of users. In determining a set of measurements which can be made available to a wide group of consumers, some degree of coordination, both in time and in space, may be needed. Coordination in the time dimension means that measurements have to be activated at known instants and last for known periods (including the possibility that they are “permanent” or “periodic”). Coordination in the space dimension means that they have to be taken in selected points among a couple of points of particular interest (e.g. for the amount of traffic volume, for the type of traffic, etc.).

Now come two questions: who are the entities whose coordination in setting up measurements may be beneficial? And which type of measurements is beneficial to coordinate?

In principle, any user could coordinate their measurement requests with anyone else if they wanted to avoid the duplication of a specific measurement. More realistically, this coordination will happen mostly among a very limited set of highly privileged users that are able to set up measurements that are guaranteed to be accepted by MPs, and whose results are guaranteed to be accepted (and kept for longer retention periods) in MAs. The

framework does not specify how this coordination happens. In the first stage, we envisage that it will happen off line among groups of network administrators, perhaps as part of a peering agreement. If such coordinations prove to be useful (both in end users' satisfaction and in reducing the overall number of on demand measurements), they will drive the need to define a more formal coordination protocol.

With regard to which type of measurements will be beneficial to coordinate, that will mostly be a function of the interested groups of users collaborating. For the initial probable groups of network engineers, we foresee that it will be useful to do so mainly for Double Point Intradomain and the Double Point Neighbour domain measurements. Typically, the results will be stored within a MA in the same domain as the MP in which they are taken. End users may make use of Double Point Intradomain and Double Point Neighbour Domain results in two ways:

- Combine them to get a rough performance information on an end-to-end path without establishing a dedicated end-to-end measurement.
- Use them in parallel with an end-to-end measurement that they are (or were) taking to detect the network portion responsible for performance degradation.

Single point measurements can be essentially SNMP counter retrieval, Netflow data, or dumps of packet traces. A coordination of the collection of these measurements may prove useful as well. Netflow or packet trace dumps introduces privacy concerns that will need to be dealt with by the policy controls of the MP and MA services.

Double Point Interdomain and Double Point Partitionable measurements are perhaps most applicable for virtual communities. For example, a community of researchers spread across multiple administrative domains but united in the use of a set of shared resources may desire coordinated measurements among those shared resources, thus forming a coordinated virtual community. In this case, it is most natural to designate a MA for the virtual community (hosted by one of the participants) and direct the stored results there. The results of the measurement are mainly of interest for the virtual community members (and their access may even be restricted to them).

The following table summarizes what we described in this chapter.

Type	Typical Storage
Single Point	MA of MP's Domain
Double Point, Intradomain	MA of MP's Domain
Double Point, Neighbour Domain	MA's of MP's Domains
Double Point, Partitionable	One MA hosted by one of the members of the virtual community

Double Point, Interdomain	One MA hosted by one of the members of the virtual community
---------------------------	--

5 Design of Services

This section discusses at a more detailed level the services defined previously. Internal components of services are identified and interactions between them in order to carry out specific tasks are brought out and also illustrated. However, the text contained in this section should be considered as “Work In Progress” and is expected to change with time depending on the needs. **Figure 5-1** shows the components that will be used in sequence diagrams within each service in order to illustrate interactions between components. Dashed arrows between blocks show a “possible/optional” interaction while plain arrows are expected interactions. Representations of services and components are colour coded in order to highlight the possible similarities in functionality. However, it’s not necessary that components with the same colour codes are identical.

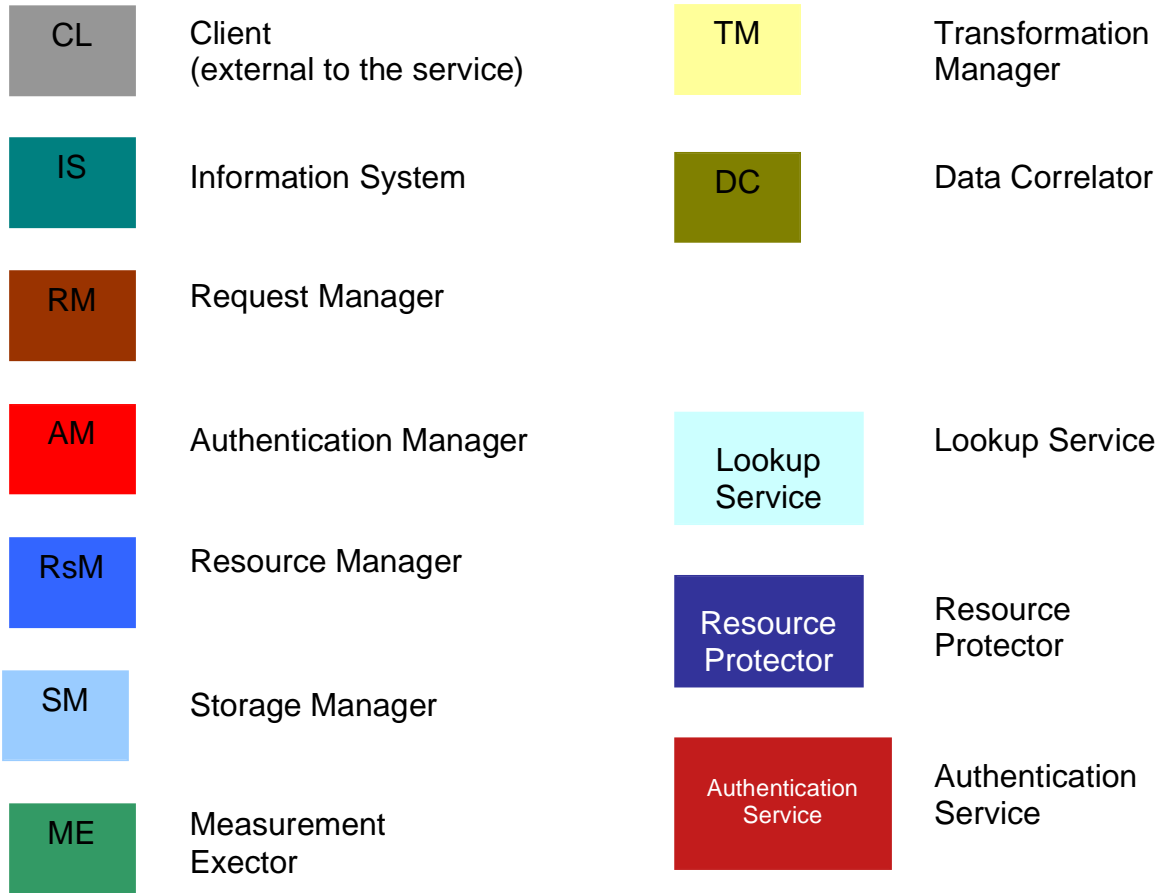


Figure 5-1: Introduction to conventions used in the section

5.1 Measurement Point Service

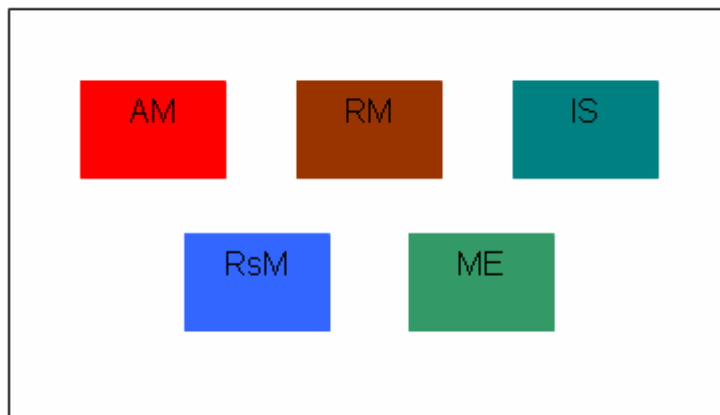


Figure 5-2: MP Service components

5.1.1 Specification of Tasks

The main tasks of the MP Service include

- **Making network measurements**
- **Publishing measurement data**
- **Allowing management of the service**

The above tasks are mostly initiated by requests coming from outside the service. The service will need to have the capability to accept the following requests via its interfaces

- Measurement requests
- Measurement data subscription requests
- Queries for capabilities, status and logs
- Management interactions

The service might have to act as a client to other entities/services to achieve some of its tasks. Below are a few situations where it will have to initiate communication with other entities.

- Register and Deregister itself
- Publish data
- Report errors
- Provide status updates

Many internal tasks would be necessary to achieve its main tasks. They are:

- Authorize requests
- Manage Resources
- Schedule Measurements
- Conduct Measurements

5.1.2 Components

The MP Service is an infrastructure consisting of several components, as shown in **Figure 5-2**. These components include:

- The Request Manager component which exchanges all requested information with outside services and other MP Service components
- The Information System component which keeps descriptive and configuration information
- The Authentication Manager which authenticates requests. It accepts authentication tokens and ensures that they are valid

Project:	GN2
Deliverable Number:	DJ.1.2.1
Date of Issue:	02/06/05
EC Contract No.:	511082
Document Code:	GN2-05-057v5

- The Resource Manager component which determines whether a proposed schedule satisfies resource constraints (these constraints are defined by configured policy and also depend on actual status of the network). Requests are analyzed and the authorization of the requests and granting of resources are based upon the rights granted to the resource requestor as defined by policy. Depending on the configuration, the Resource Manager would either take a decision by itself or communicate with an external Resource Protector Service. The Resource Manager can then be seen as a local Resource Protector.. The resource manager will contact one or several external Resource protector services (level 1), which can also contact one or several resources protectors services (level 2) and so on.... The Resource Protector of each level plays a pre-defined role. For example, Resource Protector at level 1 could be in charge of all the resources within the domain. Resource Protector at Level 2 could have a decision role based on policy of a wider domain or set of domains.
- The Measurement Executor component which starts the measurement and directs the data to the Request Manager component to be distributed to all subscribers

5.1.3 Dependencies

The MP Service is dependent upon:

- Authentication Service: The MP Service accepts one or more token types from one or more Authentication Services, effectively delegating authentication, but not authorization. Also, to determine if a particular resource requestor is allowed access to a given resource, the MP Service may need to query the Authentication Service for more specific information about the requestor.
- MA Service: The MP Service does not store data. Instead, it pushes data to all subscribers. In most practical applications there will be a local MA service, based on the same physical box as the MP service, subscribing to the results, to buffer data in case of a network partition or in case of network congestion.
- Lookup Service: The MP Service must register its existence and capabilities with one or more lookup services. It must also register in the lookup service which Authentication service(s) it trusts for authentication.
- Resource Protector Service: If any of the resources needed to perform the measurement are shared resources, a Resource Protector should be used to manage them.

5.1.4 Data Types

The MP Service deals with five types of data:

- Monitoring Data: Generated by the Measurement Executor and sent to subscribers (e.g. MA or the requestor).

- Authentication Tokens: Validated by the Authentication Manager.
- Capability Data: Specifies the types of measurements that can be gathered.
- Configuration Data: Specifies the required authorization token type and the topological placement of the MP Service.
- Resource Availability Data: Specifies whether a particular resource is available or not.

5.1.5 Interfaces and Interactions

There are two different categories of interface: ones which handle interactions between the MP Service and external entities and the ones which handle interactions between the components of the MP Service.

5.1.5.1 *Service Interfaces and Interactions*

Interactions between the service and its clients are handled by the following components:

- The Request Manager, which receives measurement requests and subscription requests from external clients and publishes measurement data to the subscribers.
- The Resource Manager, which can be configured to send resource availability and authorization requests to Resource Protector Service.
- The Information System, which publishes the MP's metadata.
- The Measurement Executor is the only component in the system that communicates with measurement tools. It must therefore understand syntax and semantics of the input and output interfaces of each measurement tool. It communicates over one common interface with the rest of the building blocks. The definition of this common interface is out of the scope of this general framework design. However, we must keep in mind that this common interface must be general enough to convey all necessary input and output information to and from all measurement tools that we want to integrate into the system. This information includes measurement results, requests to start or stop measurement, request to reconfigure the measurement tool, etc.

5.1.5.2 *Component Interfaces and Interactions*

Information exchanges between components are usually initiated by requests from clients to the service. Most of these interactions are illustrated using the following figures. Requests dealing with administration of the service are not discussed in this sub-section.

Measurement and subscription requests are received by the Request Manager and sent for verification to the Authentication Manager component. If the Authentication Manager requires more attributes of the user, depending on its configuration, it will either interact with the Authentication Service to get these attributes or it will take a local decision on whether to accept the request or not. If authenticated, the measurement schedule request is passed to the Resource Manager. The Resource Manager, depending on its policy, will either authorize the request and reserve resources to make a measurement or reject it. This policy of the Resource Manager can depend on the requestor (role, domain, etc) and also on the availability of local resources (CPU, bandwidth, no other measurements scheduled, etc.) The Resource Manager can be configured to depend on a Resource Protector Service to make this decision. In this case, the policies of the Resource Protector Service will be applied in addition to the policies of the local Resource Manager. Policies of a Resource Protector include more general factors which are influenced by the state of other resources that it governs. A common example of such factors would be a resource being used by other types of measurements. In this case, if two different types of measurements carried out between two different pairs of Measurement Point Services compete for the same resource, the Resource Protector Service, which can be seen as a central entity by these Measurement Point Services will be used for management of such resources. Resource Manager will accept a request only if one or more of the Resource Protector Services that it is configured to communicate with agree to the request. When a scheduled test is due, the Measurement Executor performs the measurement or otherwise acquires the measurement data.

The Request Manager is responsible for publishing the data to authorized subscribers. **Figure 5-3** illustrates how the service handles requests for measurements and **Figure 5-4** shows how measurements are made and data published to subscribers. **Figure 5-5** illustrates the process of registering, de-registering and providing keep-alives to a Lookup Service. This figure (**Figure 5-5**) is commonly found under most services defined later on.

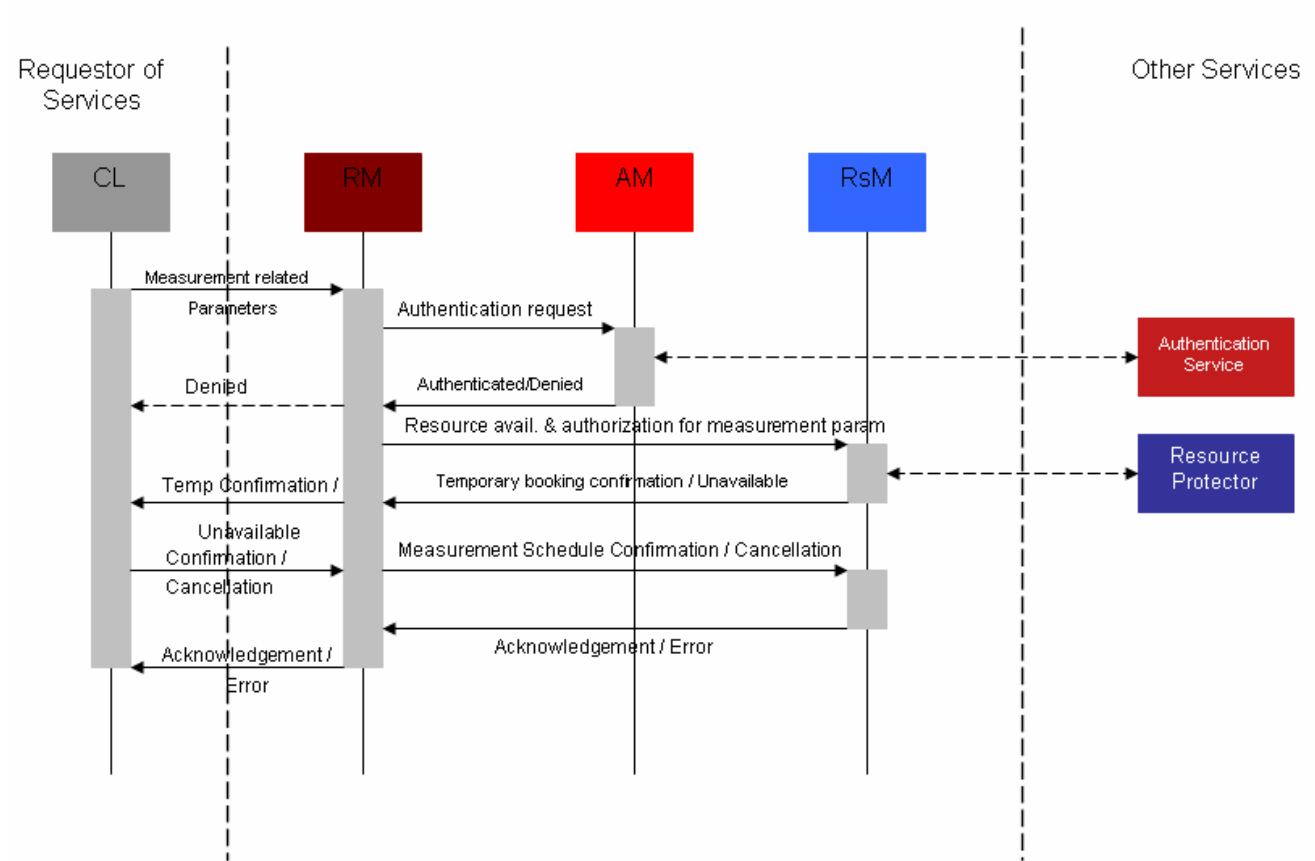


Figure 5-3: Requesting a measurement from a MP Service

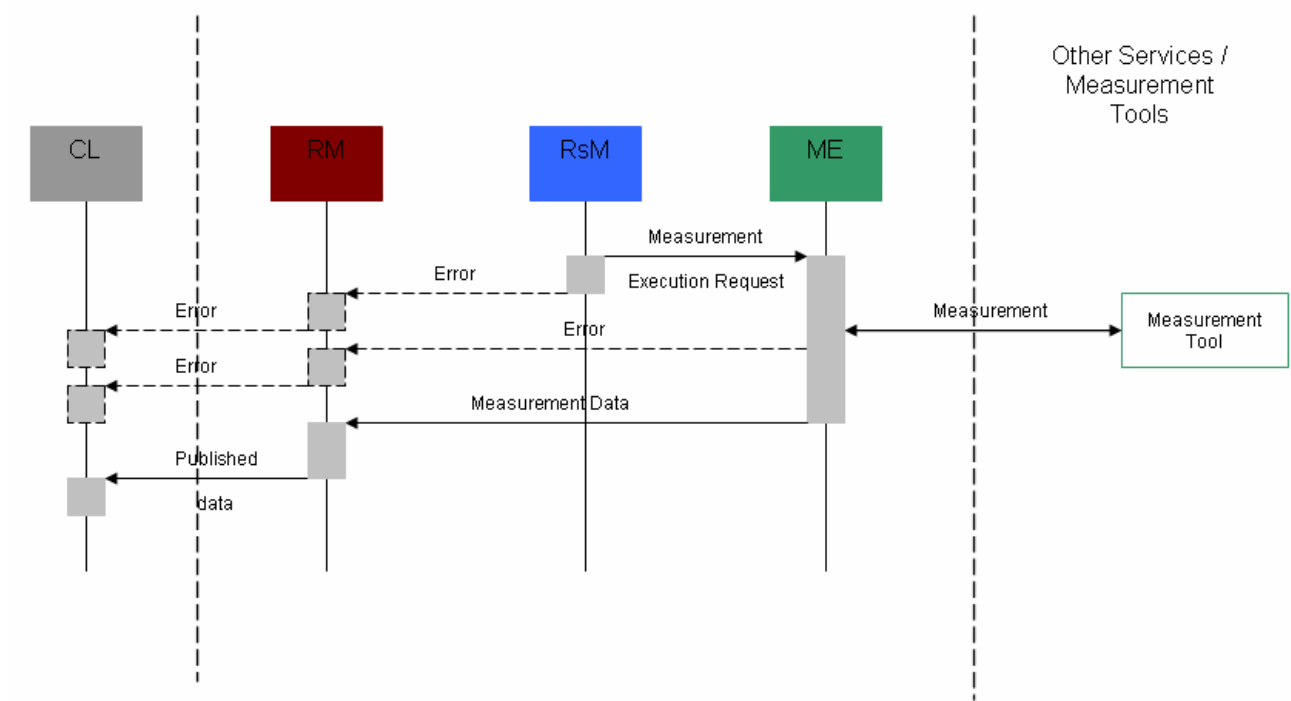


Figure 5-4: Making a measurement and providing data to the subscribers

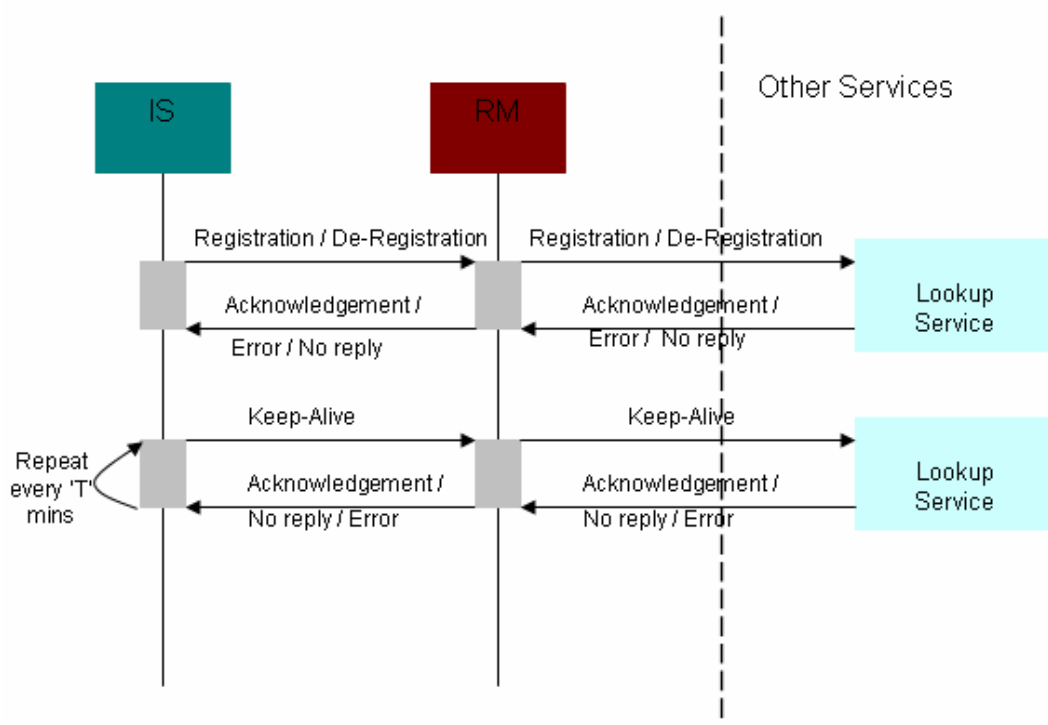


Figure 5-5: Registering, De-registering and Keep-Alives

5.2 Measurement Archive Service

The Measurement Archive service contains all the network measurement data retrieved from the monitoring architecture. The internal components of the Measurement Archive are shown in **Figure 5-6**.

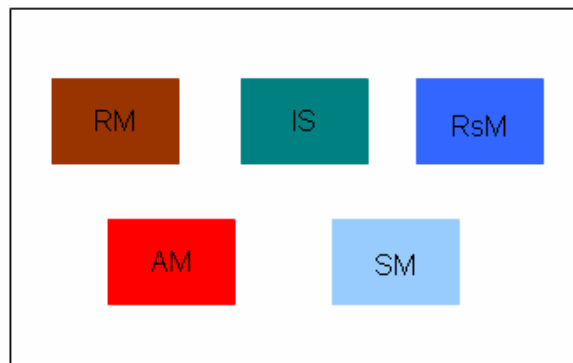


Figure 5-6: MA internal components

5.2.1 Specification of Tasks

The main tasks of a MA Service include

- **Storing network measurement data**
- **Retrieving network measurement data**
- **Allowing management of the service**

The service will need to have the capability to accept the following requests via its interfaces

- Requests for retrieval of measurement data, meta data and content information
- Measurement data storage requests
- Queries for capabilities, status and logs
- Management interactions

The service will have to initiate communications with external entities to achieve the following

- Register and Deregister itself
- Subscribe to measurement data published by MP Services

Internal tasks that might be necessary to achieve its main tasks are:

Project:	GN2
Deliverable Number:	DJ.1.2.1
Date of Issue:	02/06/05
EC Contract No.:	511082
Document Code:	GN2-05-057v5

- Validate requests
- Manage and Authorize Resources
- Management of stored data information

5.2.2 Components

The MA Service is an infrastructure consisting of several components, as shown in **Figure 5-6**. These components include:

- The Request Manager, which handles all measurement data related requests and replies with external entities.
- The Information System which keeps descriptive and configuration information.
- The Authentication Manager, which authenticates requests. It accepts tokens of authentication and ensures that they are valid. If more information about the requestor is required, it communicates with the Authentication Service to retrieve it.
- The Resource Manager component determines whether the resource is constrained by policy or availability. Requests are analyzed and access to resources (measurement data) is granted depending on the rights granted to the resource requestor as defined by policy. Usage of other resources (hardware) can also be configured into the Resource Manager. Optionally but not necessarily, the Resource Manager can rely on an external Resource Protector Service. The Resource Manager can then be seen as a local Resource Protector as explained in section 5.1.2. One of the reasons for configuring the Resource Manager to query an external Resource Protector is if a given administrative domain, for administrative purposes, wants to centralize the placement of policy definitions.
- The Storage Manager, which stores and retrieves data, as requested and if authorized. (Effectively, this is a wrapper around the local database.) It must maintain state information about where data is stored (database software, database and table), so that it will be able to fulfil the requests. This state information may be in a database directly maintained by this Storage Manager component and updated by it with each data storage request.

5.2.3 Dependencies

The MA Service is dependent upon:

- Authentication Service: The MA Service accepts one or more token types from one or more Authentication services, effectively delegating authentication, but not authorization. Also, to determine if a particular resource requestor is allowed access to a given resource, the MA service may need to query the Authentication Service for more specific information about the requestor.

- **MP Service / Transformation Service:** The MA Service does not create data. Instead, it receives data from MP and/or Transformation service(s).
- **Lookup Service:** The MA Service must register its existence and capabilities with one or more Lookup Services. It must also register which Authentication Service(s) it trusts for authentication.
- **Resource protector:** The MA could use a Resource Protector Service as a way of centralizing the management of policy definitions.

5.2.4 Data Types

The MA Service deals with five types of data:

- **Measurement Data:** Received from MP Service(s) and/or Transformation Service(s).
- **Authentication Tokens:** Validated by the Authentication Manager.
- **Capability Data:** Specifies the types of data that can be stored and/or retrieved.
- **Configuration Data:** Specifies the required authorization token type and MP Services with which it is associated.

5.2.5 Interfaces and Interactions

There are two different categories of interface: interactions between the MA Service and external entities and interactions within the components of the MA Service.

5.2.5.1 *Service Interfaces and Interactions*

Interactions with external entities are carried out by the following components:

- **Request Manager,** which handles measurement data related requests such as storage and retrieval of measurement data and provides appropriate requests.
- **Information Service,** which registers (and de-registers) service information with a Lookup Service. It also provides 'keep-alive' messages in order to refresh its information in the Lookup Service.
- **Authentication Service,** which communicates with the Authentication Service when it requires more information about the requestor.
- **Resource Manager,** which if configured, communicates with a Resource Protector for authorizing requests

5.2.5.2 Component Interfaces and Interactions

Figure 5-7 illustrates interactions between components when the service is requested for measurement data. Similar interactions are expected when requests for storage of measurement data are issued to the service. Requests dealing with administrative aspects of the service aren't discussed in this sub-section.

Received data and publication requests follow more or less the same schema as outlined for the Measurement Point (section 5.1.5.2). They are sent for validation of authentication tokens to the Authentication Manager component. If the Authentication Manager requires more information about the requestor, it contacts the Authentication Service. Authorization and Resource Availability are then taken care of by the Resource Manager. The Resource Manager, if configured, would request the Resource Protector in order to take a decision about the request. Once the request has been authenticated and authorized, it is passed on to the storage manager, which will then return measurement data (if it exists) from its data store.

Registration, De-registration and Keep-Alive messages are initiated by the Information System component as illustrated in **Figure 5-8: Registering, De-registering and Keep-Alives** and similar to the definition of the same process under Measurement Point Service.

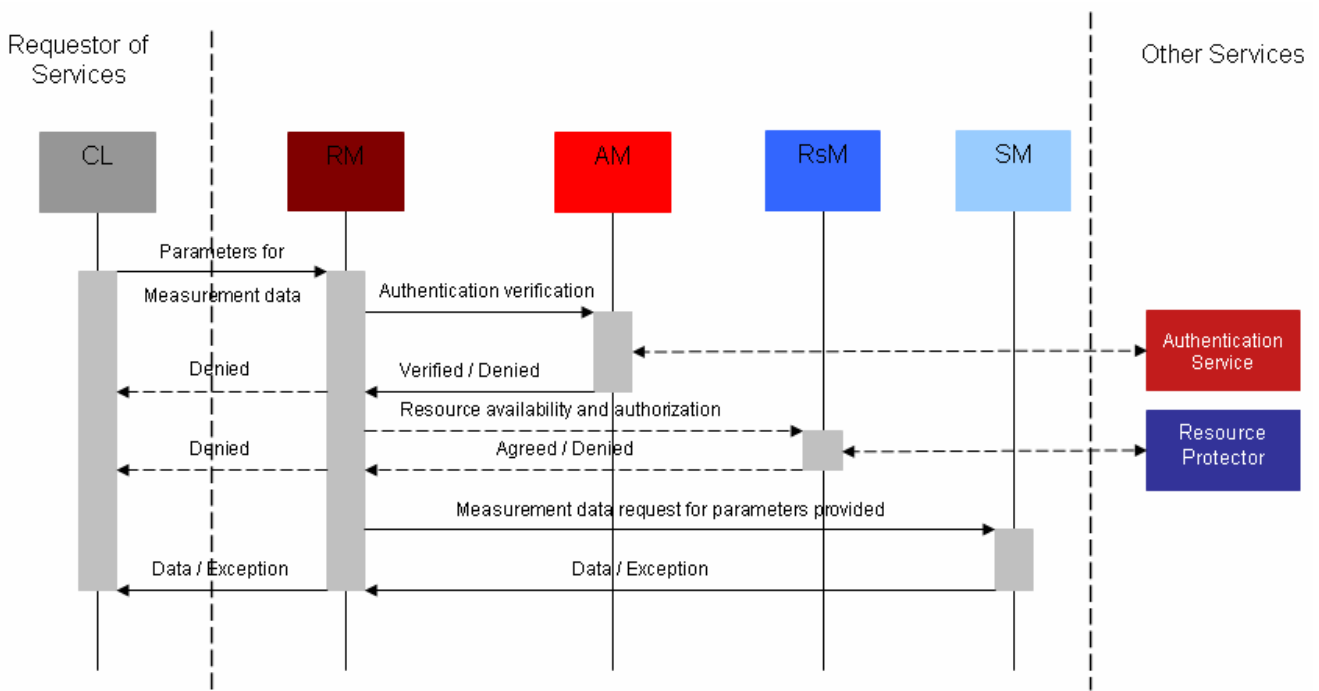


Figure 5-7: Requesting data from a Measurement Archive service

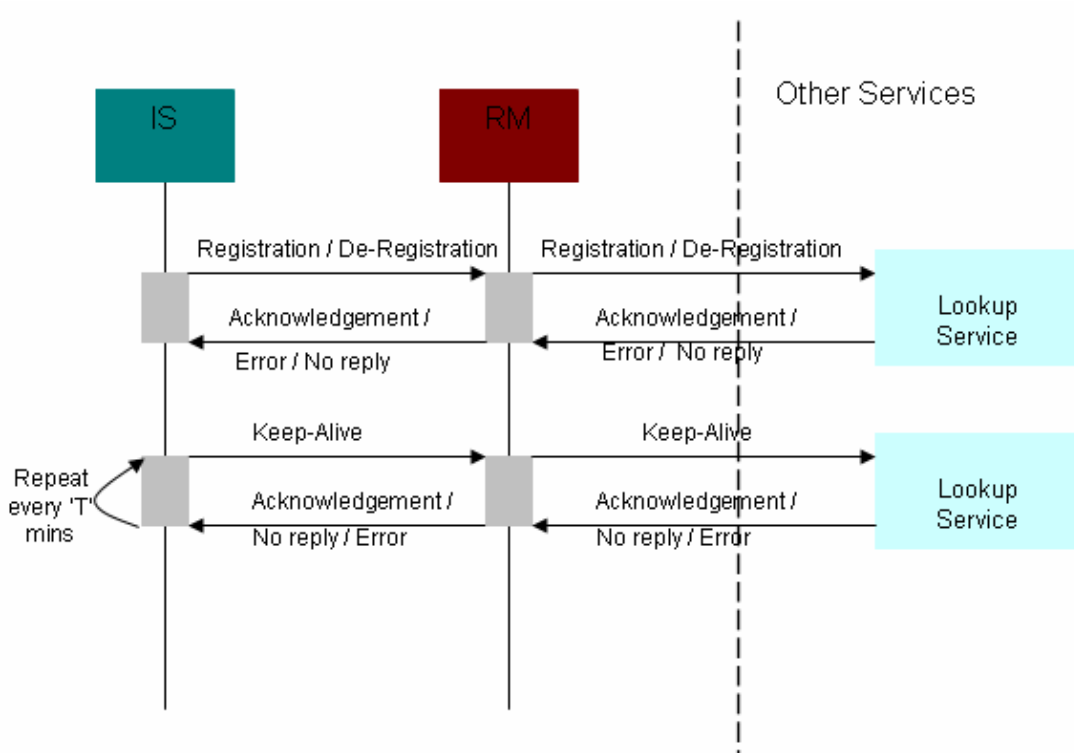


Figure 5-8: Registering, De-registering and Keep-Alives

5.3 Lookup Service

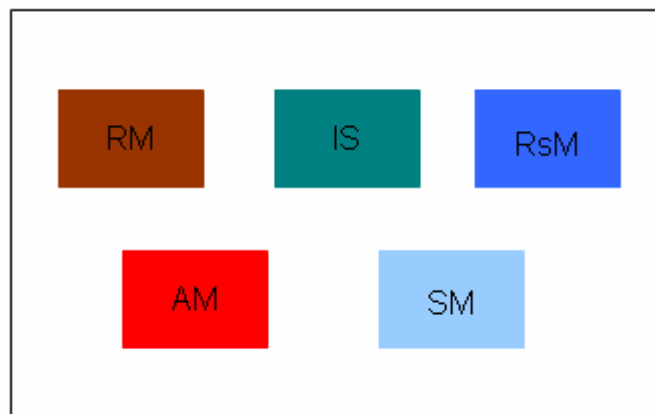


Figure 5-9: Lookup Service internal components

5.3.1 Specification of Tasks

The main tasks of a Lookup Service include

- **Storing, retrieving and refreshing information about other Services**
- **Allowing management of the service**

To do the above, the service will need to accept the following requests via its interfaces

- Requests for information on available services (Lookup Requests)
- Registration and De-registration requests
- Information refresh requests from other services (Keep-Alive messages)
- Management interactions (configurations, status, logs)
- Advertisements from other Lookup Services announcing their existence

In addition, the Lookup Service might need to make the following requests to external entities

- Requests for service related information (delegation of requests)
- Advertising its own presence to other Lookup Services

5.3.2 Components

There are several different elementary components that the Lookup Service relies upon to perform its tasks. They are represented in **Figure 5-9**.

- Request Manager, which handles Lookup requests and also Registration, De-registration and Keep-Alive messages
- Authentication component, which validates authentication tokens for certain types of requests. It also communicates with the Authentication Service if extra information about the requestor is required
- Resource manager, which if required for certain types of requests, provides necessary authorization. Communication with an external Resource Protector is possible but not expected
- Information System, which communicates with other Lookup Services (registration, de-registration and keep-alive messages)
- Storage Manager, which stores and retrieves information about services

5.3.3 Dependencies

Lookup services are not dependant on other services for their successful operation. However, a Lookup Service does require other services to register with it in order to function usefully. If a Lookup Service is configured to communicate with other Lookup Services, it will forward any unanswerable requests to such services but won't be dependent on them for it's functioning.

5.3.4 Data Types

The Lookup Service handles the following types of data:

- **Registration data:** Requests for Lookups, Registration requests, Keep-Alive messages and De-registration requests are directly linked to registration data set that a Lookup Service stores.
- **Authentication Information:** Authentication Information is provided by certain lookup requests, in certain cases for sensitive data and also for requests dealing with registration, de-registration and keep-alives. Tokens of authentication are provided during these requests, which will need to be validated by the Lookup Service.

The lookup information consists of the description of the service (MP, MA, AS, etc.). This description is provided by services themselves when they register with the Lookup Service. The information provided by services has fields, which can be seen as common among all services, as well as fields, which are specific to a particular type of service. A broad classification of all the fields that services can provide is seen below.

- **Service Provider Information:** Contains information about the provider. For example, Service provider's name, geographical location, contact information, domain name, etc.
- **Service Information:** Contains information about the service being provided such as Service type, Service parameters, etc. and also information about Service Access. For example, Service Access Point URI, Access type, etc.
- **Technology Information:** Contains information about the type of technology to be used for communications. For example, SOAP, type of schemas, version numbers, etc.

The Lookup Service will link an expiry date for all registrations received. Services will have to refresh their registration through keep-alive messages before this expiry date.

5.3.5 External Interactions

Both service providers and service consumers will contact the Lookup Service. Additionally, the Lookup Service itself is a service provider so it will participate in some of the same interactions as other service providers when

Project:	GN2
Deliverable Number:	DJ.1.2.1
Date of Issue:	02/06/05
EC Contract No.:	511082
Document Code:	GN2-05-057v5

it registers with other Lookup Services. This service will be central to the join and discovery protocols for the network of services.

5.3.5.1 Service Interfaces & Interactions

- The services in the system will register with the Lookup Service. These services (which are clients of the Lookup Service) will provide detailed information about the services they offer along with the information about authentication they require. They will effectively lease space in the directory and will have to refresh their entries to keep them current.
- Clients will make queries to the Lookup Service to determine how to find needed resources. These queries can ask for all or a portion of the information stored about a particular Service offering.
- Peer Interactions: a Lookup Service can peer with other Lookup Services to achieve the following
 - Provide wider access to the information stored within them. When other Lookup Services receive requests that they do not have data for, they delegate these requests to all such registered Lookup Services. Similarly, when requests for information about another domain is received to a local Lookup Service, the service can retrieve information from other domains through such registrations.
 - It is suitable for having a resilient architecture in case of failures.
- Protocols: the following protocols are defined for the lookup service:
 - The Join protocol defines the way services advertise themselves in a known lookup service. The format of messages for this protocol will be defined at a latter stage. It can be noted that a service can Join more than one Lookup service. One can note that this is not related to the lookup service discovery mechanism.
 - The Lookup protocol defines the queries a client can make when looking for a resource.
 - The discovery protocol is an optional protocol, making it possible for services to discover automatically one or several lookup services. When this discovery protocol is not implemented, services register a statically configured lookup service..

5.3.5.2 Component Interfaces and Interactions

Interactions between components are grouped together based on relevance and discussed below.

- Lookup Information registration and refreshals

All services register their lookup information in the Lookup Service. They also send updates of the information regularly. Lookup Service stores the information in its database and checks if all lookup entries

are up-to-date. If any entry is deprecated it should be removed from the database or an update should obsolete it. Fig. 5-10 illustrates these processes.

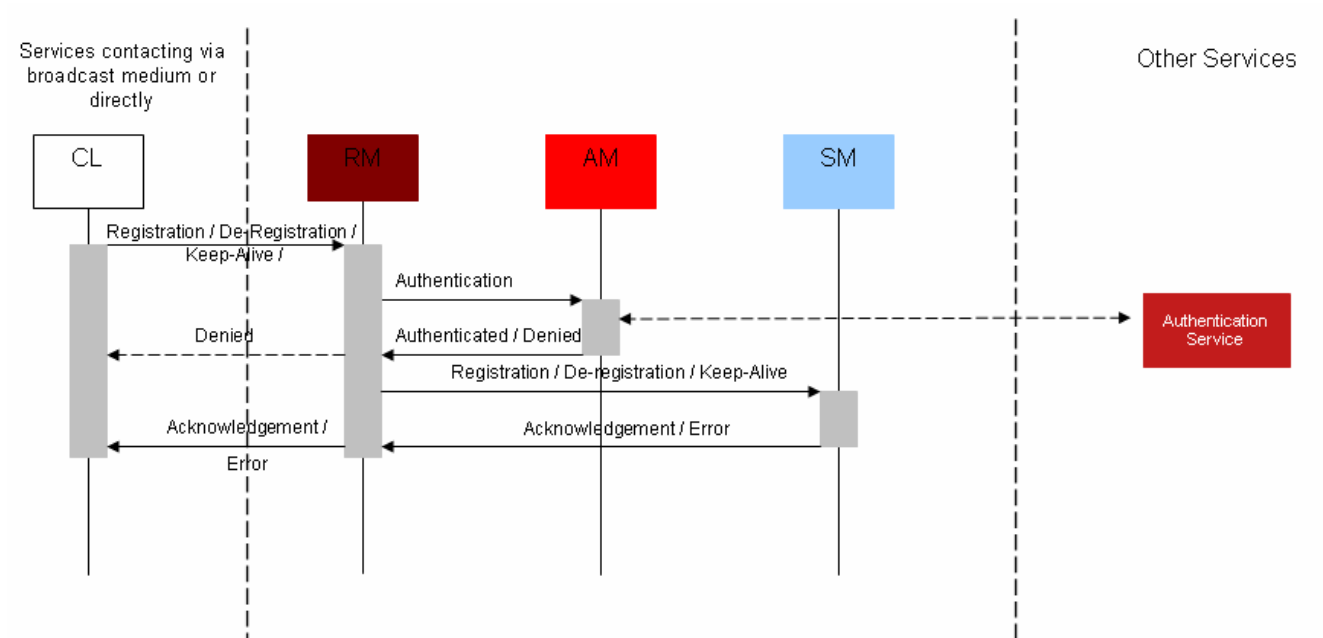


Figure 5-10: Registering / De-registering and Keep-Alives with a Lookup Service

- Lookup Information Storage
 - Lookup information database
All data obtained from a service registration message should be stored in the database. Database may be either placed locally or distributed.
 - Cache
Lookup information, which is accessed frequently may be placed in local cache. The deployment of cache is very important when a part of the lookup information is stored externally (distributed database or even other Lookup Service). Cache lets the Lookup Service response faster to requests.
 - Data refreshing
Services register themselves in the Lookup Service and then send updates regularly. Lookup information about such services has a special time stamp which let the Lookup Service determine if the information is still up-to-date. If a service stops working it won't send update information and after some time the Lookup Service will determine its lookup information as out of date.
- Lookup Information Retrieval

The Lookup Service should process client's queries, look through its data store (and cache) and respond to the user with the requested service information. The lookup information may be stored either locally or

externally. If the Lookup Service determines that the information requested by a user is not local, it can either:

- Forward the incoming request to the Lookup Service that has the lookup information
- Reply to the client that the information cannot be provided from the server requested. In this case, the requested server must give the client the necessary information to contact the Lookup Service that contains the information needed. As explained in section 4.1, all lookup services know each other, through a peer-to-peer infrastructure.

Figure 5-11 shows the process of finding other Lookup Services. Figure 5-12 and 5-13 show the processes of retrieving data from the local data store and contacting other Lookup Services respectively in order to satisfy a client's request.

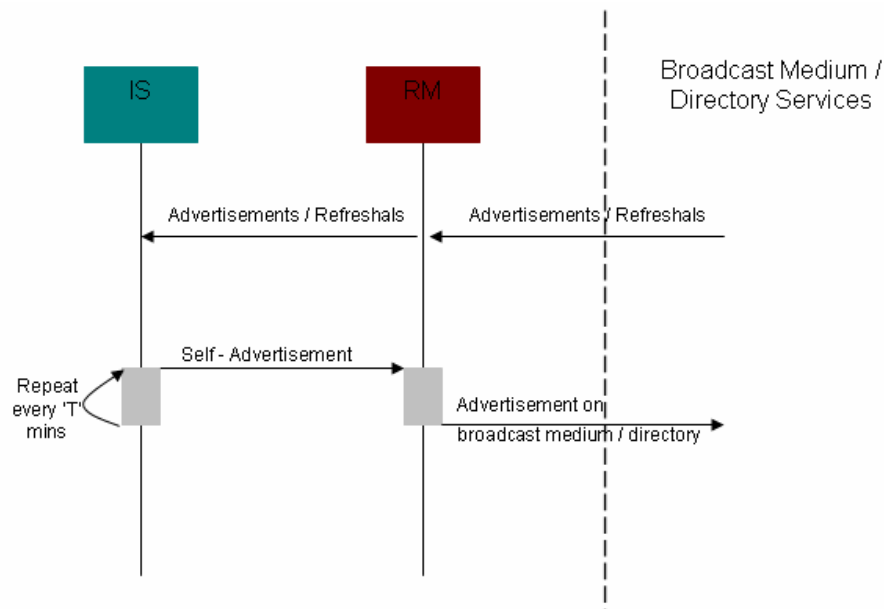


Figure 5-11: Find other Lookup Services

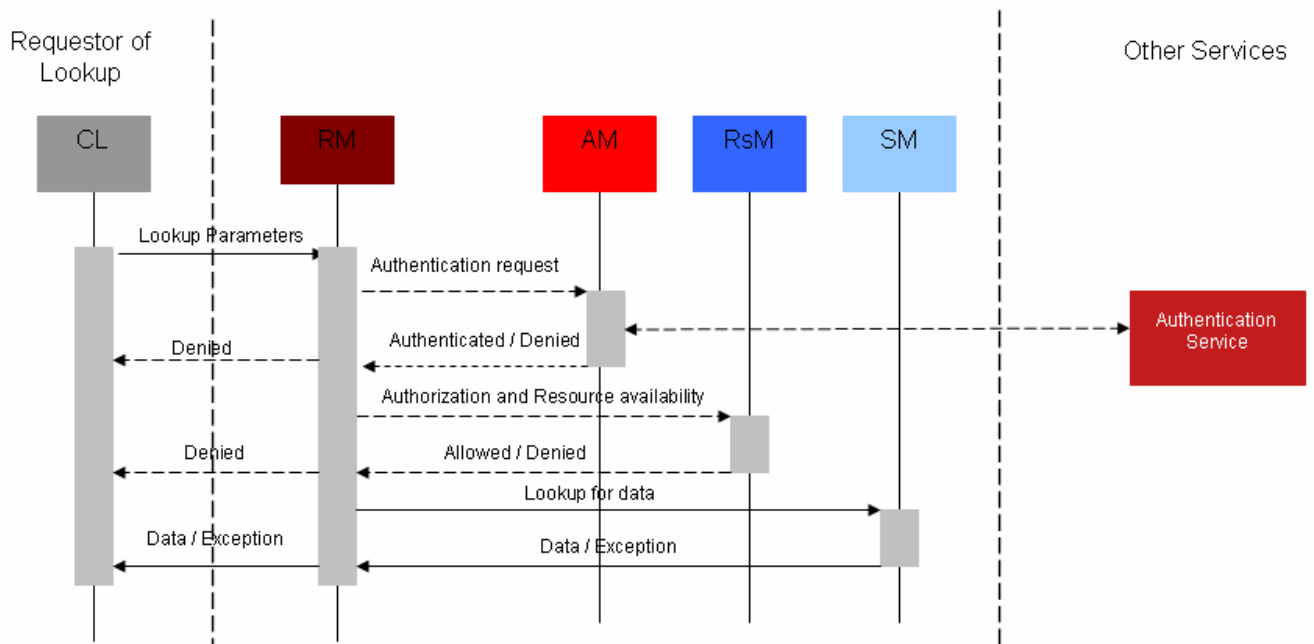


Figure 5-12: Querying for information

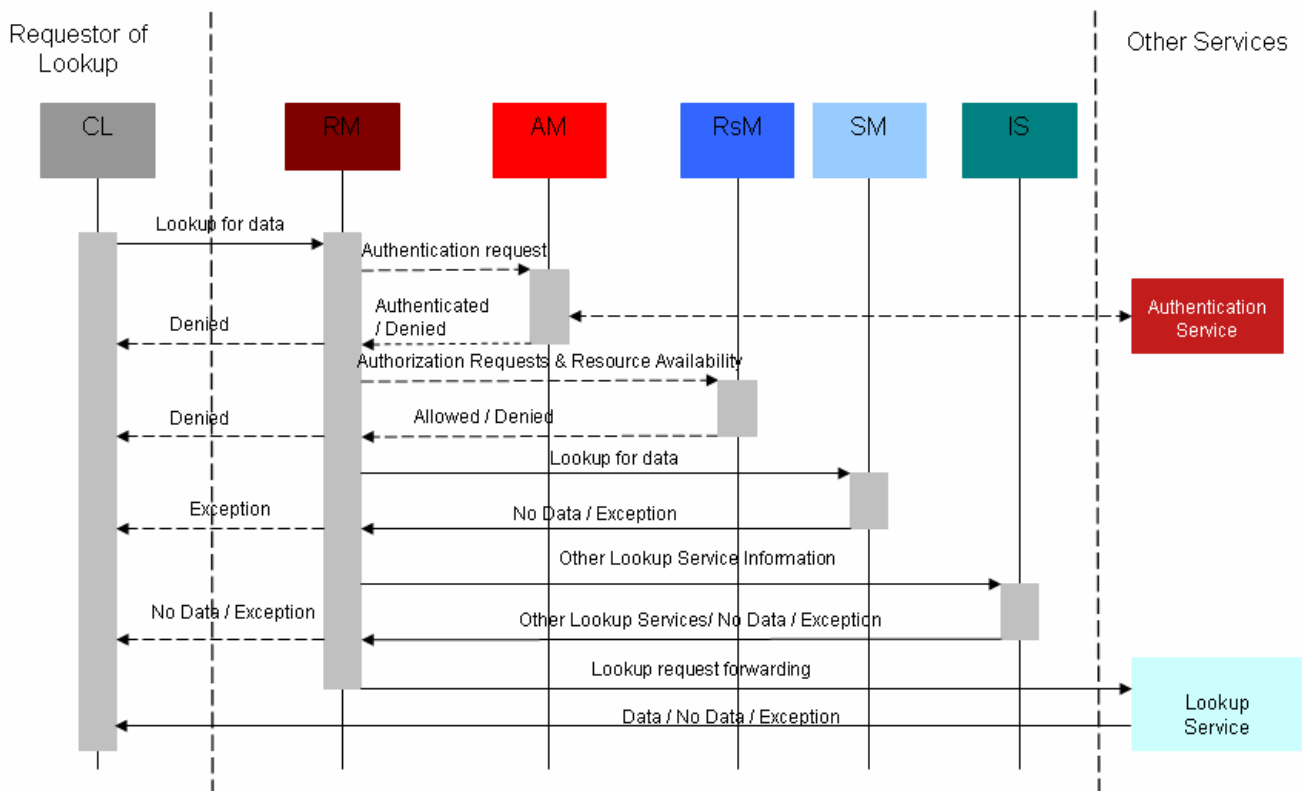


Figure 5-13: Query other Lookup Services

Project:	GN2
Deliverable Number:	DJ.1.2.1
Date of Issue:	02/06/05
EC Contract No.:	511082
Document Code:	GN2-05-057v5

- Management Interactions

An administrator of the Lookup Service should have the possibility to configure and control its work (e.g. through a web interface). The administrator may control lookup information stored in the database: add static entries, remove or modify existing ones. There may be also a possibility to influence some actions triggered by certain events (e.g. not let the Lookup Service remove some entries when they're out of date). Any more details about management interactions aren't discussed in this document.

5.4 Authentication Service



Figure 5-14: Authentication Service internal components

5.4.1 Specification of tasks

The main tasks of the Authentication service include:

- Authenticating clients and services
- Managing attribute information about specific identities
- Managing trust relationships in federations with other authentication services
- Allowing management of the service

The above tasks are mostly initiated by requests coming from outside the service. The service will need to have the capability to accept the following requests via its interfaces

- Authentication requests
- Attribute requests
- Queries for capabilities, status and logs
- Management interactions

The service might need to communicate with other entities to perform the following

- Register and Deregister itself
- Query other Authentication Servers for the attributes of a federated identity
- Report errors
- Provide status updates

5.4.2 Components of the Authentication Service

There are several different elementary components that the Authentication Service relies upon to perform its tasks. They are represented on **Figure 5-14**.

- Request Manager which handles all communications between the service and external entities
- Authentication component authenticates requests. It accepts tokens of authentication and ensures that they are valid. Because this service issues authentication tokens, this functionality will be used primarily to validate the requests coming from entities that have been authenticated by another authentication server that is part of the same federation.
- Resource Manager, which would determine if an identity from a federated authentication realm is authorized to receive an identity for this authentication realm. These decisions are taken based on configured policy and original identity of the client. The policy also dictates the type of identity that can be granted to the requestor. If the requestor is from its realm itself, then the Resource Manager associates an identity with the requestor, again based on policy.
- Storage Manager for storing identity and attribute information about those identities.

5.4.3 Dependencies

All services and clients within the framework will need to be authenticated. Therefore all services will contact the Authentication Service if they need to make any requests to other services.

The Authentication Service is dependent upon:

- Lookup Service: The Authentication Service must register its existence and capabilities with one or more Lookup Services. (Capability in this sense means it must indicate which authentication realms it is authoritative for).

5.4.4 Interfaces and Interactions

The Authentication Service in this framework is expected to be provided by JRA-5. For this reason, interfaces and component interactions will not be dealt with in this document.

5.4.5 Authentication service information

The Authentication Service is responsible for the AuthN and attribute release functionality with respect to the other services. Authentication (AuthN) also includes gathering WAYF (Where Are You From) information about requesters that choose to authenticate from federated domains (or realms). In order to do this, the Authentication Service will have to retrieve attribute information about the user from the federated realm using the tokens provided by the Authentication Service of the federated realms. These pieces can be done in part or whole by other authentication implementations and it is expected that the Authentication Service in the framework will either be the same as provided by JRA-5 or that it will act as a bridge so that the JRA-5 authentication solution can be used by the framework.

5.5 Resource Protector Service

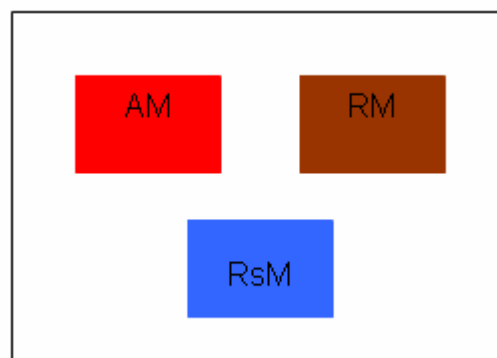


Figure 5-15: Resource Protector internal components

5.5.1 Specification of Tasks

The main task of the Resource Protector Service includes:

- Managing resources. This entails tracking resource consumption to determine availability as well as authorizing use by particular resource requestors. Time dependent resources will require scheduling functionality to determine availability.

The service will need to have the capability to accept the following requests via its interfaces

Project:	GN2
Deliverable Number:	DJ.1.2.1
Date of Issue:	02/06/05
EC Contract No.:	511082
Document Code:	GN2-05-057v5

- Authorize requests
- Queries for capabilities, status and logs
- Management interactions

The service might need to communicate with other entities to perform the following

- Register and Deregister itself
- Query Authentication Servers for the attributes of an identity (to determine the identities entitlement to the resource)
- Report errors
- Provide status updates
- Resource requests. Resource Protectors can be chained together so resource consumption can be aggregated. This allows resource owners to enforce policies across multiple real resources.

5.5.2 Components of the Resource Protector

There are several different elementary components that the Resource Protector relies upon to perform its tasks. They are represented on **Figure 5-15**.

- Request Manager which handles all communications between the service and external entities
- Authentication Manager authenticates requests. It accepts authentication tokens and ensures that they are valid.
- Resource Manager which authorize or not the request based on policy. The Resource Manager can contact an upper level Resource Protector for taking a decision.

5.5.3 Dependencies

The Resource Protector will be configured to allow specific resources to be consumed by specific roles. Information about the role of the requestor, which is identified through the Authentication token presented by the requestor, is passed along with the request. Authorization can then be made based on credentials locally applied for the requestor.

The Resource Protector is dependent upon:

- Authentication Service: The Resource Protector accepts one or more token types from one or more Authentication services, effectively delegating authentication. To determine if a particular resource requestor is allowed access to a given resource, the Resource Protector Service may need to query the Authentication Service for more specific information about the requestor.

- Lookup Service: The Resource Protector must register its existence and capabilities with one or more Lookup Services.
- Other Resource protectors: A Resource Protector can be configured to contact another Resource Protector to take a decision.

5.5.4 Data Types

The Resource Protector deals with three types of data:

- Resource Information: Information about utilization of resources, available resources and also policies on resource usage. These policies include authorization information as well.
- Authentication Tokens: Validated by the Authentication component.
- Capability Data: Specifies the types of data that can be stored and/or retrieved.

5.5.5 Interfaces and Interactions

There are two different categories of interactions: The ones between the Resource Protector and external entities and interactions within the components of the Resource Protector.

5.5.5.1 Service Interfaces and Interactions

The Request Manager exchanges information with external entities. The Request Manager accepts requests for resource availability and authorization to use the resource. Based on its configuration, the Resource Manager could request permission from other higher level Resource Protectors. Other information includes registering and refreshing information about its capabilities and status.

5.5.5.2 Component Interfaces and Interactions

Information exchanges between components are illustrated with the help of a few sequence diagrams. Requests for resource availability and authorization are treated as shown in **Figure 5-16**. In case a higher level Resource Protector needs to be contacted for permission, this is carried out as shown in **Figure 5-17**.

5.5.6 Resource Protector Information

The Resource Protector is responsible for the authorisation (AuthZ) and resource management of the framework. Multiple resources are required by many measurements and different types of measurements share some of these resources. In such cases, authorization is required at a more centralized area. This allows easy

management of such requests and also simplified processing of requests for such resources. For the above mentioned reasons, Resource Protectors are allowed to be chained. However, it will be important in deployment to only protect the resources that truly need protection or the performance of the framework will suffer. The optional interaction between the AM component and the Authentication Service is for requesting some more information (attributes) about the user, functional to take an authorization decision.

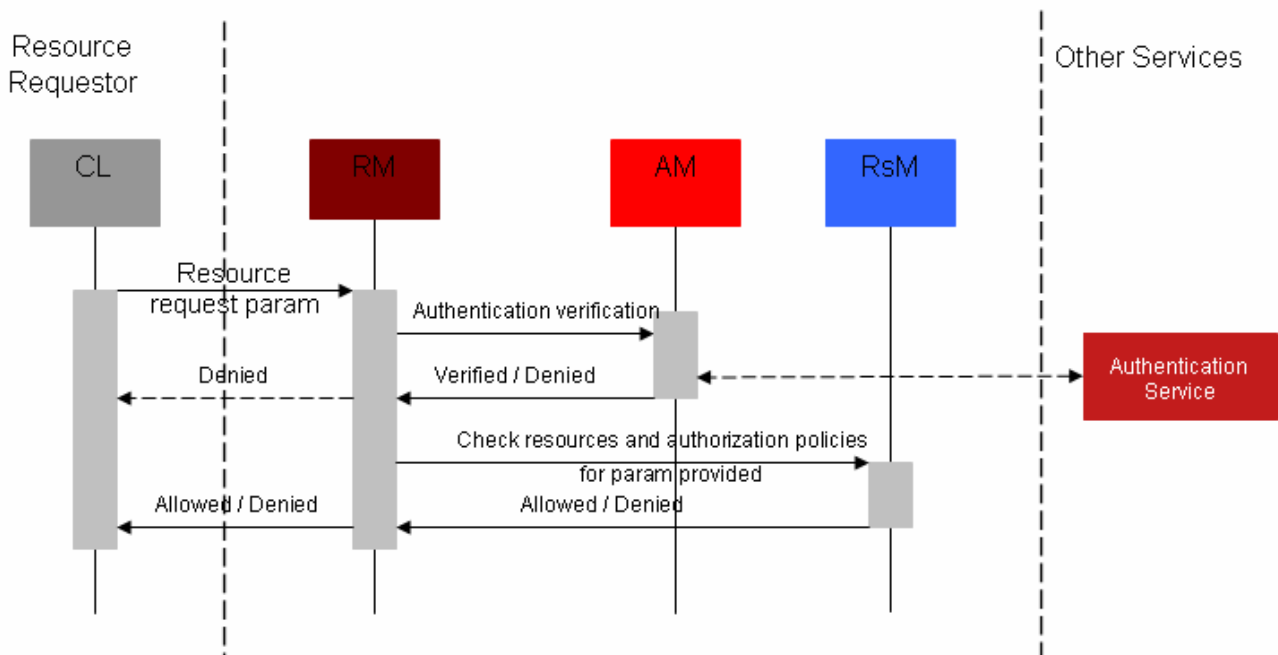


Figure 5-16: Resource Protector – Requesting resource availability

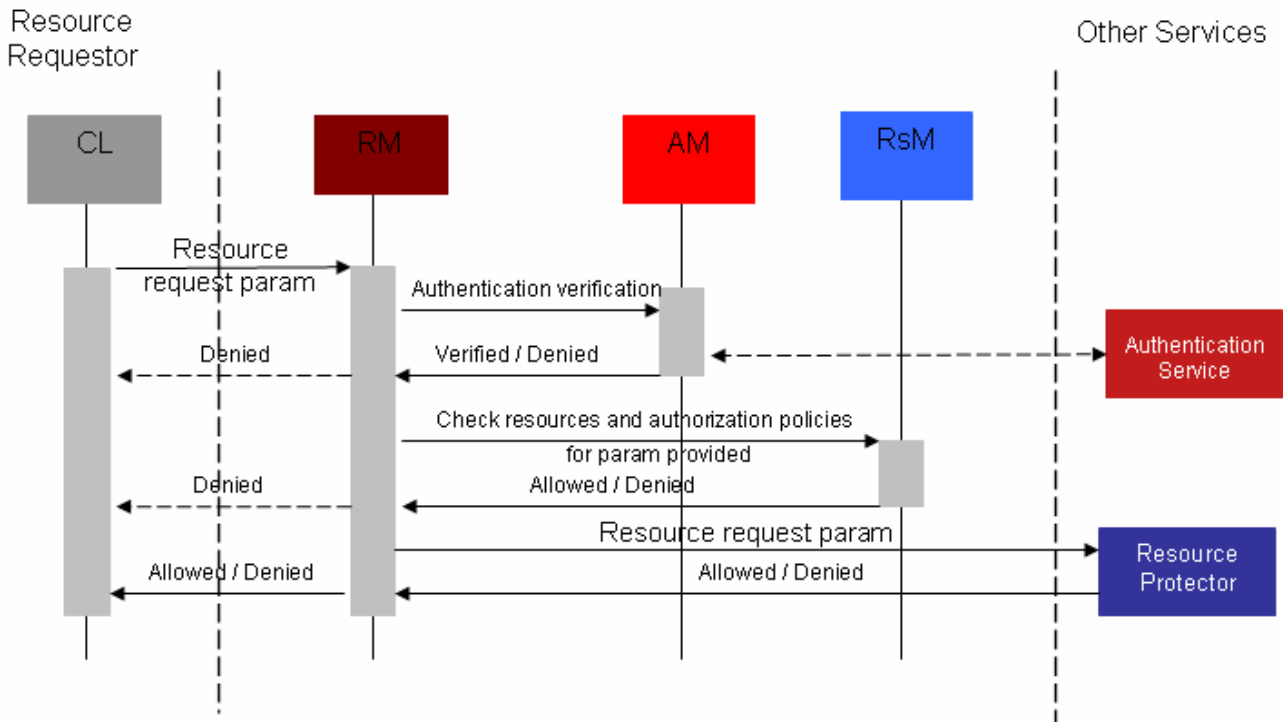


Figure 5-17: Resource Protector – contacting external Resource Protectors

5.6 Transformation Service

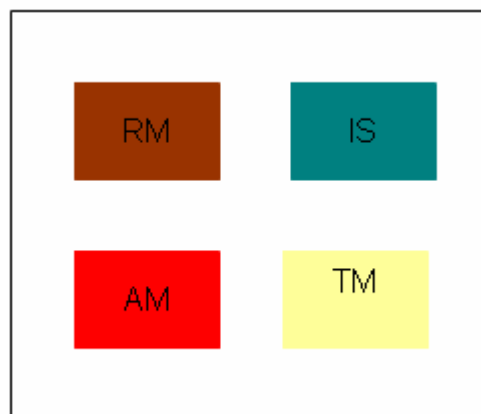


Figure 5-18: Transformation Service internal components

5.6.1 Specification of Tasks

The main tasks of a Transformation Service include

- **Setup for using the service**
- **Transforming data received from a publisher to one or more subscribers**
- **Allowing management of the service**

The service will need to have the capability to accept the following requests via its interfaces

- Service setup requests to transform data from a particular publisher onto a specified set of subscribers
- Data from publishers that needs to be transformed
- Management interactions

The service will have to initiate communications with external entities to achieve the following

- Register, De-register and provide Keep-Alives
- Publish transformed measurement data to the Subscribers

Internal tasks that might be necessary to achieve its main tasks are:

- Authorize requests
- Management of Transformation setups
- Mapping Transformation setups to identify and transform appropriate data flows
- Transformation of data

5.6.2 Components

The Transformation Service is an infrastructure consisting of several components, as shown in **Figure 5-18**. These components include:

- The Request Manager component, which exchanges all requested information with outside services.
- The Information System component, which keeps descriptive and configuration information.
- The Authorization Manager, which authorizes requests. It also accepts tokens of authentication, analyzes the current request in the context of policy rules, and grants/denies authorization.
- The Transformation Manager, which does the actual work of transforming data based on its capabilities

5.6.3 Dependencies

The Transformation Service is dependent upon:

- **Authentication Service:** The Transformation Service accepts one or more token types from one or more Authentication Services, effectively delegating authentication, but not authorization.
- **Lookup Service:** The Transformation Service must register its existence and capabilities with one or more Lookup Services. It must also register which Authentication Service(s) it trusts for authentication.

5.6.4 Data Types

The Transformation Service deals in three types of data:

- **Measurement Data:** Received from Measurement Point Service(s) and/or Transformation Service(s).
- **Authentication Tokens:** Evaluated by the Authorization component with respect to policies.
- **Setup Data:** Provided by its clients and used to relate between publishers of data and subscribers of transformed data

5.6.5 Interfaces and Interactions

5.6.5.1 Service Interfaces and Interactions

The Transformation Service can be used by clients wishing to see data in a different format than the ones that are available or provided by other services. It can also be strongly coupled with one or more services, which by default want to export data in a format other than those used. Although any client can make use of a particular Transformation Service by 'Looking them up' and using appropriate credentials, the following clients can be listed as the most likely ones.

- Topology Service
- Measurement Archive Service
- External Clients

Basically the Transformation Service will be used to transform data. It will be a subscriber for some data and a publisher for the transformed version of the same. No data is generated by the service by itself. The exact specifications of the Transformation Service will dictate exactly which actual "Services" are interacted with. The following list describes some possible data sources or publishers:

- Measurement Point Service

Project:	GN2
Deliverable Number:	DJ.1.2.1
Date of Issue:	02/06/05
EC Contract No.:	511082
Document Code:	GN2-05-057v5

- Measurement Archive Service
- Topology Service

5.6.5.2 Component Interfaces and Interactions

Information exchanges between components are illustrated with the help of a few sequence diagrams. Initial setup required is illustrated in **Figure 5-19**. **Figure 5-20** shows how data is transformed and published onto subscriber/s. **Figure 5-21** shows how the service registers, de-registers and provides Keep-Alives to a Lookup Service.

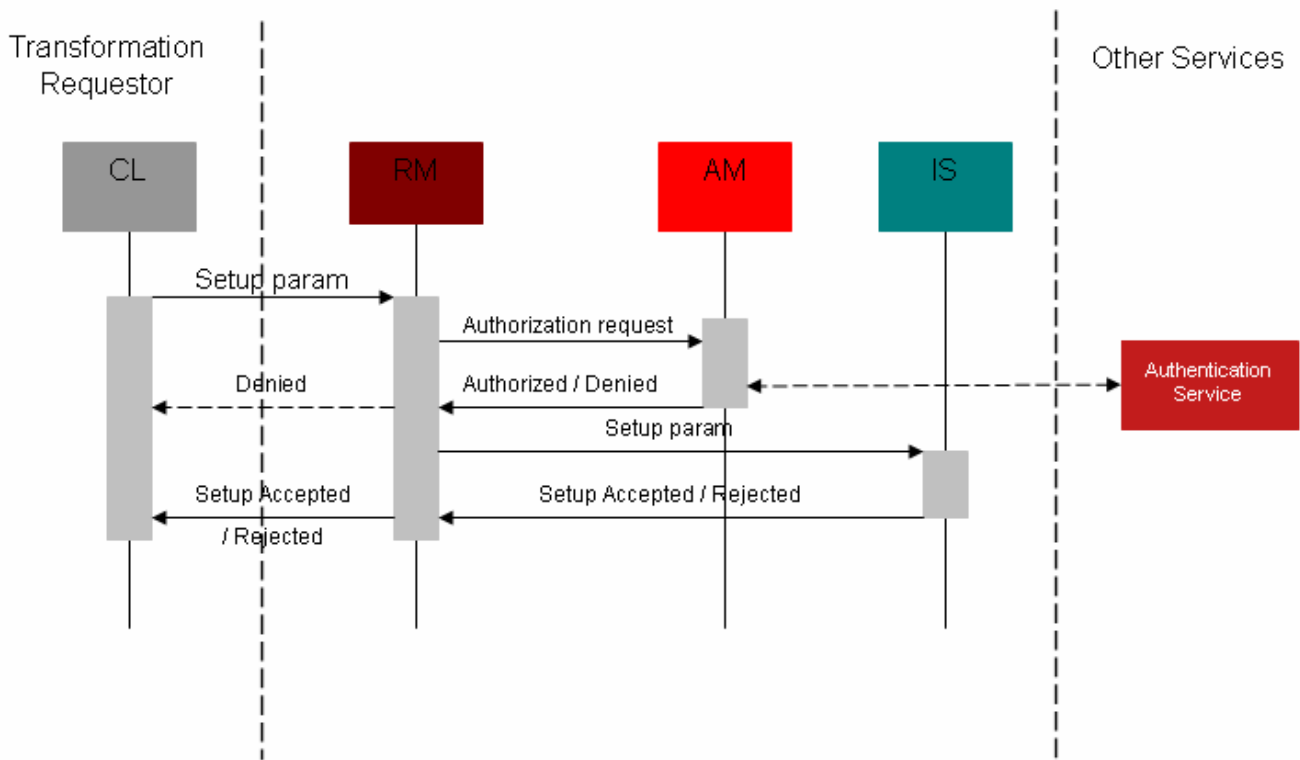


Figure 5-19: Setup for a Transformation

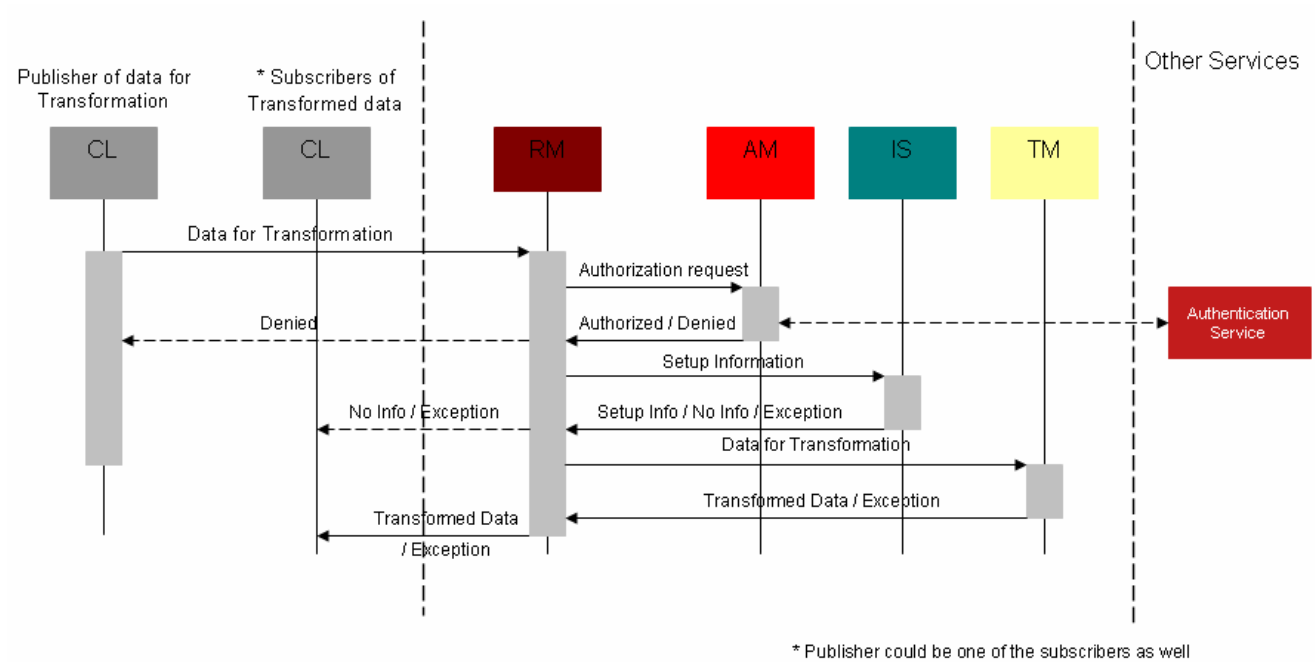


Figure 5-20: Transformation of Data by the Transformation Service

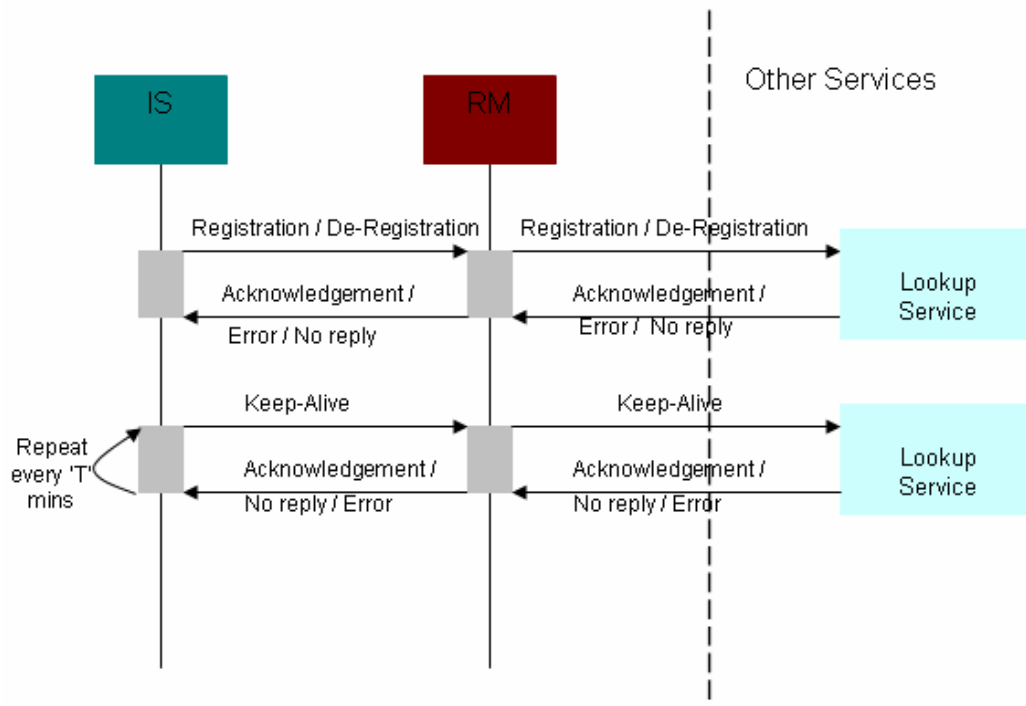


Figure 5-21: Registration, De-registration and Keep-Alives

5.7 Topology Service

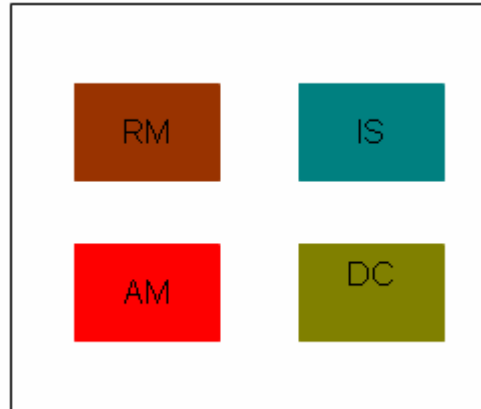


Figure 5-22: Topology Service internal components

5.7.1 Specification of Tasks

The main tasks of a Topology Service include

- **Retrieving topology related information from other services**
- **Correlating topology related data and publishing it into a Measurement Archive Service**
- **Allowing management of the service**

The service will need to have the capability to accept the following requests via its interfaces

- Subscribed data from publishers
- Management interactions

The service will have to initiate communications with external entities to achieve the following

- Register, De-register and provide Keep-Alives
- Publish topological data to a topologically oriented Measurement Archive Service

Internal tasks that might be necessary to achieve its main tasks are:

- Authorize requests
- Correlate measurement data collected from various sources and also configured into the service itself

5.7.2 Components

- The Topology Service is composed of several components as shown in **Figure 5-22**.
- The Request Manager component, which exchanges all requested information with outside services.
- The Information System component, which keeps descriptive and configuration information.
- The Authorization component, which authorizes requests. It also accepts tokens of authentication, analyzes the current request in the context of policy rules, and grants/denies authorization.
- The Data Correlator component, which correlates measurement data and transforms it into topology data

5.7.3 Dependencies

The Topology Service is dependent upon other services, which can provide topology related measurement data. It is also dependent on a topologically oriented Measurement Archive Service in order to store topology data. If information presented through the tokens of the publisher are insufficient, it can contact the Authentication Service in order to verify the authenticity of the publisher. In order to find data sources, which can supply it with topology related measurement data, it will need to make use of the services of the Lookup Service as well.

5.7.4 Data Types

The only type of data that this service deals with is topology data. Topology data can consist of actual topology information provided by sources (configured information, etc), which provide this information directly. Examples of such data include location, GPS coordinates, nearest router, etc. Topology data can also be obtained by transforming topology related data, which can be received via some network measurements. For example, an active measurement like a traceroute or polls of routing tables can be used to create topology information.

5.7.5 Interfaces and Interactions

5.7.5.1 Service Interfaces and Interactions

Clients external to the system (i.e., clients other than services) would not be able to interact with the Topology Service directly. Topology data correlated by using data from various sources is stored in a topologically oriented Measurement Archive Service, which can be used by all types of clients to retrieve this data. The Lookup Service could ideally refresh any topology data it uses by subscribing to the Measurement Archive Service storing topology data.

Summarizing, the topology data would communicate with different services that can provide topology related data. It would also communicate with the Lookup Service to find such services and would need to publish correlated data onto a Measurement Archive Service.

5.7.5.2 Component Interfaces and Interactions

Information exchanges between components are illustrated in **Figure 5-23**. It illustrates how topology related data is collected, correlated and topology data is stored in a Measurement Archive Service.

5.7.6 Topology Service Information

Topology Service will use the Lookup Service (and possibly MA Services) to publish the topology information. The trade-off here is between API and protocol complexity and data complexity. One real challenge in designing the Topology Service will be in coming up with reasonable data models that allow the topological constructs to be queried in an adequate way. As the Topology service is one of the additional (optional) services, further design will be presented in the next version of this document.

The topology data model must contain a flexible hierarchical node concept including domains/networks, regions, locations/sites, network entities and components. The topology database in the Lookup Service will contain objects that represent each Observation Point in the infrastructure like routers, interfaces and processors. A network level connection may be decomposed to a link level subnet and aggregated to higher-level connections between domains.

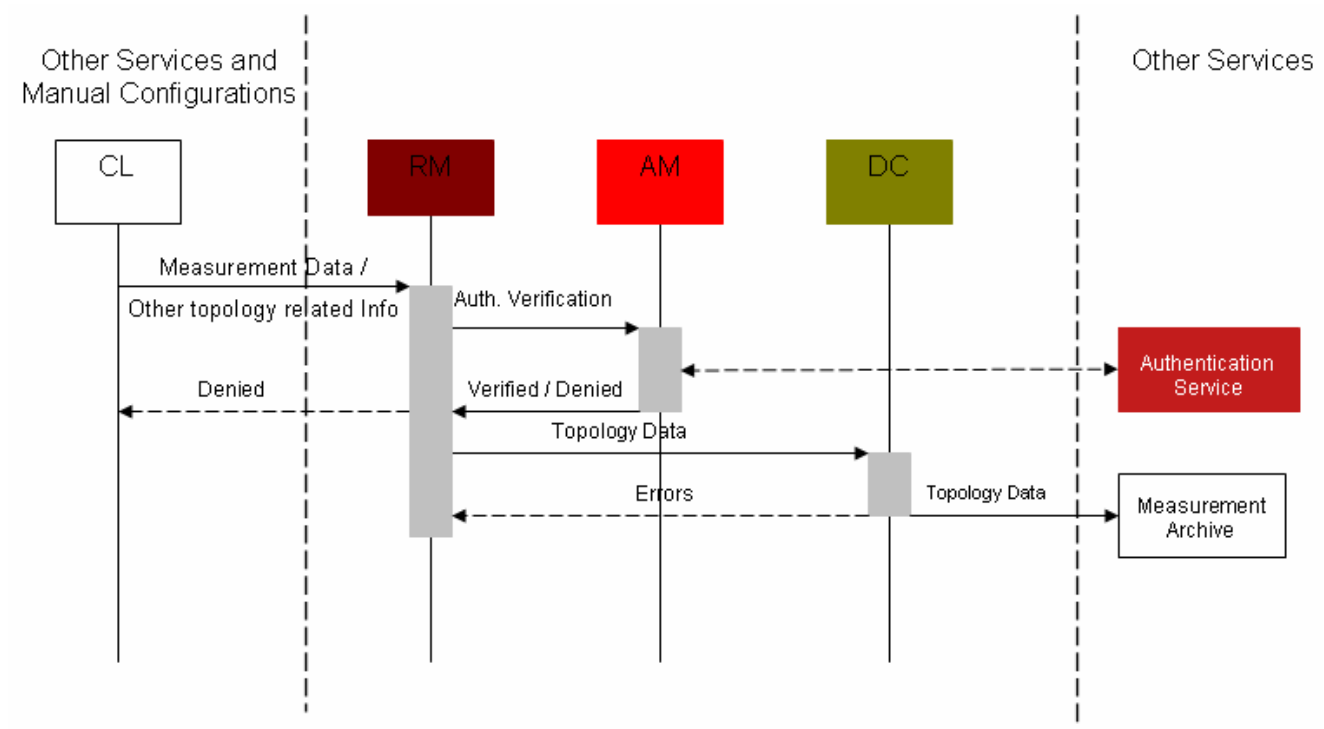


Figure 5-23: Correlating Topology data

6 Architecture validation against existing measurement capabilities

This section illustrates how existing measurement capabilities can fit into the architecture discussed in previous sections. However, the reader shouldn't assume that the scope of the architecture is limited to the ones discussed. These should be treated as example deployments of an architecture, which has been designed to be flexible enough so that a wide variety of measurement capabilities can be a part of it.

This section can also be treated as an introduction to the development process. It has been included in this document with the aim of providing a starting point for teams working on prototypes or other development procedures.

Further text in this section are divided into subsections, each handling a particular type of service and discussing a possible process of integrating existing measurement tools/capabilities within the services previously discussed.

6.1 Existing Measurement Capabilities and the MP Service

An MP Service needs to provide a certain set of functionality to its clients. These functionalities have already been discussed in section 2.1 and also listed below. To be considered as an MP Service, any measurement tool/capability is required to provide these functionalities.

- Interface to handle measurement requests and an interface to output measurement data. These also include necessary security features such as authorization, etc
- Collect measurement data that are generated either by active or passive measurements
- Provide measurement data to the client in the following ways
 - Directly to the requestor
 - Via a Transformation Service or a pipeline of Transformation Services
 - To a pre-configured MA
 - A chain of the above
- Interface to register, de-register and refresh its status with a 'Lookup Service'

Project:	GN2
Deliverable Number:	DJ.1.2.1
Date of Issue:	02/06/05
EC Contract No.:	511082
Document Code:	GN2-05-057v5

- Capability to manage resources locally and also be configurable to talk to external 'Resource Protectors'

All existing measurement tools/capabilities offer the basic functionality of providing measurement data (some of them do so by querying external entities such as routers, etc. Such entities will be seen as transparent in this context). It is important to note that the interfaces presented by the tools to provide this functionality need to be standard i.e., they need to follow specifications given by the MP Service.

The following subsections aim to list out the functionalities that would be needed for each type of measurement capability/tool to act as a MP Service. Some capabilities/tools might already provide some or all the functionalities expected. In such cases, if any recommendations for alterations are necessary in order to conform to the specifications of an MP Service, then they will be addressed as well. The basic idea of a MP Service providing a common interface irrespective of the type of capability is upheld at all times.

Based on the set of capabilities, measurement tools can be broadly classified into two types of analysis tools; those that allow active measurements; and tools that are passive in regards to measuring the network. While active measurements provide the capability to make a custom-specified measurement between (usually) two or more measurement tools (mostly of the same type), passive measurement tools would only allow specification of parameters that would be used in capturing and deducing measurements from real network traffic or in retrieving network information status. Although most passive measurements do not need co-ordination between any two MPs involved to provide measurement data, exceptions do exist. However, in case of active measurements, the co-ordination to perform a measurement is essential.

Using the above classification, further text in this section will discuss different types of tools such as DFN-IPPM⁴, OWAMP⁵, BWCTL⁶, etc., and illustrate how some of them can be integrated into the framework by adhering to the specifications of an MP Service.

6.1.1 Active Measurements

6.1.1.1 Delay and Loss Measurements

Delay and loss measurements, using active measurement techniques, can be made with the help of DFN-IPPM and OWAMP tools. The following text underlines the required set of functionalities and discusses how these tools either already provides the functionality or would need some extra work to provide the functionality.

- Requesting a measurement

⁴ The DFN-IPPM measurement boxes perform tests of one-way delay, jitter, and packet loss. These tests require time to be perfectly synchronized on the different boxes, using GPS, PZF receivers.

⁵ There are 2 tools in OWAMP that can be used to make measurements: owping and powstream. They perform active test based using the OWAMP protocol and post the results in a database.

⁶ BWCTL is a command-line tool that is a wrapper around lperf for making active TCP throughput tests. It can be used to schedule easily regular lperf tests over the network. It can also be use to facilitate the tests between 2 boxes in different network domains.

A measurement request can be of two types, on-demand measurements (the ones that might come from a visualization or analysis tool) and regular measurements (like a NOC might use for quality assurance). An interface for such requests will be specified to have a common way to talk to all MP Services.

In the case of DFN-IPPM, a configuration file is used by the tool to handle requests. This configuration file is read by the tool to make measurements. Additional functionality, which makes use of specified technologies (e.g. Web Services) and understands request messages in standard formats (e.g. NMWG specified) is necessary. Once the request is understood, the user authenticated, and the request authorized, the configuration file on the DFN-IPPM measurement tool will need to be updated accordingly.

The OWAMP measurement tool accepts requests for measurements directly. Additional functionalities would be necessary to accept requests via an interface compliant to NMWG specified request schemas.

- Authentication and Authorization

All requests presented to the MP Service would need to be first checked to ensure that they come from an already authenticated user. Some measurement tools could already provide such kind of functionality with the help of an accepted security mechanism. In case this functionality is missing or is not acceptable for some reason then introducing an acceptable solution is essential.

Authorization of requests is carried out locally. For this, the MP Service will need to take decisions based on its policies. These policies should ideally be configurable to reflect future changes. In both cases, the MP Service could have the option of further communicating with the “Authentication Service”.

DFN-IPPM provides basic encryption and authentication via SSH. Further enhancement to this authentication methodology is required. Authorization and policies aren't supported either. This brings up the need for adding extra functionalities such as Authentication Manager (to handle authentication) and Resource Manager (to handle Authorization of resources) either in addition or in lieu of already existing SSH.

OWAMP uses a configuration file to manage authorization. The configuration file is based on the usage of user classes where each user is assigned to a particular class. Netmasks are also grouped into these classes. A user is identified through shared AES keys or less securely by source IP address.

- Information while making a measurement

Most measurement tools/capabilities log the errors that occur while making a measurement. Such kind of logging is definitely needed but pushing this information to the client might be necessary in order to keep the client informed and also to provide periodic updates about the requested measurement. It is important to note that standardisation of such messages across all types of MP Services is essential to uphold transparency.

Currently, DFN-IPPM generates diagnostic output, which is forwarded to the client via a (permanent) TCP connection. This has to be moved over to an event based (NMWG specified) schema compliant

Project:	GN2
Deliverable Number:	DJ.1.2.1
Date of Issue:	02/06/05
EC Contract No.:	511082
Document Code:	GN2-05-057v5

error reporting system. It is predicted that Web Services technology (SOAP based communications) will be used for implementing such a type of system.

A small set of negative response values are provided by the OWAMP tool to the client which might need to be further extended depending on requirement. Also, the OWAMP daemon sends diagnostic information to syslog that could be intercepted for this purpose. This would also need to be modified into an event based schema.

- Temporary caching of data, pushing data

Some tools temporarily cache measurement data and others output it immediately when it is generated. In both cases, the format in which data is required could be different from the one being it is being provided. This can be achieved by either formatting the data received from the tool/capability (if not already formatted) within the MP Service itself, or configuring the tool to make use of an appropriate "Transformation Service".

The DFN-IPPM measurement tool stores data locally for an almost indefinite amount of time. Data can currently be output as ASCII files. This brings about the need for translating from the format used in flat files to the format and technology (XML) specified by NMWG. Data 'push' mechanism isn't currently supported by the tool but the client has to request for data. Currently, SSH key based authentication is used during this process. These techniques will have to be migrated to a web service (SOAP) based push mechanism, which uses secure communications and also authenticates itself to the published/subscriber interface.

The server side of the receiver of the OWAMP stores temporarily caches measurement data. Data is then requested by the client part of the tool. AES based authentication is used during this process.

- Resource management

Measurement tools/capabilities would have a local resource management to handle their resources. But for various reasons, communications with external "Resource Protectors" might be essential. In such cases, permission from such Resource Protectors will be sought before agreeing to make a measurement.

Administration of resource management (internal to the MP Service) to configure its policies is seen as an essential feature that needs to be provided by the MP Service.

Resource management in DFN-IPPM is carried out at a centralized entity. Since all requests for measurements go via this entity, it is aware of the resources used by measurements. A list of available and used timeslots is maintained for this purpose. Local resource management is not available on the tools. This arrangement has to be moved over to a Resource Manager – Resource Protector communication model. The current central entity can be configured to be a Resource Protector. A Resource Manager local to the Measurement Point service has to be put in place, which will contact the central Resource Protector before agreeing to make a measurement. Policies will have to be configured

Project:	GN2
Deliverable Number:	DJ.1.2.1
Date of Issue:	02/06/05
EC Contract No.:	511082
Document Code:	GN2-05-057v5

at both ends (Resource Manager and Resource Protector) and all decisions made will be based on these policies.

As previously mentioned, a configuration file on the server side of the OWAMP tool manages resources and also takes care of authorization. The tool is not able to contact external Resource Managers currently.

6.1.1.2 *Stress type Achievable Bandwidth Measurements*

Active throughput measurements can be made with the help of tools such as Iperf, nuttcp, and BWCTL. The following text briefly discusses how some of these tools can be incorporated into the network.

- Requesting a measurement

Iperf and nuttcp have no control protocol as such. To run one of these tests the user must start the application on two hosts with the correct options to make them communicate with each other. A request protocol would need to be wrapped around the tools and used to execute each side of the test.

BWCTL accepts requests for measurements directly. Additional functionalities would be necessary to accept requests via an interface compliant to NMWG specified request schemas.

- Authentication and Authorization

All requests presented to the MP Service would need to be first checked to ensure that they come from an already authenticated user. Some measurement tools could already provide such kind of functionality with the help of an accepted security mechanism. In case this functionality is missing or is not acceptable for some reason then introducing an acceptable solution would be essential.

Authorization of requests is carried out locally for which the MP Service would need to take decisions based on the policies. These policies should be configurable to reflect future changes. In both cases, the MP Service could have the option of further communicating with the "Authentication Service".

Iperf and nuttcp have no authentication or policy built into them other than the ability to run them on different ports to provide some anonymity and obscurity.

BWCTL uses a configuration file to manage authorization. The configuration file is based on the usage of user classes where each user is assigned to a particular class. Netmasks are also grouped into these classes. A user is identified through shared AES keys or less securely by the source IP address of the client.

- Information while making a measurement

Most measurement tools/capabilities log the errors that occur while making a measurement. Such kind of logging is definitely needed but another requirement could be pushing this information to the client and

Project:	GN2
Deliverable Number:	DJ.1.2.1
Date of Issue:	02/06/05
EC Contract No.:	511082
Document Code:	GN2-05-057v5

letting the client know periodically that the request measurement is ongoing. It is important to note that standardisation of such messages across all types of MP Services is essential to uphold transparency.

Because Iperf and ntttcp are command-line tools, diagnostic output comes in the form of errors reported to the console. An MP configured to run those tools would have to read those errors and translate them into an event based (NMWG specified) schema compliant error reporting system.

A small set of negative response values are provided to the BWCTL client which might need to be further extended depending on requirements. Also, the BWCTL daemon reports errors to syslog, which could be intercepted and used as a source of data in an event based error reporting system.

- Temporary caching of data, pushing data

Some tools temporarily cache measurement data and others output it immediately when it is generated. In both cases, while outputting data, the format of the data has to be standard across all MP Services. This could be achieved either by formatting the data received from the tool/capability (if not already formatted) within the MP Service itself or configuring the tool to make use of an appropriate "Transformation Service".

Iperf and ntttcp return their results to the command line. Therefore the MP will have to capture the results of each test.

BWCTL implements a peer control connection between the measurement points conducting the test. Both the sender and receiver receive the results from both sides of the test. There are call-back functions in place to allow the BWCTL daemon to do any processing on the results. The client also receives the results from both the sender and the receiver of the test. The data will need to be converted into a format that can be represented within the framework.

- Resource management

Measurement tools/capabilities could probably have a local resource management to handle their resources. But for various reasons, communications with external "Resource Protectors" could be essential. In such cases, permission from such Resource Protectors would need to be sought before agreeing to make a measurement.

Administration of resource management (internal to the MP Service) to configure its policies is seen as an essential feature that needs to be provided by the MP Service.

There is no resource management built into Iperf or ntttcp.

As previously mentioned, a configuration file on the server side of the BWCTL tool manages resources and also takes care of authorization. The tool is not able to contact external Resource Managers currently.

Project:	GN2
Deliverable Number:	DJ.1.2.1
Date of Issue:	02/06/05
EC Contract No.:	511082
Document Code:	GN2-05-057v5

6.1.2 Passive Measurements

6.1.2.1 Information from Network Equipment

Information from network equipment could be retrieved directly from the network equipment; or via passive probes and alike; or via other tools, which periodically retrieve and store data from the above equipment or probes. An example of a tool that periodically retrieves data would be RRD based tools. Retrieving data directly from the network equipment introduces the need to use protocols that can communicate with the network equipment (or use tools which have the capability to do so). Examples would include SNMP and NetFlow.

- RRD based tools (Round Robin Database based tools)

Many network monitoring tools make use of Round Robin Databases internally to store network measurement data. Examples of such tools are Cricket [Cricket] and RRDTOol [RRDTOol]. MRTG [MRTG] is a similar network monitoring tool, which can be configured to use RRDTOol to store data. By wrapping an MP Service around such RRD based tools, the user will have the capability to request the tool to collect data for a particular network element and also provide the collected measurement data through standardized interfaces. An ideal migration of such tools would be to wrap the configuration part of the tool as an MP Service as described above and the data storage in RRD as an MA Service. In this case, interactions between the MP Service and the MA Service will be standardized and will replace the currently used proprietary communications. The MA Service wrapping the RRD storage will be seen as one of the subscribers for data collected by the MP Service. Another alternative to this migration strategy is wrapping the RRD based tools and RRD data store as an MP Service. One of the subscribers to this MP Service can be another MA Service, which stores data using conventional (SQL based) database. In both cases, the MP Service maintains a list of subscribers and will push measurement data that it retrieves onto those subscribers. Subscribers can specify the type of data, the sources of data and also the frequency with which data should be pushed to them.

- MP Services for commonly used protocols and interfaces

Many protocols and interfaces are currently being used to retrieve network information. Some very common protocols are SNMP, NetFlow, and CLI (Command Line Interface). These protocols and interfaces could be used in our context to retrieve information about network topology and network equipment information (looking glass functionality) as well as other network characteristics such as packet drops and other error indicators. MP Services designed for such purposes provide the required translation between different protocols and standard external interfaces presented by these services.

An important factor for such services is providing frequent or scheduled data retrievals. Requests from clients could either be 'One time' only or subscription for periodic data publishing. MP Services for the above can provide both.

For protocols like SNMP, the MP Service will translate requests to Object IDs (OID). To do so, it will have a repository of OIDs along with a mapping of network elements to such OIDs. Adding new translation capabilities would be introducing new OIDs to the repository and the MP Service should make this

possible as well. In the case of CLI, the MP Service will need to know the exact sequence of commands that must to be presented to the network device to retrieve information from it.

6.1.2.2 Packet Trace Capturing

An important property of passive monitoring is that it also *processes* packet traces directly in measurement points, rather than just capturing them. In high-speed networks (~10Gb/s), it is not possible to capture and store all packets and process them afterwards in software. Time critical processing must be done in monitoring hardware and only filtered data or computed results can be passed to the host computer and out of the monitoring point. The function of a passive monitoring point is to find out something from passively observing user data on user request and report these findings back as measurement results.

The following paragraphs summarize initial thoughts on what needs to be done to integrate SCAMPI⁷ passive monitoring platform with JRA1 infrastructure.

- Requesting a measurement

The primary interface to the SCAMPI platform is MAPI (Monitoring Application Programming Interface). Monitoring applications are written on top of MAPI. An application can open one or more flows. A flow is initially a packet trace consisting of all packets coming to a specified network adapter or stored in a file. The application can then apply monitoring functions to open flows. These functions can include: header filtering, sampling, payload searching, computing statistics, composing NetFLOW records, anonymization etc. Each function can be implemented either directly inside a hardware monitoring adapter (which is preferable for a high-speed network) or inside MAPI, which automatically uses either hardware or software implementation depending on the used monitoring adapter. Using a hardware implementation implies the necessity to download corresponding instructions to the hardware monitoring adapter.

To integrate the SCAMPI platform within this framework, the Measurement Executor of the Measurement Point Service will need to communicate with SCAMPI's MAPI interface. This will provide access to single point measurements. To satisfy the need for multi-point measurements, either two MAPI interfaces can be contacted through two Measurement Point Services or the work on DiMAPI (Distributed MAPI), which is going on to support multipoint measurements, can be utilized by configuring one Measurement Point Service to talk to multiple SCAMPI tool instances.

- Security – authentication and authorisation

SCAMPI uses its own authentication daemon. A possibility to use this authentication for this framework purposes will need to be investigated. Another possibility (probably more transparent) is to use a common Authentication and Authorisation external to SCAMPI. The basic usage of the already existing authentication system in SCAMPI can be made use of by configuring MEs of Measurement Point Services to use this system along with any necessary additions to support JRA1 needs.

⁷ SCAMPI provides hardware and middleware for enhanced flow data monitoring for links. More information can be retrieved on <http://www.ist-scampi.net>

- Information while making a measurement

It is possible to read continuous results of an applied MAPI monitoring function. Depending on what interim information is required, it could possibly be obtained via MAPI. In order to provide this data to the client, the ME of the Measurement Point Service will have an open interface towards MAPI, which will relay these live feeds after any necessary formatting to the client.

- Pushing data

SCAMPI supports a basic data retrieval (poll interface) scheme and additional functionalities are required on the MP Service in order to push data in the necessary manner.

- Resource management

SCAMPI has its own internal Resource Management. The possibility of linking it with external Resource Managements along with trust issues with such resource management needs to be investigated.

6.2 Measurement Storage and the MA Service

Archiving data retrieved from MP Services is essential for understanding network performance trends (thus helping the network planning process) and speed up the troubleshooting of network performance problem by looking at the past history.

The MA Service offers this functionality. A few issues related to database storage are listed below.

- Data must be available in a standardized format. Internal storage may be different as required by the nature of the data itself. Additional data formats may also be available to provide for more efficient distribution of data, but the standardized format is a requirement for interoperability.
- Data archiving, backups can be achieved through many solutions – reflections, hardware backups, etc.

6.3 Other Services

Most of the services that need validation against existing infrastructure have already been analysed in the previous sections. Other services such as Lookup Service, Transformation Service, Topology Service and Measurement Archive Service are either independent of existing infrastructure or they are new in comparison to existing techniques and wouldn't need to be validated. Many technologies such as Web Services, UDDI, JINI, JNDI, etc. are available to help with deployment of such techniques. The analysis of these technologies and validation of other remaining services based on this analysis is considered to be beyond the scope of this document.

GEANT2 General Monitoring Framework Design

Architecture validation against existing measurement capabilities



Project:	GN2
Deliverable Number:	DJ.1.2.1
Date of Issue:	02/06/05
EC Contract No.:	511082
Document Code:	GN2-05-057v5

7 Conclusions

This document presents the General Framework Design (GFD) which will be used by both GN2-JRA1 and the Internet2 piPEs activity to provide monitoring information to their users. The GFD will most likely evolve in the future as the integration of new type of tools (both measurement and visualisation), and new features will steer the improvement to the GFD.

The next design steps are the detailed design of the three base services described in the GFD (Measurement Point, Measurement Archive, Lookup services) and the specification of the interfaces between those services. Those tasks will be written in deliverable DJ1.2.2⁸ with the exception of the Authentication service will be provided by GN2-JRA5 activity. The detailed design of the other services (Resource Protector, Transformation and Topology services) will depend on their needs and resource availability.

Both GN2-JRA1 and Internet2 piPEs have felt the need for a prototype to be built by mid-July 2005 in order to:

- Validate the choices made for the General Framework Design process (services oriented)
- Test and evaluate the chosen technology for implementing the services
- Implement a visualisation to test the data retrieval process.

The services which will be implemented are the Measurement Point, Measurement Archive, Lookup Service for link utilisation and link capacity. The services implemented will have a reduced scope: a restricted set of functionalities and metrics provided in comparison with a fully featured service. The data provided will be retrieved from within several domains, but the services will act as if those data were coming from a single domain. A “mock-up” AA service will be implemented if JRA5 will not have delivered the AA service by that time. The prototype will then lead at a later stage to the MJ1.3.1 “Single Domain Pilot Implementation” (November 2005), which will covers more metrics, tools and services features.

⁸ System architecture modules design

8 Acronyms

AES	Advanced Encryption Standard
AM	Authentication Manager
API	Application Programming Interface
AuthN	Authentication
AuthZ	Authorisation
CL	Client
CLI	Command Line Interface
DC	Data Correlator
DMAPI	Distributed Monitoring Application programming Interface
DoS	Denial of Service
IS	Information System
JNDI	Java Naming and Directory Interface
LS	Lookup Service
MA	Measurement Archive service
MAPI	Monitoring Application programming Interface
ME	Measurement Executor
MP	Measurement Point service
NMWG	Network Measurements Working Group
NOC	Network Operation Centre
NREN	National Research and Educational Network
OID	Object Identifier
RM	Request Manager
RP	Resource Protector service
RRD	Round Robin Database
RsM	Resource Manager
SM	Storage Manager
SNMP	Simple Network Management Protocol
SOA	Service Oriented Architecture
SSH	Secure Shell
TCP	Transport Control Protocol
TM	Transformation Manager
UDDI	Universal Description, Discovery and Integration
XML	Extensible Markup Language
WAYF	Where Are You From

9 References

9.1 GN2 project documents

[DJ1.1.1] - GN2 deliverable DJ1.1.1 - Requirement report

[DJ5.1.2] - GN2 deliverable DJ5.1.2 - Documentation on the GÉANT2 Roaming Requirements

GN2 JRA1 - Design WI - Architecture study document

9.2 Related technology references

Service discovery in ubiquitous computing environments”, Simon (Chih-Chieh) Han:

<http://www.item.ntnu.no/fag/ttm3/presentations/ServiceDiscoveryProtocols.pdf>

JINI Technology:

<http://btobsearch.barnesandnoble.com/booksearch/isbnInquiry.asp?btob=Y&isbn=0130333859>

Shibboleth Arch:

<http://shibboleth.internet2.edu/docs/draft-mace-shibboleth-arch-protocols-05.pdf>

Lionshare (P2P with AA):

<http://lionshare.its.psu.edu/main/info/docspresentation/LionShareWP.pdf>

9.3 Measurement tools and frameworks references

Pipes:

<http://e2epi.internet2.edu/e2epipes/docs/piPEs-arch20040607.pdf>

Advisor:

<http://dast.nlanr.net/Projects/Advisor/>

Project:	GN2
Deliverable Number:	DJ.1.2.1
Date of Issue:	02/06/05
EC Contract No.:	511082
Document Code:	GN2-05-057v5

MonaLisa:

<http://monalisa.cacr.caltech.edu/home.html>

OWAMP:

<http://e2epi.internet2.edu/owamp/>

BWCTL:

<http://e2epi.internet2.edu/bwctl/>

DFN IPPM:

<http://www-win.rrze.uni-erlangen.de/ippm/index.html.en>

Pathfinder: (GN1 Deliverable 8.3: Update on Multi-Domain Monitoring - Year 4)

<http://www.geant.net/upload/pdf/GEA-04-081v8.pdf>