Title:	Document Version:
Deliverable D3.1 **IPv6 QoS Measurement Specification**	7.5

Project Number:	Project Acronym:	Project Title:
IST-2001-37611	6QM	IPv6 QoS Measurement

Contractual Delivery Date:	Actual Delivery Date:	Deliverable Type* - Security**:
30/05/2003	05/12/2003	R – PU

* Type: P – Prototype, R – Report, D – Demonstrator, O – Other
** Security Class: PU- Public, PP – Restricted to other programme participants (including the Commission), RE – Restricted to a group defined by the consortium (including the Commission), CO – Confidential, only for members of the consortium (including the Commission)

Responsible and Editor/Author:	Organization:	Contributing WP:
David Diep	HIT	WP3

Authors (organizations):

Miguel Angel Díaz (Consulintel), Jordi Palet (Consulintel), Guido Pohl (Fokus), Florian Schreiner (Fokus), Emile Stephan (FT), Lidia Yamamoto (HEL), Kiminori Sugauchi (HIT).

Abstract:

This first WP3 deliverable aims at providing the specifications of the 6QM Measurement System Prototype. The present document defines the baseline for the future prototype development.

Keywords:

IPv6, Measurement, Prototype specifications, QoS.

Revision History

The following table describes the main changes done in the document since its creation.

Revision	Date	Description	Author (Organization)
v0.1	03/03/2003	Document creation	Kiminori Sugauchi (HIT)
v1.0	16/06/2003	Content update and new major version number	David Diep (HIT), Guido Pohl (Fokus), Emile STEPHAN (FT)
v2.0	18/06/2003	Update section 3,4 and 6 and new major version number	David Diep (HIT), Emile STEPHAN (FT)
v2.1	18/06/2003	Template issue	David Diep (HIT)
v3.0	18/06/2003	New major version number	David Diep (HIT)
v3.1	18/06/2003	Existing QoS measurement systems	David Diep (HIT) Lidia Yamamoto (HEL)
v3.2	18/06/2003	Conclusion section	David Diep (HIT)
v3.3	18/06/2003	“Initial Metrics” section integration	David Diep (HIT) Guido Pohl (Fokus)
v4.0	18/03/2003	Template and new major version number	David Diep (HIT)
v4.1	18/06/2003	“6QM Prototype Components” integration	David Diep (HIT)
v5.0	18/06/2003	New major version number	David Diep (HIT)
v5.1	19/06/2003	Revisions in data export and summary sections Minor corrections in “Passive measurement metrics definition”	Guido Pohl (Fokus) Lidia Yamamoto (HEL)
v5.2	19/06/2003	Multiple editorial changes	David Diep (HIT)
v6.0	19/06/2003	Minor modifications and validation of changes	David Diep (HIT)
v6.1	19/06/2003	Minor change	Kiminori Sugauchi (HIT)
v6.2	19/06/2003	Corrections to metric section. Final review.	Lidia Yamamoto (HEL)
v6.3	20/06/2003	Minor corrections	Miguel Angel Díaz (Consulintel)
v6.4	24/06/2003	Final Review	Jordi Palet (Consulintel)
v6.5	01/02/2004	Revision	David Diep (HIT)
v6.6	13/05/2004	Revision concerning inter-domain aspects	Lidia Yamamoto (HEL)
v6.7	27/05/2004	Added appendix on inter-domain measurements	Lidia Yamamoto (HEL)
v7.0	18/06/2004	Accept all the changes	David Diep (HIT)
v7.1	16/07/2004	Final Review	Jordi Palet (Consulintel)
v7.2	15/10/2004	Update/Addition to reflect reviewer’s comments	Guido Pohl (FOKUS)
v7.3	27/10/2004	Update content about prototyping plan	David Diep (HIT)
v7.4	22/11/2004	Update section 7	David Diep (HIT)
v7.5	05/12/2004	Final review	Jordi Palet (Consulintel)

Executive Summary

This first WP3 deliverable aims at providing the specifications of the 6QM Measurement System Prototype. In order to create these specifications WP3 tried to reuse as much as possible the concepts and guidelines developed in the WP2. As defined in the project technical annex the target is to develop a prototype measurement system capturing IPv6 packets in order to perform QoS measurements. Consequently, the scope of WP3 is to address QoS measurements by means of passive measurement techniques. However, as shown in WP2 active measurement techniques are very important for network operators. Therefore, even if this technology is not in the original scope of the 6QM project, the current prototype specification addresses the possibility to include some active probing in the 6QM measurement infrastructure as a scope extension.

Before specifying the prototype, this deliverable gives an overview of some relevant existing QoS measurement systems and compares them. This comparison points out that the active measurement is traditionally covered by many existing systems unlike the passive measurement. Moreover, this study shows a need for passive QoS measurement systems in multi-point mode and a need for systems combining both passive and active techniques.

This deliverable also fixes the 6QM measurement system scope by defining several levels of priorities for the prototype functions – referred as “Core Scope”, “Extended Scope”, “Advanced Scope” and “Future development”. The definition of those boundaries addresses a strong need to clarify the prototype direction and to help in assigning the future development resource.

Once this background information is clarified, the main part of the document presents the prototype specification by defining the initial metrics to be used in the prototype and by providing the system overview. The result of the system overview is to provide the basic system components defined as “6QM Measurement Manager”, “6QM Evaluator” for the management and respectively data collection component. This system overview also defines some generic meter components, which could be passive or active (optionally). Then the deliverable describes the mandatory components in more detail, their external interface and their internal structure. Finally, the document shows the relation between WP3 work and WP2 requirements.

The present document defines the baseline for the prototype development. As far as the WP3 is concerned the next steps are the definition of the prototype detailed design and the development itself. The next deliverable (D3.2) will extensively address those issues.

Table of Contents

Table of Figures

1. INTRODUCTION

This document is the first deliverable in WP3. The purpose of this document is to provide the specifications of the 6QM Measurement System Prototype. In order to create these specifications WP3 tried to reuse as much as possible the concepts and guidelines developed in WP2. The requirements developed by WP2 were especially taken as a major input.

As defined in the project technical annex the prototype is oriented towards the development of a measurement system capturing IPv6 packets in order to perform QoS measurements; in other words, the scope of the WP3 is to address QoS measurement by means of passive measurement techniques. However, as shown in WP2, operators also need active measurement techniques. Therefore, even if this technology is not in the original scope of the 6QM project, the current prototype specification addresses the possibility to include some active probing in the 6QM measurement infrastructure as a scope extension.

This deliverable is structured by the follow sections:

- Section 2 “Existing QoS Measurement Systems”: This section provides a comparison of existing measurement systems.
- Section 3 “6QM Measurement System Scope”: This section provides some usage scenario for a measurement system and especially defines the boundaries of the 6QM prototype.
- Section 4 “Initial Metrics for 6QM Measurement System”: This section describes the metrics that are planned to be developed in the prototype.
- Section 5 “6QM Prototype System Overview”: This section addresses the system overview and the functional architecture of the prototype.
- Section 6 “6QM Prototype Components”: This section defines more clearly the components, their external interface and their internal structure.
- Section 8 “Relation Between WP3 and WP2”: This section shows the relation between the prototype system and the WP2 requirements.

In addition to the above sections, an appendix on inter-domain measurements is presented in Section 10. In the original project plan for WP3, the provision of a complete solution to the difficult inter-domain problem had not been initially foreseen. As a result of the 1st 6QM project review in Brussels, July 2003, it was required to incorporate the inter-domain aspect among the developments in WP3. In response to this review, which also required the resubmission of the present deliverable D3.1, the 6QM consortium started to investigate the state of the art and devise solutions to the problem of requesting, setting up and exporting measurements across different administrative domains. Research performed at HEL concluded that this requires negotiation techniques, and a proposal was put together to apply existing approaches to automated negotiation for agent-based systems in this context [Yama04].

2. EXISTING QOS MEASUREMENT SYSTEMS

Section 2.4 of Deliverable D2.2 included extensive tables comparing several QoS measurement products. In the present section, we attempt to provide a broader insight on existing measurement systems, not only commercial products but also research tools. We classify and compare the systems, and provide a motivation for further developments.

Since the number of existing measurement tools is overwhelming, we filter with respect to D2.2 Section 2.4, describing only the systems that seem the most relevant with respect to the 6QM topics. For a comprehensive list of tools, see the CAIDA web site [Cai02].

We define as QoS measurement system any hardware or software system able to measure one or more of the network performance metrics listed in Deliverable D2.1, such as RTT (round-trip time), jitter, packet loss, one-way delay, one-way jitter, one-way packet loss, throughput, and goodput. Some systems may comply with standards specified by the IETF (IPPM working group) or ITU-T as described in Deliverables D2.8 and D2.6 respectively. A few systems are now available that offer IPv6 support. These systems are the most related to 6QM and will be more closely studied.

We recall that QoS measurement systems may be passive or active. Passive systems perform measurements based on captured packets, and do not inject extra traffic in the network, while active systems measure the performance based on probe packets that are injected in the network. Hybrid systems are possible that support both active and passive mode, or use a combination of both.

From an architectural point of view, measurement systems can be classified according to the number of observation points required to produce a given metric. A single-point or 1-point measurement system requires a single observation point. An example is the simple “ping” tool: It sends an ICMP packet from a given source host to a given destination and measures the time to receive a response, obtaining the RTT to the destination. The source host that generates the “ping” packet is the single measurement point.

An example of 2-point measurement system is a passive one-way delay measurement system, which captures packets from two distinct points in a given path, timestamps them, and calculates the one-way delay as the difference between the two timestamps observed for the same packet.

A multi-point or n-point system requires several observation points. An example is to measure the performance of different branches of a multicast tree, including receivers, intermediate nodes, etc. Another example is the measurement of spatial metrics (see D2.1 Section 6 and [Ste02b] for details), for example, to measure spatial one-way delay and the corresponding one-way delay trajectory several points of measure are required to obtain the delays of individual portions of the path. In [Ste02a] a number of multicast metrics are defined as a set of spatial metrics.

Usually a n-point system also supports i-point measurements, with $0<i<n$. For example, some 2-point systems measuring one-way delay also support other metrics such as throughput, which requires only one point of measure. Another example is a multicast measurement system working in unicast 2-point mode.

One-point systems are the easiest to implement. Measures can be performed from a single machine, with no need to coordinate or synchronize with other parts of the network. It comes at

no surprise that here is where we find the biggest amount of commercial products and tools already available. Many of these systems are derived from protocol analyzers, comprehensive tools that are able to scrutinize packets at several layers of the protocol stacks, supporting numerous protocols. An example in this category is Agilent Advisor [Agi02]. It will be briefly described later on.

The 6QM proposal is mainly focused on 2-point systems therefore they will receive special attention. Fewer systems are available in this category, since they are more difficult to implement and operate. Multi-point systems are still mainly a research subject, although a few commercial products have recently appeared. The state of the art in this area will be summarized at the end of the section. In addition the document will present a product interesting because of its data export scheme.

2.1 Single-point Systems

Numerous single-point systems exist. This document presents a selection of a few systems judged as relevant in the 6QM context.

- Ping [Ping] is perhaps the most common and simple tool for active performance measurements. It uses ICMP Echo_Request/Echo_Reply pairs to estimate the RTT between two sites. It also reports packet losses, and works in broadcast and multicast mode. IPv4 and IPv6 versions are available.
- Pathchar [Pathc][Downey99] is a software program for the active estimation of path characteristics including bandwidth, delay, average queue and loss rate of every hop between a given source-destination pair on the Internet. It works by sending multiple probe packets of several sizes per hop, starting with TTL=1 and then incrementing the TTL until the destination is reached. For each hop, pathchar collects statistics that allow estimation of path characteristics. Although pathchar has been designed to be used in the global Internet, it must inject a significant amount of probe traffic in order to obtain meaningful results. Therefore, it can have a negative impact on the performance of the network in case it becomes extensively used. Pathchar takes a long time to accumulate sufficient statistics to obtain measurement results therefore it is not suitable for real-time or semi-real time measurements. Pathchar development seems to have stagnated since 1997. In addition to the source code unavailability, the documentation is almost non-existent. Meanwhile Pchar [Pchar] emerged as a new independent tool based on the same principles as Pathchar, but now with frequent maintenance and IPv6 support.
- Agilent Advisor is a network analyzer station produced by Agilent Technologies [Agi02]. It covers all layers of the protocol stack and is used to measure network performance, for SLA testing and troubleshooting. The Agilent Advisor platform is an integrated PC that performs the measurements and stores the measured data locally. The data can be later used within Windows compatible applications.
- QoSmetrix [Qosm] is a company that produces distributed performance measurement systems for traffic analysis, SLA conformance monitoring, troubleshooting and related operational activities. It provides a single point probe to make passive traffic analysis.

2.2 Two-point Systems

A two-point measurement system requires two observation points. A typical example is a tool that measures one-way delay. This metric requires clock synchronization between the two measurement points. This is usually achieved using GPS timing. Therefore, several two-point

systems include built-in GPS modules or offer them as options. Now this document presents a selection of a few 2-point systems, which are passive or active.

- RIPE (Réseaux IP Européens) [RIPE] is an open collaborative forum intending to ensure the necessary coordination for the operation of the wide area Internet within the RIPE region (covering mostly Europe). The RIPE Network Coordination Centre (RIPE NCC) is an Internet Registry that provides allocation and registration of global IP addresses and domain names for Internet sites within the RIPE region. Since the year 2000, RIPE NCC offers the TTM Service (Test Traffic Measurements) [RIPETTM] that collects measurement data from sites in order to enable a proactive monitoring of the network. This service requires a test box that is installed at the measured site and managed by RIPE in a centralized fashion. A RIPE test box is an active measurement probe built using a PC with GPS input. RIPE TTM measures one-way delay, one-way packet loss, traceroutes and bandwidth capacity. The delay and loss measurements comply with IPPM standards. IPv6 is fully supported since February 2003. RIPE's centralized database and ROOT-analysis server [ROOT] supply their participants with IPPM compliant IPDV, Long-Term-Trend, Packet Delay Frequency, Routing Vector distribution and many more statistical information generated for all measured end-to-end (testbox-to-testbox) links. Further integration of Available-Bandwidth measurements is planned. Currently there are around 60 boxes deployed. Fully meshed measurements are performed, i.e. from every testbox to every others. Although users have the option to deselect a link, the scalability problem of storing O(n2) measurements in a database is still an issue to be solved.
- SmartBits and SmartFlow are two products by Spirent Communications [Spirent]. When used together they allow QoS measurements. SmartBits is a network analysis station that can be used to "test, simulate, analyze, troubleshoot, develop, and certify network infrastructure". Spirent also provides applications for the SmartBits platform, "to test QoS and analyze the performance and behavior of the new breed of policy-based network devices". One of those applications is SmartFlow that enables IPv6 measurements and QoS testing, among other test types. SmartFlow IPv6 test solutions include standards conformance tests, performance (loss, latency and throughput), routing tests, translation tests, and tunneling tests.
- Brix System [Brix] is a suite of products that support SLM and SLA management and verification. There are three main products: Brix 100™ Verifier, Brix 1000™ Verifier, and Brix 2500™ Verifier. These are probes that can be used for 2-point performance measurements. They can be managed using a command line connection or via the BrixWorx central server software. Brix 100™ Verifier is a probe designed for customer-side QoS measurement; it supports both passive and active modes. Brix 1000™ Verifier are passive probes designed for provider-side measurements; they include an optional GPS module. Brix 2500™ Verifier is an enhanced version of Brix 1000™ Verifier for Gigabit Ethernet and OC-3 ATM interfaces.
- Netperf [Netperf] is a software tool for active measurements developed and distributed free of charge by Hewlett-Packard. It is designed for benchmarking purposes. It measures TCP and/or UDP throughput performance, and request/response latency between two hosts. Iperf [Iperf] is very similar to Netperf and supports IPv6. Iperf is newer and more up to date, however it is more complex to use.
- IPMM2 is the result of some cooperation between Hitachi Ltd. And FhI Fokus about QoS in IPv4 networks. IPMM2 is a passive metering system dedicated especially to the computation of one-way delay, but also including some functions for measuring throughput and RTP loss. This project was mainly oriented towards the development of a software meter. The system also includes a basic measurement server. Currently this prototype is neither a commercial product nor open software, however it is planned to open the source in near future.

- MGEN [MGEN] and trpr – MGEN is a command line tools that supports active measurements implementing a generator for multiple flows (currently UDP/IP) supporting basic statistical transmission schemes. The control of MGEN is either script or socket driven. An MGEN receiver logs the packet parameters to a log file. Other tools, for instance, can analyze this file typically with trpr (TRace Plot Real-time). Trpr analyses MGEN log files for the following metrics: delay, delay variation, packet loss over time. MGEN is under continues development and enhancement.
- OpenIMP is an initial implementation of a measurement system for one-way-delay measurements using passive means. It is freely available through Fraunhofer FOKUS. The system is server-based enabling remote control of a set of distributed meters. The system is operated by a graphical user interface running on a web server.

Systems that support two-point measurements typically have a basic architecture composed of measurement points scattered within the measured network and a centralized management station that collects the measured data, computes the corresponding metrics, and stores or displays the results so that the network manager can monitor the state of the network. Such centralized architecture is easy to manage and control, however it is not scalable to a large amount of collected data. In operational environments, over-provisioning a dedicated measurement infrastructure typically solves this scalability problem. This solution is simple but very expensive. For example, in [Fraleigh02] the deployment of a passive monitoring infrastructure over the Sprint IP backbone is reported. The authors report an amount of more than one terabyte of collected data per day. These data are stored in a large tape repository. They also report that about 12 hours are needed to transfer the compressed data from the remote systems to the repository.

2.3 Multi-point Systems

Multi-point systems are still a research subject, with a few commercial products appearing on the market. We describe some of the work in this area, which is more relevant to 6QM.

A multi-point system requires several observation points. A typical example is QoS measurement at several points of a multicast tree. Another example is future inter-domain measurements involving spatial metrics: Multi-point measurements will play a key role in this case, since measurements from each domain must be consolidated in order to obtain consistent end-to-end results. Now there seems to be no tool able to provide such inter-domain multi-point measurements. A lot of standardization and development work is necessary before this can happen.

The technical challenges in the design of multi-point systems are mainly related to the coordination and synchronization among the various probes, and how to manage all the information that is generated. Centralized management architecture does not scale in this case. The central server can quickly become a severe bottleneck. The problem is worse than in the case of 2-point systems: Since each metric requires several measurement points, much more data might be generated in total.

The scalability problem is common to centralized management systems. Solutions based on hierarchical management have been proposed, such as [Subram00] but they increase complexity and management overhead. Decentralized network management [Kahani97][Cabri01][Shen03] is an area of active research, however it still has not produced sufficiently simple and controllable tools to become widely accepted in operational networks.

Concerning the current market offer, some tools are able to perform multicast measurements within the same domain, for example QoSmetrix as described below.

QoSmetrix [Qosm] is a company that produces distributed performance measurement systems for traffic analysis, SLA conformance monitoring, troubleshooting and related operational activities. Their systems are compliant with IPPM for both UDP and TCP metrics measurements, and include support for IPv4 and IPv6. Their systems are also compliant with RTP measurement framework, and include support for IPv4 and IPv6 in unicast and multicast. The current solution provides two commercial products used in combination: NetWarrior and the NetAdvisor. NetWarrior is a hybrid active used to send or receive measurement traffic. It includes an embedded GPS receiver and patented technology for accurate time stamping. NetAdvisor is a web-based management and monitoring station used to configure and analyze the measurements. QoSmetrix works in close collaboration with France Telecom.

In the context of 6QM FTR&D purchased QoSmetrix probes, which are currently in deployment over partners IPv6 networks. The initial measurement system is made of 3 probes located on VTHDv6 and at Madrid Consulintel premises.

2.4 Data Export Issue

In future heterogeneous environment it is essential to have a common standard protocol for the data export from the measurement elements to a management station or to a data repository. IETF IPFIX working group drives this standardization activity. The base protocol chosen by the working group is Netflow v9. NetFlow is a Cisco feature available on most Cisco routers, and recently in Juniper routers. FlowCollector and cflowd are tools that comply with NetFlow. They are not oriented towards QoS measurements but towards accounting and billing. However, 6QM must be aware of them since IPFIX standardization will also apply to QoS measurements. Therefore, we mention these tools for informational purpose:

- Cisco FlowCollector [CisFcoll] is a collector for Netflow packets. It collects and aggregates Netflow data from several Exporter devices such as routers and switches that export NetFlow data records.
- Cflowd [Cflowd] is a flow analysis tool currently used for collecting NetFlow data. It provides different tools to analyze Netflow data and provides helpful information for network planning, network monitoring, network analysis and security-related areas.

According to the IPFIX Requirements draft [Quittek03], an Exporter device may be a router, probe or other device. However, to the best of the authors' knowledge there seems to be no QoS measurement systems available that export or handle data in NetFlow format.

2.5 Summary

Figure 2-1 provides a comparison among the QoS measurement tools above described. It has been adapted from Tables 2-9 and 2-10 of D2.2, now with focus on WP3 aspects.

The comparison table points out several interesting points about the market offer:

- Most of the QoS solutions are active solutions. Indeed the active measurement is traditionally covered by many existing systems unlike the passive measurement.
- Concerning the support of IPv6 QoS, among active solutions, several solutions already support it. Such is not the case for passive solutions.

- Passive systems do not support standard metrics (actually this is explained by the lack of appropriate metric definition).
- Globally multi-point systems are just emerging for both active and passive solutions.
- The table shows a few systems combining active and passive techniques however the scheme for combination is unclear, usually the combination is not redundant that is to say active and passive techniques identifies different kind of network characteristics typically QoS for active and accounting for passive.

Tool	Pass. Active [1]	IPv6 [2]	Metrics	IPPM ITU [3]	Measur. Server [4]	Availability
1-point						
Ping [Ping]	Active	Yes	Round-Trip-Connectivity Round-Trip-Delay Round-Trip-Packet-Loss	Yes Yes No	No	Free of charge.
Pathchar [Pathc]	Active	No	Round-Trip-Connectivity Round-Trip-Delay One-Way-Throughput Round-Trip- Packet-Loss	Yes Yes No No	No	Free of charge but no document-ation, no source code.
Pchar [Pchar]	Active	Yes	Round-Trip-Connectivity Round-Trip-Delay One-Way-Throughput Round-Trip- Packet-Loss	Yes Yes No No	No	Free of charge.
Agilent Advisor [Agi02]	Both	No	Round-Trip-Connectivity Throughput Round-Trip-Delay Packet-Loss	Yes No Yes No	No	Commercial
2-point						
RIPE TTM [RIPETTM]	Active	Yes	One-Way-Connectivity One-Way-Bandwidth-C One-Way-Delay One-Way-Packet-Loss One-Way-IPDV	Yes Yes Yes Yes Yes	At RIPE, analysis based on ROOT [ROOT]	Probes can be purchased at the cost of the hardware
Spirent SmartBits + SmartFlow [Spirent]	Both	Yes	One-Way-Connectivity One-Way-Throughput One-Way-Delay One-Way-Packet-Loss	Yes No Yes Yes	No	Commercial
Brix System [Brix]	Both	No	One-Way-Connectivity One-Way- Throughput One-Way-Delay One-Way-Packet-Loss	Yes No Yes Yes	BrixWorx	Commercial
Netperf [Netperf]	Active	Yes	One-Way-Connectivity One-Way-Delay One-Way-Throughput	Yes Yes No	No	Free of charge.

[1] *Passive, Active, or Both.*
[2] *IPv6 support.*
[3] *IPPM and/or ITU Compliant.*
[4] *Measurement Server included.*

Tool	Pass. Active [1]	IPv6 [2]	Metrics	IPPM ITU [3]	Measur. Server [4]	Availability
Iperf [Iperf]	Active	Yes	One-Way-Connectivity One-Way Throughput One-Way-Delay One-Way-Delay-Variation One-Way-Packet-Loss	Yes No Yes Yes Yes	No	Free of charge.
IPMM2	Passive	No	One-Way-Delay Throughput RTP loss	No No No	Yes	Meter will be available for 6QM
MGEN & trpr	Active	Yes	One-Way-Delay One-Way-Delay-Variation One-Way-Packet-Loss	Yes Yes Yes	Yes	Free of charge.
OpenIMP (initial)	Passive	Yes	One-Way-Delay One-Way-Packet-Loss	No No	Yes	Free of charge.
n-point						
QoSmetrix [Qosm]	Active	Yes	One-Way-Connectivity One-Way-Delay One-Way-Packetloss Jitter (Ipdv) Round-Trip-Connectivity Round-Trip-Delay Round-Trip-Packet-Loss 80 RTP metrics	Yes Yes Yes Yes Yes Yes Yes Yes	NetAdvisor	Commercial

Figure 2-1: Comparison of QoS Measurement Tools

This analysis shows clearly, what the weak points are in the market offer. Those weak points are the following:

- The lack of passive solution supporting IPv6.
- The lack of multi-point systems.
- The lack of systems combining active and passive technique in a redundant way.

Most of the vendors focus their offers on the probes. This is especially true when the probes are hardware based. They can potentially provide a server side but as it is not the main focus consequently, this server side is not that advanced.

The market niche targeted by the 6QM is tailored according to the market offer. Consequently the research is going to:

- Contribute to advance the state of the art in passive methods, by proposing new passive metrics definitions and by developing an original passive system for IPv6.
- Contribute to multi-point systems.
- Take advantage of existing active solution to provide advanced features.

As a conclusion, an ideal solution would combine a strong passive multi-point infrastructure with an existing active solution in order to reach a versatile system with a minimal R&D effort. The prototype scope definition will define clearly in which extent such a system is feasible. This section about the existing products is essential for our project because within 6QM there is a strong feeling that there is no efficient research without a good understanding of the existing or potential needs.

3. 6QM MEASUREMENT SYSTEM SCOPE

The purpose of this section is to define the direction concerning the prototype and to identify the challenges regarding its development.

3.1 Usage of QoS Measurement

The purpose of this section is to highlight the potential usage of the measurement system. The first part describes a simplified service deployment scenario and the second part points out typical measurement usage regarding the described scenario.

3.1.1 Typical Service Deployment Phases

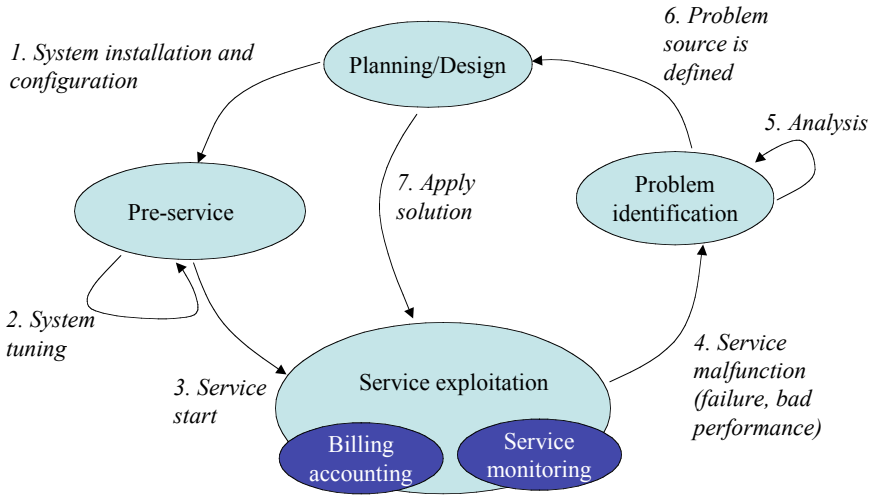

Figure 3-1: VoIP Deployment Phases

The Figure 3-1 illustrates an exemplary VoIP service deployment. The initial phase is the planning phase: The service provider designs, installs and configures its service (1). In pre-service phase, the service provider proceeds with tests and system tuning to meet the service level objectives (2). In this phase the service itself is not open to real customer traffic yet, consequently the provider needs to generate test traffic – ideally with similar characteristics as the deployed service – in order to perform the system tuning. Once the service is considered ready, the provider can provide its service to real customers then starts the exploitation phase (3). The figure stresses especially two aspects of the service exploitation, which are the accounting and the service monitoring. The accounting can be used for applications such as billing for example, or also for capacity planning. Indeed, a service provider may want to account the volume of real-time data that its customers transmitted in order to generate a consumption-based bill. Mobile carriers typically use this billing model for example. Another aspect of the service exploitation is the service monitoring which can be divided into the fault monitoring and the QoS measurement. On the top of the QoS monitoring a classical application is obviously the SLA

conformance. The service monitoring may detect some service failures or some bad performance in the service, and then trigger some alarm on the provider side. In such a case the provider must proceed with some problem identification by the mean of performing tests including measurements and analysis (5). Once the source of the dysfunction is identified (6), the provider needs to find a solution in order to fix the problem and then to apply it (7). Whether the applied changes do succeed can be observed again by service monitoring.

3.1.2 Measurement System Usage

In this service deployment phases there is some strong need for a measurement system in:

- The pre-service phase: The carrier needs to know if a service fulfills the performance requirement, this could be addressed by active measurements as there is no user traffic in the system
- The service exploitation phase.
- Accounting: The carrier needs to compute the volume of transmitted data, this must addressed by passive measurements as the volume values must reflect the amount of data actually transmitted by the customer
- QoS Monitoring: In order to monitor its service quality it is obvious that the carrier needs a QoS measurement system, which could be based on active and passive techniques. If the SLA conformance is involved as explained in the WP2 it is required to use passive measurement technique in order to reflect what the customer really experiences for its traffic.
- The problem identification phase: The carrier needs to know what the service problem is and where the problem precisely occurs. This troubleshooting can be based on previously collected data and on some additional spatial metrics as shown in WP2.

3.2 6QM Measurement System Scope

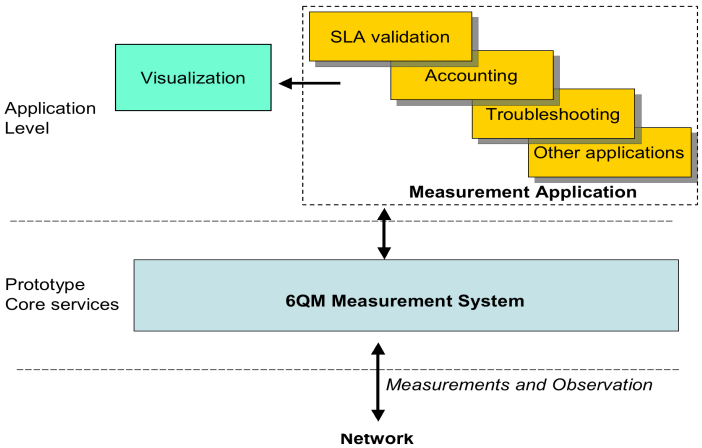

Figure 3-2: Prototype Interaction

The scope of the 6QM Measurement System is to provide some specific QoS measurement services for IPv6 networks. As illustrated on the Figure 3-2, the prototype will provide to upper layer applications a set of QoS metric results, those applications can be related to SLA validation, troubleshooting or accounting for example. The prototype will also provide the configuration interface necessary to setup the measurement tasks. The main concern about the prototype design is to allow the system to be as much flexible as possible in order to allow

further enhancements. Therefore, the presented architecture will propose a design based on an extensive functional decoupling. The applications may also be combined with visualization applications to display their output. In order to display the metric result it is planned to include in the prototype some basic visualization functions at least.

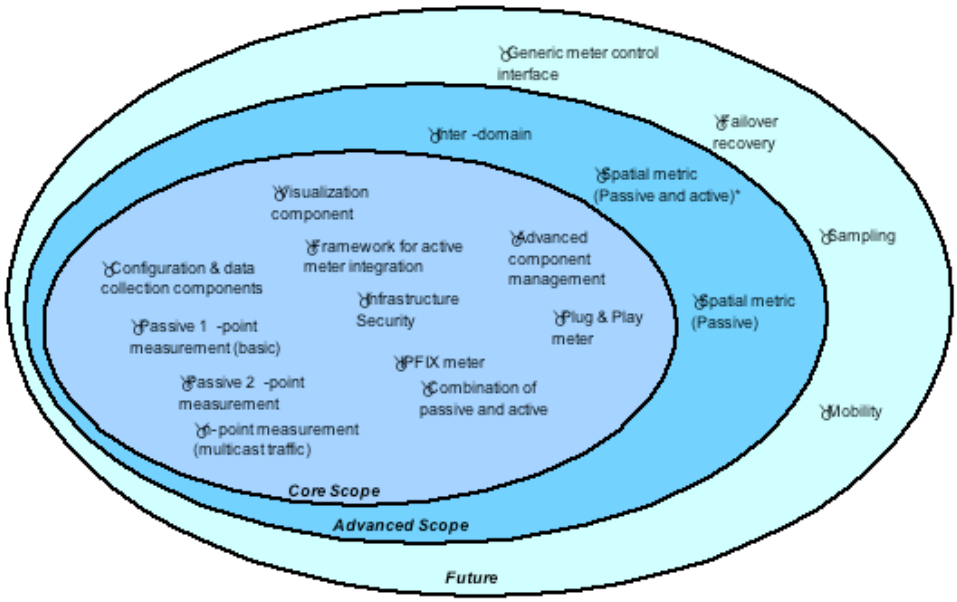

Figure 3-3: 6QM Measurement System Scope

The development of a measurement infrastructure is very challenging and can consist in the development of a wide range of features and functions so that there is a strong need to define some clear boundaries for the project. Those boundaries will define the baseline for some prioritization concerning the infrastructure development.

For the 6QM prototype, WP3 proposes a classification of the main features sorted by the following categories, according to the outcome of the 1st project review (July 2003):

- Core Scope: This is the set of functions that must be supported by the prototype in order to fulfill the project objectives.
- Advanced Scope: This is an extended set of functions that enhances the measurement infrastructure capabilities and that could be included in the prototype however there is no actual commitment for this prototyping.
- Future: This corresponds to a set of items that have been considered as interesting and that could be included in future development.

Figure 3-3 presents this classification. The functions are categorized as follow, for the Core Scope:

- Configuration and data collection components corresponds to the functions of setting up a measurement, collecting and storing the measurement results.
- Passive 1-point measurement refers to the capacity of computing metrics requiring one point of measurement (e.g. volume).

- Passive 2-point measurement refers to the capacity of correlating results from two measurement points to measure the performance of this segment of the path.
- n-point measurement refers to multipoint measurement for delay related metrics with in mind multicast traffic. The challenge consists mainly in configuring the meters consistently and managing the correlation.
- Visualization refers to the graphical displaying of results.
- Framework for active meter integration refers to the integration of active metering capacity.
- Infrastructure security refers here to the security of exported data from the meters to the collecting components.
- IPFIX component refers to the development of a meter component compliant with IPFIX
- Combination of active and passive refers to a scheme correlating both active and passive measurement.
- Advanced component management addresses the robustness and reliability of the infrastructure. The challenge consists in managing the components state, activity and resource consumption and to detect any component failure in order to create a robust system.
- Plug & Play meter refers to advanced management capabilities at the server and at the meter allowing a meter to register itself to a measurement server and to send its measurement capabilities along with the additional information required for a proper meter management.

For the Advanced Scope:

- Interdomain refers to the interdomain measurement presented in WP2. In intra-domain measurements the controlling components have complete knowledge about the location and characteristic of meters and send commands to them to perform measurement tasks. The same assumptions do not hold across administrative boundaries. In the case of inter-domain measurements, each domain must request measurement tasks from the other domain, which might agree or not to perform the required task, depending on local policies, peer agreements among providers, local resource availability, and so on. This problem has been studied within 6QM and a potential solution is proposed in [Yama04].
- Spatial metric (Passive and Active) aggregation refers to the possibility to combine active and passive techniques in order to get aggregated measurements as presented in WP2. The challenge consists in the combination of multiple and heterogeneous meters at the server side for both configuration and data collection.
- Spatial metric (Passive) aggregation refers to the possibility of combining multiple passive meters on the same path in order to get sub-paths delay but with pure passive meters.

For the Future potential development:

- Generic meter control interface addresses the development of an extensible and flexible command language dedicated to the meter control. The extensibility and flexibility become an issue especially when the meters under control are heterogeneous that is very likely to happen in practice and in future deployment.
- Failover recovery refers to recovery schemes in case of collector component failover.
- Sampling addresses some technique allowing the reduction the exported data at the meter.
- Mobility refers to the reconfiguration of the measurement when mobility mechanisms are involved.

The full implementation of the Core Scope should provide the following draft data sheet:

- Target network: OC-3 capacity level.
- Measurement capacity:
  - 1-point capacity: raw packet capture, volume of transmitted traffic.
  - 2-point capacity.
    - Passive metric: one-way-delay, jitter, loss (based on innovative packet identifier generation for optimized bandwidth usage).
    - Active metric: one-way-delay.
  - Support measurement for multicast traffic.
  - Innovative combination of active and passive measurement.
- Features:
  - Bandwidth usage reduction technique (packet identifier).
  - GUI for measurement setting and data visualization.
  - Secured data exportation from meters.
  - IPFIX compliant meters.
  - Simplified meters management and meter auto-registration.
- Requirement:
  - PC (Linux).
  - GPS receiver (for synchronization), NTP.

This gives a first draft for the prototype datasheet at this stage.

4. INITIAL METRICS FOR 6QM MEASUREMENT SYSTEM

This section clarifies the initial metrics to be used in the prototype in order to reflect important network QoS parameters such as one-way delay, delay variation and one-way loss. The focus is put on the passive measurement, as it is the core scope of the 6QM project as defined in the technical annex. The document will explain the limitation of existing standard definitions as far as passive measurement is concerned. Consequently, the document will propose some definitions of the metrics to be used in order to address the lack of appropriate standards. Those definitions will be presented in conjunction with the presentation of a framework for passive measurement including the main building blocks involved in the implementation of the chosen metrics. This will enable us to introduce the techniques planned to gain the measurement data. The last part of the section will list the active metrics that should be supported for an active meter integrated in the prototype along with the list of the spatial metrics of interest in case of development.

4.1 Passive Measurement and Existing Standards

The documents of Work Package 2 have already introduced the existing metrics defined within the scope of RFC 2330 “Framework for IP Performance Metrics” (IPPM) by the IETF and within the ITU-T Recommendation Y.1540 (formerly I.380) “Internet Protocol Data Communication Service – IP Packet Transfer and Availability Performance Parameters”.

Before we tend to the initial metrics of the 6QM Measurement System we want to make a brief judgment to which degree the standardized metrics for three of the most important QoS parameters delay, delay variation, and loss on hand are suitable and applicable to passive measurement methods we will use in the scope of 6QM.

4.1.1 IPPM Metrics

The metrics defined in IPPM related documents are mainly proposed for active measurements – there are several specific points that make their purpose obvious. The metrics are defined on different levels:

- First, there is always a definition for a single instance of a measurement (e.g. for one packet or packet pair).
- Secondly, the singleton metric is extended to a sample metric.
- Finally, from the sample metric important statistical parameters such as mean and percentiles are derived.

The important difference between the active and the passive measurement methodology is their purpose. By nature, the active methodology uses a more pro-active approach where from an active measurement a conclusion is drawn about the QoS parameters within the network. In order to exclude the possibility that injected test traffic is synchronized to the behavior found in the network or the network behavior is affected by periodically injected test packets active measurements (and their metrics), will take samples at point in time according to random distributions, suitable to grant certain properties, of the sample. Namely, unbiased or asymptotically unbiased estimation of the unknown population parameter.

Then on the other hand, passive measurements are inherently non-intrusive so they cannot change the network state. Because of the non-intrusive characteristic of passive measurements, there is no need to select samples from the traffic with the intention used by active methods (with

the exception of reducing data which we do not consider currently). Since the passive methodology takes a complete measurement, that is no sample, we do not need to demand any estimator properties.

It should be apparent that the (sample) metrics for one-way delay (RFC 2679), for packet delay variation (RFC 3393), for the one-way packet loss metric (RFC2680) of the IPPM Framework are not applicable to a passive measurement method. However, we will profit from the defined singleton metrics.

4.1.2 ITU-T Metrics

The metrics defined in the ITU-T Recommendation are not exclusively dedicated to active measurement methods. The requirements of the metric definitions are difficult to fulfill. For instance, the one-way delay metric (IP packet transport delay) applies to the combined set of successfully delivered and corrupted packets. When applying hashing methods where packets are identified by calculated Ids, it is difficult to decide whether a packet is corrupted or lost when relying on matching or non-matching Ids at the measurement points. Similar statements can be made for the packet error ratio and the packet loss ratio; also spurious packets will be difficult to distinguish from corrupted packets.

4.2 Passive Measurement Metrics Definition

We now present the passive measurement metrics we plan to support exemplarily with the 6QM measurement system. This does not mean to restrict the general approach of the system to the initially selected metrics. Since the existing metrics will not match the measurement method we aim for passive measurements, we proposed alternative definitions appropriate for real passive measurements.

In the following, we assume that the IP packets are well formed in the following sense:

- The length of the packet is in the header and equals the sum of IP header length, plus the payload length.
- The IP packet is of version 6 or version 4.
- In case of tunneling IPv6 over IPv4, these properties should hold for the tunnel (i.e. IPv4).
- If we intend to filter flows based upon predicates of the transport layer (e.g. ports) we probably have to demand that IP packets are no fragments unless we provide a filter that can handle such fragments.

In addition, we consider unidirectional flows of traffic from a source host destined for a destination host. For passive two-point measurements (such as one-way delay), we will refer to the measurement point closer to the source of the traffic as reference measurement point (MP^{ref}) and refer to the measurement point closer to the destination of the traffic as observation measurement point (MP^{obs}). Consequently, any packet of an examined flow will arrive first at the reference point MP^{ref} and later at the observation point MP^{obs}. For one-point measurement, there will be only the observation measurement point MP^{obs} (see Figure 4-1).

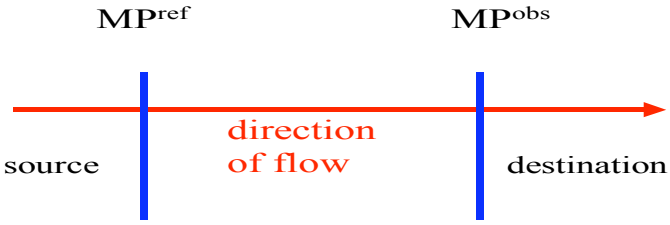

Figure 4-1: Measurement Frame

In favor of performance gain, data reduction, and privacy reasons we intend to use hash techniques to identify packets on reference and observation measurement point by an ID determined based on packet contents. The identifier for a packet P_i will be ID_i. Specifically the symbol for application of a selected packet identifier generator onto the content or parts of P_i at the reference measurement point MP^{ref} is $ID(P_i^{ref}) := ID_i^{ref}$ and will be $ID(P_i^{obs}) := ID_i^{ref}$ at the observation measurement point MP^{obs}. We will use similar conventions for the timestamp of packet P_i – it will be referred to as t_i^{ref}, the point in time packet P_i (more precise P_i^{ref}) is captured at the reference measurement point MP^{ref} and the timestamp is referred to as t_i^{obs}, the time at which the packet P_i (more precise P_i^{ref}) is captured at the observation measurement point MP^{obs}, respectively. Please note that a packet P_i can be observed at the reference measuremement point and is then referred as P_i^{ref}; and it can be observed at the observation measurement point and is then referred as P_i^{obs}.

As for the metrics consisting transport delay of a packet, strictly we are interested in "wire times" as mentioned in the IPPM Framework. Therefore, we adopt from the IPPM Framework the definition of wire arrival time as occurrence of the first bit of a packet P at a specific measurement point. (Note: For passive measurements, there is no need to define a wire exit time because packets are captured only, that is, they arrive at a measurement point.)

In the subsequent metric definitions some symbols are repeatedly used – therefore we present them prior:

- T_{start} measurement start time based on Coordinated Universal Time (UTC).
- T_{stop} measurement stop time based on UTC.
- T_{window} is the maximum expected delay for a packet to proceed from the reference measurement point to the observation point.

4.2.1 6QM One-Way Delay (6OWD)

Parameters:

- T_{start}, T_{stop},T_{window}

Singleton metric:

- The singleton metric exists for a packet P_i^{ref} with $T_{start} \le t_i^{ref} \le T_{stop}$ if and only if there exists a packet P_j^{obs} verifying:
  - $ID(P_i^{ref}) = ID(P_j^{obs})$

- and $T_{start} \leq t_j^{obs} \leq T_{stop} + T_{window}$
- $t_j^{obs} - t_i^{ref} < T_{window}$

- if the singleton metric exists:
  - The singleton metric is defined as $OWD_i := dt_i := t_i^{obs} - t_j^{ref}$
  - The singleton metric tuple (t_i^{ref}, dt_i) is defined too.

6OWD metric is the ordered set of all existing singleton metric tuples.

From this metrics, any statistical analysis is possible including minimum, maximum, arithmetic mean, standard deviation median, percentiles or the distribution and cumulative distribution.

4.2.2 6QM Delay-Variation (6DV)

The delay variation metric uses the 6OWD metric defined above.

The delay-variation will be calculated for selected pairs of packets P_i^{obs} and P_j^{obs} (I < j) ordered by timestamp, such that the corresponding one-way delay singletons OWD_i and OWD_j exist. The packet pairs are selected according to a default selection function, which we choose to be consecutive captured packets within the measurement interval, i.e., j := i+1. This is similar to the first selection function in the IPPM metric specification for packet delay variation (RFC 3393). If required, additional filtering rules may be specified to support multiple selection functions.

Singleton metric:

- The delay-variation will be calculated for selected pairs of packets P_{i-1}^{obs} and P_i^{obs}, i>0, with existing OWD_{i-1} and OWD_i .
- The delay-variation singleton is the tuple $DV_i := (t_i^{ref}, ddt_i)$ with $ddt_i := dt_i - dt_{i-1}$.

The 6DV metric is the ordered set of all singletons DV_i that can be calculated from 6OWD.

From this metrics, any statistical analysis is possible including minimum, maximum, arithmetic mean, standard deviation median, percentiles or the distribution and cumulative distribution.

4.2.3 6QM One-Way Loss (6OWLOSS)

Parameters:

- T_{start}, T_{stop},T_{window}

Singleton metric:

- The singleton metric exists for all packets P_i^{ref} with $T_{start} \leq t_i^{ref} \leq T_{stop}$.
- The singleton metric is the tuple $(t_i^{ref}, loss_i)$ with $loss_i = 1$ for packet P_i^{ref} if no packet P_j^{obs} exists, such that
  - $ID(P_i^{ref}) = ID(P_j^{obs})$
  - and $T_{start} \leq t_j^{obs} \leq T_{stop} + T_{window}$
  - and $t_j^{obs} - t_i^{ref} < T_{window}$
- The singleton metric is the tuple$(t_i^{ref}, loss_i)$ with $loss_i = 0$ for a packet P_i^{ref} if at least one such packet P_j^{obs} exists.

(That roughly means that if for a specific packet at the reference point no matching packet ID can be found at the observation point within the time window then the packet is considered lost.)

The 6OWLOSS metric is the ordered set of all singleton metrics.

Reports to this metric could include absolute number of losses, a time series of all singletons (t_i^{ref}, l_i)as a loss pattern, average loss ratio (total lost over total observed), loss ratio per given interval, loss distribution.

4.3 Passive Measurement Framework

There is a need to setup a framework for the passive measurement. The following part defines the foundation blocks for this framework and proposes a coordination scheme for coherent measurements.

4.3.1 Building Blocks

4.3.1.1 Packet Identifier Generation

For passive multipoint measurements, there is often the need to identify a specific packet as it crosses different spread measurement points. Multipoint measurements have been introduces to analyze the one-way delay behavior of Internet links.

Considerations for Generating Packet Identifiers:

One particular problem that arose from the usage of multipoint measurements was the aforementioned need to identify packets. Purely passive measurement methods neither do indicate packets nor modify them in a way. Consequently, an identification of packets must base on existing information (fields) of the packet itself. Since the identification should be unique in a way such that a specific packet can be recognized on different measurement locations. It seems obvious that information that changes as the packet travels along its path cannot be utilized when deducing an identifier that have to be the same on all measurement points. Used information must be either invariant throughout the path or must be predictable somehow.

Then in order to distinguish one packet among a set of packets one will rather use information that is highly variable between packets – constant or nearly constant information are of no use in the process of generating an identifier. The demand to have the ability to clearly identify a packet is the demand for high uniqueness or the identifier – ideally with no collisions between identifiers of packets within a large set. The probability for collisions depends on several factors: Namely, the distribution of bit sequences taken into account when generating the packet identifier, the generator function for the identifier, the size allocated for the packet identifier. In order to decrease the probability for collisions of the identifier one can use additional input in the generation process in form of bytes of the packet payload ([DuGr00]).

As identifier generators there are a few functions worth considering: Concatenation of unprocessed selected header fields, one-way hash functions ([DuGr00], MD5, SHA-1, etc.), checksums, cyclic redundancy checks (CRCs), compression functions.

Prototype Implementation:

We will now introduce the generation of packet identifiers that we will use in the prototype implementation. The prototype will generate identifier for IPv6 packets, as well as for IPv4 packets. Both variants feed the appropriate packet content into the following CRC generator polynom: $G(x) = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x^1 + x^0$. This

generator is known as AAL5 CRC32 or as Frame Check Sequence (FCS) in Ethernet frames. The packet content that serves as input for the generator is presented in Figure 4-2 for the case of IPv4 and in Figure 4-3 for IPv6, respectively.

For IPv4 packets the total length field (2 bytes), the identification field (2 bytes), the protocol number (1 byte), the source and destination address (4 bytes + 4 bytes). Additional input of up to 20 bytes is taken from the payload (data) if existing.

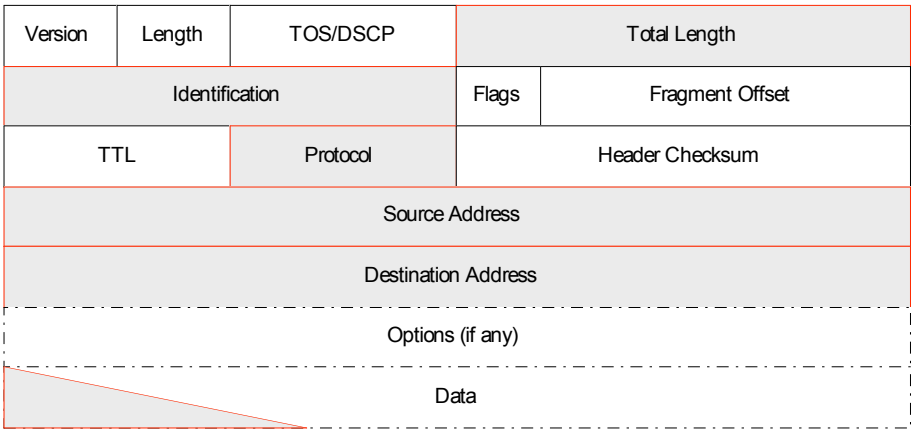

Figure 4-2: IPv4 Packet Content for Generating Identifier

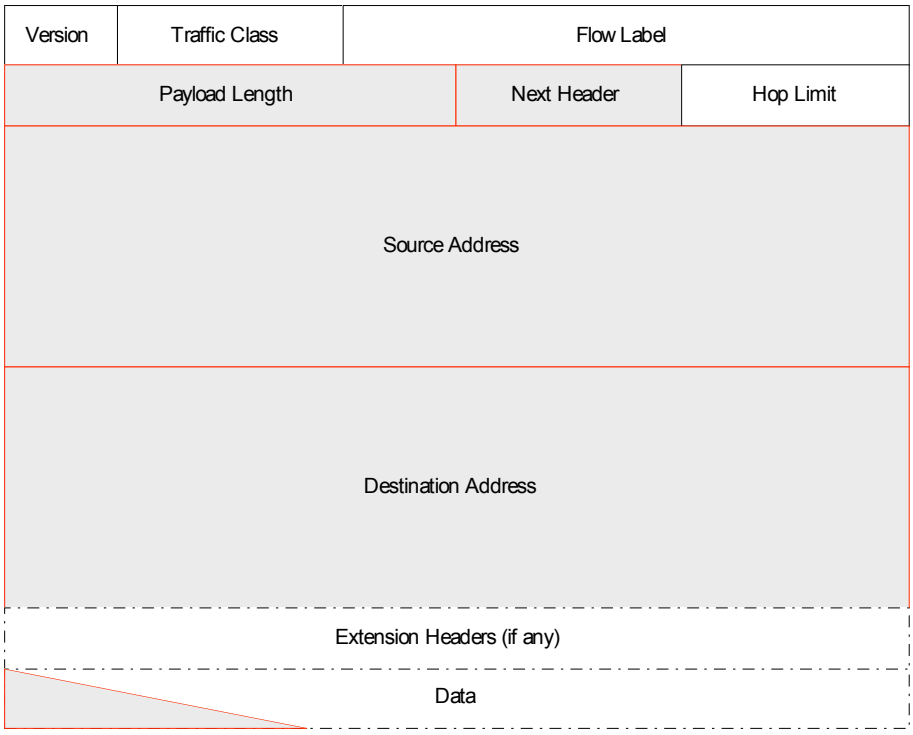

Figure 4-3: IPv6 Packet content for Generating Identifier

The identifier for IPv6 packets uses the Payload Length field (2 bytes), the Next Header field (1 byte), Source and Destination Address (16 bytes + 16 bytes).

The Next Header field needs special treatment. The identifier generation routine will skip any IPv6 Extension Headers until it finds a value for TCP or UDP protocol in the Next Header field. So, this field will be effectively used like the Protocol field of IPv4 packets. More data for

generating the packet identifier is again taken from the payload (data) – up to 20 bytes are used if existing.

4.3.1.2 Packet Matching

The correlation between measurement point results is based on packet matching at a central correlation point. This section describes the possible scenario and the associated actions.

The correlation point receives two sets of data:

- One containing the list of packet identifiers and timestamps generated at the reference measurement point.
- One containing the list of packet identifiers and timestamps generated at the observation measurement point.

The Figure 4-4 shows the correlation scenario for a given packet P belonging to a flow under observation. The proposed analysis is based on the proper coordination proposed and detailed in the following section about measurement coordination. It has to be mentioned that the case where ID(P) does not exist in the set of data from the reference measurement point should not occur when the proposed coordination is respected and when the reference measurement point is close enough to the source.

	ID(P) exists in the set of data from the reference measurement point	**ID(P) does not exist in the set of data from the reference measurement point**
ID(P) exists in the set of data from the observation measurement point	Sufficient data is available for the OWD computation	The packet P is considered as lost because the OWD is impossible to compute (in order to verify the delay upper bound value)
ID(P) does not exist in the set of data from the observation measurement point	The packet P is considered as lost	The packet P is not detected and not considered

Figure 4-4: Correlation Scenario

Exemplarily for two point measurement as one-way delay and one-way loss, the metric evaluation process for a specific packet P_i^{ref} is demonstrated in Figure 4-5. The correlation and metric computation algorithm takes the packets from the reference trace file one after the other. For the currently selected packet P_i^{ref} with $ID(P_i^{ref})$ it looks for an identical ID in the trace file from the observation measurement point. In case a) of Figure 4-5 there is a packet P_j^{obs} with a matching ID. Therefore, the delay, namely td_i, can be calculated from the timestamps of P_i^{ref} and P_j^{obs} – of course packet P_i^{ref} is not a lost packet. Case b) represents the situation where either no matching packet ID is found at all or there is a matching ID at a point violating the time window. Note that matching packet Ids can be present due to the limited properties of the ID generator that produces identical Ids for two different input packets. Consequently the width T_{window} of time window that coincidences with the notion of the maximum expected delay for a packet should be as tight as possible to decrease the ID collision probability for this interval.

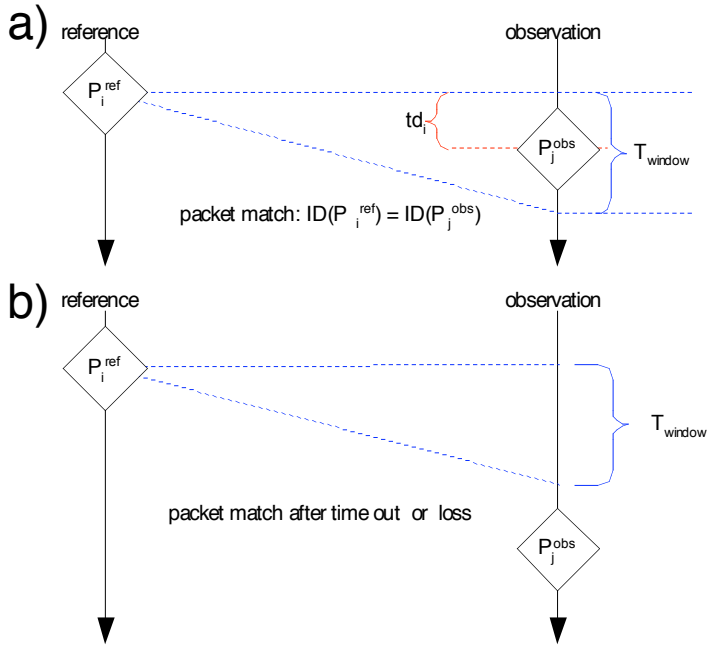

Figure 4-5: Packet Lookup for Two-point Passive Measurement

4.3.2 Measurement Coordination

This following section addresses the need of coordination in multi-point environment the scope of the measurement correlation is restricted to the offline measurement – i.e. the correlation is not proceeded immediately but after the end of some capture period.

The Figure 4-6 proposes a clear picture of the measurement coordination in multi-point mode. The observation meter schedules take care of the window interval as defined in the 6QM metrics in order to take into account the time taken by a packet to be transfer in the network for the reference point to the observation point. Another important issue is the scheduling of the correlation. The correlation process has to be started strictly after the end of a capture period at the observation meters. Besides the correlation, starting time has to take into account other parameters, which are:

- The export schedule of the measurement points (data may not be exported immediately).
- The transfer delay of the measurement data from the measurement points to the correlation point.
- Data availability delay at the correlation point – i.e. the time required between receiving the measurement data and making them available for a correlation component.

The minimum duration between the end of capture and the correlation start is deeply dependent on the measurement system and its configuration. Consequently, it is the operator responsibility to select the appropriate parameters. The responsibility of a measurement system is to provide the configuration services enabling this coherent coordination.

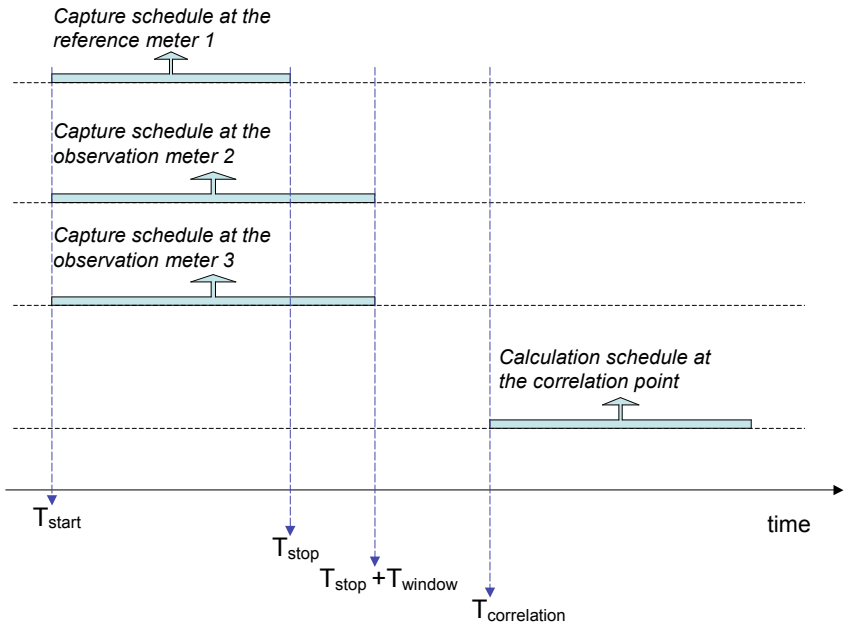

Figure 4-6: Offline Measurement Coordination

4.4 Additional Metrics

In addition to the basic metrics described above, 6QM will also take into account existing and on-going standardization efforts on other relevant metrics such as connectivity, round-trip delay, spatial metrics, etc. For more information on metrics needed to compute SLAs, see D2.1 Section 3. It includes a full overview of most standardized IETF and ITU metrics, as well as other useful metrics which are still not standardized. Deliverable D2.1 Section 6 expresses the needs for spatial metrics and describes an initial proposal for spatial one-way delay standardization. The spatial metrics defined are active ones, therefore work on passive spatial metrics is still open. Extension to such metrics is subject of current work and discussions within the 6QM project. Our decisions and results on this matter will be presented in the next deliverable.

4.5 Summary

As a conclusion, in the course of this document we defined appropriate metrics particularly tailored for the passive measurement and showed how the measurement methods of the 6QM system can be fulfilled in order to meet the requirements of existing metric definitions. We sketched techniques as packet identifier generation and measurement data correlation for two-point passive measurement methods that will be included in the implementation. Those funding blocks are generic enough to serve as the basis for multi-point correlation and spatial metric computation as each of those measurements can be fragmented in a set of two-point correlations.

The implementation of the suggested metrics will representatively demonstrate the capabilities and operation of the 6QM system. When we succeeded demonstrating the effectiveness of the system additional effort can be dedicated to develop and implement more metrics.

5. 6QM PROTOTYPE SYSTEM OVERVIEW

The purpose of this section is to provide the initial 6QM QoS Prototype system overview. The chosen approach is the top to bottom methodology. First, the document provides a high-level system view with the core components and their interactions. Then this high level view is mapped into a more practical and detailed view providing for each core components their functional blocks.

5.1 High-level System Overview

The following part describes the 6QM Quality of Service Measurement System at a high abstraction level. In order to get an overview of the contributing building blocks and the interfaces we present an illustration of the 6QM Measurement System with Figure 5-1.

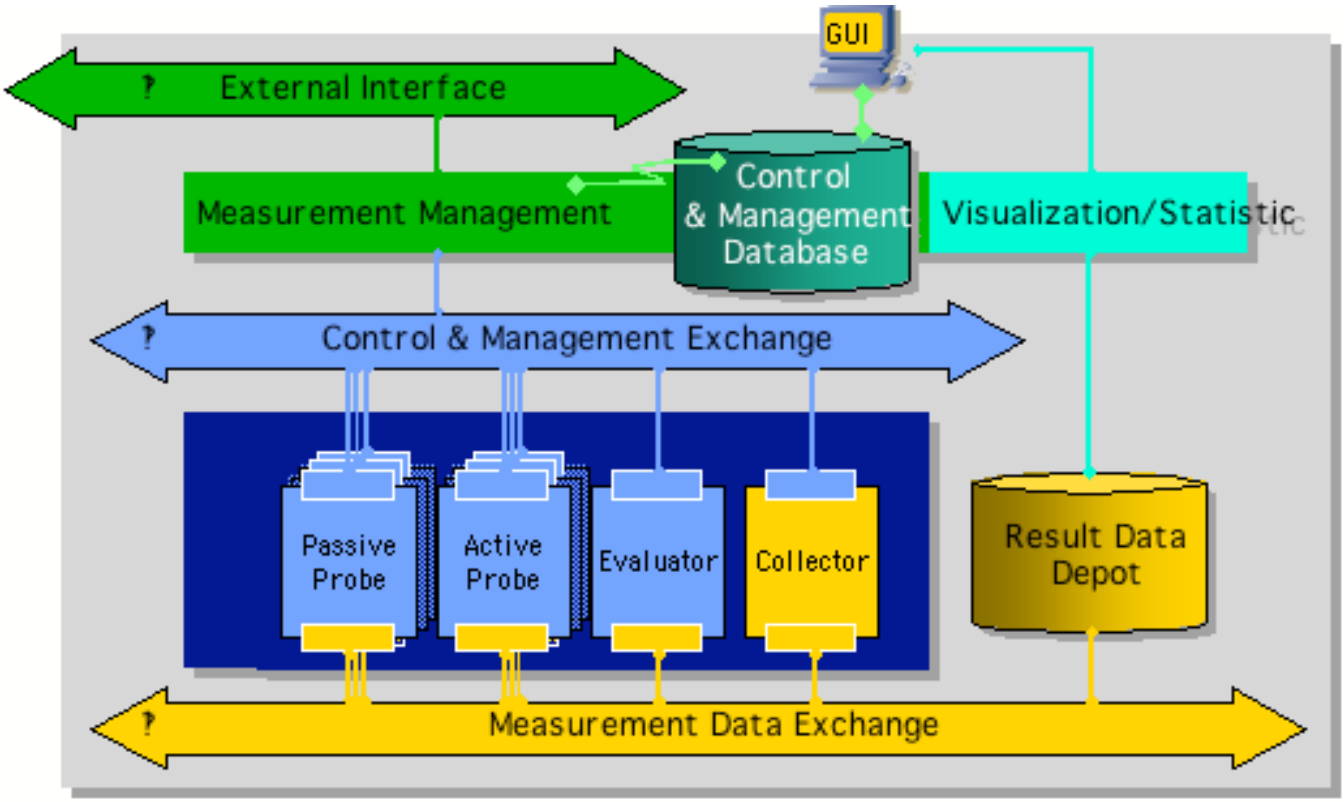

Figure 5-1: 6QM Measurement System (high-level view)

The illustration identifies classes of components of the 6QM measurement system and the functionally different principal interfaces (indicated by broad arrows) that connect the components.

Starting from top of Figure 5-1, the graphical user interface (GUI) empowers users to configure components as well as the overall system. Explicitly, this includes setup of QoS measurements and affected components. In addition, configuration information and measurement management data are presented to the user. Therefore, the GUI itself acts upon a control and management database. The measurement system state that also covers information and status of controlled components is modeled within this database.

As a second main functionality, the user can access and behold the results of executed measurements through the GUI. Components of the class visualization/statistic carry out calculations and run statistical analysis over basic measurement data won afore. Thus, gaining higher order aggregated or summed up results.

The measurement management component is of central significance for the measurement system. As for the Graphical User Interface, there is a strong coupling between measurement management component and control and management database. The main role of the control and management plane is the invocation of tasks on distributed components (of class servers).

How will control and management information be passed on to and gathered from system components? This question will lead us to the control and management exchange interface (represented as the broad blue arrow ②, connecting the control and management plane and most of the remaining system components). The class of server components will accept control messages from the control and management component via the control and management exchange interface.

As for the term server: The class of servers contains components that offer services to the measurement system whereupon where the control and management component acts as a client. Among those components are, or instance, active and passive probes, collectors and evaluators.

A collector component will receive result data from other components of the system via a data exchange interface (see broad yellow arrow ③ in the figure). The collector transforms accepted data into a representation format used within the result data depot and stores it. Every server component has an exporter subcomponent that makes use of the data exchange protocol used for the interface and that is compatible to the collector.

5.2 Functional Architecture

The objective of this part is to describe in more details the initial system architecture. In order to do so the abstract components presented previously are mapped into the 6QM prototype components. The components are also refined into some basic functional blocks. The result of this functional decomposition is presented on the Figure 5-2. Here we address only the core components of the 6QM prototype, not the one related to the application level. As presented on the illustration, the core components are: The 6QM Measurement Manager, the 6QM Evaluator, the passive meter and the active meter.

The following paragraph is going to refine each component with respect to his main functions. This division into functional blocks will be used as based information for our component detailed design later on.

The 6QM Measurement Manager addresses the following functions:

- Enables to be interfaced with an external configuration component such as a GUI (referred as “interface box on the figure”).
- Stores the configuration state and management information of the system.
- Generates the commands to the system components.
- Monitors the other system components to detect any sign of failure.

The passive meter component addresses the following functions:

- Captures and timestamps the packets according to some flow criteria configured by the Measurement Manager. This is the most critical part of the passive meter.

- Analyzes the measurement data before processing the export.

The active meter component addresses the following functions:

- At the sender generates and timestamps the test packets according to configuration defined by the measurement manager.
- At the receiver captures and timestamps the packets belonging to the test traffic according to the measurement configuration.
- Analyzes the measurement data before processing the export.

The 6QM Evaluator addresses the following functions:

- Collects the measurement data as an IPFIX collector.
- Stores the measurement data.
- Processes additional calculation typically when time correlation is necessary.

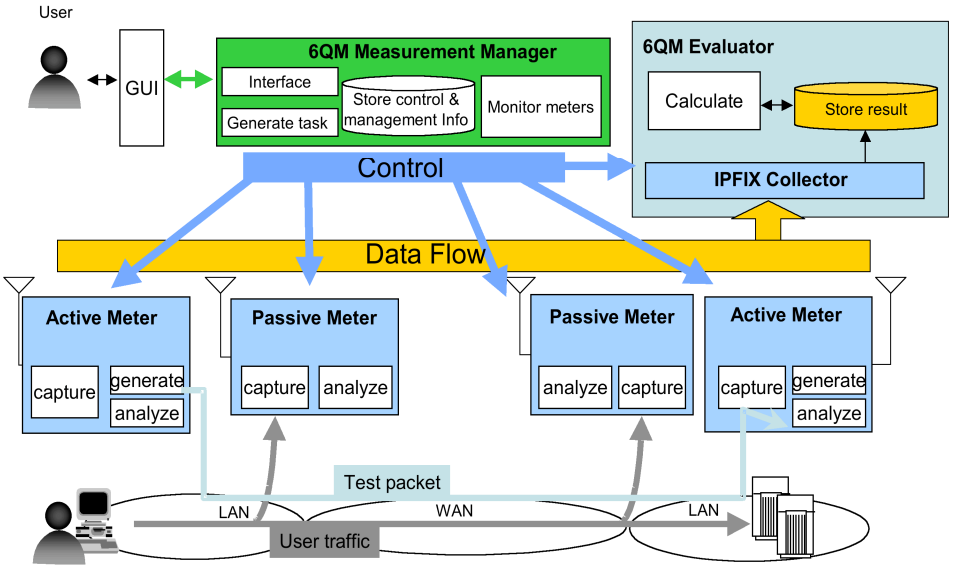

Figure 5-2: 6QM Initial Functional Architecture

This system architecture is a proposal for a flexible and decoupled measurement system targeting the capture of IPv6 packets and the insertion of precise timestamp information. This architecture defines the baseline for the 6QM measurement system by providing the fundamental required components. The following sections address in more details each component presented in this section.

6. 6QM PROTOTYPE COMPONENTS

This section defines more clearly the components, their external interface and their internal structure.

6.1 Passive Meter

This part proposes to address the component details through the description of the component internal structure and then the internal component interactions. In addition the external component interface is defined.

6.1.1 Internal Structure

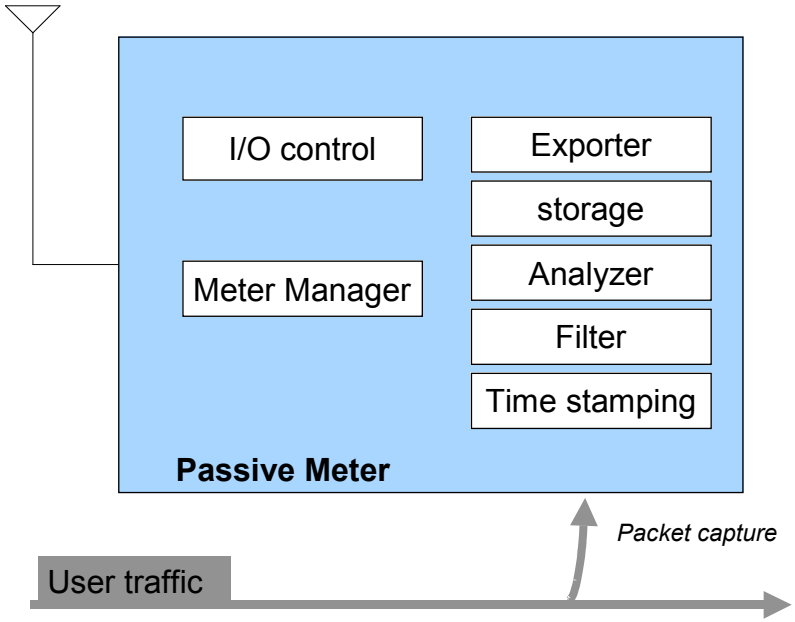

Figure 6-1: Passive Meter Functional Blocks

This section defines the features, which is planned to develop in the 6QM prototype system concerning the passive meter part. The identified functions blocks are illustrated on the Figure 6-1. The components are:

- The control I/O module.
- The time stamping module.
- The filter.
- The Analyzer.
- The storage module.
- The exporter.
- The Meter Manager.

This section gives more details about the passive meter system components. The Control I/O is addressed in the section about the "external control interface".

6.1.1.1 Packet Capture, Time Stamping, Sampling and Filtering

The passive meter performs a copy of the packet without affecting the observed traffic. Using some promiscuous mode features at the meter network interface performs this.

The captured packets are time stamped by the time stamping module and then filtered according the some filtering rules at the filter module (also referred as packet classifier in the document). The packet classifier will support the classification criteria presented in the Figure 6-2, it will also allow the combination of those parameters in order to build a complex filtering rule. Those filtering criteria are relevant in order to define various flow filtering rules. The flexibility offered by this filtering component will allow the support of various applications.

Filtering parameters	Related WP2 requirements
IPv4 or IPv6 source address.	MI1.1
IPv4 or IPv6 destination address.	MI1.2
IPv4 ToS field content / IPv6 Traffic class	MI1.3
Flow label field content.	MI1.4
IPv4 Protocol field content / IPv6 Next header field content	MI1.5
Port number.	MI1.7

Figure 6-2: Packet Filtering Parameters

One of the targeted applications is clearly SLA validation therefore in order to get an accurate measurement of what a user experiences, all the user traffic will be used to perform the QoS measurement. For this reason the sampling capability is not planned to be included in the prototype as a first approach.

6.1.1.2 Synchronization

In order to perform multi-point correlation an accurate synchronization scheme is required as far as delay measurement is concerned. The passive meter will implement GPS synchronization or NTP synchronization for the purpose of synchronization; both methods apply to the host system of the meter.

The chain from synchronization to actual time-stamping an event can bear a couple of uncertainties or errors. For software solutions, one has to be careful in the design, as there are three main factors that influence the time stamping:

- Synchronization offsets (between different measurement components). The amount can be reduced by means of using GPS (best with Pulse-per-Second signal) or NTP. However, since the host clocks are disciplined to the aforementioned sources, the disciplining algorithms must settle before they are in phase and frequency with the reference clock. Furthermore, especially NTP must be configured properly and the accuracy is limited.

- Latencies: On scheduled non-real-time (multitasking) operating systems can add latencies between actual occurrence of an event and the time stamping of it – the latency is furthermore a function of the machine load. Either implementing a time stamping mechanism within the kernel or the usage of a real-time operating system for the timestamp process can minimize the effect. Network interface cards that usually buffer several packets before they release an interrupt and deliver the packets to the operating system, also introduce latencies.
- Resolution of timestamps. The resolution is a minor problem as high-resolution tick counters from modern CPUs can be used to interpolate timestamps for every packet. Otherwise, due to imperfect resolution it will look as if two packets arrived at identical times.

Although [ApDv] discusses the careful design of time stamping methods for active measurement methods, similar effects on passive measurements are obvious since in both cases the time stamping process is affected.

Exemplarily and for informational purposes, we close the discussion on time stamping by presenting some packet inter-arrival times for different Ethernet types (Figure 6-3).

Ethernet Network Bandwidth [Megabits per second]	Mean Packet** Interarrival Times [us]
10	67.2
100	6.72
1000	0.672
** 64 byte packet size + 64 bit inter-frame arrival time + 64 bit preamble (source [SKNETGE])	

Figure 6-3: Packet Inter-arrival Times for Various Ethernets

The accuracy of the time-stamping process is crucial as it directly influences the usefulness of won data. Timing uncertainties in the order of time intervals one tries to measure will result in measurement errors or “nonsense” results.

6.1.1.3 Analyzer

The purpose of the analyzer is to reduce the bandwidth necessary for the exported measurement data.

In order to decrease the amount of data exported by the passive meter, the Analyzer module is in charge of some data preprocessing when applicable. Typically for accounting functions, the meter is able to compute the volume of captured data on a packet base or on a bytes base. The meter can handle this computation by itself without any correlation with any other measurement points. Indeed the meter does not need to send the captured packets to a central server. Consequently, this computation will be carried out in the meter itself and only the metric result will be sent in order to reduce the bandwidth required for the exported data.

In addition to accounting functions the analyzer will be able to generate a packet identifier which is to be associated with a time-stamp value. The packet identifier and the time-stamp will be exported at a central server for correlation in order to compute multi-point time correlation (e.g. passive one-way-delay measurement). Here the main concern is the packet identifier generation

scheme. The packet generation is going to be based on the generation scheme defined in the "Initial Metrics" section.

Concerning the packet identity generation, an advantage of IPv6 over IPv4 is that a packet can be fragmented only at the source host sending the packet and the reassembly is handled at the destination host. In IPv4, routers along the path could fragment a packet but as described in [Zseby01] this fragmentation could raise some serious issues concerning the packet identity matching. Indeed if at a first passive meter a packet is captured, and if this packet is fragmented or re-fragmented before reaching the second passive meter, the packet identity generation will lead to different result at each observation point. Consequently, the traditional packet identity matching is not possible in order to compute delay related metrics. With IPv6 packets such a problem should not occur.

For the sake of clarity, the detailed list of the analyzer computation capabilities is presented in the Figure 6-4. This list includes both accounting functions and packet identity generation function. The Analyzer will be developed as a flexible component allowing an easy integration of new computation modules.

Computation Modules	Related WP2 Requirement	Description
Packet Counter	MI6.1	Count the number of packet per flow and timestamp the first and last packet
Byte Counter	MI6.2	Count the number of bytes per flow and timestamp the first and last packet
Packet Throughput Estimator		Compute a packet throughput estimation over the computation period and timestamp the first and last packet (based on packet counter)
Octets Throughput Estimator		Compute a bandwidth estimation over the computation period and timestamp the first and last packet (based on byte counter)
Packet Identifier Generator		Compute a list of packet identifiers associated with timestamps for all the captured packets

Figure 6-4: Analyzer Computation Modules

6.1.1.4 Meter Manager

The meter manager is the central meter component. The meter manager is in charge of the meter configuration, other component control and task scheduling. More precisely this component provides:

- The management of the task list, which is combined with a scheduler in order to decide when a command should be started or stopped.
- The interpretation of the 6QM Measurement Manager Commands in order to configure the other internal passive meter components.
- The management and reporting of the metering process status (details in Figure 6-5) and task list – part of Extended Scope: Advanced component management.

- The management and reporting of the measurement capabilities – part of Advanced Scope: Plug & Play meter.

Information	Comment
Version	General information
Host name	General information
Starting time	General information
Number of running tasks	Useful for advanced resource management
CPU usage	Useful for advanced resource management
Free storage space	Useful for advanced resource management

Figure 6-5: Metering Process Status Information

The features of the Meter Manager are listed in the Figure 6-6.

Meter Manager Features	Related WP2 Requirements
Manage the measurement task list	
Manage the task scheduling	MI1.12
Configure the others meter components for IPv4 or IPv6 measurements	MI1.13, MI1.14
Manage and report meter resource (i.e. the metering process status information) and task list	MI6.9
Manage and report meter measurement capabilities	

Figure 6-6: Meter Manager Features

6.1.1.5 Exporter and Storage

Concerning the data export part as explained previously, the current plan is to make it be conformant with IPFIX component this way the prototype can take advantage of the standard under development. However as the IPFIX export protocol is not defined clearly yet consequently this part may require an alternate protocol. The alternate exportation scheme will at least use a reliable protocol and support the push mode.

The existence of an intermediate storage component is justified at the meter because the data may not be configured for an immediate export so that the meter has to store temporary the measurement data until the execution of a data export operation. The concern here will mainly be the size of the storage and the management of this storage capacity in order to avoid any storage memory shortage.

6.1.2 Component Interaction

This part proposes to give the overview of the component interaction in the case a new measurement task configuration:

- The control I/O receives a new measurement task and sends it to the Meter Manager.

- The Meter Manager receives, processes the command and adds this new task in its scheduler.
- When the task is scheduled to start the Meter Manager configures the other components according the measurement configuration in other words the Meter Manager is in charge of the whole coordination between components.

The component relation is illustrated on the Figure 6-7 where the thin black arrows represent the control commands and the wide arrow represents the data flow.

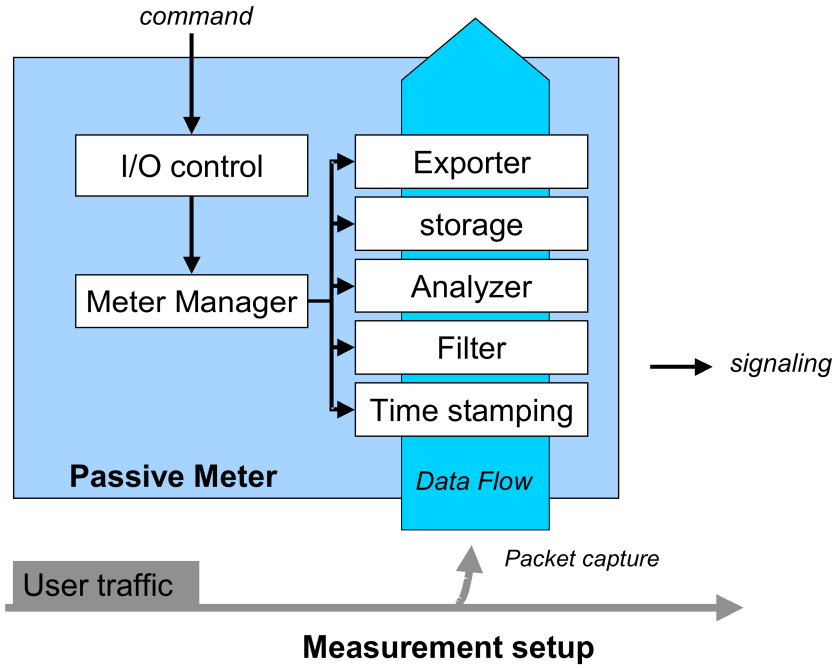

Figure 6-7: Passive Meter Component Interactions

6.1.3 External Control Interface

The signaling interface will use a reliable transport protocol in order to have a reliable signaling. Now no security scheme has been discussed among partners as far as the prototype signaling protocol is concerned but the security could be provided by using a dedicated management network, separated from the public network, as a simple solution.

The signaling protocol will be generic and flexible enough to enable several types of meters. The features of the control interface are more precisely defined in the Figure 6-8.

For a more precise analysis, the meter control interface will accept three commands:

- Add: This command configures a new measurement task.
- Del: This command deletes a measurement task.
- GetInfo: This command requests some meter information.

Signaling interface Features	Related WP2 Requirements
Enable to configure both IPv4/IPv6 measurement tasks	MI7.3
Enable to send meter task list and status – part of Extended scope: Advanced component management	
Enable a meter to send its metering process status information as defined in Figure 6-5– part of Extended Scope: Advanced component management	
Enable to send the meter measurement capabilities–- part of Advanced Scope: Plug & Play meter	
Enable the meter to register itself to the measurement system this function will be referred as auto-registration – part of Advanced Scope: Plug & Play meter – under investigation	

Figure 6-8: Meter Signaling Interface Features

Command Name	Parameter	Comment
add	taskName (m)	This name must be unique within a meter
	IPVersion (o)	This value defines the kind of IP packets under observation. The default value refers to IPv6
	filteringRule (m)	This is a combination of one or several parameters defined in the Figure 6-2 in order to select the packets belonging to a flow
	metric (m)	This value is unique and refers to the metrics defined in the Figure 6-4
	interval (m)	This value defines the time interval for the data export
	schedule (o)	This is a combination of starting date & time plus duration
	target (o)	This specifies the address and optionally port number of the collector
	outputName (m)	This value is the name given for the data output identifier (e.g. the filename)
	IPFIXSupport (o)	This is a yes/no parameter specifying whether the output should be an IPFIX output or not (this is useful only in the case where a native export scheme is defined)
The meter returns a confirmation message or an error code		
del	taskName (m)	This is the unique task identifier
The meter returns a confirmation message or an error code		
getInfo	meteringProcess /taskList/ capability (m)	getInfo commands requires exactly one of the mentioned parameters (see the text for function details)
The meters returns a complex message or an error code		

Figure 6-9: Passive Meter Interface

Those commands are described more precisely in Figure 6-9. In the parameter column (m) indicates that the parameter is mandatory and (o) indicates that the parameter is optional. As presented in the table the getInfo function accepts a request value, which is one the following: MeteringProcess, taskList, capability. This command aims at providing Extended Scope and Advance Scope functions, the precise functions are the following:

- "getInfo meteringProcess" returns the information defined in Figure 6-5.
- "getInfo taskList" returns the list of the configured tasks along with their description and status.
- "getInfo capability" returns the list of the supported computation modules as defined in Figure 6-4 along with the indication for IPFIX support and the auto-registration feature.

6.2 Active Meter

The active measurement is a part of the Advanced Scope for the prototype. In order to adjust this scope extension with the project resource, the prototype will take advantage of an existing IPv6 active measurement system. Consequently, the effort on this part will mainly be an integration effort. The active component will be considered as a "black box" providing some primitives measurement services for configuration and measurement export. The selected active solution to be integrated will be presented in D3.2.

As a conclusion, the 6QM project will neither specify nor develop the active component. The 6QM prototype will target the development of a comprehensive framework able to take advantage of both passive and active components – as a part of the Advanced Scope the commitment on this activity will be tightly dependent on the project available resource as described previously in the prototype scope section.

6.3 6QM Measurement Manager

This part proposes to address the component details through the description of the component internal structure and then the internal component interactions. In addition, the external component interface is defined.

6.3.1 Internal Structure

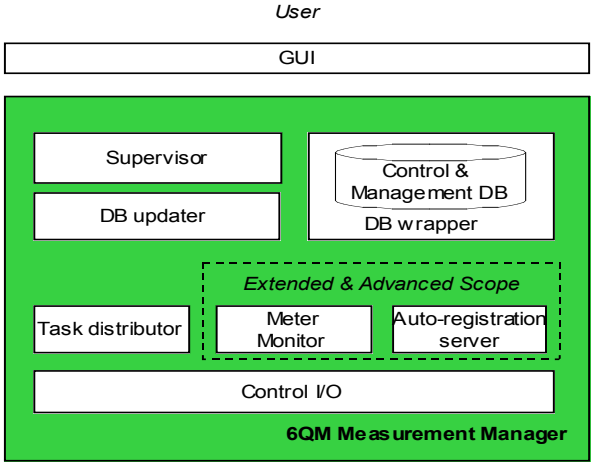

Figure 6-10: 6QM Measurement Manager

The Measurement Manager functionality is addressed in WP2 as "configuration management". Before describing into more details this component, it is necessary to make a distinction between a component task and a measurement task. The measurement task consists in a set of configuration parameters input in the manager component in order to get some metric results. A component task is a set of configuration parameters input by the manager component in the other components in order to create the intermediate results necessary to compute a metric. In order to explain the difference between the two concepts let's take an example, in the case of a passive one-way delay measurement, the user inputs a measurement task into the manager- this can be considered as a high level command. Then the manager component is going to interpret and divide the measurement task into some component tasks in order to generate the meter commands typically. So in this example the manager will request the passive meters to generate packet identifiers and will configure the component in charge of the data correlation.

The 6QM Measurement Manager is in charge of configuring and managing the measurement system. It includes the functions to generate the component tasks to distribute them and to manage the components.

The Measurement Manager internal components are presented in the Figure 6-10. The internal components are:

- The Supervisor: This module takes care of the interaction between the components and the external interface. In the initial 6QM measurement system this module will be interfaced with a GUI.
- The Control and management DB: This is the central part of the measurement system. As the name implies this database holds configuration states and management information of the system. It allows keeping the state of the system components and the measurement task information. More precisely this component stores: The components description (including their description and capabilities), the measurement tasks and the components tasks.
- DB Wrapper: This is a component interfacing the system with the Control and management DB. The DB Wrapper handles all the requests to the DB.
- DB Updater: This component is introduced to keep the consistency between the control and management database and the real system state.
- The Task Distributor: Task distributor is in charge of generating the component tasks and configuring the other system components (the meters and the 6QM Evaluator).
- The Control I/O: This component sends the commands generated by the Task distributor and the Meter Monitor to the measurements components. The signaling exchange is based on a reliable transport protocol to ensure a reliable transmission of commands.
- The Meter Monitor: This component will poll the meters to make sure that they are reachable or alive. Moreover, this component will be able to request the measurement task list or resource information within each component in order to keep track of the meter status – part of Extended Scope: Advanced component management.
- Auto-registration server: This component is still under investigation, it will provide the services necessary for the meter auto-registration – part of Advanced Scope: Plug & Play meter.

In the initial Measurement Manager, the inter-domain issue and the security aspect about accessing the Measurement Manager by the user are not addressed.

6.3.2 Component Interaction

The following section provides more details about the component interaction. The Figure 6-11 presents the Measurement Setup as an exemplary scenario. One key point of the design is that the task scheduling is managed at the meter itself consequently this reduces the complexity at the manager side. However, the Control & Management DB still needs to be updated concerning parameters such as the activity of a task for example; this is independent from any external component interaction. This issue is addressed by the DB Updater component – this component relies on the Meter Monitor component too.

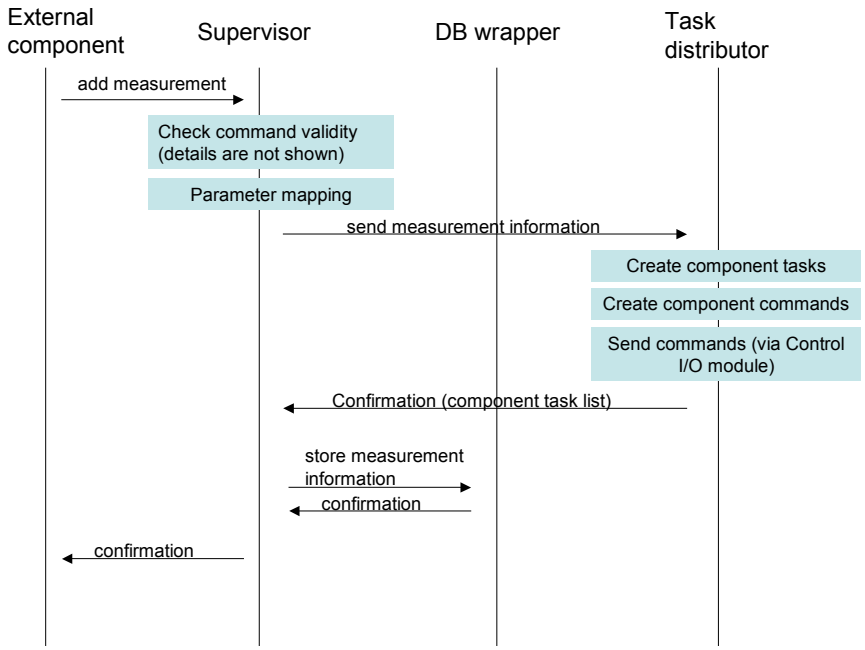

Figure 6-11: Measurement Setup

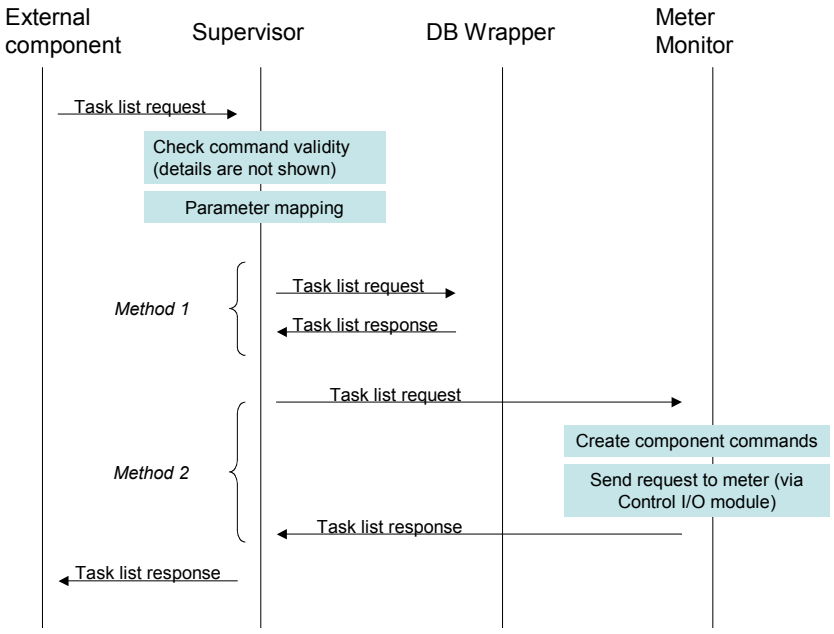

Figure 6-12: Meter Task List Request

The Figure 6-12 presents the request of the task list of a meter as an exemplary management function but this message sequence is also the same for the other management information requests concerning a given component. The illustration shows two methods: One simply using the DB to get the meter list (method 1) and one requesting directly task list to the meter (method 2). The method 1 is clearly faster and simpler but the combination of both methods could help checking some inconsistency between the DB and the real system. The method 2 relies on Extended Scope functions allowing sending management requests to the meter.

The Figure 6-13 presents the registration of a new component in the system. In the future this is simple registration could be enhanced by using the Meter Monitor to check that the new component really exists and to check the component characteristics provided at the registration.

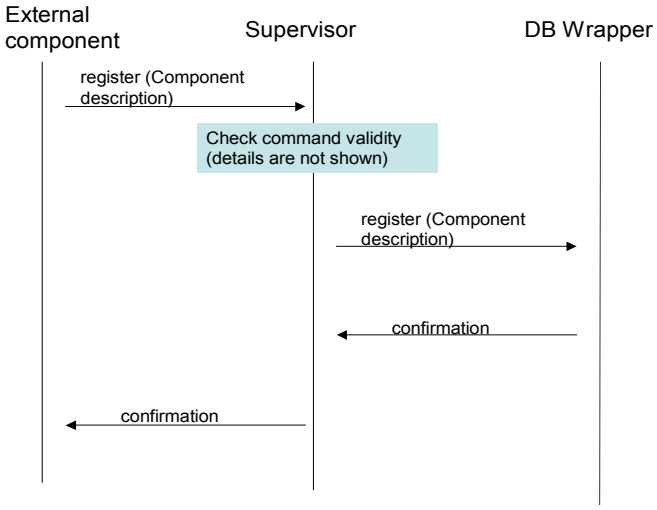

Figure 6-13: Component Registration

6.3.3 External Control Interface

The 6QM Measurement Manager offers an interface to a configuration component such as a GUI. The set of services offered by this component is presented in the Figure 6-14.

For a more precise analysis, the meter control interface will accept six commands divided in two groups one addressing the measurement configuration, and the other one addressing the management operations

The measurement configuration commands are:

- addMeasurement: this command allows configuring a new measurement task.
- delMeasurement: this command allows to deleting a measurement task.

The Figure 6-15 provides a more detailed description of those commands. In the parameter column (m) indicates that the parameter is mandatory and (o) indicates that the parameter is optional.

6QM Measurement Manager Services	Related WP2 Requirements
Enable to configure both IPv4/IPv6 measurements	I9.2
Enable to register/deregister a component	
Enable to request component list	
Enable to request a given component task list and task status – part of Extended scope: Advanced component management	
Enable to request for a given component the host status – part of Extended Scope: Advanced component management	
Enable to request for a given component the metering process information as defined in Figure 6-5– part of Extended Scope: Advanced component management	
Enable to request for a given meter the measurement capabilities – part of Advanced Scope: Plug & Play meter	

Figure 6-14: 6QM Measurement Manager Interface

The management commands are:

- getInfo: This command allows the request of some component information.
- getMngtInfo: This command allows to request management information such as the component list.
- register: This command allows registering a new component.
- deregister: This command allows to remove a component from the manager authority.

The Figure 6-16 provides a more detailed description of those commands. In the parameter column (m) indicates that the parameter is mandatory and (o) indicates that the parameter is optional.

Command Name	Parameter	Comment
addMeasurement	taskName (m)	This name must be unique within a manager
	meterSource(m)	The value for this parameter is a meter identifier identifying the source meter
	meterSourceSchedule (o)	The value of the parameter is a schedule where a schedule is a combination of starting date & time plus duration at the source meter
	meterList (o)	This list contains from one to n meter identifiers in case of n-points measurement. The number of accepted meters depends on the metric to be computed. The meters refer to the destination meters
	scheduleList (o)	List of destination meters schedules
	evaluatorID (m)	This identifies the 6QM Evaluator to be used
	evaluatorSchedule (o)	The value of the parameter is a schedule where a schedule is a combination of starting date & time plus duration at the source meter
	IPVersion (o)	This value defines the kind of IP packets under observation. The default value refers to IPv6
	filteringRule (m)	This is a combination of one or several parameters defined in the Figure 6-2 in order to select the packets belonging to a flow
	metric (m)	This value is unique and refers to the metric identifiers
	exportInterval (m)	This value defines the time interval for the data export from meters
	computationInterval (m)	This value defines the time interval for the data computation at the 6QM Evaluator
	evaluatorOutputName (m)	This value is the base identifier for the data output identifier at the 6QM Evaluator
	meterOutputName (m)	This value is the base identifier for the data output identifier at a meter
The meter returns a confirmation message or an error code		
delMeasurement	taskName (m)	This is the unique task identifier
The meter returns a confirmation message or an error code		

Figure 6-15: 6QM Measurement Manager Task Configuration Commands

Command Name	Parameter	Comment
getInfo	componentID (m)	This identifier is mandatory and defines uniquely the component of interest
	taskList/meteringProcess/capability/hostStatus (m)	Those parameters refer to the actions associated with the "getInfo" command of the passive meter interface and the 6QM Evaluator interface, except for hostStatus which provides information about the host of the measurement component
	DB (o)	This is a yes/no parameter. If the value is yes the response is based strictly on the DB information, if the value is no the response is based on direct meter interrogation. The default value is no
The meter returns a complex message or an error code		
getMngtInfo	meterList/evaluatorList (m)	This commands returns the list of meters or 6QM Evaluators
The meter returns a complex message or an error code		
register	ComponentInfo (m)	This command registers a new meter or a new 6QM Evaluator in the system – see Figure 6-17
The meter returns a confirmation message or an error code		
deregister	componentID (m)	This command removes a meter or a 6QM Evaluator from the manager authority
The meter returns a confirmation message or an error code		

Figure 6-16: 6QM Measurement Manager Management Commands

Parameter	Comment
componentID	This name must be unique within a manager
type	The value of type defines the type of the component: Active meter, passive meter, Evaluator
description	Component description
address	IP address
Port	Controller port
NetworkID	Network identifier
metrics	Metric list

Figure 6-17: Basic Component Description

6.4 6QM Evaluator

This part proposes to address the component details through the description of the component internal structure and then the internal component interactions. In addition, the external component interface is defined.

6.4.1 Internal Structure

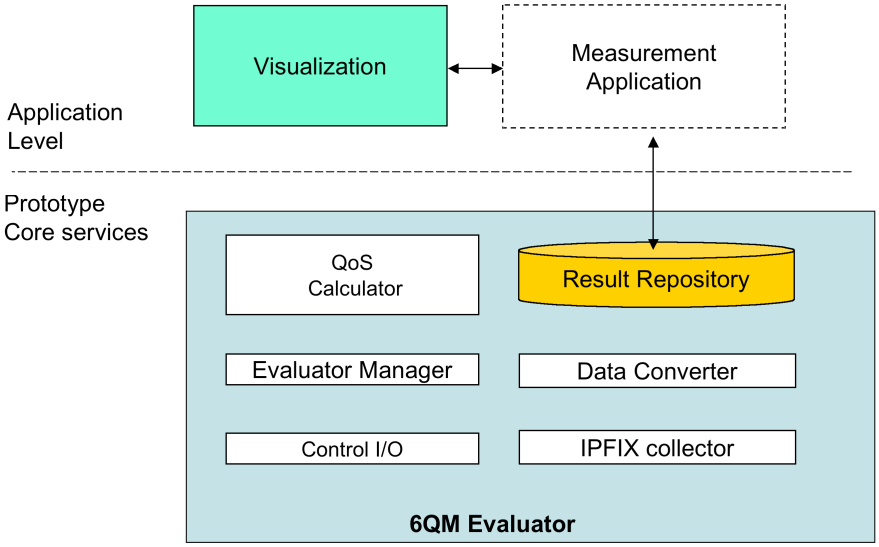

Figure 6-18: 6QM Evaluator

The 6QM Measurement Evaluator component is related to the "collector" defined by WP2. But in the prototype this component will address only basic functions. The detailed structure of the 6QM Measurement Evaluator is presented on the Figure 6-18. The illustration reminds the distinction made between the primitive services offered by the measurement system and the application itself. Indeed the 6QM Evaluator presents measurement results to upper layer application such as SLA, troubleshooting or accounting applications, which can be process further data analysis and afterwards represent their results with visualization components.

6QM Evaluator Features	Related WP2 Requirements
Accepts flow information compliant with IPFIX (remark: IPFIX is still work in progress)	I9.1
Processes the format conversion from IPFIX exported data to file based storage in a format suitable for post processing (e.g. list of comma separated values)	
Stores measurement data in a persistent repository	C1.6
Associates results to measurement and measurement components via database entries	
Correlates the measurement data	

Figure 6-19: 6QM Evaluator Features

The 6QM Evaluator components are:

- IPFIX collector: The task of the IPFIX collector is to accept measurement results from distributed measurement components.

- Data Converter: The purpose of this component is to convert the measurement data from IPFIX format into a format suitable for further processing.
- Control I/O: This module accepts the command for the 6QM Measurement Manager.
- Evaluator Manager: This module is the central part of the component. It is in charge of the configuration, other component control and task scheduling.
- Result Repository: This component ensures the persistent storage of the measurement data.
- QoS calculator: The 6QM Evaluator purpose is to provide measurement results but in case of passive delay measurement some correlations are necessary in order to provide the metrics results. So when applicable this component provides this correlation. The details of the QoS Calculator supported computation modules are described in Figure 6-20. Those metric computations will be based on packet identifier matching techniques as defined in the "initial metrics" section.

Computation Modules	Comment
one-way delay	Provides a list of delays with timestamps
packet delay variation	Provides a list of delay variations with timestamps
Packet loss	Provides a list of packet loss values computed over a time interval

Figure 6-20: QoS Calculator Modules

6.4.2 Component Interaction

This part proposes to give the overview of the component interaction in the case of a new measurement task configuration:

- The control I/O receives a new measurement task and sends it into Evaluator Manager commands
- The Evaluator Manager receives, processes the command and add this new task in its scheduler
- Then the Evaluator Manager configures and controls the other components.

The component relation is illustrated on the Figure 6-21 where the thin black arrows represent the control commands and the wide arrow represents the data flow. The design of this component is very similar to the design of the passive meter.

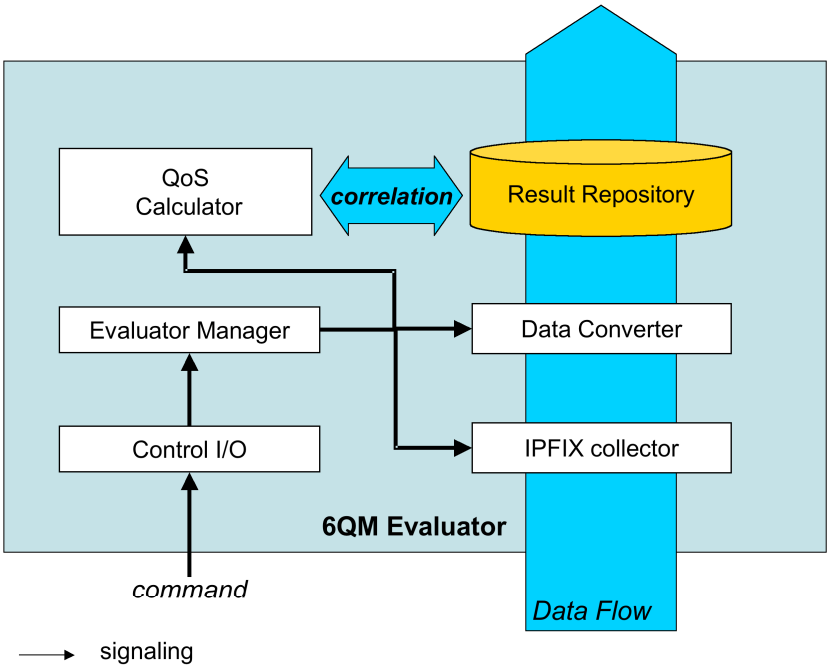

Figure 6-21: 6QM Evaluator Component Interaction

6.4.3 External Control Interface

The signaling interface will use reliable transport protocol in order to have a reliable signaling. The control interface supports:

- The configuration of both IPv4/IPv6 measurement tasks.
- The ability to send task list and task status (part of Extended scope): Advanced component management.
- The ability to send the metering process information as defined in Figure 6-5 (part of Extended Scope): Advanced component management.
- The ability to send the computation capabilities.

For a more precise analysis, the control interface will accept three commands:

- add: This command configures a new measurement task.
- del: This command deletes a measurement task.
- getInfo: This command requests of some component information.

Those commands are described more precisely in Figure 6-22. In the parameter column (m) indicates that the parameter is mandatory and (o) indicates that the parameter is optional. As presented in the table the getInfo function accepts a request value, which is one the following: meteringProcess, taskList, capability. This command aims at providing Extended Scope functions. The precise functions are the following:

- “getInfo meteringProcess” returns the information defined in Figure 6-5
- “getInfo taskList” returns the list of the configured tasks along with their description and status
- “getInfo capability” returns the list of the QoS Calculator supported computation modules

Command Name	Parameter	Comment
Add	taskName (m)	This name must be unique within the component
	meterSourceOutputName (m)	This parameter is referring to the source meter output
	meterDestinationOutputNameList (o)	This is a list of parameters referring to the destination meter output
	IPVersion (o)	This value defines the kind of IP packets under observation. The default value refers to IPv6
	metric (o)	This value is unique and refers to the metrics requiring some correlation
	interval (o)	This value defines the time interval for the data calculation
	schedule (o)	This is combination of starting date & time plus duration
	outputName (m)	This value is the name given for the data output identifier typically the filename
returns a confirmation message or an error code		
Del	taskName (m)	This is the unique task identifier
returns a confirmation message or an error code		
getInfo	meteringProcess /taskList/ capability (m)	getInfo commands requires exactly one of theentionedd parameters (see the text for function details)
returns a complex message or an error code		

Figure 6-22: 6QM Evaluator Interface

This interface is a draft proposal and it is very likely to be changed to follow the evolution of the IPFIX standard and to fit our requirements.

7. JUSTIFICATION FOR USAGE OF OPENIMP AND MGEN

We already introduced OpenIMP within this document. OpenIMP stands abbreviated for Open Internet Measurement Project and has been internally initiated at FOKUS. The development of the Open Internet Measurement Project originated from the intention to make a measurement platform publicly available; so as to promote the persuasion that measurements in computer networks can provide valuable insight into the processes of the examined network. The hardware and prerequisites required to run the measurements should be off-the-shelf components. A more sophisticated solution with dedicated measurement hardware (e.g. hardware-based probes) for more demanding cases is not precluded by such an approach.

A basic implementation of OpenIMP has been started by FOKUS prior to start of the 6QM project. This first realization contained passive meters that are remotely controlled by a central measurement controller instance. Measurement data were pushed from the probes towards a collecting component and a data repository and presented on the GUI.

In the scope of 6QM we decided to further develop the initial OpenIMP implementation since a framework of basic building blocks have already been implemented.

At the time of initial implementation up to the time of writing there were no measurement tools that implemented both passive and active measurement methodology combined with a simple user interface and which are then also freely available and modifiable.

The reasons which pushed the consortium to use the OpenIMP as a based for the prototype system are the following:

- OpenIMP architecture is close to the 6QM specifications so it can serve as a proper base for the prototype.
- OpenIMP has been developed within Fokus so Fokus already has an extensive knowledge of the system which will facilitate the further development.
- OpenIMP is extensible for further functions or integration of active measurement component.
- OpenIMP relies on freely available component (apache, mySQL…).
- OpenIMP is an open source software.

These considerations are strong reasons affirming to advance with the development of OpenIMP within the project scope of 6QM.

Concerning the active measurement part, many active components are already available as a consequence our policy was to integrate an existing component and to create added value functions from it. We decided to use MGEN as an active component as it is light, freely available and simple to integrate. This approach does not prevent to replace this component with a more advanced active component in the future if required.

The direction chosen was to have a software-based plate-form as opposed to a hardware based plate-form because in end-to-end measurements, the number of measurement points can be high and we need a solution that can be cost effective. Our intention was to reuse existing components as much as possible in order to achieve this goal.

8. RELATION BETWEEN WP3 AND WP2

The following tables are extracted from the WP2 requirements they intend to show in which extent WP3 plans to fulfill the WP2 requirements. Those tables refer to the "passive meter", "configuration manager" and "the "collector" respectively as defined in the WP2.

The interpretation is as follow for the column "addressed in prototype":

- An "X" means that the requirement is currently planned to be addressed in the prototype.
- A "~" means that the requirement will partially be addressed in the prototype.
- An empty case means that the prototype does not consider this requirement.

8.1 Passive Meter

Type of requirement	RID	Requirement	Level of requirement	Addressed in prototype
Measurement Operations Traffic Copy	MI0.1	Ability to perform packet capturing in order to obtain a copy of the traffic without introducing modifications in the original traffic.	Must	X
Measurement Operations-Classification	MI1.1	Ability to classify packet according to IPv4 or IPv6 source address.	Must	X
	MI1.2	Ability to classify packet according to IPv4 or IPv6 destination address.	Must	X
	MI1.3	Ability to classify packets according to IPv4 ToS field content / IPv6 Traffic class	Must	X
	MI1.4	Ability to classify packets according to IPv6 flow label field content.	Must	X
	MI1.5	Ability to classify packets according to the IPv4 Protocol field content / IPv6 Next header field content	Must	X
	MI1.6	Ability to classify packets according to Transport addresses.	Must	X
	MI1.7	Ability to classify packets according to previous packets information within a flow.	Must	
	MI1.8	Ability to classify packets according to BGP information (Destination AS, Source AS).	May	
	MI1.9	Ability to classify tunneled packets (v4 over v6, v6 over v4)	Should	
	MI1.10	Ability to classify packets according to incoming interface.	Should	
	MI1.11	Ability to perform classification operations at line-rate	Should	
	MI1.12	Ability to perform classification operations within fixed duration bounds.	Should	X
	MI1.13	Ability to configure classification process	Must	X

		with classification parameters		
	MI1.14	Ability to perform IPv6 and IPv4 configuration consistently.	Should	X
Measurement Operations- Time-Stamping	MI2.1	Ability to time-stamp the first packet of a flow	Must	X
	MI2.2	Ability to time-stamp the last packet of a flow	Must	X
	MI2.3	Ability to perform time-stamp operations before other operations.	Should	X
	MI2.4	Ability to perform time-stamp operations after classification or sampling.	May	
	MI2.5	Ability to perform time-stamp operations on a remote device.	Should not	
	MI2.6	Ability to indicate time-stamping source as well as time-stamping source characteristics (resolution)	Must	
	MI2.7	Ability to choose time-stamping source if several available	Should	X (choice between NTP or GPS depending on meter configuration)
	MI2.8	Ability to perform time-stamping operations at line-rate	May	
	MI2.9	Ability to perform time-stamping operations within fixed duration bounds.	Should	X (performed by a task scheduler)
	MI2.11	Ability to synchronize clocks from a single source.	Must	X (when using NTP)
	MI2.12	Support several clock synchronization sources	Should	
	MI2.13	Support several clock synchronization methods	May	X (GPS or NTP)
Measurement Operations- Sampling	MI3.1	Ability to perform systematic sampling	Must	
	MI3.2	Ability to perform random sampling	Should	
	MI3.3	Ability to perform hash based sampling	May	
	MI3.4	Ability to perform stratified sampling	May	
	MI3.5	Ability to perform classification before sampling	Must	
	MI3.6	Ability to perform sampling before classification	May	
	MI3.7	Ability to configure sampling process with sampling parameters	Should	
	MI3.8	Ability to perform sampling operations at line-rate	Should	
	MI3.9	Ability to perform sampling operations	Should	

		within fixed duration bounds.		
Measurement Operations-Coordination	MI5.1	Ability to perform pre-defined sequences of time stamping, classification and sampling operations.	May	
	MI5.2	Ability to express any sequence of time stamping, classification and sampling operations.	May	
	MI5.3	Ability to indicate if sequences are impossible to execute according to measurement architecture and timing model.	Should	
	MI5.4	Ability to optimize operation placement depending on the sequence to execute.	May	
	MI5.5	Ability to start and stop measurement operations given specific time conditions.	May	
	MI5.6	Ability to start and stop measurement operations when a specific event is detected.	May	
Accounting operations	MI6.1	Ability to account number of packets per flow	Must	X
	MI6.2	Ability to account number of bytes per flow	Must	X
	MI6.3	Ability to account duration of flow	Must	X
	MI6.4	Ability to classify flows according to their type.	Should	X
	MI6.5	Ability to account packets based on their actual size	Must	
	MI6.6	Ability to account IPv6 packet based on the payload length	Must not	
	MI6.7	Ability to deal with fragmented packets.	Must	
	MI6.8	Ability to compute fragmentation rate of flow.	May	
	MI6.9	Ability to measure measurement cost (CPU/memory consumption)	May	X
Measurement operations configuration	MI7.1	Ability to retrieve flow information. This flow information complies with IPFIX requirements. [Qui02]	Must	X
	MI7.2	Ability to provide full packets.	May	
	MI7.3	Ability to perform measurement configuration and to retrieve measurement results remotely.	Must	X
	MI7.4	Ability to pull results from measurement devices to measurement manager.	Must	
	MI7.5	Ability to push results from measurement devices to measurement manager.	Should	X (addressed by the collector)
	MI7.6	Ability to perform exports operations depending on the type of flow (Long lived, Short lived).	Should	
	MI7.7	Ability to perform measurement operations	May	

		configuration and measurement through a single interface. (MP side)		
	MI7.8	Ability to perform measurement operations sequences configuration through the same interface.	May	
	MI7.9	Ability to signal or detect failure or dysfunction of any component of the system.	Must	
	MI7.10	Configuration and result retrieval protocol is loss and error resilient	Should	X
	MI7.11	Support several measurement operations in parallel.	Should	X
	MI7.12	Support several measurement requesters.	May	X (addressed at the measurement manager)
	MI7.13	Ability to express measurement conditions (type of clock synchronization, clock resolution, value of results) for the acceptation of measures.	May	
	MI7.14	Ability to report resources consumption regarding a measurement operation.	May	~ resource consumption is globally reported it is not task based
	MI7.15	Support several collectors for fail over operations	Should	
Impact on network traffic	MI8.1	The impact of passive measurement operations on the traffic measured is negligible.	Must	X
	MI8.2	The impact of passive measurement operations on existing network devices is negligible.	Should	X
	MI8.3	The impact of traffic measurement configuration on the traffic measured is negligible.	Must	X
	MI8.4	The impact of traffic measurement configuration on existing network devices is negligible	Should	X
	MI8.5	Remote management operations have a negligible effect on existing traffic.	Should	X

Figure 8-1: WP2 Passive Meter Requirements

The Figure 8-1 shows the WP2 requirements, which are planned to be fulfilled by the 6QM prototype.

8.2 6QM Measurement Manager

Type of requirement	RID	Requirement	Level of requirement	Addressed in prototype

Measurement operations configuration	I9.1	Flow information complies with IPFIX requirements. [Qui02]	Must	X
	I9.2	Ability to perform active and passive measurement configuration and to retrieve measurement results from measurement devices	Must	X (addressed by a combination of Measurement Manager and Collector)
	I9.3	Ability to perform configuration and measurement retrieval through a single interface.	Should	X (addressed by the GUI)
	I9.4	Ability to perform measurement operations sequences configuration through the same interface.	Must	
	I9.5	Support several measurement operations in parallel.	Must	X
	I9.6	Support several measurement requesters.	Must	X
	I9.7	Support several requests from several requesters simultaneously.	Must	X
	I9.8	Ability to advertise measurement capacities (measurement points, measurement point capacities)	Should	X (Information of measurement components are stored within control and management database)
	I9.9	Configuration interface enables administrator to express measurement conditions (type of clock synchronization, clock resolution, value of results, maximum duration, measurement location, measurement method …) for the acceptation of measures.	Should	~
	I9.10	Ability to report resources consumption regarding a measurement operation along with measurement results.	Should	
	I9.12	Ability to report measurement conditions and limitations along with. This include clock synchronization, sampling method, classification method, computation method, type of measure (active, passive) …	Should	
	I9.13	Ability to provide measurement results through several methods. (Flow based/ Active measurement) – Several results would be provided.	May	
Result Storage	I10.1	Ability to store measurement results in separate DB.	Should	X (collector issue but the Measurement Manager stores the result filenames in a DB)

	I10.2	Ability to query DB to retrieve past measurement.	Should	X (collector issue)
	I10.3	Ability to combine new and past measurement results (e.g. statistical values) through DB queries	May	X (collector issue)
MP configuration	I11.1	Ability to translate measurement configuration in MP configuration.	Must	X
	I11.2	Ability to translate MP measurement results to common format results.	Must	
	I11.3	Ability to pull results from measurement devices to measurement manager.	Must	
	I11.4	Ability to push results from measurement devices to measurement manager.	Should	X (addressed at the collector)
	I11.5	Ability to report failure or dysfunction of any component of the system.	Must	X
	I11.6	Configuration and result retrieval protocol is loss and error resilient.	Should	X

Figure 8-2: WP2 "Configuration Management" Requirements and 6QM Prototype

The Figure 8-2 shows the WP2 requirements, which is planned to be fulfilled by the 6QM prototype.

8.3 6QM Measurement Evaluator

Type of requirement	RID	Requirement	Level of requirement	Addressed in prototype
Measurement Operations- Time-Stamping	C0.1	The Collector has the ability to check remote time-stamping resolution (Cross Check with other measurement source)	May	
Measurement operations: Configuration	C0.2	The Collector has the ability to request the measurement operations in other domains (operation fully performed in foreign domain).	Must	
	C0.3	The Collector has the ability to initiate measurements starting in mother domain and finishing in a foreign domain (cross domain measurement).	Must	
	C0.4	The Collector has the ability to receive synchronous measurement results from other domains.	Must	
	C0.4	The Collector has the ability to share measurement definitions between domains.	Must	
Standardization	C0.5	Measures indicate if the measurement metrics complies with standards and which standards it complies to.	Must	
	C0.6	The measures indicate if the measurement methodology complies with standards and which standards it complies to.	Should	

	C0.7	The Collector has the ability to indicate that a specific metric is not supported or a specific measurement request is not possible.	Must	X (managed at the Measurement Manager)
Publisher/Directory Service	C0.8	The Collector has the ability to advertise measurement capacities (measurement points, measurement point capacities)	Must	X (addressed at the Measurement Manager)
	C0.9	The Collector has the ability to advertise measurement capacities (measurement points, measurement point capacities) of foreign partner domains.	May	
	C1.0	The Collector has the ability to identify and log measurement requests.	Must	
Proxy Server	C1.1	The Collector has the ability to provide “proxy” measurements to other domains for n points measurements.	May	
	C1.2	The Collector has the ability to request “proxy” measurements from other domains for n points measurements.	May	
	C1.3	The Collector has the ability to export measurement to other domains asynchronously. Periodic flow export/flow beginning-end notification.	Should	
	C1.4	The Collector has the ability to receive asynchronous measurement results from other domains.	Should	
Broker	C1.5	The Collector has the ability to find an appropriate service that will satisfy a client’s request. This service may on different machines in the same domain or it may be in external domains. To the requesting client, the Collector’s Broker functionality is transparent. The client neither knows, nor should it care, how the service is provided.	Must	
	C1.6	The Collector stores QoS information in a persistent repository.	Must	X
Authentication Service	C1.7	The Collector provides an authentication service to user who are accessing the system	Must	
Access Control Service	C1.8	The Collector provides access control to any client that is attempting to access a service in the QoS measurement system.	Must	
Service Activation	C1.9	The Collector provides Service activation functionality for all clients interacting with the QoS Measurement system. This means that the service for a particular request may be activated upon demand.	Must	
Persistent Service	C2.0	The Collector stores QoS measurements in a persistent repository.	Must	X

Figure 8-3: WP2 Collector Requirements and 6QM Prototype

The Figure 8-3 presents the WP2 requirements that are planned to be fulfilled in the 6QM prototype.

9. SUMMARY AND CONCLUSIONS

This first WP3 deliverable provides the initial specifications of the 6QM Measurement System Prototype targeting QoS measurement for IPv6. It provides a passive measurement system as proposed in the 6QM technical annex. However, the proposed architecture is flexible enough to potentially integrate some active meters in order to fulfill possible carrier requests.

This deliverable shows a need for passive QoS measurement systems in multi-point mode and a need for systems combining both passive and active techniques. Another output of this deliverable is to fix the 6QM measurement system scope by defining several levels of priorities for the prototype functions. The definition of those boundaries addressed a strong need to clarify the prototype direction and to help in assigning the future development resource. Then the main part of the document defines the prototype itself by defining the initial metrics to be used by the prototype and by providing the system overview with the basic system components.

In addition, the deliverable describes the mandatory components in detail, including their external interface and their internal structure.

The present document defines the baseline for the prototype development consequently it will be reused extensively during the WP3. As far as the WP3 is concerned the next steps are the definition of the prototype detailed design and the development itself. The next deliverable D3.2 will extensively address those issues.

10. APPENDIX: INTER-DOMAIN QOS MEASUREMENTS

This section is a summary and update of the article presented in [Yama04]. We start with an introduction to the main issues in inter-domain measurements and then propose a solution based on automated negotiation architecture.

The ability to perform inter-domain QoS measurements is crucial to provide reliable and high quality services. However today, monitoring an arbitrary end-to-end path today is difficult and restricted, and the obtained information is very limited and inaccurate. Since no central authority controls all domains, inter-domain monitoring is inherently distributed and decentralized. Cooperation among domains cannot be taken for granted, and pre-configured measurement tasks might not suit the need for fast response time required in applications such as troubleshooting and detection of attacks. It is necessary to foster cooperation between providers for the execution of measurement tasks and the provision of the corresponding results. This can only be achieved with a solution that respects each provider's own policies and constraints, and at the same time offers the necessary safety, security and privacy features.

We propose to apply automated negotiation techniques [Jennings2001][Klein2003] as a way to dynamically agree on which QoS parameters may be monitored across domains, depending on the resources available within each domain, the current network conditions, the trust levels among providers, and their respective policies and constraints, including security and privacy constraints.

Automated negotiation mimics human negotiation processes to reach agreements on one or more issues. The idea is to use this technique to agree on parameters for the set-up of measurement tasks across domains and upon demand. We believe that a well-designed automated negotiation mechanism could enable on-demand agreements for the dynamic set-up of measurement tasks across domains, similar to the way goods can be purchased in electronic markets. This could act as an incentive for cooperation, as providers that cooperate to offer monitoring results would be in a better position to offer higher quality services appreciated by customers, and to promptly react to customers' requests.

As a first step towards this goal, we have identified the potential protocols and strategies that could be applied, and mapped monitoring parameters to them. After that we have defined an architecture that enhances the basic 6QM measurement system described in Section 5 in order to cope with the inter-domain case. The next steps, which are currently in progress, are to refine the architectural design, the negotiation protocol and strategy, and to integrate a proof-of-concept implementation of the main aspects into the existing 6QM prototype. These next steps will be reported in Deliverable D3.3, under the revised specification of the 6QM measurement system.

10.1 Architecture for the Inter-Domain Measurement System

Recalling and updating from 6QM Technical Annex (Sections 9.6.2.2, pages 35 to 40) inter-domain QoS measurement can be divided into three phases:

- Measurement set-up: This phase comprises the initial agreement between providers involved in a given measurement task, and the corresponding configuration of the elements involved in the requested measurement. The measurement set-up agreement may be part of an SLA between customer and network provider, or may be established later on, when new measurements are needed.

- Measurement task execution: In the case of passive measurements, passive meters located at strategic positions in the network, or meter components within routers, are activated to run a specified measurement task. In the case of active measurements, an important part of task execution is the recognition and treatment of standard test packets by active probes [Ste02c].
- Measurement result exportation: After a measurement task is executed, standard formats are needed for the exchange of measurement results so that they can be unambiguously interpreted in different domains

Currently none of these three phases is completely automated nor sufficiently reliable. Measurement set-up and exchange of measurement results across domains is still relatively rare in practice, falling far short of what is needed for a quick response to new service demands, for a reliable troubleshooting, for reactive QoS-based services, attack detection, etc.

The proposed inter-domain measurement architecture is based on proxy agents working on behalf of their respective domains. The proxy agents negotiate agreements for measurement set-up and export. An agent-based automated negotiation mechanism is responsible for the measurement set-up process. The mechanism requires a standard negotiation protocol for the exchange of negotiable parameters, and a private agent strategy able to make decisions on behalf of the domain. After an agreement is reached as a result of the negotiations, the agents perform the corresponding configuration tasks within their respective domains, in order to execute the agreed measurement tasks and to further export the results in the agreed export format.

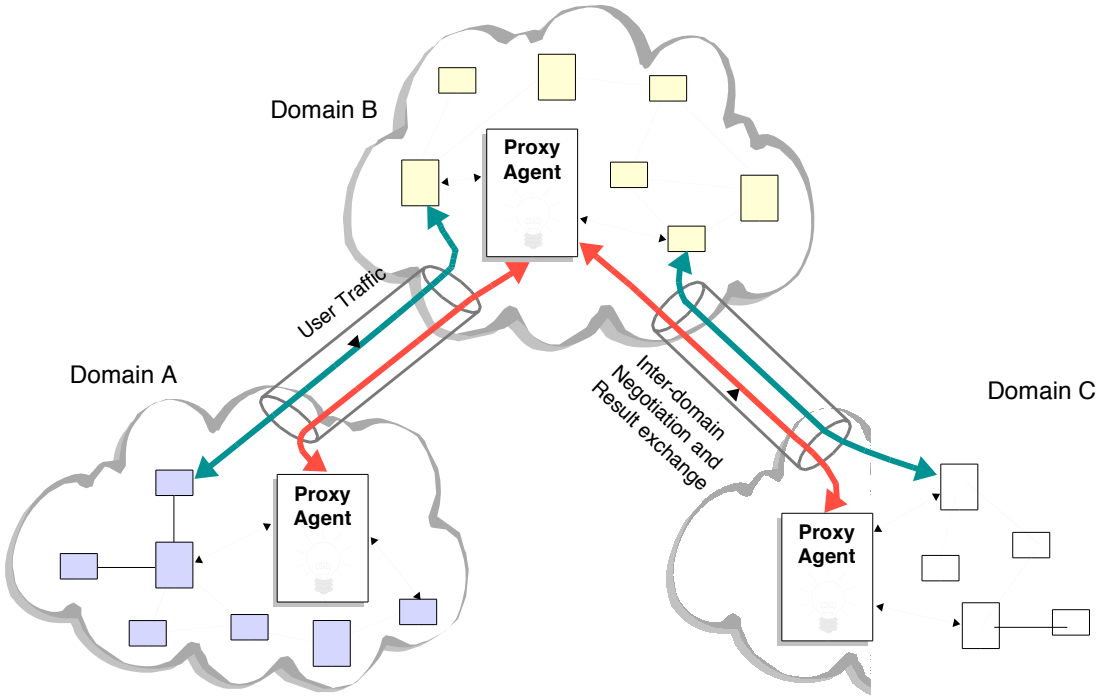

Figure 10-1: Inter-domain measurement architecture: Inter-domain interactions

A sketch of the architecture is presented in Figure 10-1. A proxy agent that is in charge of all the inter-domain negotiations, and of retrieving the associated results represents each domain. The agents communicate with each other using a standardized language and transport protocol. An agent dynamically obtains information from other elements within the domain, about the domain

policies and current network conditions, and uses this information to make decisions. Figure 10-1 also depicts the fact that domains might share a single transmission pipe between themselves, stressing the importance of controlling the agent traffic such that it does not interfere with the traffic from the real users. The domain's proxy agent negotiates on the measurement parameters, accuracy, amount of data to be exported, such as to respect local policies and current network conditions, mainly to avoid overloading the network with measurement data. After an agreement is reached, the agent issues commands within its domain to set up the corresponding measurement tasks.

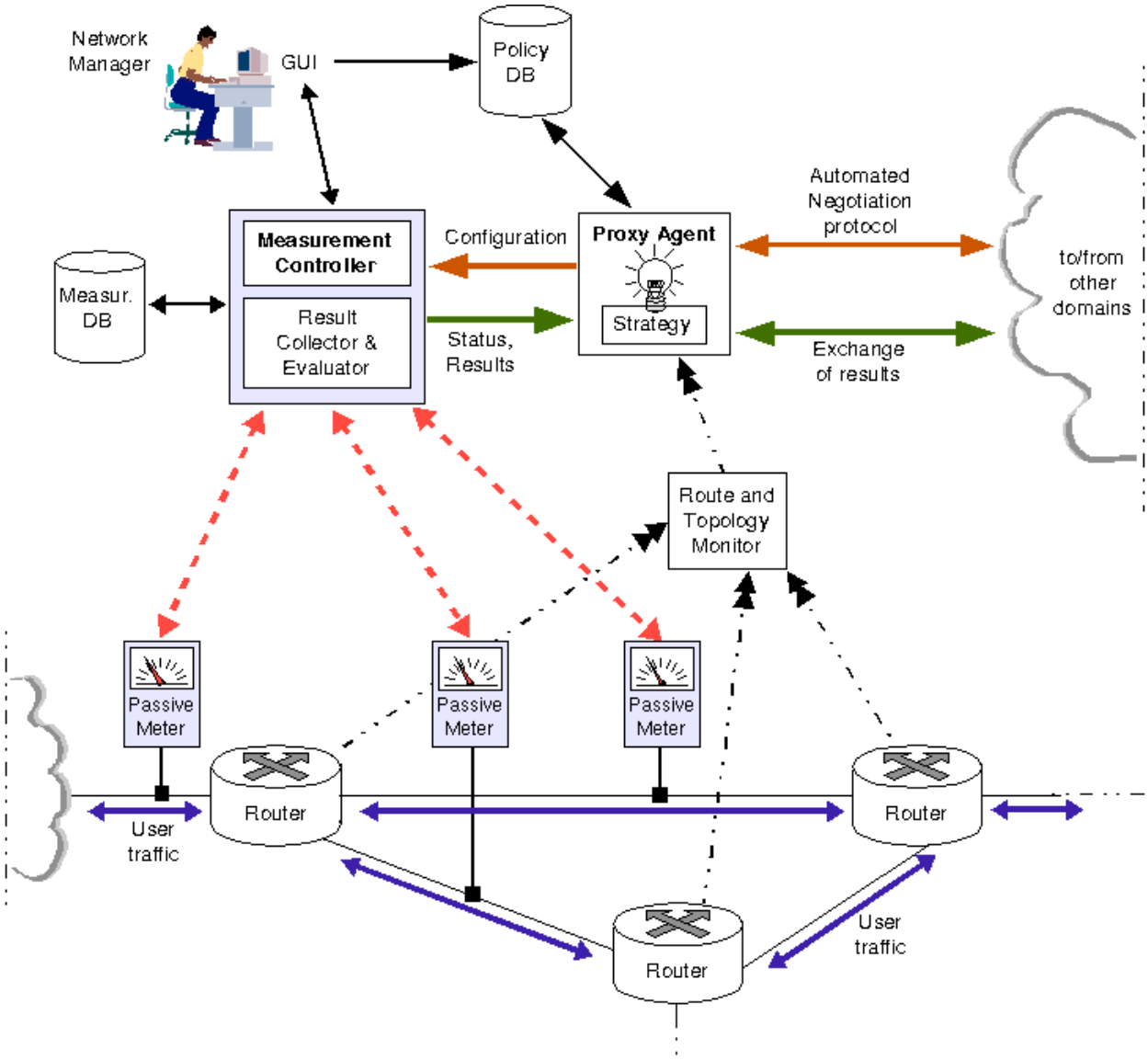

Figure 10-2: Inter-domain measurement architecture: Inside a domain

Figure 10-2 shows a more detailed view of the architecture, inside a domain. This figure is important to see how the 6QM measurement system can be extended with the proxy agents for inter-domain measurements. The measurement controller, collector/evaluator, and passive meters are part of the 6QM measurement architecture defined in Sections 5 and 6.

For intra-domain measurements, the network manager specifies the desired measurements via a web-based user interface, which communicates with the measurement controller. To activate new measurement tasks, the controller issues measurement commands to the passive meters.

During task execution, at specified intervals, the meters send their measurement data to the collector/evaluator, which then calculates the corresponding metrics and stores the results in the measurement database. The network manager may then visualize the results via the user interface.

For inter-domain negotiations, the Proxy Agent may receive new requests from the network manager, requiring a given service from another domain. The agent may also receive requests from other domains requesting a given service.

With respect to the pure intra-domain case described in Sections 5 and 6, in the proposed architecture for inter-domain measurements the internal measurement elements within the domain (measurement controller, collector/evaluator, meters) now always interface with the proxy agent, instead of directly with the user via a web user interface. This introduces transparency in the set-up of new measurement tasks. Once the agent agrees on tasks, or receives orders from the network manager to set up tasks, it communicates directly with the underlying 6QM system by automatically generating and issuing the necessary commands. With this approach, the same measurement system can be used transparently for either pure intra-domain tasks or for more complex inter-domain ones.

After receiving a negotiation request for an inter-domain measurement task, the agent must determine whether the request should be granted, denied, or if a counter-proposal should be generated. This is part of the agent's negotiation strategy. In order to make such a decision, the agent needs information about the current state of the domain in terms of policies and dynamic network conditions. It obtains policy information from the policy database, and meter information via the measurement database. The topology and route monitor provides information on network conditions to the agent in a transparent way, using an interface that is independent on routing or network management protocols. Note that the complete design and implementation of topology and route monitoring software is a complex task outside the scope of 6QM. Fortunately however, there are solutions already available that could be used: A topology discovery module for both intra-domain (OSPF) and inter-domain (BGP) routing protocols has been developed for the INTERMON project and is presented in [Avallone2004]. INTERMON also has a topology collection tool for BGP-4 and corresponding XML data structure to represent inter-domain topology information [Aranda2004].

If the internal policies determine that the request must be denied, the agent sends a denial message to the requesting entity and goes no further. Otherwise, using the obtained meter and routing information, the agent is able to determine which measurement points should be activated for a given measurement task. It then uses meter information again to check whether the concerned points have enough resources to perform the task. Based on the requested parameters, the agent may be able to estimate the amount of data that will be exported, and evaluate whether this amount can be supported with current resources. When available this information can be of great assistance in the decision process. After a decision is made, the agent generates and issues the corresponding commands to the measurement controller, which processes them as if they had come directly from the user interface to the network manager. This helps automating the process of measurement set-up across domains. We now describe the automated negotiation mechanism in more detail.

10.2 Automated Negotiation Mechanism

The mechanism is divided into three parts:

- Negotiable parameters: within the set of all parameters requested for a given service, only a few might be negotiable.
- Negotiation protocol: defines the semantics of the messages to be exchanged, how they are encoded and transported over the network.
- Negotiation strategy: specifies the agent's internal decision algorithms used to obtain the desired negotiation outcomes.

10.2.1 Parameters of the monitoring service

The INTERMON project [Intermon] has defined a document format for the Specification of Monitoring Service (SMS) [Boschi2004], which contains the necessary parameters for inter-domain QoS monitoring. This is exactly what we need in 6QM in order to specify the requested measurement tasks. We are working to keep our list of parameters essentially compatible with the INTERMON SMS format, however we wish to adapt it to the specific case of automated on-demand measurements, in which the recipients of the measurement results might not be human beings directly but software agents intended to interpret the results in order to perform diagnosis or other tasks.

We do not attempt to provide an exhaustive list of parameters. It is also important that the specified format be open enough to accommodate new parameters that might be incorporated in the future.

An agent requesting monitoring service from another domain must specify at least the following parameters in the negotiation request message:

- **Flow description**: describes the flow to be monitored in terms of rules to be applied to the monitored packets to identify the flow, such as source address, destination address, port numbers, protocol, and other packet fields. It is important to be able to measure traffic aggregates, and not only single flows: this is crucial in inter-domain measurements where a huge amount of flows traverse a transit domains.
- **Time schedule**: start and end of monitoring task.
- **Metrics**: the performance parameters to be measured, e.g. one-way delay, loss, jitter, throughput, average packet or bit rate over a specified interval, etc. Each metric may have the following associated attributes:
    - **Notification threshold**: value that triggers a notification to the client domain when exceeded.
    - **Report schedule**: interval for sending periodic reports to the client domain.
- **Report format**: format in which measurement results should be sent to the requesting domain. A suitable standard format, or set of standards according to each metric, must still be agreed upon. Starting points are for instance [Pohl2003][Stephan2003] [Dantonio2003].

The INTERMON SMS format contains other information not treated here:

- **Scope**: ingress and egress points of the traffic flow. In our case, the proxy agents determine these points from the source/destination address or prefix specified in the Flow description, together with route information provided by the Route Monitor.
- **Report destination address** (e-mail, postal, fax,...): this is oriented toward delivery of results to a human customer. In our case, the results are delivered to the agent that requested the service.
- **Security parameters** (authentication data and encryption service): In our case we assume that the Proxy Agent runs over a secure transport connection, such that the

domain identification of the peer agent can be assumed to have already been properly authenticated. Moreover the communication over the secure connection is assumed to be encrypted for privacy when needed.

We must now define which parameters are negotiable, among the previously selected ones (flow description, time schedule, metrics, notification threshold, report schedule, and report format). A non-negotiable parameter must be accepted as is, otherwise the measurement task becomes infeasible. On the other hand, a negotiable parameter admits some flexibility within a range of values, in which the measurement task remains feasible but with different accuracy or resolution.

In principle, the flow description and metrics cannot be negotiated. One could imagine that for a flow described in terms of a network prefix, the prefix length could be negotiated: a longer prefix would mean that less packets are captured, and depending on the purpose of the measurement task this could be sufficient. However this is difficult to quantify in practice. For simplicity we will not consider this possibility.

The other parameters (time schedule, notification threshold, report schedule, report format) are all negotiable in general: A shorter time schedule, or a shift in time schedule, can make a measurement task acceptable for a server domain, while still useful for the client domain. The notification threshold can be adjusted in order to raise less alarms. The report schedule interval can be increased in order to reduce that amount of exported data. The report format can be chosen such as to generate an acceptable amount of data.

All the negotiable parameters go in the direction of saving resources by reducing the amount of information exported. Other parameters should be added to this list. The most important one is accuracy information: the domains must agree on the exact precision of the results in order to be able to interpret them in an unambiguous manner. The precision obviously also has an impact on the amount of information exported, since higher precision values require larger fields to hold them. Sampling and filtering parameters can also be added to control the trade-off between the amount of information obtained and the resources needed.

Besides the parameters of the monitoring service, there are also parameters related to the negotiation itself. The most important parameter is the deadline of the negotiation (timeout).

10.2.2 Negotiation Protocol

The proposed negotiation protocol is essentially an instance of the FIPA Iterated ContractNet Interaction Protocol [FipaIcn]. The choice of an existing protocol has the advantage of dispensing the network community from a potentially long standardization process.

FIPA Iterated ContracNet defines the exchange of messages between an Initiator Agent and one or more Participant Agents. The Initiator issues a Call For Proposals (‘*cfp’* act) to every Participant. Within a given deadline, each Participant may refuse the *cfp* (‘*refuse’* message) or reply with a proposal (‘*propose’* message). The Initiator evaluates all the received proposals and responds with ‘*reject-proposal’*, ‘*accept-proposal’*, or a new, revised *cfp*. In the latter case, a new iteration takes place, with new proposals being evaluated, and so on, until an agreement is reached (i.e. at least one of the proposals is accepted), or the Initiator decides to stop (i.e. reject all proposals, either because they are not satisfactory or because a deadline is reached).

The Iterated ContractNet protocol is very generic and does not specify details of the negotiated parameters. In the case of measurement services it is not necessary to issue multiple calls for multiple agents (that would be the case in a service provisioning request, for example, in which the client domain would issue several concurrent calls to competing domains, in order to choose

the most interesting service offer). Based on this, we have refined the contents of the messages to be exchanged as:

- ***Request***: This is the first message sent from the domain that requests the monitoring service (client domain) to the domain that is expected to provide the service (server domain). It is equivalent to a *cfp*, but specialized for this service. It contains the selected monitoring service parameters described in Section 10.2.1. The format is:

  request(seqno,service)

  where: *seqno* is a sequence number that uniquely identifies the current request within the negotiation; *service* contains the list of <variable,value> pairs that describe the desired characteristics of the requested service, in terms of the selected parameters of Section 10.2.1: flow description, starting time, finish time, list of metrics with corresponding notification threshold and report schedule if any, and report format.
- ***Propose***: This message is issued by the server domain to say that it is willing to offer the requested service. However it may suggest changes in one or more of the negotiable parameters of a previous *request* message. The format is:

  propose(seqno,proposal)

  where *seqno* is the sequence number of the corresponding *request* message, and *proposal* is a list (possibly empty) of <variable,value> pairs containing the new proposed values for a number of parameters. If the list is empty it means that all requested parameters have been accepted. The client agent may or may not accept the proposal. If accepted, it issues an *Accept* message for *seqno*, otherwise it may issue a *Reject* message or a new *Request* message with revised parameters (and a new *seqno*).
- ***Accept***: This message indicates that the previous proposal has been accepted. It successfully terminates the negotiation. Format:

  accept(seqno)

  where *seqno* is the identifier of the corresponding *Propose* message. The of the negotiation is the *Request* whose *seqno* is mentioned in the proposal, modified with the new parameter values proposed in corresponding *propose* message.
- ***Refuse***: This message is issued by the server domain in order to categorically refuse a previously issued *Request* message. This message causes the negotiation to abort. Format:

  refuse(seqno)

  where *seqno* is the sequence number of the corresponding *Request* message.
- ***Reject***: This message is issued by the client domain in order to say to the server domain that it rejects its previous proposal. Format:

  reject(seqno)

  where *seqno* is the corresponding proposal identifier. This message causes the negotiation to abort unsuccessfully.

In addition to the abovementioned parameters, all messages contain a *negid* (Negotiation Identifier) parameter (not shown) to uniquely identify a given negotiation between the two agents involved. It can be formed, for instance, by concatenating the requesting agent's Autonomous System (AS) number with a locally generated sequence number. This allows for multiple negotiations to be handled in parallel.

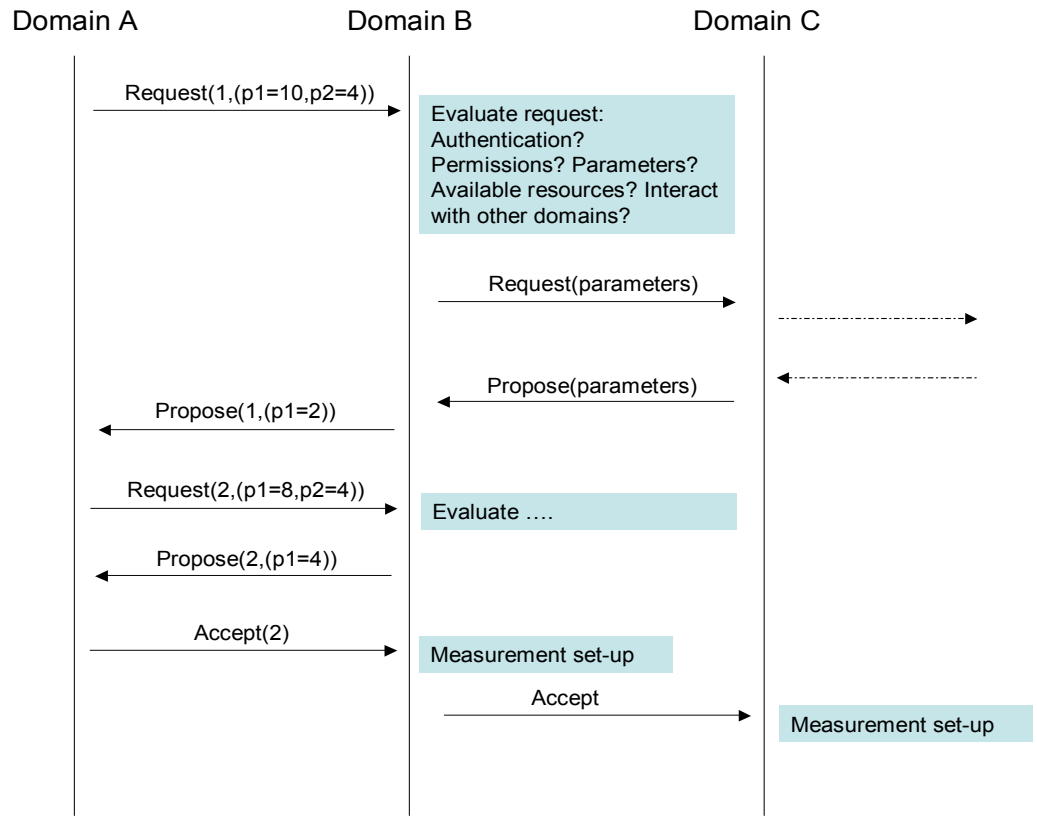

Figure 10-3: Typical inter-domain negotiation session

An example of a typical negotiation interaction is shown in Figure 10-3. Domain *A* (the initiator or client) requests a measurement service with two negotiable parameters *p1* and *p2*. From the topology monitoring information and the flow description, the agent in domain *A* discovers that the next domain on the path of the flow to be measured is domain *B*. It then sends a negotiation request to domain *B* with the desired values for each parameter (i.e. optimum from *A*'s point of view). The agent from *B* evaluates the request, and concludes that it can make an offer to domain *A*, provided that the next domain on the path, which is domain *C*, agrees to provide the service to cover the remaining path. It then negotiates with domain *C*. Domain *C* on its turn, might have to negotiate with other downstream domains before being able to make an offer to *B*. This is a cascade negotiation process involving multiple domains.

After *B* and *C* reach an agreement, *B* can make an offer to *A*. The new proposal takes into account the outcome of the negotiation between *B* and *C* and is also more advantageous from *B*'s point of view (changing the value of *p1* and accepting *p2* as is). After evaluating *B*'s proposal, the agent from *A* decides to send a new request with some modified parameters, hoping to achieve a better deal. *B* then proposes a compromise solution (*p1*=4) which is finally accepted. After that the agent from *B* issues the necessary commands within its domain such that measurement set-up can take place according to the negotiated parameters. It also issues an acceptance message to *C* so that the service can be established on *C*'s side.

10.2.3 Agent Strategy

The agent strategy is not part of the negotiation protocol, therefore can be kept secret. It is indeed in the best interest of each domain to do so, since the agent that has a good negotiation strategy can win competitive advantage by negotiating agreements that are highly beneficial for the domain's owner. Moreover, an agent should not reveal its negotiation deadline, since its opponent could exploit this knowledge to push its own selfish interests (for instance, by offering

a very high price to an agent with a short deadline, hoping that the agent accepts the offer because it is in a hurry to reach an agreement).

Extensive studies on negotiation strategies are available from literature. We have selected a few deemed suitable for the case of a network performance measurement service. First of all, since the agents have deadlines, we restrict ourselves to those strategies especially designed for time-constrained agents. The strategies described in [Faratin2002][Fatima2002] seem very suitable, also because they are able to deal with multiple issues. While the strategy in [Fatima2002] evaluates each issue independently, strategy [Faratin2002] considers the trade-off among different issues.

We are currently mapping the parameters of the service as listed in Section 10.2.1 to the selected strategies [Faratin2002][Fatima2002]. We should soon develop a proof-of-concept prototype in order to evaluate these strategies experimentally over a running IPv6 network.

10.2.4 Security

Before any negotiation can start, the first step is the security check, which is independent of the negotiation strategy adopted. This is not part of the negotiation itself, but we mention it for completeness, and to emphasize the importance of security in inter-domain interactions. Permission are checked to verify whether a given site or user is authorized to perform a given measure. This should follow the policies of each site, which in turn may reflect laws and regulations. For example, ISPs may want to monitor traffic from their clients, however individual users or organizations should be prevented from monitoring traffic from third parties to respect confidentiality and security. In any case users should be prevented from eavesdropping content from other users, so any request to monitor payload information should be denied.

10.3 Summary and Next Steps

In this chapter a flexible architecture for inter-domain measurements based on automated negotiation has been proposed. The next step is to implement a proof-of-concept prototype in order to validate the proposed approach in a quantitative way, to refine the specification of the message exchange standard languages and protocols between domains, and to test different negotiation strategies in practice. In particular, we would like to verify whether the delays incurred by the negotiation process are realistic enough to provide a responsive service compatible with the time frame of the applications requiring measurement services. This is mainly a concern when cascade negotiations involving multiple domains are needed.

Another question is how effective this mechanism is to control resource usage within domains. One of the main goals of the proposed system is to avoid overload and consequent low performance of the measurement system. This requires careful monitoring of resource usage, calibration of the measurement equipment, and an estimation of the traffic that will result from a given measurement task. These are difficult tasks, and more research is needed to provide the answers and methodologies necessary to accomplish them.

In order to be pragmatic, we plan to start with a simple prototype that is able to request measurement tasks from other domains using the INTERMON SMS format described in [Boschi2004]. This provides a standard way to request measurements and specify the desired export format. After that, the negotiation of a single parameter will be implemented and tested. This parameter is likely to be the report schedule, i.e. the interval for exporting periodic measurement results. For the same amount of measurement data, the size of this interval determines the promptness of the results and the amount of information that must be stored

within the server domain. After that, multiple parameters will be introduced, including a simple sampling capability (e.g. select one packet out of N) to keep the amount of information exported within reasonable bounds.

11. REFERENCES

[Agi02] Internet Advisor, Agilent Technologies, http://www.agilent.com/, 2002.

[ApDv] Attila Pástor, Darryl Veitch: High Precision Active Probing for Internet Measurements; 2001

[Aranda2004] Pedro A. Aranda, Paul Malone et al., «Acquisition, Modelling and Visualisation of Inter-Domain Routing Data», In Proceedings of 2nd International Workshop on Inter-Domain Performance and Simulation (IPS 2004), Budapest, Hungary, March 2004.

[Avallone2004] S. Avallone, S. D'Antonio, M. Esposito, A. Pescape, S.P. Romano, «A Topology Discovery Module based on a Hybrid Methodology», In Proceedings of 2nd International Workshop on Inter-Domain Performance and Simulation (IPS 2004), Budapest, Hungary, March 2004.

[Boschi2004] Elisa Boschi, Salvatore D'Antonio, Giorgio Ventre, «Inter-domain Communication and Data Exchange», In Proceedings of 2nd International Workshop on Inter-Domain Performance and Simulation (IPS 2004), Budapest, Hungary, March 2004.

[Brix] Brix System, http://www.brixnetworks.com/, 2002

[Cabri01] Giacomo Cabri, Letizia Leonardi, Franco Zambonelli, "Mobile Agent Coordination for Distributed Network Management", Journal of Network and Systems Management, Vol. 9, Number 4, December 2001.

[Cai02] Workload Measurement Tools Taxonomy, CAIDA, http://www.caida.org/tools/taxonomy/performance.xml, 2002.

[Cflowd] "cflowd: Traffic Flow Analysis Tool", http://www.caida.org/tools/measurement/cflowd/

[CisFcoll] "NetFlow Flow Collector", on-line documentation by Cisco Systems, http://www.cisco.com/univercd/cc/td/doc/product/rtrmgmt/nfc/

[Danalis03] Antonios Danalis and Constantinos Dovrolis, "ANEMOS: An Autonomous Network Monitoring System", Proceedings of Passive and Active Measurement Workshop (PAM 2003), La Jolla, CA, USA, April 2003.

[Dantonio2003] S. D'Antonio, M. Esposito, M. Gargiulo, S.P. Romano and G. Ventre, «A Component-based Approach to SLA Monitoring in Premium IP Networks», First international workshop on Inter-domain performance and simulation (IPS 2003), Salzburg, Austria, February 2003.

[Downey99] Allen B. Downey, "Using pathchar to Estimate Internet Link Characteristics", Proceedings of ACM SIGCOMM'99, Boston MA, USA, September 1999.

[DuGr00] Nick Duffield, Matthias Grossglauser: Trajectory Sampling for Direct Traffic Observation, Proceedings of ACM SIGCOMM 2000, Stockholm, Sweden, August 28–- September 1, 2000.

[Faratin2002] P. Faratin, C. Sierra and N. R. Jennings, "Using similarity criteria to make issue trade-offs in automated negotiations", Journal of Artificial Intelligence, Elsevier Science, Volume 142, Number 2, 2002, pp. 205-237.

[Fatima2002] Shaheen S. Fatima, Michael Wooldridge, Nicholas R. Jennings, "Multi-Issue Negotiation Under Time Constraints", First International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2002), Bologna, Italy, July 2002.

[FipaIcn] FIPA Iterated Contract Net Interaction Protocol Specification, SC00030H, December 2002, Foundation for Intelligent Physical Agents (FIPA), http://www.fipa.org

[Fraleigh02] Chuck Fraleigh et al., "Design and Deployment of a Passive Monitoring Infrastructure", Proceedings of Passive and Active Measurement Workshop (PAM 2001), Amsterdam, April 2001.

[GrDM98] Ian D. GRAHAM et al.: Nonintrusive and Accurate Measurement of Unidirectional Delay and Delay Variation on the Internet; July 1998

[Intermon] INTERMON: Advanced architecture for INTER-domain quality of service MONitoring, isualiz and isualization, IST Project IST-2001-34123, http://www.ist-intermon.org

[Iperf] Iperf Version 1.6.3, The TCP/UDP Bandwidth Measurement tool, NLANR, http://dast.nlanr.net/Projects/Iperf, Oct. 2002.

[Jennings2001] N. R. Jennings, P. Faratin, A. R. Lomuscio, S. Parsons, C. Sierra, M. Wooldridge, "Automated negotiation: prospects, methods and challenges", International Journal of Group Decision and Negotiation, Volume 10, number 2, pp. 199-215, 2001.

[Kahani97] M. Kahani, and H. W. P. Beadle, "Decentralized Approaches for Network Management", SIGCOMM Computer Communication Review, Volume 27, Number 3, July 1997.

[Klein2003] Mark Klein, Peyman Faratin, Hiroki Sayama, Yaneer Bar-Yam, "Protocols for Negotiating Complex Contracts", IEEE Intelligent Systems Journal, Special Issue on Agents and Markets, Volume 18, Number 6, pp. 32-38, 2003.

[MGEN] Multi-Generator Toolset homepage http://mgen.pf.itd.nrl.navy.mil

[Netperf] Netperf homepage, Hewlett-Packard, http://www.netperf.org/

[PaMF02] Konstantina Papagiannaki, Sue Moon, Chuck Fraleigh, Patrick Thiran, Fouad Tobagi, Christophe Diot: Analysis of Measured Single-Hop Delay from an Operational Backbone Network; 2002

[Pathc] Van Jacobson, PathChar, "Path Characteristics", Internet path characterization tool, http://www.caida.org/tools/utilities/others/pathchar/, ftp://ftp.ee.lbl.gov/pathchar/, 1997.

[Pchar] Bruce A. Mah, "pchar: A Tool for Measuring Internet Path Characteristics", http://www.employees.org/~bmah/Software/pchar/, 2001.

[Ping] Mike Muuss, "The story of the ping program", http://ftp.arl.mil/~mike/ping.html, 1999.

[Pohl2003] G. Pohl, L. Mark, C. Schmoll and T. Zseby, "IPFIX Export of packet information for QoS Meaurements", October 2003, IETF Internet Draft (work in progress), draft-pohl-pktid-00.txt, expires May 2004.

[Qosm] QoSmetrix, http://www.qosmetrix.com/

[RFC2330] V. Paxson et al.: Framework for IP Performance Metrics; RFC2330; May 1998

[RIPE] RIPE, Réseaux IP Européens, htt p://www.ripe.int

[RIPETTM] RIPE NCC Test Traffic Measurements (TTM) Service, http://www.ripe.net/ttm/

[ROOT] "ROOT: An Object-Oriented Data Analysis Framework", http://root.cern.ch

[Shal2002] Shalunov et al.: A One-way Active Measurement Protocol; Internet-Draft <draft-ietf-ippm-owdp-05.txt> "work in progress"; August 2002

[Shen03] Chien-Chung Shen, Chavalit Srisathapornphat, and Chaipon Jaikaeo, "An Adaptive Management Architecture for Ad Hoc Networks", IEEE Communications Magazine, February 2003.

[SKNETGE] White Paper SysKonnect: SK-NET GE–- Performance and Reliability for Gigabit Ethernet Server Connections; 1999; (www.syskonnect.com)

[SnPS01] Alex C. Snoeren, Craig Partridge, Luis A. Sanchez, Christine E. Jones, Fabrice Tchakountio, Beverly Schwartz, Stephen T. Kent, W. Timothy Strayer: Single-Packet IP Traceback; 2001

[Spirent] Spirent SmartBits and SmartFlow, by Spirent Communications, http://www.spirentcom.com/

[Ste02a] E. Stephan, "IPPM Multicast metrics measurement", Internet Draft, draft-stephan-ippm-multicast-metrics-00.txt, work in progress, February 2002.

[Ste02b] E. Stephan, “IPPM spatial metrics measurement”, Internet Draft, draft-stephan-ippm-spatial-metrics-00.txt, work in progress, September 2002.

[Ste02c] E. Stephan, “IPPM measurement signature”, IETF Internet Draft (individual submission, work in progress), draft-stephan-ippm-test-packet-header-01.txt, October 2002.

[Stephan2003] E. Stephan and J. Jewitt, “IPPM reporting MIB”, October 2003, IETF Internet Draft (work in progress), draft-ietf-ippm-reporting-mib-05.txt.

[Subram00] R. Subramanyan, J. Miguel-Alonso, J.A.B. Fortes.”A scalable SNMP-based distributed monitoring system for heterogeneous network computing”, SC2000, Dallas, Texas, USA, November 2000.

[Yama04] Lidia Yamamoto, «Automated Negotiation for On-Demand Inter-Domain Performance Monitoring», In Proceedings of 2nd International Workshop on Inter-Domain Performance and Simulation (IPS 2004), Budapest, Hungary, March 2004.

[Zseby01] Tanja Zseby, Sebastian Zander, Georg Carle, “Evaluation of Building Blocks for Passive One-way delay Measurements”, Proceedings of Passive and Active Measurement Workshop (PAM 2001), Amsterdam, April 2001.

[ZsZC01] Tanja Zseby, Sebastian Zander, Georg Carle: Evaluation of Building Blocks for Passive One-way delay Measurements; 2000