




Project Number:	IST-1999-10077
Project Title:	 Adaptive Resource Control for QoS Using an IP-based Layered Architecture
Deliverable Type:	PU - public

Deliverable Number:	IST-1999-10077-WP2.3-SPU-2301-PU-R/b0
Contractual Date of Delivery to the CEC:	Sept 30, 2000
Actual Date of Delivery to the CEC:	Sept 29, 2000
Title of Deliverable:	Report on the development of measurement utilities for the first trial
Workpackage contributing to the Deliverable:	WP 2.3
Nature of the Deliverable:	R – Report
Editor:	Felix Strohmeier (SPU)
Author(s):	Heinz Dörken, Joachim Mende, Ralf Widera (DTA); Markus Isomäki, Hanna-Maija Karjalainen, Tero Kilkanen (ELI); Bernhard Hechenleitner, Ulrich Hofmann, Ilka Miloucheva, Felix Strohmeier (SPU)

Abstract:	This report gives an overview about the measurement utilities available for the 1 st trial and their use.
Keyword List:	AQUILA, IST, QoS, Internet, Measurements

Executive Summary

This deliverable describes the status of the development process of the measurement utilities for its usage in the first trial. Until D2301 the development concentrates on the basic implementation with a measurement information database and simple traffic generators simulating different types of application oriented synthetic load. Part of this deliverable will be a guideline for integration of the measurement utilities into the first trial. Hence, target audience for this deliverable are the members of WP3.1 and WP3.2.

Table of Contents

1	INTRODUCTION.....	8
2	DISTRIBUTED MEASUREMENT ARCHITECTURE	10
2.1	MEASUREMENT REQUIREMENTS IN AQUILA.....	10
2.2	DESIGN CONSIDERATIONS	11
2.3	MEASUREMENT METHODOLOGY.....	15
2.4	PERFORMANCE METRICS	17
2.5	MEASUREMENT COMPONENTS.....	18
2.5.1	Synthetic Flow Generator	18
2.5.2	Active Network Probing Tool.....	19
2.5.3	Router QoS Monitoring Tool.....	20
2.5.4	Measurement Database	21
2.6	BENCHMARK AND QOS VALIDATION	21
3	STATE OF THE ART OF PERFORMANCE MEASUREMENT IN THE INTERNET	24
3.1	AQUILA DMA COMPARED TO STANDARDISATION EFFORTS AND CURRENT RESEARCH	24
3.2	PERFORMANCE METRICS IN THE INTERNET.....	28
3.2.1	General comparison.....	29
3.2.2	One-way-Delay.....	30
3.2.3	(One-way) Delay variation.....	31
3.2.4	One-way-Packet-Loss	32
3.2.5	Other parameters.....	33
3.2.5.1	Round-trip delay	33
3.2.5.2	Service availability.....	33
4	MEASUREMENT DATABASE.....	34
4.1	DESIGN GOALS.....	34
4.2	DBMS EVALUATION.....	35
4.2.1	Specific company requirements	35

4.2.2	System design and software tools	35
4.2.3	Scalability	36
4.2.4	Performance.....	37
4.2.5	Quality	37
4.2.6	Security.....	38
4.2.7	Costs	38
4.2.8	Functional evaluation.....	38
4.2.9	Conclusion	38
4.3	DATABASE DESIGN	38
4.3.1	Database Model	39
4.3.2	Description of Database Entities.....	40
4.3.2.1	Entity „user”	40
4.3.2.2	Entity „test”	40
4.3.2.3	Entity „flow”	40
4.3.2.4	Entity „traffic”	40
4.3.2.5	Entity „distribution”	40
4.3.2.6	Entity „parameter”	41
4.3.2.7	Entity „path”	41
4.3.2.8	Entity „hop”	41
4.3.2.9	Entity „mgmtinfo”	41
4.3.2.10	Entity „result”	41
4.3.2.11	Entity „packet”	41
4.3.2.12	Entity „eventlog”	42
4.3.2.13	Entity „rsvp”	42
5	MEASUREMENT TOOLS.....	43
5.1	SYNTHETIC FLOW GENERATION TOOL (CM TOOLSET)	43
5.1.1	CM Toolset Architecture	43
5.1.1.1	Distributed Agents	44

5.1.1.2	CM Server.....	44
5.1.1.3	User Interface (WWW-GUI).....	45
5.1.2	CM Toolset Functions	47
5.1.2.1	List Tests.....	47
5.1.2.2	Add new Tests.....	49
5.1.2.3	List/Add new Hosts.....	50
5.1.2.4	List Parameters.....	51
5.1.2.5	Add Parameters	51
5.1.2.6	Edit Parameters	51
5.1.2.7	List/Add new Users.....	52
5.2	ACTIVE NETWORK PROBING TOOL	52
5.2.1	Architecture of the T-Nova toolset	53
5.2.1.1	Client.....	53
5.2.1.2	Master	54
5.2.1.3	Offline display.....	59
5.2.2	AQUILA architecture of the T-Nova toolset	60
5.3	ROUTER QOS MONITORING TOOL.....	62
5.3.1	Statistics Retrieval	63
5.3.1.1	Command Line Interface.....	63
5.3.1.2	Simple Network Management Protocol	68
5.3.1.3	Summary	71
5.3.2	General architecture of the monitoring tool	72
5.3.2.1	routerinit	72
5.3.2.2	collector	72
5.3.2.3	analyzer.....	74
5.3.2.4	Shell script	74
5.3.3	Other notes.....	74
6	SYSTEM INTEGRATION	75

6.1	HARDWARE AND SOFTWARE REQUIREMENTS	75
6.1.1	Management Station	75
6.1.2	Measurement Agents	76
6.1.3	Web Server	76
6.1.4	Database	77
6.1.5	Routers.....	77
6.1.6	Performance of load generators on different end systems.....	77
6.1.6.1	Performance of different operating systems	78
6.1.6.2	Properties	82
6.1.6.3	Conclusion	84
6.2	PROPOSED EXAMPLE SCENARIOS.....	84
6.2.1	Simple QoS Measurement Scenario	84
6.2.2	Admission Control Scenario	84
6.2.3	Observing the QoS properties of traffic flows by probing	85
7	ABBREVIATIONS	88
8	GLOSSARY.....	90
9	REFERENCES.....	99

Table of Figures

FIGURE 2-1: MEASUREMENT REQUIREMENTS RELATED TO AQUILA NETWORK ARCHITECTURE	11
FIGURE 2-2: GENERAL STRUCTURE OF CONTROL SYSTEM	11
FIGURE 2-3: OVERVIEW OF THE MEASUREMENT ARCHITECTURE AND RELATIONSHIP TO AQUILA RCL.....	13
FIGURE 2-4: COMPONENTS OF THE DISTRIBUTED MEASUREMENT ARCHITECTURE.....	14
FIGURE 2-5: END-TO-END SYNTHETIC FLOW GENERATION.....	19
FIGURE 2-6: ACTIVE NETWORK PROBING FOR OBTAINING THE PATH PERFORMANCE.....	20
FIGURE 2-7: ROUTER QoS MONITORING TOOL.....	21
FIGURE 2-8: BENCHMARK AND VALIDATION LAYER.....	22
FIGURE 4-1: CONCEPTUAL DATABASE MODEL (ER DIAGRAM)	39
FIGURE 5-1: CM TOOLSET ARCHITECTURE.....	44
FIGURE 5-2: CM TOOLSET MENU AND TITLEPAGE.....	46

FIGURE 5-3: EXAMPLES OF POP-UP WINDOWS..... 46

FIGURE 5-4: DETAILED TEST VIEW..... 47

FIGURE 5-5 CM TOOLSET RESULTGRAPHS..... 49

FIGURE 5-6: ARCHITECTURE OF THE T-NOVA TOOLSET 53

FIGURE 5-7: GUI OF THE MASTER PROGRAM..... 54

FIGURE 5-8: CLIENT DIALOG..... 56

FIGURE 5-9: CONNECTION DIALOG 56

FIGURE 5-10: ONLINE DISPLAY OF THE „ONE WAY DELAY ”..... 59

FIGURE 5-11: ONLINE DISPLAY OF THE „DELAY VARIATION” 59

FIGURE 5-12: OFFLINE DISPLAY OF THE ONEWAY DELAY..... 60

FIGURE 5-13: AQUILA ARCHITECTURE OF THE T-NOVA TOOLSET 61

FIGURE 5-14: AQUILA CONFIGURATION EXAMPLE..... 61

FIGURE 6-1: PHYSICAL ARCHITECTURE FOR THE FIRST TRIAL 75

FIGURE 6-2: TEST ASSEMBLY 78

FIGURE 6-3: QOS MONITORING 86

Table of Tables

TABLE 2-1: PROJECTION OF APPLICATION END-TO-END QOS TO PERFORMANCE METRICS..... 18

TABLE 3-1: GENERAL COMPARISON OF IETF AND ITU-T 30

TABLE 3-2: COMPARISON OF ONE-WAY-DELAY 30

TABLE 3-3: COMPARISON OF PACKET DELAY VARIATION 31

TABLE 3-4: COMPARISON OF PACKET LOSS 32

TABLE 3-5: COMPARISON OF ROUND-TRIP-DELAY 33

TABLE 5-1: CONNECTION PARAMETERS 57

TABLE 5-2: INFORMATION IN THE CONNECTION REGION..... 58

TABLE 5-3: PARAMETER NUMBERING 74

TABLE 6-1: WINNT TO WINNT 79

TABLE 6-2: WINNT TO WIN95..... 80

TABLE 6-3: LINUX TO SUN 81

TABLE 6-4: LINUX TO LINUX..... 82

TABLE 6-5: OVERVIEW OF OPERATING SYSTEMS 83

TABLE 6-6: EXAMPLE SCENARIO FOR ADMISSION CONTROL..... 85

1 Introduction

As QoS becomes a more and more important topic in the IP-world, measurements are necessary to monitor the delivered quality and network states relevant for resource control as well as to test QoS networks, whether they are able to provide their more or less guaranteed service quality. This has to be done within most realistic scenarios and as precise as possible.

Metrics, that describe the QoS-Parameters (delay, delay variation, packet loss,...) are currently within the standardisation process (IETF, ITU) to have a common understanding about the measurement results. Network states which have to be monitored are specified in proprietary or standard MIBs.

The work-package “Distributed QoS Measurements” is responsible for development of measurement methodology and tools for QoS measurement and monitoring of QoS-related parameters in the framework of AQUILA, which is specified in [D1101][D1201][D1301].

A distributed measurement architecture (DMA) is designed considering the AQUILA component architecture, including ACA, RCA and EAT and the defined network service concept. In this context, the measurement architecture will

- provide the validation of the end-to-end QoS provision in AQUILA based on mappings between the *measured* end-to-end-QoS, the *used network service* (PremiumCBR, PremiumVBR, ...) and the end-to-end QoS which is *required by the user*.
- offer monitoring of state information (e.g. packet loss) for the resource control layer.

To be able to offer, guarantee and account for the network services in the AQUILA architecture, the DMA provides the following components and functionalities:

- To be able to define measurement scenarios of arbitrary complexity the DMA is **scalable**.
- For comprehensive measurements and performance analyses, the measurement architecture integrates **different measurement techniques like active network probing, passive monitoring and synthetic flow generation**.
- To systemise and manage QoS measurements on different levels concerning end-to-end and intermediate flows, a **measurement database** is designed to be able to **store and reproduce measurement scenarios** and to **enable correlation analysis** based on the measurement results.
- Measurement tools are offered **for different layers** such as transport and network (router) layer and therefore enable **mapping between the QoS measured** on the different layers.

- **Synthetic flow generation** for different types of applications required for accurate performance analyses of end-to-end QoS mapping of the applications to the network services (“validation”).

The implementation of the measurement methodologies considers standardisation efforts and current research (e.g. standardised metrics, MIBs, ...).

This document is intended to give a report on the status of the designed and implemented measurement tools as well as their integration in the AQUILA architecture and therefore is structured as follows:

Chapter 2 contains the description of the distributed measurement architecture and some background about validation and control components.

Chapter 3 describes the state of the art of the current research regarding metrics and measurements.

Chapter 4 describes the measurement database, which will be employed. It contains the evaluation criteria of the DBMS as well as the database design.

Chapter 5 contains the description of the developed measurement tools and their current state of implementation.

Chapter 6, the system integration discusses the required hardware and software platforms for the measurement tools and describes example scenarios to be applied in the 1st trial.

The document is completed with the glossary, the abbreviations and the references sections in chapters 7, 8 and 9.

2 Distributed Measurement Architecture

2.1 Measurement Requirements in AQUILA

The distributed measurement architecture is built in the framework of AQUILA for scalable, distributed end-to-end QoS provision in the Internet [D1201].

The main focus of AQUILA is the implementation of different network services based on DiffServ mechanisms. To support network operation and validate the concepts of different QoS levels measurements are needed.

Measurements have to:

1. evaluate and validate

- the **realised quantitative values** for QoS parameters of single flows using the network services (network service request via EAT or directly to ACA)
- the **effectiveness of service provisioning** by AQUILA resource and admission control components (e.g. evaluation of the number of accepted flows and the under- and over-provisioning of their QoS).

The network services in AQUILA are seen as products for which customers have to establish contracts, so called Service Level Agreements (SLAs) [D1101]. The contract based on the selected network service depends on the application requirements, reservation requirements, periods of reservation time, etc.

The network services are mapped into traffic classes which characterise different types of network traffic flows [D1301]. The information elements single rate, dual token bucket, single token bucket, sliding window and their parameters describe the traffic. Based on these traffic descriptions, the reservation requests are computed by the admission control. The reservation request is accepted or denied depending on the network load.

One measurement requirement is to analyse the impact of these mechanisms and their parameters to the end-to-end QoS of the applications. For a systematic approach for the validation of the different mechanisms in use synthetic workload generators are needed. For the 1st trial generic load generators will be available.

2. support network operation and resource control

i.e. a measurement database and performance analysis capabilities can be used immediately by the network operator to maintain and enhance the network operation (“online analysis”). Therefore measurements are required to obtain performance information about

- the networks paths

- the network elements

The possibility to access the measurement information (describing paths performance and loads) enables the AQUILA components (RCA, ACA, EAT) to realise effective control loops.

The required measurements of AQUILA are summarised in Figure 2-1:

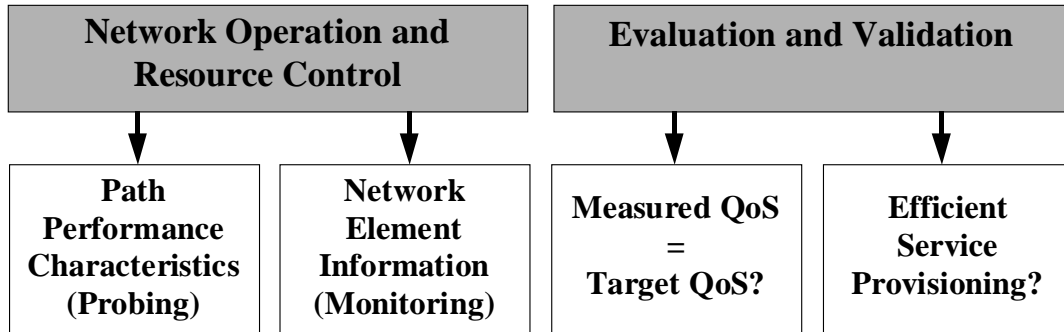


Figure 2-1: Measurement requirements related to AQUILA network architecture

2.2 Design Considerations

The modular design follows the general block structure of a control system with the components shown in Figure 2-2.

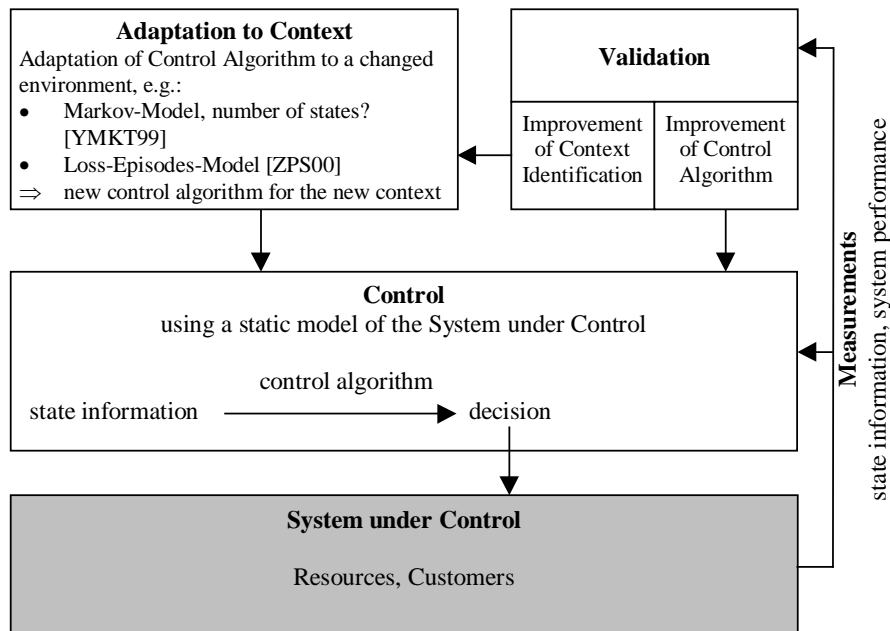


Figure 2-2: General structure of control system

The DMA is designed to realise the following components of this structure:

- Emulation of end-user applications (customers) as a part of the “System under Control”
- Measurements:
 - state information as input to the control algorithms (e.g. path performance, network traffic, etc.)
 - system performance (end-to-end QoS for single and aggregated flows) as input to validation

Regarding the measurement requirements the distributed measurement architecture is modular designed to reach the following goals:

- to emulate end-user applications and to characterise the mapping of their end-to-end QoS requirements into network services in a quantitative way, i.e. measurement of performance metrics for generated single and aggregated flows.
- to maintain and store measurement information of different types (end-to-end performance, path metrics, network traffic, etc.) in order to be used for a mapping of the user end-to-end QoS to the network service requirements.
- to offer online access to measurement information, e.g. to enhance the network operation by the use of path performance metrics or packet drops in network elements.
- to provide time continuous performance analysis of measured end-to-end QoS of the network services with their mappings to traffic classes and resource reservations in order to enhance SLAs and to predict the performance of the network services.

These goals are achieved by combining the following new tools for distributed active and passive measurements interacting via using a common, shared measurement database:

- **End-to-end flow generators for single and aggregated flows** based on distributed agents for end-to-end performance measurement. The goal is generation of synthetic traffic emulating typical end-to-end applications (e.g. VoIP, streaming audio, streaming video, WEB access, etc.). To implement these generators, **passive measurements of real applications**, which should use the AQUILA network, are done to obtain traces for traffic generation and to derive traffic models. Existing models of other research activities are considered.
- **Active network probing agents** for obtaining path performance metrics (e.g. packet delay or packet loss as defined by IPPM) for characterisation of reachability/connectivity as well as performance metrics of particular paths and links in the access and core networks.
- **Router QoS monitoring** for collecting measurement information concerning traffic classes and router characteristics from the core routers and edge devices. As the single and aggregate flows are mapped into traffic classes, this tool measures the performance characteristics of network elements concerning these traffic classes.

The main intention of storage of performance information in a measurement database is to use these information for end-to-end QoS validation. The database can also be accessed directly by the RCL, which may use this information for a better distribution of resources within its individual Resource Pools and to retrieve the current condition of the network. Figure 2-3 illustrates the concept of the measurement tools with their measurement database and the relationship to the resource control layer of AQUILA. The single components of the architecture are described in detail in chapter 2.5, the measurement database is described in chapter 4. The implementations of the tools are described in chapter 5.

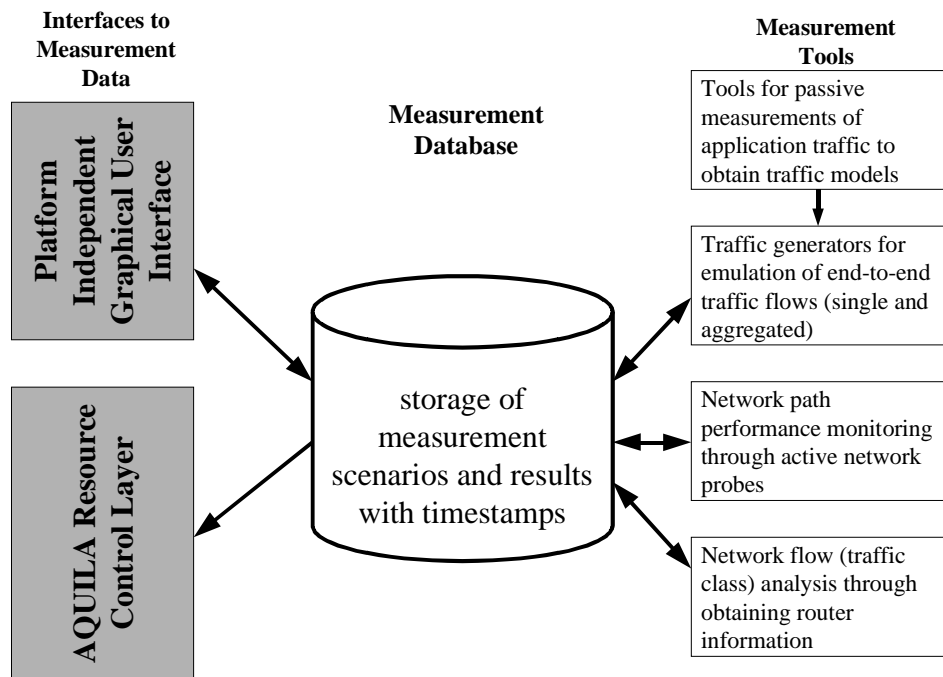


Figure 2-3: Overview of the measurement architecture and relationship to AQUILA RCL

The network engineer can control the measurement tools via a platform independent graphical user interfaces (GUIs). The GUIs are communicating with the measurement database in order to:

- specify measurement scenarios
- obtain results from the measurement scenarios
- obtain specific measurement database views for performance analysis.

The graphical user interfaces are similar but specialised for the different tasks:

- to specify measurement scenarios for emulated synthetic end-to-end flows and obtain the results
- to specify measurement scenarios for active network probing and monitor the results.

- to obtain router measurement statistics

The measurement database can be accessed by the AQUILA RCL components via a standardised interface (e.g. SQL, JDBC).

The distributed functional components of the measurement architecture and communication flows between them are illustrated in Figure 2-4.

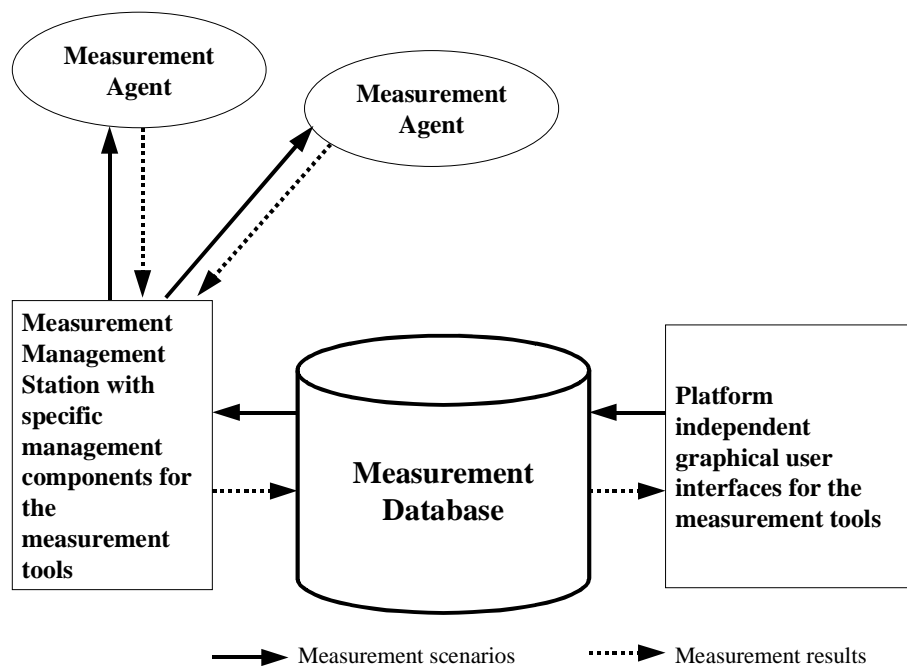


Figure 2-4: Components of the distributed measurement architecture

Distributed measurement agents are requested from management components to perform specific measurements. For each type of measurements, a specific measurement process at the measurement agents is responsible. The interaction of measurement agents and management components is done via the following transactions:

- a **request** of a management component to a measurement agent orders the execution of a specific measurement
- a **response** of a measurement agent to a management component includes the measured metrics (measurement result).

The management components are requesting the measurement database periodically (e.g. once per second) to get new scenarios and translate them into commands for measurement agents. After successful execution of the scenario, the management components collect the results of the measurement agents to the specific scenario and write them into the database.

The management components and the database can be distributed in the network architecture, but for performance reasons it is useful to locate them at one site. The measurement agents and the user interfaces are distributed dependent on their usage.

2.3 Measurement Methodology

[RFC2330] describes the framework for measurement methodology in the Internet and IP performance metrics.

The measurement methodology in AQUILA is based on active and passive measurement techniques. With active measurements the measurement tool injects defined measurement packets into the network, which deliver results that are evaluated online or afterwards (active probes). Passive measurements rely on data traffic related parameters without additional packets, e.g. packet loss collected by routers (passive probes). The results of passive measurements are gathered from different network elements. This produces overhead load on the network, that has to be considered.

Performance metrics are measured on different levels of the Internet architecture. The measurement of performance metrics uses injected test traffic as well as projections of metrics from lower level measurements.

To minimise the uncertainties/errors in measurements and to understand and document the sources of uncertainty/errors [RFC2330], the measurement environment is based on the use of dedicated machines for accurate measurements (e.g. GPS for one-way delay measurement) and a well known measurement environment (operating system, workstation, router configuration). Chapter 6 gives an overview about the hard- and software requirements of the measurement environment.

Performance analysis of end-to-end QoS of applications and/or synthetic traffic generation emulating applications are understood better when path performance measurements or router statistics are available. In AQUILA the measurement methodology is based on the integration of different techniques:

- synthetic flow generation for end-to-end performance measurements (emulating applications with synthetic flow models for single or aggregated flows),
- active network probing for path performance measurement and
- collection of QoS statistics from network elements by using passive measurements.

In the case of the AQUILA measurement architecture the network elements are mainly routers and the statistics obtained are dependent on the functions performed by the routers. Statistics obtained from the routers can be used in combination with active network probing and end-to-end measurements. They provide additional information for example on the bottlenecks within the network and can help to pinpoint the location in the core network where provision of network resources should be rethought (i.e. they can tell you *where, when and why* the

packets are dropped). At the edge of the network it is beneficial to see how the marking and the dropping of packets has been performed by the edge routers.

In addition, statistics can be used for approximation of edge to edge QoS if the topology of the network is known when the monitoring action is performed. It must be kept in mind that there is always a limit how often statistics can be obtained (and transmitted) without deteriorating considerably the performance of the router and its links. Statistics can be, in principal, retrieved from routers using several mechanisms:

- FTP if the accounting data is recorded to a file,
- Telnet/console/ssh using commands (vendor specific) which show the status or statistics of certain variables,
- standardised SNMP to retrieve the values of MIB objects.

The problem with SNMP is that usually features in the routers are implemented before MIBs even if they are private (vendor specific) and with MIBs defined by the IETF, standard MIBs, it tends to take even longer. In addition, the current SNMP protocol is not suitable for transmitting bulk traffic since there is no flow control (SNMP uses UDP) and Object identifiers take a lot of overhead in the transport packet. The file recording implementations are usually used for billing purposes or for long time scale monitoring. It is not the best choice for real time monitoring. The commands given through command line interface are very vendor specific, but at the moment it was seen as the only possibility for QoS related statistics retrieval in the AQUILA trial network since necessary MIBs are yet to be implemented in the routers.

The management of measurement scenarios and results is enabled by using a common measurement database for the different measurement methodologies. This allows to

- correlate active and passive measurements carried out at different layers (achieved by the use of timestamps)
- map performance results (measured end-to-end QoS) to reserved resources (i.e. the AQUILA network service specifications)
- predict end-to-end user QoS in a given scenario of user requirements and network resources
- repeat measurements (e.g. to compare between different devices and implementations)

To summarise, the AQUILA measurement methodology is derived from the needs of the AQUILA project considering the status of the work of the IETF and ITU and implements new components regarding to resource control and validation concepts.

2.4 Performance Metrics

The measurement methodology is based on the use of standard metrics (considering the IETF IPPM WG and ITU SG13) for performance measurement in the Internet and measurement statistics derived from them. The IPPM working group has defined a set of performance metrics, sampling techniques and associated statistics for transport-level or connectivity-level measurements. This chapter gives a short overview about considered metrics, chapter 3.2 describes the metrics in detail.

The IPPM framework document [RFC2330] discusses numerous issues around sampling techniques, clock accuracy, resolution and skew, wire time versus host time, error analysis, etc. The IPPM working group has defined and is defining several metrics and their associated statistics:

- connectivity metric [RFC2678]
- one-way delay metric [RFC2679]
- one-way packet loss metric [RFC2680]
- one-way loss pattern sample metrics [IPPM-LOSS-PATTERN], Draft
- round trip delay metric [RFC2681]
- instantaneous packet delay variation metric [IPPM-IPDV], Draft
- periodic stream metric [IPPM-NPMPS], Draft
- throughput metric [IPPM-BTC-FRAME][IPPM-TRENO-BTC], *expired Drafts* which are currently discussed for update.

Metrics on the application-level (e.g. for emulated traffic flows) are defined application dependent and are derived from lower-level measurements (also known as “projection” of a metric [RFC2330]). Table 2-1 describes this projection of the application-level metrics to the performance metrics for selected applications.

Application	Application-level metric	Performance metric
VoIP	call set-up time voice quality, e.g. ITU-T P.861 [ITU861]	round trip delay end-to-end packet loss, end-to-end delay and delay variation
File Transfer	delivery time	throughput
Streaming Audio	audio quality (compression, ...)	end-to-end packet loss, end-to-end delay and delay variation

Streaming Video	video quality (compression, frame rate, resolution, ...), e.g. MPQM [BDTL96]	end-to-end packet loss, end-to-end delay and delay variation
-----------------	--	--

Table 2-1: Projection of application end-to-end QoS to performance metrics

2.5 Measurement Components

For each of the three different measurement methodologies, one tool is responsible to implement the functionality. The three different tools, that implement the distributed measurement architecture are:

- the **Synthetic Flow Load Generators** for measurement of end-to-end QoS of synthetic single and aggregated flow generation,
- the **Active Network Probing Tool** for path performance monitoring,
- the **Router QoS Monitoring tool** collecting management information per traffic class for workload analysis on the routers.

The **measurement database** facilitates performance analysis and the correlation of the measurement results gathered from the different tools.

2.5.1 Synthetic Flow Generator

The intention of synthetic flow generators is to **emulate emerging Internet applications** like VoIP or Audio/Video Streaming according to the AQUILA applications. Therefore the load, which is produced by these applications is measured and analysed in a separate task. Traffic models can be derived from the measurements and implemented in the synthetic flow generator, which emulates the traffic afterwards. The end-to-end QoS of this emulated traffic is measured and evaluated.

The synthetic flow generator has to fulfil two major tasks:

- **Single flow generation** at the end-systems based on the emulation of user applications.
- **Aggregated flow generation** for the emulation of loads in the core and access networks.

The interaction of single and aggregate flow generation can be used for performance analysis of traffic control and resource reservation concepts.

The measurement scenario embedded into the AQUILA network architecture for synthetic end-to-end traffic generation is shown in Figure 2-5.

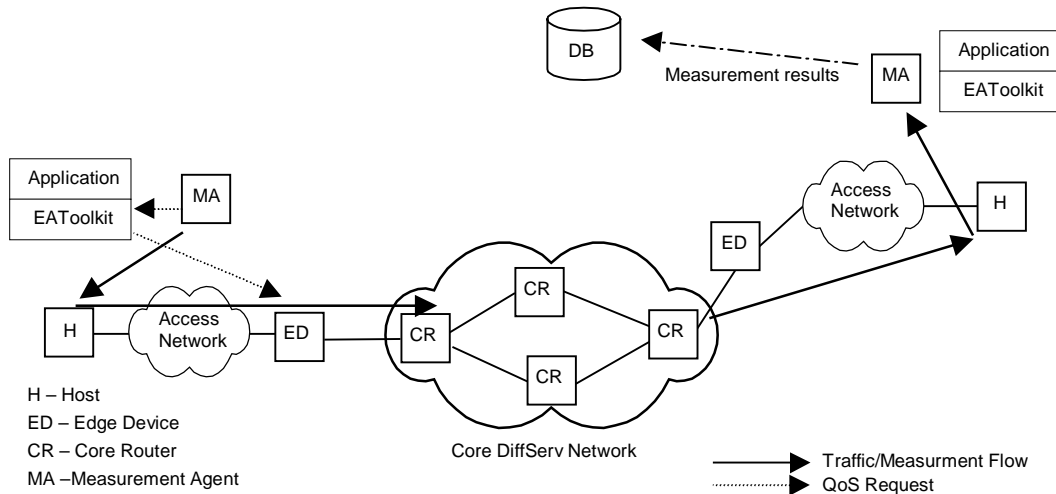


Figure 2-5: End-to-end synthetic flow generation

A sender MA generates a synthetic traffic flow and the corresponding receiver MA processes the received traffic packets with regard to performance metrics like packet delay, packet delay variation, packet loss rate. The performance metrics are stored in the measurement database. This allows offline post-processing of the achieved and comparison with the expected performance.

2.5.2 Active Network Probing Tool

Like the synthetic flow generator, the active network probing tool measures end-to-end QoS parameters between a defined pair of sender and receiver. The idea of these active measurements is to **inject small independent measurement packets into a network** to get “online” results of the achieved performance metrics like end-to-end packet delay, packet delay variation, packet loss rate etc. Therefore the active network probing tool contributes to network performance monitoring during the network operation.

The following measurements are fulfilled by this tool:

- Reachability/connectivity of particular paths in the network architecture
- Measurement of unidirectional properties using IP Performance Metrics (end-to-end delay and packet loss) for paths, subpaths and links (on the network layer).
- Monitoring of the path performance (e.g. one-way delay, packet loss)

The results can be used for example to analyse the path performance, to detect of changes in the path status or to study the asymmetry of end-to-end delay and packet loss.

These active measurements called “Probing” would be carried out by Measurement Agents (MA) of the DMA, whereby a single MA is associated with a host. Figure 2-6 shows the scenario of active network probing.

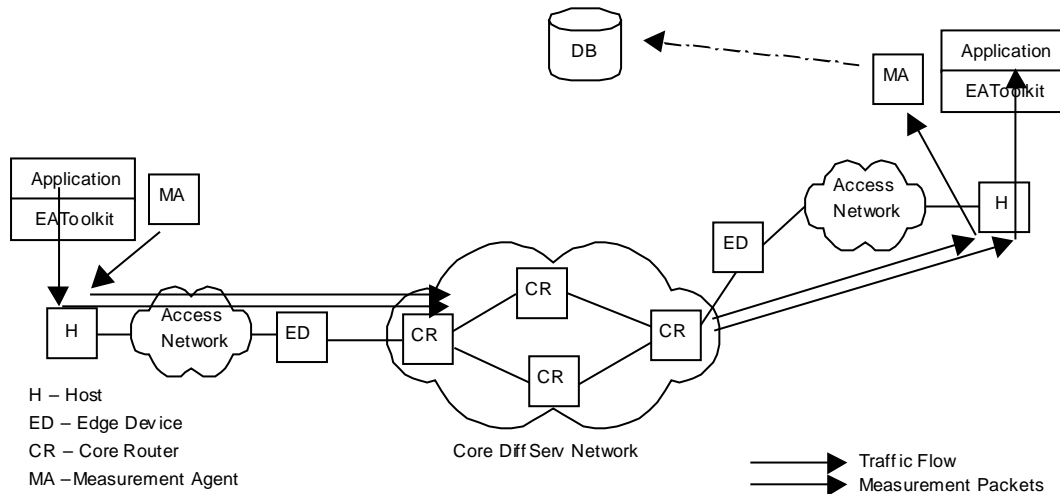


Figure 2-6: Active Network probing for obtaining the path performance

The results of active path performance monitoring could be used by the EAT to provide “on-line” performance feedback to the customer and the RCL (e.g. the actual delay), for instance in case the end-to-end delay of a path has changed through router failure or routing table change.

2.5.3 Router QoS Monitoring Tool

The router QoS monitoring tool is designed to obtain **statistics from the router about traffic classes** into which the flows in AQUILA are mapped. The correspondence between single and aggregated flows to traffic classes is given by the SLA or network specification.

The tool uses passive measurements. It works based on Cisco routers and measures the workloads of different traffic classes at the router. It performs following tasks in the measurement architecture:

- Measurement and analysis of router workload evoked by the metering/marketing processes at the router.
- Collection of performance statistics for each of the traffic classes on the routers like instantaneous and average queue size, number of packet losses, number of conforming/non-conforming packets, etc.
- Collection of path workload information by monitoring all or a subset of Core Routers (and Edge Devices) periodically and storing the information to the measurement database.

The degradation of the router performance caused by the monitoring process has to be investigated.

Figure 2-7 shows the scenario of management information collection by the means of passive measurements.

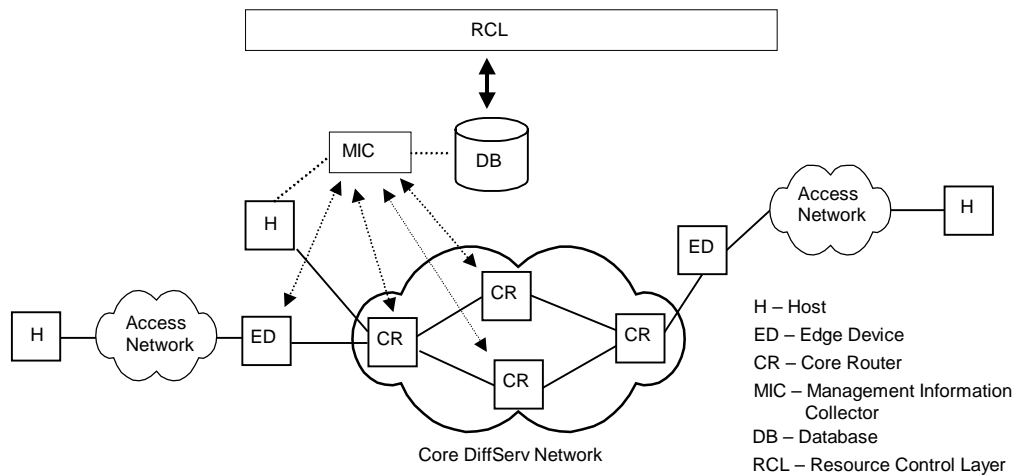


Figure 2-7: Router QoS Monitoring Tool

2.5.4 Measurement Database

The measurement tools in AQUILA are interacting using the measurement database. The correlation between the stored information (collected by the different tools of the DMA) is possible because all tools are storing timestamps for the measured metrics. Dependent on the tool, different kinds of information from the measurement database can be correlated:

- the synthetic flow load generator stores the network service specification per flow and the measured end-to-end QoS
- the active network probing tool collects path analysis information
- the router QoS monitoring tool gathers workload information of all or a subset of the core routers and/or edge devices periodically and stores it into the measurement database.

The stored information in the measurement database can be directly accessed by the AQUILA RCL. Appropriate visualisation of the measurement data gathered from the different tools can be used for a **performance analysis** of the network.

2.6 Benchmark and QoS Validation

The methodology to enhance end-to-end QoS will be studied on scenarios based on application traffic emulation and aggregated flows. A performance analysis is required, which implies the consideration of all measurement results and the selected AQUILA network service specification.

The analysis of the performance of the control algorithm is a stepwise approach, beginning with a simple systematically structured generation of flows and ending with a realistic sce-

nario with stochastic arrivals and QoS-demands of the flows. These scenarios are called **benchmark**.

In order to accomplish benchmark tests, which are defined by a fixed test scenario and can be used to evaluate the settings of several AQUILA QoS mechanisms, and furthermore to validate promised end-to-end QoS, an additional layer called Benchmark and Validation Layer is integrated into the AQUILA architecture.

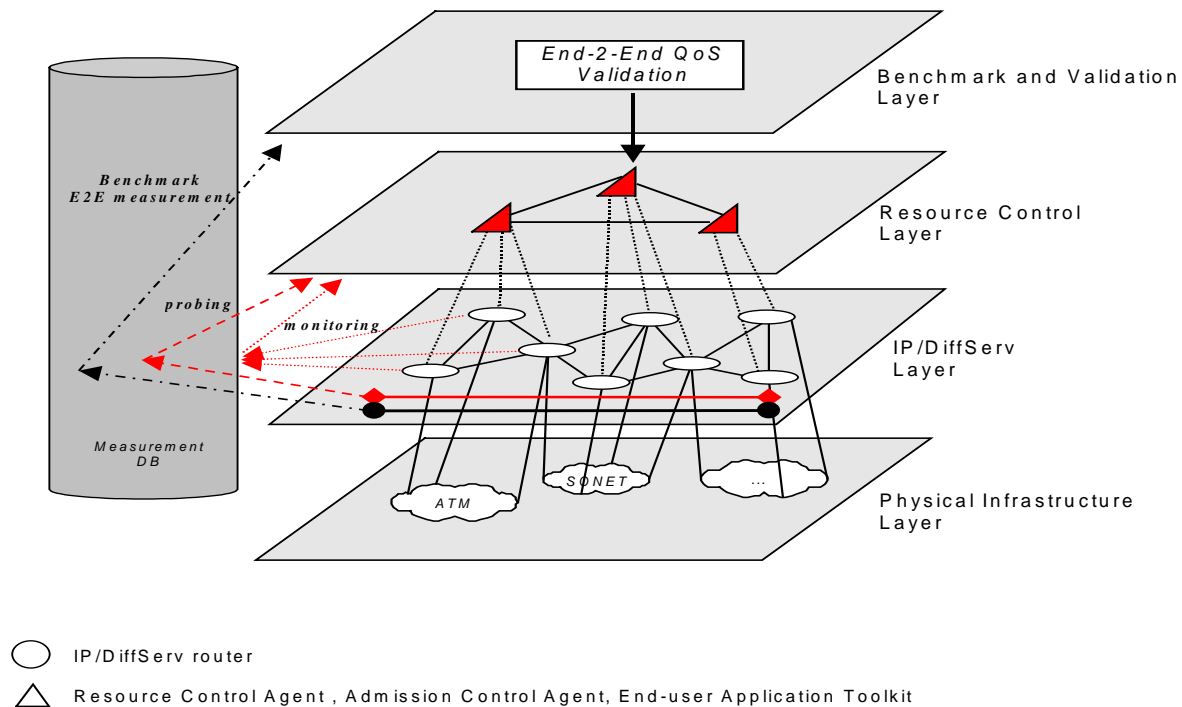


Figure 2-8: Benchmark and Validation Layer

The Benchmark and Validation Layer can evaluate the used RCL control algorithm and provide feedback that can be used to adjust this algorithm. In AQUILA there are different kinds of validation each of which require measurement scenarios and experiments. Measurement scenarios for the validation of the services in the AQUILA architecture are aimed at:

- Validation of SLA (mapping of required application QoS to network services) during network operation. If the required end-to-end QoS is not guaranteed, it could be possible to improve the end-to-end user QoS based on the measurement database. When there is no network fault on the measured path and the application flow is processed as expected by the routers, the SLA, i.e. the contract for the application can be changed and another network service can be assigned (i.e. changing reservation request or even the traffic class).
- Validation of required performance metrics for end-to-end emulated flow (application) generation. The focus is the mapping of the application type to a traffic class and a resource reservation on a given path in the network architecture. Unidirectional and bi-directional scenarios are possible.

- Validation of required performance metrics for aggregated flow generation where one kind of application is used. The focus is to study the performance of traffic classes when the aggregated flow is built by the same emulated application type. Here a lot of scenarios are possible, aggregation on the same path, aggregation on paths sharing same routers, study of unidirectional and bi-directional aggregation through the routers.
- Validation of used synthetic traffic models by collection of traffic parameters of typical applications like data rates, burst size, ...

Validation concerning AQUILA components:

- For the validation of AC (admission control) an incrementing number of traffic flows is generated until the ACA blocks the request of an additional traffic flow. The test result is the number of the defined flows which have been accepted by AC. This test could be accomplished for any combination of traffic flows and network service requests.
- Validation of traffic control like traffic conditioners, queue management [RFC 2309], scheduling).
- Resource Reservation validation (to validate provisioning).

Further validation is based on the accuracy of the measurement architecture including interaction between the synthetic flow generator, the active network probing tool and the router QoS monitoring tool with the measurement database.

Validation of the overall network performance, considering end-to-end provision of QoS and network utilisation will include the following characteristics

1. How many flows of a specific traffic class have been accepted?
2. Which QoS did the accepted flows experience?
 - How many of the accepted flows were QoS-satisfied?
 - How many of the accepted flows were not QoS-satisfied?
 - Level of QoS which was achieved for each single flow. For specification of QoS levels refer to [D1301].

3 State of the Art of Performance Measurement in the Internet

3.1 AQUILA DMA compared to Standardisation Efforts and Current Research

Much work has already occurred within the IETF which has a direct influence on the development of the performance measurement methodology in the Internet on different layers and environments. Internet measurement standardisation efforts are found in the frameworks of working groups which are dealing with recommendations and proposals for the Internet Community (RFCs, Internet-Drafts). The work of following IETF WGs is related to the DMA of AQUILA:

- **IPPM (IP Performance Metrics)** – definition of performance metrics, sampling techniques and associated statistics for measurement of performance in the Internet (see chapters 2.4 and 3.2). In addition to the specification of a framework and the definition of metrics, the IPPM WG also discusses application-level measurements e.g. in the Internet Draft “Network Performance Measurement for Periodic Streams” [IPPM-NPMPS].
- **BMWG (Benchmarking Methodology)** – benchmark specification and methodology to validate performance characteristics of various internetworking technologies and services built from this technologies.
- **RMONMIB (Remote Network Monitoring)** – developed an extensive, passive monitoring capability defined in [RFC1757] and [RFC2021]. Initially, the monitors collected statistics at the MAC layer, but now they have been extended to monitor at the application level and a definition of an overall performance monitoring architecture for applications in the Internet is defined in several Internet drafts: application typing and relevant metrics ([RMONMIB-APMMIB]), transaction level statistics collection and reporting ([RMONMIB-TPMMIB]) and the overall application performance monitoring system' capabilities [RMONMIB-APMCAPS]. For a synthetic flow generation it is important to consider the document “A Framework for Synthetic Sources for Performance Monitoring in Internet” which was recently proposed as Internet Draft [COLE-SSPM].
- **SNMPCONF (Configuration Management with SNMP)** – specification of effective methods for using the SNMP Framework to accomplish configuration management. Currently this working group is focused on the SNMP Configuration for policies [SNMPCONF-PM]. For synthetic probes there is the need to have configuration of a) a single probe, b) several probes, c) source and destination probes and d) intermediate probes. In addition, it may be necessary to configure any or all of these combinations simultaneously.

- **RTFM (Realtime Traffic Flow Measurement, inactive)** – was concerned with issues relating to traffic flow measurements, usage reporting for network traffic and Internet accounting. The work in this group was focused on passive measurements of user traffic and is therefore related to the monitoring work within the RMONMIB working group. Documents exist which describe the requirements and a traffic flow measurement architecture [RFC2063] and a traffic flow MIB [RFC2064]. A further document ([RFC2123]) describes the experiences in implementing and using the architecture and the MIB.
- **DISMAN (Distributed Management)** – defining a set of 'active' tools for remote management. Such tools are pingMIB, DNS Lookup MIB, tracerouteMIB, scriptsMIB and expressionMIB. Related to AQUILA are the pingMIB and tracerouteMIB which define an active probe capability, primarily for the remote determination of path and path connectivity [DISMAN-REMOPSMIB]. There are some performance related metrics collected from the pingMIB and one could conceivably use these measurements for the evaluation of a limited set of performance statistics.

In addition to the above mentioned IETF working groups a number of measurement research projects are working on tools and architectures that could be compared to AQUILA in specific points.

The main part of current research is focussed on **network performance monitoring** (especially path performance analysis) and **wide-area network measurement infrastructures** [Surveyor], [CAIDA], [NIMI], [Lab99], [MINC], [McBr00], [RIPE], [WAND]. For measurements and performance monitoring of application end-to-end QoS based on SLAs more proprietary tools are used. The SLAs are more controversial in the community since they depend on the application types. Compared with existing measurement infrastructures and tools for the Internet, the AQUILA DMA has a lot of flexibility, because it integrates different functionality and therefore can be used for network planning, analysis and operation.

The interpretation and analysis of collected data by tools like Surveyor [Surveyor] and AMP [McBr00] is done through post-analysis. These tools are primarily used for network research and planning.

Measurement architectures with real-time performance analysis capabilities can be used immediately by the network operator to maintain and enhance the network services. The Network Operation Center discussed in [GuHa00] is based on the concept of a common database management system (DBMS) with additional properties to automate performance analysis in real-time mode.

Some similarities of AQUILA to NIMI [NIMI], [PAM00] is the scheduling mechanism for the measurements to definite times and the scalability concept. Principle difference is the design goal. While NIMI is rather a command and control system for managing measurement tools, the AQUILA DMA is designed to be a scalable active and passive measurement architecture with different tools using a common measurement database.

The active network probing in AQUILA has similarities to the Surveyor measurement infrastructure discussed in [KaZe99]. In brief, surveyor is based on network monitoring using ac-

tive probes, provision of long-term performance data, measurement of unidirectional properties, the usage of dedicated machines for accurate measurements and the usage of database. The active network probing in AQUILA can use some Surveyor experience, for instance a study of asymmetry, even if the interaction of measurement tools in AQUILA has another architecture. Based on the obtained path metrics such as packet loss implications for end-to-end QoS [BSU98], [TTA00] are possible.

Another task of the network path measurement in AQUILA is the loss of reachability/connectivity. Modern techniques for reachability/connectivity are considered in [OMKN00] which proposes synthesising wide area fault information and locating the point of error by monitoring ICMP messages. The accuracy of measurement methodology for unidirectional delay and delay variation is addressed in [GDM98]. The assessment of performance metrics can be done using passive and active measurements [Horn00].

The CAIDA measurement environment [Claf99], [CAIDA], [CMP98] uses the skitter tool [Skitter] whose purpose is similar to the interaction between the active network probing and the router QoS monitoring in AQUILA. The difference is, that the skitter tool is not only used for gathering data of performance on specific paths, but also for dynamic reachability/connectivity analysis as well as discovering and depicting the global Internet topology. The network monitoring is combined in CAIDA with workload measurements and traffic flow matrices (tables which store how much traffic is flowing from a given source to a destination network). In AQUILA traffic measurement is extended with traffic class filtering.

AQUILA performance monitoring methodology using active and passive measurements is similar to the NLANR Network Analysis Infrastructure (NAI). NAI also combines active and passive measurements for the purpose of network path monitoring:

- active measurements with recording of metrics (Active Measurement Project [NLANR] and Internet Performance Measurement Protocol [IPMP98])
- passive measurements where data being transmitted over a network link is captured and analysed,
- control flow monitoring (SNMP and router data collection).

The performance of networks and networking protocols in local and wide-area networks can be studied focussing on routing characteristics, such as BGP Convergence and Internet failures (Internet Performance Measurement and Analysis [Lab99]).

The Multicast-based Inference of Network-internal Characteristics (MINC) project is developing and implementing methods to determine performance characteristics in the interior of a network from edge measurements. The basis of the method is that correlation between performance degradation on different paths can be used to infer the extent of performance degradation on their intersection. The principal innovation lies in the use of multicast probes exchanged between measurement servers; these exhibit such correlation inherently [MINC].

The router QoS monitoring tool provides measurements on queues and therefore measures the amount of packets from flows that are mapped to the traffic classes based on the Cisco routers. Emerging Cisco routers support the “NetFlow accounting” concept. With “NetFlow accounting a router counts packets and bytes for all packets with matching source and destination IP addresses, transport protocol and TCP/UDP port numbers (if applicable). Packets with matching parameters are defined as a “flow”. The collected information is not kept in the router for a long time, but exported to management workstation as soon as possible. The monitoring of traffic on the routers depends on the specific NetFlow Switching and Accounting mechanism implemented in the router. Earlier schemes for traffic flow measurements are based on NeTraMet [RFC20649], [RFC2123], [RFC2063] and IP accounting based on traffic matrices between observed pair of hosts and networks.

The AQUILA measurement architecture is also related to performance monitoring of real applications. A software toolkit that makes it easy for networked applications to report the performance they obtain as they communicate with distant Internet hosts and remember this information for later use is SPAND – Shared Passive Network Performance Discovery [Stem99], [SSK97], [SSK98], [SSK00]. As clients communicate with other hosts, they generate Performance Reports that summarise how their connection progressed. For example, clients may report the average throughput of a TCP connection or the download time for a particular web page. These Performance Reports are sent to a per-domain Performance Server who acts as a repository of all the Performance Reports for the clients in the domain. Later on, other clients contact the Performance Server with a Performance Query, asking about the performance to a distant network site. The Performance Server replies with a Performance Response which gives the average performance (and its variance), that was seen by clients in this domain. The key advantages of this approach are:

- Clients share the information they collect, which enables clients to learn from the past behaviour of other clients.
- Measurements are passive. Unlike other methods like Pathchar, SPAND relies only on traffic generated by applications and the results place a minimal load on the network.
- The measurements are application-specific and end-to-end. Other systems must assure that their network probes accurately reflect the way in which applications use the network. Because SPAND relies on application-to-application traffic, application-level performance is measured.

Resource monitoring for network-aware applications to obtain relevant information about their execution environment is addressed in the Remos project [Remos]. By using the measurement database in AQUILA, it is possible to develop a similar resource and performance monitoring interface.

Another research point is measurement accuracy. Standard hardware and operating systems have fundamental differences at measurements and have to be handled with care. Some sources of errors taken from experience [LMH00] and research [CDG00] are:

- Loss of packets can occur at end-system by the operating system which does not report the loss,
- Buffering at the Network Interface Cards (NIC) delays the earlier packets and shapes the traffic.
- Loss of packets at the NIC when the source application is generating the traffic at very high speed.

3.2 Performance Metrics in the Internet

In the near future QoS will be a critical differentiator among service providers. It's an important prerequisite for building standardised service offerings. QoS is a term, chiefly used to measure a specified set of performance attributes typically associated with a service. In the IP network environment, IP QoS refers to the performance of IP packets flowing through one or more networks. Given the current drive toward greater performance and reliability on the Internet, the ultimate aim of service providers is to deliver end-to-end, guaranteed (absolute, statistically) IP QoS to user traffic on IP networks – including data, video, voice,... The first step toward meeting this goal, a clear definition of QoS, is a critical prerequisite. With this aim in mind, QoS can be characterised by a small set of measurable parameters:

One-way delay – also known as *latency*; refers to the interval between transmitting and receiving packets between two reference points.

(One-way) **Delay variation** – also called *jitter* refers to the variation in time duration between two or more consecutively received packets taking the same route.

Packet loss – the rate at which packets are discarded during transfer through a network; packet loss typically results from congestion.

Throughput – the rate at which data is transmitted in a network.

The definitions of the above parameters and statements concerning the measurement methods can be found in several documents of the IETF (Internet Engineering Task Force) and the ITU (International Telecommunication Union):

IETF (IPPM-Working-Group):

- Framework for IP Performance Metrics (RFC 2330)
- Drafts for
 - One-way Delay
 - Packet Loss
 - Packet Delay Variation

- (Round Trip Delay)
- (Connectivity)

ITU-T (Study Group 13):

- Recommendations „IP Packet Transfer an Availability Performance Parameters” (I.380 renamed to Y.1540) and „IP Performance Objectives and Allocations” (I.381 renamed to Y.1541)
- Definitions for:
 - IP Packet Transfer Delay (IPTD)
 - IP Packet Delay Variation (IPDV)
 - IP Packet Loss Ratio (IPLR)
 - (IP Packet Error Ratio (IPER))
 - (Spurious IP Packet Rate)

Nobody wants multiple standards for the same thing. But are ITU and IPPM producing documents for the same thing? The following shows the overlap and the differences concerning the definitions of the parameters and the measurement methods.

3.2.1 General comparison

The ITU and IETF have common goals, but a different emphasis: the ITU-T strives to evaluate a service while the IETF measures the network.. The ITU has taken pains to use some IETF vocabulary, e.g., host, link. but there are other areas where there is only approximate equivalence (e.g., ITU network section versus an IPPM cloud; focus on corresponding events versus the fate of a single packet). Other terms have no correspondence. For example, I.380 has a notion of a IP packet transfer reference event; IPPM has the „wiretime” notion.

In general, differences between IETF and ITU-T derive from different backgrounds; the ITU-T documents historically have a telephony origin, while the authors of the IETF documents have a computer systems background. The purpose of ITU-T is to allow various providers to talk in a common language about performance, thus it does not concentrate on performance within a network. So ITU-T define required quantities precisely, while IETF discusses about implementation. ITU-T wants to evaluate service and wants to exclude unfair uses, IETF on the other hand wants to measure network quantities and avoid biased sampling.

Table 3-1 makes a comparison of the different emphasis as described in [IETF98] and [IETF99]:

IETF	ITU-T
<ul style="list-style-type: none"> • Focus: measure network • Normative. • Active measurements, without forbidding passive tools. • Avoid biased samples • Measurement methods is the IPPM area. • Relative QoS. • Precisely define required quantities • Characterise „absolute“ behaviour • Singleton measurement 	<ul style="list-style-type: none"> • Focus: evaluate service • Normative and Informative. • Focuses on passive measurement methods, but active measurements are not forbidden. • Exclude unfair uses • Abstract definitions. • Discuss implementation issues • Define grade of service • Overall measures (statistical)

Table 3-1: General comparison of IETF and ITU-T

3.2.2 One-way-Delay

IETF	ITU-T
<p>For a real number $(t_2 - t_1)$, the one-way-delay from a source to a destination at t_1 is $(t_2 - t_1)$ means that source sends sent the first bit of an IP packet to the destination at a time t_1 and that the destination received the last bit of that packet at time t_2.</p> <p>The one-way-delay from a source to a destination at t_1 is undefined (informally, infinite) means that a source sent the first bit of an IP packet to destination at time t_1 and that destination did not receive that packet. [RFC2679]</p>	<p>The one-way-delay is defined for all successful and errored packet outcomes across a basic section or an network section ensemble (NSE). The one-way-delay is the time, $(t_2 - t_1)$ between the occurrence of two corresponding IP packet reference events, ingress event (IPRE₁) at time t_1 and egress event (IPRE₂) at time t_2, where $(t_2 > t_1)$ and $(t_2 - t_1) \leq T_{\max}$. If the packet is fragmented within the NSE, t_2 is the time of the final corresponding egress event.</p> <p>T_{\max} is the maximum lifetime of an IP packet. [ITU380]</p>

Table 3-2: Comparison of one-way-delay

Comparison:

Both definitions describe the same methodology to measure the one-way-delay.

But there are a few differences: ITU-T generates events when an observed IP packet crosses the measurement points (ingress and egress) and compares the times of both events. IETF works with timestamps, which were placed in test IP packets. The advantage of the IETF method is the direct interpretation of the result at the destination.

ITU-T allows errored packets, IETF on the other hand defines that the last bit has to be received at the destination side, so only non-corrupted packets are allowed. IETF makes a more detailed and exact description how to measure the one-way-delay. It's easier to compare timestamps in the same packet (IETF) than to characterise corresponding events.

3.2.3 (One-way) Delay variation

IETF	ITU-T
<p>The instantaneous packet delay variation (ipdv) is:</p> <p>For a pair of IP packets:</p> <p>The IP packet delay variation of a pair of IP packets, that are transmitted from the measurement point MP₁ to the measurement point MP₂, is the difference between the one-way-delay measured for the second packet and the one-way-delay measured for the first packet of the pair.</p> <p>For a stream of packets:</p> <p>The Instantaneous Packet Delay Variation of an IP packet, inside a stream of packets, going from the measurement point MP₁ to the measurement point MP₂, is the difference of the one-way-delay of that packet and the one-way-delay of the preceding packet in the stream. [IPPM-IPDV]</p>	<p>The End-to-end 2-point IP packet delay variation is defined based on the observations of corresponding IP packet arrivals at ingress and egress measurement points (MP) (e.g. MP₁, MP₂). These observations characterise the variability in the pattern of IP packet arrival reference events at the egress MP with reference to the pattern of corresponding reference events at the ingress MP.</p> <p>The 2-point packet delay variation (v_k) for an IP packet k between a source and a destination is the difference between the absolute IP packet transfer delay (x_k) of the packet and a defined reference IP packet transfer delay, $d_{1,2}$, between those same MPs: $v_k = x_k - d_{1,2}$. The reference IP packet transfer delay, $d_{1,2}$, between the source and the destination is the absolute IP packet transfer delay experienced by the first IP packet between those two MPs. [ITU380]</p>

Table 3-3: Comparison of packet delay variation

Comparison:

The ITU-T definitions are based on delay variation as defined for ATM cells. ITU-T works with the same methodology as described in the chapter one-way-packet-delay. Events are gen-

erated at the ingress and the egress, when packets pass the MPs. So there is a problem how to define the reference delay time of the packets. One the one hand you could take the delay of the first packet of a sequence, on the other hand you could take the average one-way-packet-delay as the reference time. In general the ITU-T definition appears in the appendix of the papers. These appendices are informative, not normative and can still be changed. IETF uses active test packets and can make an exact definition for the ipdv. There are also differences how statistics were made. ITU-T considers two methods: interval-based and quantile-based. An example alternate definition of IP delay variation is given in the appendix II of ITU-T Y.1541, which is the new name the of ITU-T I.381. IPDV may be defined as the maximum one-way-delay minus the minimum one-way-delay during a given short measurement interval. Several values of IPDV are measured over a large time interval, comprising of several short measurement intervals.

3.2.4 One-way-Packet-Loss

IETF	ITU-T
<p>The value of a IP-one-way-packet-loss is either a zero (signifying successful transmission of the packet) or a one (signifying loss).</p> <p>The IP-One-way-Packet-Loss from a source to a destination at t_1 is 0 means that the source sent the first bit of a IP packet to the destination at time t_1 and that the destination received that packet.</p> <p>The IP-One-way-Packet-Loss from a source to a destination at t_1 is 1 means that the source sent the first bit of a IP packet to the destination at time t_1 and that the destination did not receive that packet.</p> <p>Several statistics can be made over a sample of packets. [RFC2680]</p>	<p>IP-one-way-packet-loss ratio is the ratio of total lost IP packets outcomes to total transmitted packets in population of interest. A lost packet occurs when a single IP packet reference event at a ingress MP results in a misdirected outcome or when some of all the contents of that packet do not result in any IP reference event at any egress MP within the time T_{max}.</p> <p>T_{max} is the maximum lifetime of an IP packet. [ITU380]</p>

Table 3-4: Comparison of packet loss

Comparison:

There is also a difference in the methodology of measurement. ITU-T separates in successful, errored, lost and spurious packets, so loss can't be compared with loss of IETF, where a packet either receives the destination or not.

3.2.5 Other parameters

3.2.5.1 Round-trip delay

IETF	ITU-T
<p>For a real number $(t_2 - t_1)$, the round-trip-delay from a source to a destination at t_1 is $(t_2 - t_1)$ means that the source sent the first bit of a IP-packet to destination at t_1, that destination received that packet, then immediately sent a IP-packet back to the source, and that the source received the last bit of that packet at time t_2.</p> <p>The round-trip-delay from a source to a destination at t_1 is undefined (informally, infinite) means that the source sent the first bit of a IP packet to the destination at time t_1 and that (either the destination did not receive the packet, the destination did not send a IP packet in response, or) the source did not receive that response packet.</p> <p>The round-trip-delay between source and destination at t_1 means either the round-trip-Delay from source to destination at t_1 or the round-trip-Delay from destination to source at t_1. [RFC2681]</p>	<p>No definition</p>

Table 3-5: Comparison of round-trip-delay

Comparison:

ITU-T describes no methodology to measure round-trip-delay.

3.2.5.2 Service availability

ITU-T defines a parameter „service availability“ which has no counterpart in the IETF. IETF defines on the other hand metrics for measuring „connectivity“. [ITU380][RFC2498]

4 Measurement Database

A central part of the measurements made in AQUILA is the measurement database, which stores the measurement scenarios on the one hand and the according results on the other hand.

The measurement database has been derived and extended from the already existing database used within CM Toolset (see chapter 5.1).

4.1 Design Goals

The major goal is to design a database, which is capable of storing measurement scenarios and the according measurement results.

As already stated, there are three different methods of measurement in AQUILA namely router QoS monitoring, active network probing and synthetic flow generators. The database should be able to store measurement results retrieved by these methods.

The main reasons for having a database are:

- to compound performance information of different measurement tools and to find out dependencies between them.
- to integrate different flow measurement concepts (aggregated flow, single flow)
- test scenarios are repeatable: a comparison between the same tests in different conditions is possible.
- consistent storage of measurement results: results are stored integrative, so that results are comparable. Furthermore each results set is assigned to a specific test scenario, which is stored in the database, too.
- accessibility of measurement data: There is a defined interface to the measurement data, so that the results can be post-processed (e.g. statistical analyses) with arbitrary tools. Also correlation analyses between different measurement results can be done.

There are mainly two client applications accessing the database:

- graphical user interface: The user gets access to the database via a web-based GUI, which allows him to specify measurement scenarios and view measurement results.
- management station: A management station is responsible for the distribution of tests to the measurement agents and therefore has to look up the database for new entered test scenarios. Another task of the management station is to retrieve the measurement results from the measurement agents and to write this results to the according tests into the database.

Another design goal was to keep the database flexible for further enhancements (e.g. definition of complex load generators, extensions in monitoring information, changing of the DBMS, ...).

4.2 DBMS evaluation

The different test-tools developed and used within WP2.3 are extensively based on the use of a suitable database. Controlling the tools as well as storing measurement results is performed via this database.

Therefore, it is important to know the demands that are made on this database. Most of these requirements can be ignored in small networks or test environments. But if the measurement tools shall be designed that they can also be employed in real production networks with several hundred nodes the requirements may affect the choice between different database implementations and platforms (e.g. server type, operating system).

This chapter list some requirements. The main focus will be on small test environments, as will be the case for the AQUILA trials, but some hints will also be given for the evaluation of possible databases for large production networks.

4.2.1 Specific company requirements

There may be strong reasons for a certain platform and implementation that are based on specific company requirements. In general it can be assumed that there are already databases in use to perform network operation (accounting, user data, ...). The logical consequence would be to use the same database platform also for the new measurement control and results.

This requirement has certainly be taken into account for production networks. In the case of AQUILA the partners within WP2.3 have experience with MySQL.

4.2.2 System design and software tools

The overall system design also includes the necessary post-processing of the raw measurement data like aggregation of measurement data and different statistical evaluations. The post-processing can be performed with or without involvement of the database. For example data aggregation can be done by the measurement equipment itself or by post-processing tools running either on the database server or on another host/platform. This influences the data volume to be stored in the database and also the required performance of the database. The following questions arise:

- Which tasks are performed where (database server, web server, clients, measurement tools)?
- Which programming tools are available?
- Is simple scripting ok? For database management? For post-processing?

- And concerning the database design itself: How much programming effort is necessary?
- Can a general SQL interface be assumed?

Within AQUILA a simple and efficient solution should be realised. A first aggregation of measurement data, if necessary, should be performed by the measurement equipment, e.g. calculation of one result that is to be stored in the database out of 10 raw samples. Further data aggregation should be done by separate tools, e.g. calculation of an hourly / daily / weekly (...) average value.

A script-based implementation of the database management and of the visualisation of the measurement results is possible. In order to be flexible with regard to further extensions and porting the system for employment in large networks an SQL interface should be used.

4.2.3 Scalability

One issue concerning scalability is the possible size of the database. For the use in large production networks some hundred nodes have to be supported. Even assuming that not all nodes have to be equipped with measurement clients their number will still be in the magnitude of 100. Further assuming that the measurement clients will be fully meshed (each client has a connection to all other clients), 4 traffic classes and 1 sample / connection / traffic class / 1 minute, there will be 40000 samples / minute that have to be stored in the database. This results in about 58 Mio. samples / day. The questions now are:

- Is data aggregation performed and where?
- How many days shall the (raw ?) samples be kept before aggregation and / or deletion?
- Will the measurement traces be stored within the database or in separate files? Or will the traces be deleted after generating aggregation data?

As a consequence of the above calculation, an aggregation of the raw samples to (for example) 15-minute-values seems necessary, if large networks shall be monitored. A further aggregation of these 15-minute-values to hourly / daily average values can be done after some days in order to keep the database at a reasonable size. As the trial networks within AQUILA will not have so many nodes, data aggregation is not that necessary.

Another issue concerning scalability is the number of clients that have to be supported by the database:

- number of clients that write into the database
- number of clients that read the database

Very likely this will not be a problem, because according to the current system design all reading from and writing to the database is done by the master station(s) and by the web

server(s). It can be assumed, that there will only be a limited number of master stations and web servers (if not just one).

4.2.4 Performance

The necessary performance of the database implementation is another important requirement. This depends on several factors:

- A multi-user environment can be assumed. As was mentioned in chapter 4.2.3, the number of clients will be limited though, i.e. master station(s) and web browser(s).
- The results stored in the database will / can be used for network monitoring by both operator and customers. This can lead to a large number of requests to prepare and process statistic data.
- Requests to the database should be answered within an appropriate time. What will be the requirements concerning monitoring tasks by operators and customers?
- The foreseeable size of the database (see chapter 4.2.3) should not lead to performance degradation.
- Depending on the implementation the post-processing may have impact on the performance of the database.

Again, the requirements with respect to database performance should be quite easily met in the case of AQUILA. the number of simultaneous requests will be limited and the database will remain rather small.

For employment in large networks a hardware and software platform that can be expanded if need arises is preferable.

4.2.5 Quality

One topic that has to be taken into account could be termed „quality“. This includes several subtopics:

- **Availability:** Considering that a network operator relies on the results of the measurement tool for network monitoring the availability of the database is a very important issue.
- **Stability:** The amount of data in the database and frequent transactions (writing / reading) must not lead to instability.
- **Backup:** There must be some kind of backup mechanism covering the control data and results. This can be accomplished by separate tools or can be an inherent feature of the database platform.

All above requirements can surely be met sufficiently by standard equipment in the case of AQUILA. For large production networks these issues have to be analysed in more detail.

4.2.6 Security

Security is of course a very important issue. Definitely, the view an operator has on the measurement results may be more extensive than what is available to the customers. This must be achieved by the database itself or by the access methods via master station and web server.

4.2.7 Costs

The overall costs are another factor that have to be considered. They can be split up into different parts:

- Hardware costs: mainly server platform.
- Software costs: system software (operating system), database software and support tools (programming tools, ...).
- Design and implementation of the database and the necessary tools (post-processing, ...).
- Possibly also porting an early prototype implemented for test purposes to a production platform.

The main requirement for AQUILA with respect to costs is: keep it cheap, as far as hardware and software investments are concerned. This leads to standard components like PCs as hardware platform, MySQL as database software, apache as web-server.

4.2.8 Functional evaluation

A very good functional comparison of database implementations can be found on the MySQL or TCX web page [Crash-me]. The charts show that MySQL is a very good SQL-implementation with all necessary functions and data types.

4.2.9 Conclusion

This chapter has shown some requirements and aspects that should be considered in the selection process of a specific database platform. Further work is needed for employment of the measurement system in large production networks.

4.3 Database Design

This chapter describes the developed database model and its entities with their main attributes.

4.3.1 Database Model

The database model was designed using PowerDesigner 6 from Sybase. The conceptual database model (or Entity-Relationship Diagram, Figure 4-1) shows the 13 defined entities and the relations between them. From the conceptual model the physical model is derived, which gives information about how the tables are structured in the database and where the foreign keys from neighbored tables are stored to keep the relations. The physical database model is different according to the used DBMS. For the 1st trial the MySQL DBMS [MYSQL] will be used. From the physical database model the „create table“-directives are constructed.

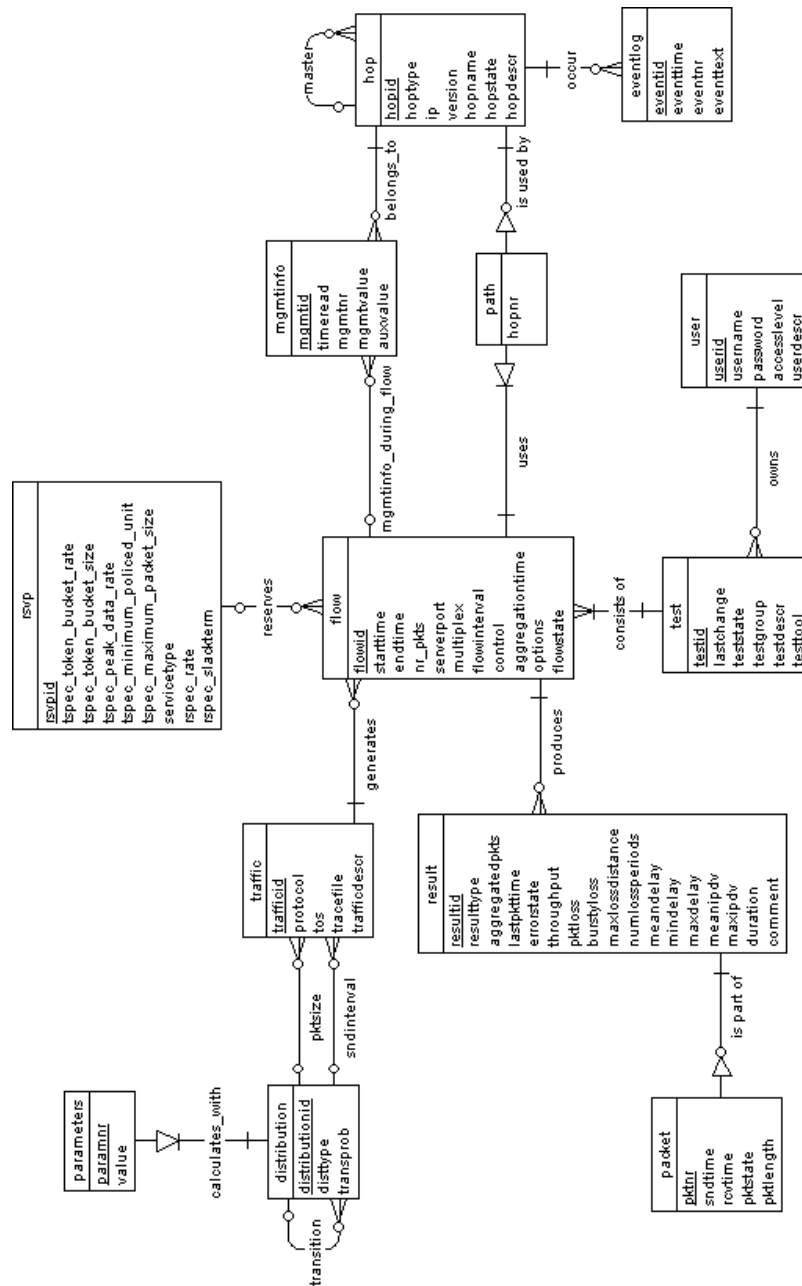


Figure 4-1: Conceptual database model (ER diagram)

4.3.2 Description of Database Entities

4.3.2.1 Entity „user”

Only registered users should be able to execute measurements. Therefore each measurement (or test) must be assigned to a specific user. Users identify themselves by providing a user-name and a password. Additionally not all users have the same access rights to the database (e.g. adding further traffic parameters or users) therefore an access-level for the user has to be specified.

4.3.2.2 Entity „test”

This entity has the general information about a test. A test can include one or more flows with different traffic parameters. It is also possible to join more tests together to a group of tests. A test state defines the actual condition of the test.

4.3.2.3 Entity „flow”

The „flow” is the central part of the measurement database. One flow is a part of a test and is related to its results, the traffic parameters (load generator) to be used, the hops which are passed (at least a sender and a receiver) and maybe to an RSVP reservation.

4.3.2.4 Entity „traffic”

The entity traffic specifies the behaviour of the traffic, which is generated for the flows and the corresponding traffic class. It is linked with the tables „flow” and „distribution” (twice, once for the distribution of the packet size and once for the distribution of the sending interval). Traffic can be generated either with the definition of a distribution for the send interval between the packets and a distribution of the packet size or with the specification of a tracefile. The tracefile can be used, if no model for a traffic exists, but some packet traces are available e.g. from „tcpdump”. Different definitions of traffic sets can be seen as different load generators that are used for the flows.

4.3.2.5 Entity „distribution”

This entity stores information about the distributions of the packet size and the sending interval. The database model is kept flexible to specify arbitrary different types of distributions. The type of the distribution (e.g. constant, exponential, n-burst) has to be specified („dist-type”). To be able to define transition probabilities between different distributions, a relation to this entity itself has been created. Because of their different number, the distributions’ parameters are stored in a separate table.

4.3.2.6 Entity „parameter”

This table stores information about the parameters of the distributions of the packet size and the sending interval. It provides a high flexibility for distributions with more parameters (e.g. „n-burst” needs 5 parameters). According to the types of distribution a number of parameters has to be defined.

4.3.2.7 Entity „path”

This entity links the flows with the hops. Therefore it contains information about the path of the flow (using an increasing hop-number for the hops in the flow). A flow must be defined with at least 2 hops (sender, receiver). To get the path for a measurement, it is possible to make a „traceroute” before and after the test. If the routes before and after the test are identically, the measurement route will be the same with a high probability. It has to be noticed, that routes can differ for each class. Therefore an adapted „traceroute” has to be used.

4.3.2.8 Entity „hop”

This entity has general information about each hop. A hop could be a host, a router or a measurement management station (the hoptype gives information about that). Hops are identified with their IP address, so a router can have one entry for each of its IP addresses.

4.3.2.9 Entity „mgmtinfo”

This entity is intended to store selected management information of the hops. Each of the relevant management information is stored with a timestamp and/or with a relation to a specific flow. The type of the information is identified by the mgmtnr.

4.3.2.10 Entity „result”

The result entity stores the measurement results of each flow. Depending on the options of the flow, more or less results for each flow are stored. Note that each flow could have more than one result, if it was a multiplexed flow or result aggregation is done. Results from one flow can be aggregated after a specified time to save storage requirements and calculation power. Result aggregation has to be done by the measurement management station, the flow description contains information about the aggregation interval.

4.3.2.11 Entity „packet”

Within the packet entity sending and receiving times and a state of every single measurement packet is stored. The timestamps in this table have a precision of microseconds (μ s) and are stored as UNIX-timestamps.

4.3.2.12 Entity „eventlog”

This entity stores events, which occur at the hops (e.g. GPS ready, GPS not available, ...). The events are stored with a timestamp for a logging functionality.

4.3.2.13 Entity „rsvp”

This entity is used to be able to define an RSVP reservation for the flows. The properties of the RSVP reservation can be reused by several flows. If an RSVP reservation should be performed, the receiving measurement agent has to initiate the reservation. This functionality will not be used by the first trial measurement utilities.

5 Measurement Tools

There are different measurement tools available for different types of measurement scenarios. Some important measurement tools are already described in [D1101]. The tools which are developed by the project partners for AQUILA are explained in this chapter.

5.1 Synthetic Flow Generation Tool (CM Toolset)

The CM Toolset (Communication Measurement Toolset) is a prototype of a tool for testing and measuring the quality of end-to-end TCP/IP communication channels. It is a development of a joint project between Telekom Austria, Polytechnical University Salzburg and Salzburg Research Forschungsgmbh, which was supported by the Austrian research fund „FFF”.

The CM Toolset offers measurement features for an evaluation of IP networks. It provides a management platform to handle the measurement scenarios and to measure the IP performance parameters for different transport protocols including their multiplexing. The different parameters (e.g. packet size, parameters of the load generators) can be adjusted. The results and the parameters of the measurements are stored in a database. The impact of different network configurations, protocol parameters and QoS parameters can be analysed. CM Toolset is intended to generate traffic, that emulates real applications, therefore different traffic models are implemented.

Tests in CM Toolset are controlled via a web-based user interface, so that the client requirement is reduced to an Internet browser like Netscape. The user can specify measurement scenarios, which consists mainly of the sending and the receiving host (specified through their IP addresses) and the load, that should be generated the hosts (specified by different parameters like protocol, packet size, packet inter-arrival time, etc).

The measurement results are throughput, packet loss, one-way delay [HPH00] and the instantaneous packet delay variation.

This chapter describes the current implementation of the CM Toolset. Because of the integration of the CM Toolset architecture into the distributed measurement architecture of AQUILA, it has to be mentioned, that the user interfaces of the different measurement tools will be merged and therefore changes will occur. The current implementation of the CM Toolset is extended by features, that will be provided for the AQUILA project.

5.1.1 CM Toolset Architecture

As shown in Figure 5-1, CM Toolset consists of 3 main components. The distributed measurement agents, which are responsible for the execution of the tests, the CM Server, which hosts the database server, the web server and the cmcaller and a client with a browser, which is used for the test management.

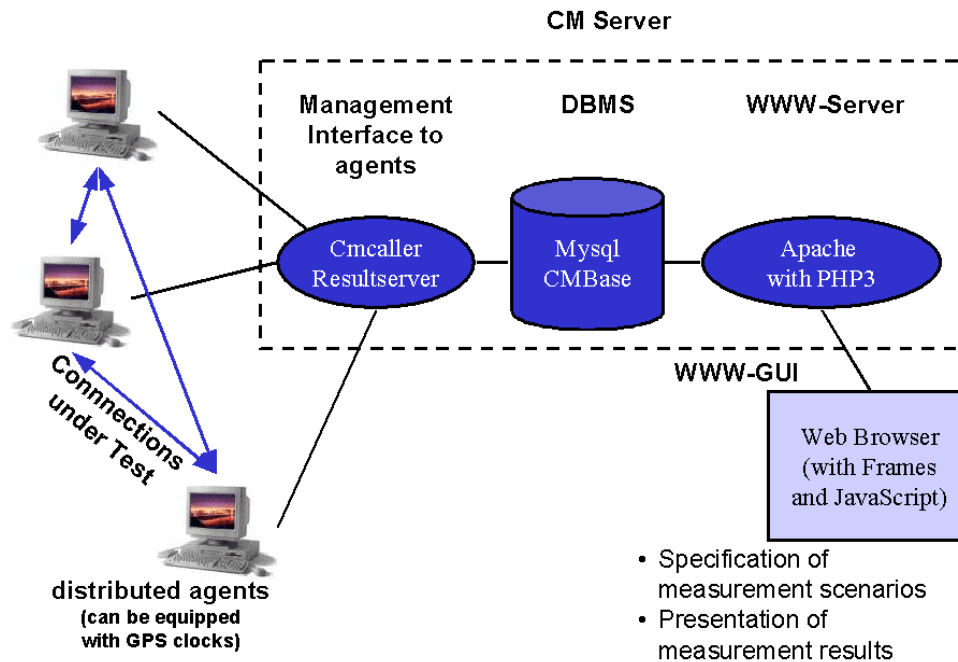


Figure 5-1: CM Toolset Architecture

5.1.1.1 Distributed Agents

The distributed measurement agents are processes running on hosts which are used for the end-to-end measurements. They can be started either as sender or as receiver. All information for execution of tests are passed to them from the cmcaller. The agents can be equipped with GPS-clocks to synchronise their internal clocks for enabling one-way delay measurements.

To use a hosts as a distributed measurement agents, a daemon program, called „cmdaemon“ has to be started. Also the so-called „generator“ must be installed on these machines. The agents are currently designed for the operating system Linux, root access rights are NOT required for running these agents. A bootdisk with the necessary binaries is also available.

5.1.1.2 CM Server

The communication measurement server builds the heart of CM Toolset. It runs three subsystems: the database management system (CM Base), the cmcaller and the web server for the client access. It is also possible to run these tasks on different machines, but it is suggested to integrate them in only one machine.

CM Base

Currently the MySQL database (www.mysql.org) is used for the management of scenario information and measurement data. It is designed for storing the test scenarios to be executed and the associated measurement results. The database has been redesigned for the AQUILA project to have a common platform also for the monitoring as well as for the probing tool.

CM Caller / Resultserver

The CM caller builds the Interface between the daemons and the database. It „calls“ the database, whether new entries of tests exists, which have to be executed. If a test with a current start-up time exists, the CM caller reads out the connection table and sends the parameters of the test to daemons of the given hosts.

When a measurement process has ended, the receiving station sends the measurement results back to the CM Server. The process, which is listening for results is the resultserver, which stores the measurement results into the database according to the measurement scenario.

Web Server

The web server, which is the standard apache server with the PHP3 module, provides the database access for the users. The data, which is manipulated by the users via a web browser (using standard HTML forms), is written into the database from the web server using the script language PHP3 (www.php.net).

5.1.1.3 User Interface (WWW-GUI)

As user interface, every web browser which supports frames and JavaScript can be used (e.g. Internet Explorer, Netscape Navigator). We have tested the system on Linux and Windows using Netscape 4.5 resp. 4.7. The web browser is used for editing new tests and for viewing the results of the completed tests.

As shown in Figure 5-2, the GUI is divided into 2 parts. The menu which is placed on the top of the browsers window and the main window, where the forms for data input are displayed. Additionally to these two frames in the browser there are some extra pop-up windows which contains additional information.

Menu

The menu is divided into „user functions“ and „administrator function“. User can list/add tests, list hosts and list specified parameter sets. Administrators are additionally able to add/edit/delete the users, the hosts and the test parameters. The functions are described in detail below.

Main Window

The main window is used for displaying the relevant data and forms which are needed to use CM Toolset. The content of the main window is dependent on the chosen menu item.

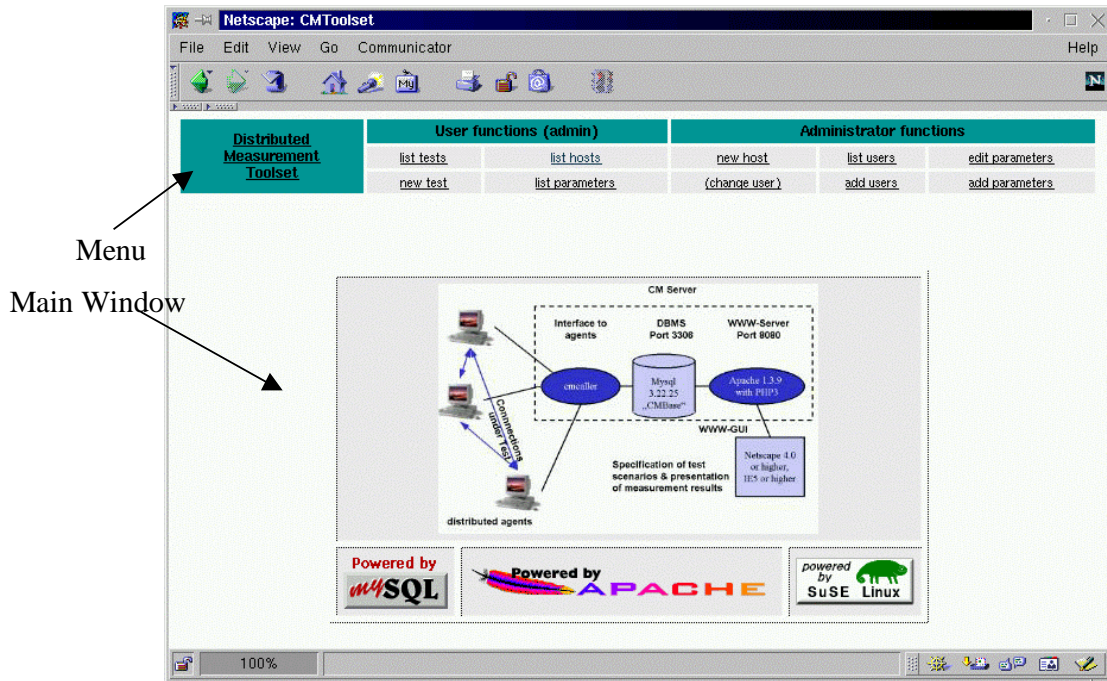


Figure 5-2: CM Toolset menu and titlepage

Pop-up Windows

There are two kinds of additional popup windows within CM Toolset. One displays the detail view of parameter sets, the other one displays the scenario image (Figure 5-3, left window: detailed parameter view, right window: scenario image). These windows are displayed only on demand and can be closed at any time.

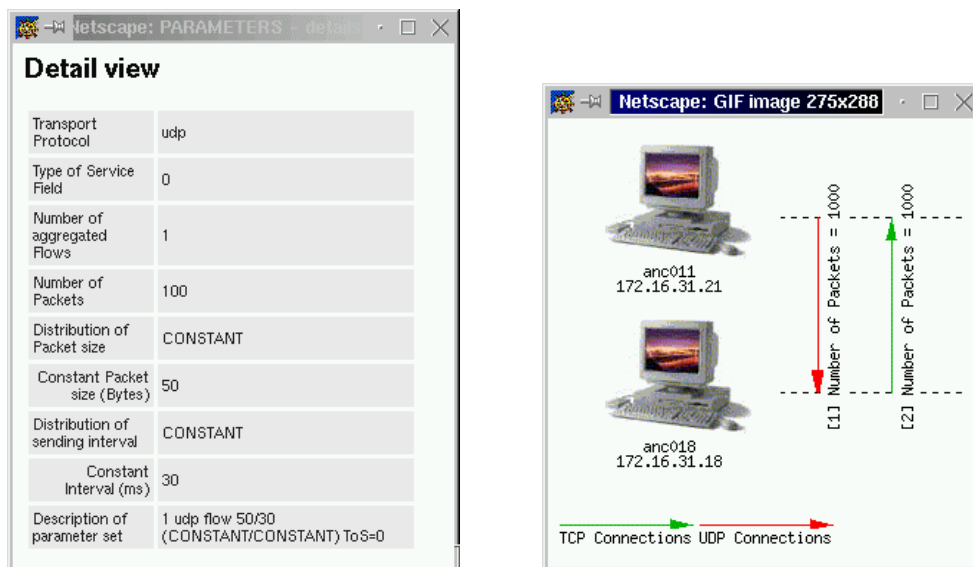


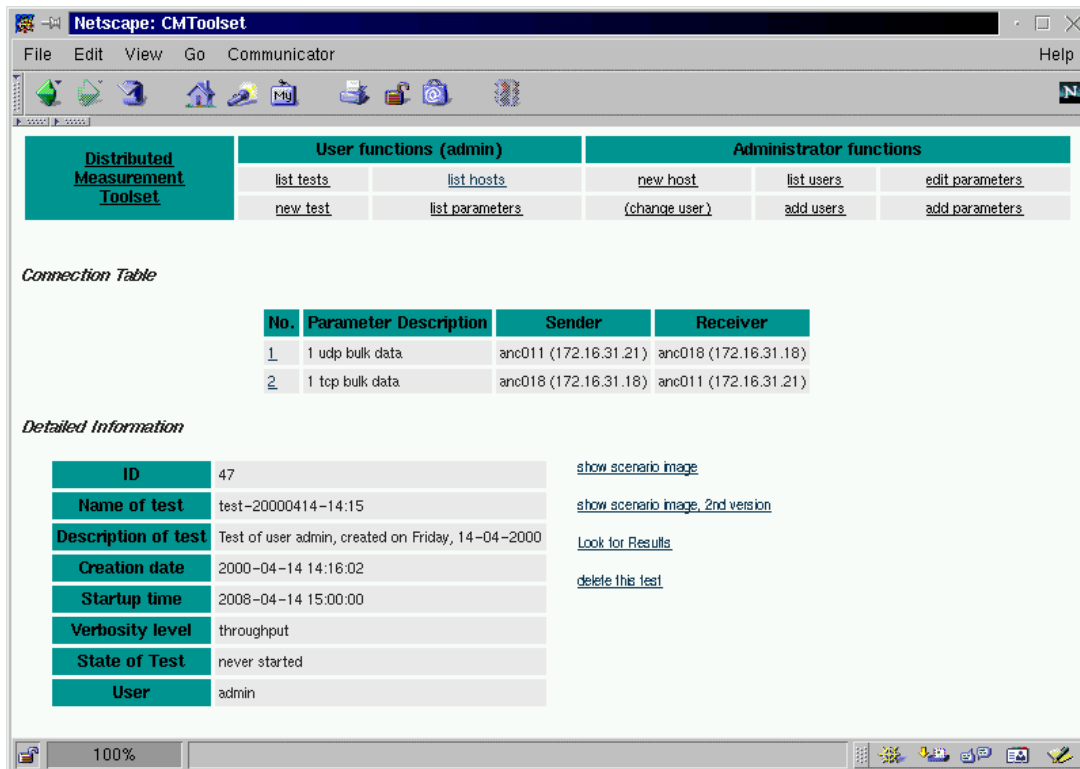
Figure 5-3: Examples of pop-up windows

5.1.2 CM Toolset Functions

5.1.2.1 List Tests

This function displays an overview of the specified tests from the current user. „Name” (the specified name of the test), „Test Description” (a short description of the test), „Start-up time” (the date/time, when the test will be/was started), „State” (which represents the numbers of ready connections of the test) are listed. From this view it is possible to repeat the test immediately (which is a useful function, if someone wants to execute the same test under different network conditions, there is no need for re-editing this test), and to delete the test with its results. A click on the test number leads the user to a detailed test view.

The detail view of the test consists of the connection table, which lists the several connections (sender, receiver and used parameters) of the test. The connection number is a link to the detailed view of the parameters (see also Figure 5-3). Further it shows some other information about this test (description, verbosity level,...). On this window there are links to the scenario image and the measurement results. See Figure 5-4 for an example.



The screenshot shows a Netscape browser window titled "Netscape: CMToolset". The browser's menu bar includes "File", "Edit", "View", "Go", "Communicator", and "Help". The toolbar contains various icons for file operations and navigation. The main content area is divided into several sections:

- Distributed Measurement Toolset**: A sidebar menu.
- User functions (admin)**: A table of links including "list tests", "new_test", "list hosts", and "list parameters".
- Administrator functions**: A table of links including "new host", "(change user)", "list users", "add users", "edit parameters", and "add parameters".
- Connection Table**: A table listing test connections with columns for "No.", "Parameter Description", "Sender", and "Receiver".
- Detailed Information**: A table showing test details such as ID, Name of test, Description of test, Creation date, Startup time, Verbosity level, State of Test, and User. It also includes links for "show scenario image", "show scenario image, 2nd version", "Look for Results", and "delete this test".

Figure 5-4: Detailed test view

From the detailed test view, a link to the test results is available. Results are provided, when the test (or at least one connection of the test) is ended and the receiving measurement agent has sent back its results via the “resultserver” to the database. The detailed test view provides a link to the results of the test. If that link is followed, the results from the test are displayed.

On the top the general results of the test are listed (state and errorstate). Also a field for typing a comment to these results is provided.

Below the general results, a list of the ready connections with the results throughput, packet loss and duration for each connection is displayed. Depending on the specified verbosity level (see below), there are also two links for each ready connection. One for a graphical presentation of one-way delay and the instantaneous packet delay variation of each packet and one for retrieving the raw data (a file, which contains packet number, sending and receiving time of each packet). Note that the absolute one-way delay is only meaningful, when the sender's and the receiver's clocks were synchronised.

The following graphs (examples shown in Figure 5-5) of the results are currently implemented:

- overall result graph: Displays the one-way delay per packet of all connections in this test, where the x-axis represents the time.
- one-way delay over time: Displays the one-way delay per packet of a single connection, where the x-axis represents the time.
- one-way delay over packets: Displays the one-way delay per packet of a single connection, where the x-axis represents the packet number. Lost packets are indicated with a cross.
- instantaneous packet delay variation ("jitter") over packets: Displays the IPDV values (according to the IETF recommendation) of a single connection, where the x-axis represents the packet number. Lost packets are indicated with a cross.

All of the graphs can be obtained either as png-graphic (for a browser) or as postscript files.

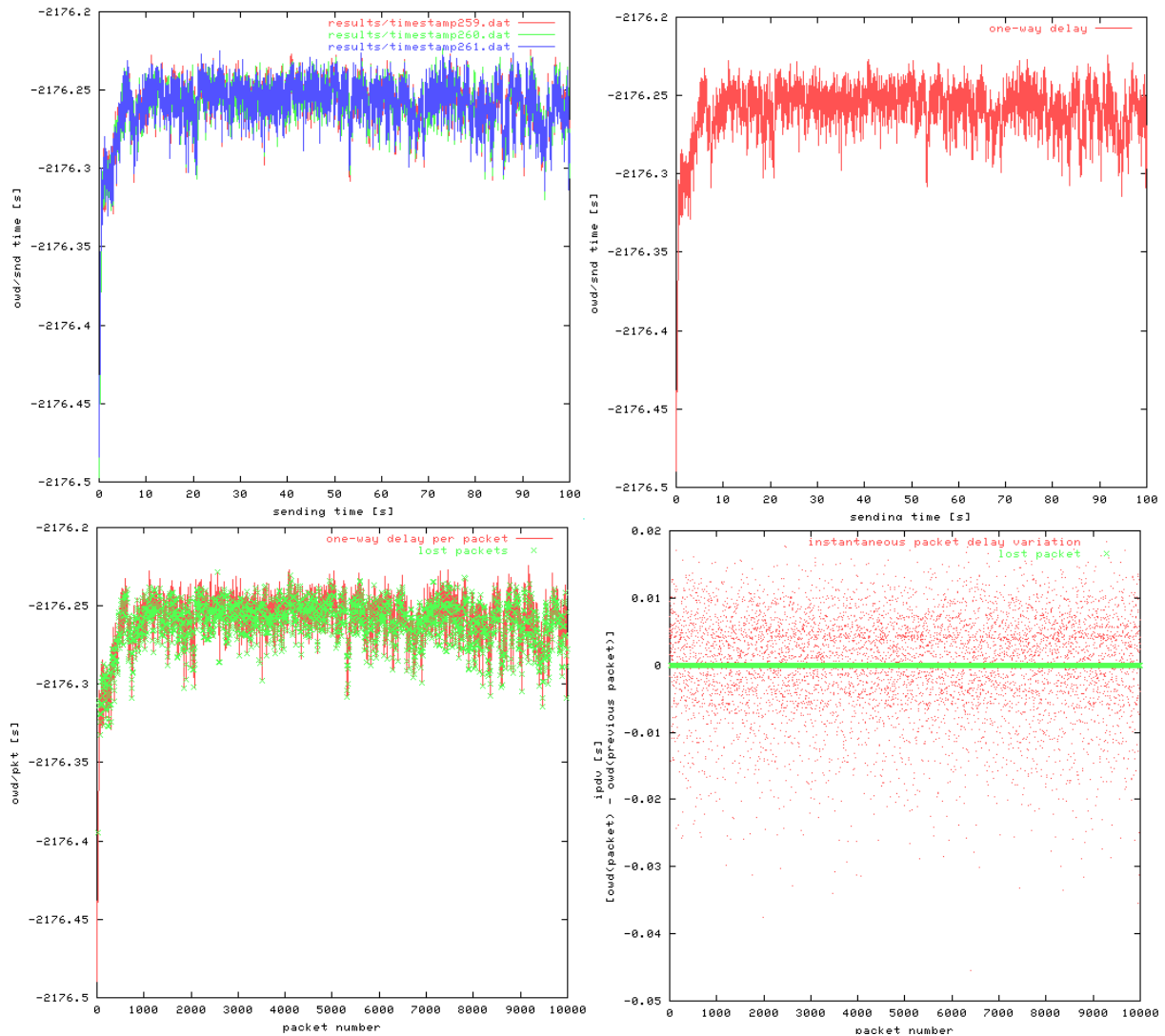


Figure 5-5 CM Toolset Resultgraphs

5.1.2.2 Add new Tests

To add a new test, 2 forms have to be filled out. The first form needs general information about the new test. This is:

- **Testname:** This can be filled with the desired name for the test. A default name is given.
- **Description:** This can be filled with a description for the test. A default description containing the name of the active user and the current date is given.
- **Start-up time:** The time, when the test shall be executed. There are two possibilities to fill this point: It is possible to specify, in how many minutes the test shall be started (0 for an immediate start after completion of the test specification), or to give an absolute time,

when the test shall be started (this applies, when the above field stays empty). Default selection is the current day, rounded up to the next full hour.

- Periodicity: This specifies, how often the test should be started and which time intervals is between the tests. (Note: using this option causes multiple entries of tests in the database. Each of these tests is further processed separately.)
- Verbosity Level: This will specify the amount of result data, that will be stored on the database server. Possible gradations are:
 - Throughput only
 - Throughput and packet loss
 - Throughput, packet loss and the timestamps of all packets

„Timestamp“ means, that for each transmitted packet the sending and receiving time is stored in the results. From this information the graphs for one-way delay and delay variation are derived.

If the first form is filled out, the button „Add connections“ leads to the next form, where the several connections of the tests can be specified. A connection between a pair of hosts is equal to UDP/TCP-flows. The first column displays the number of the connection. In the second column the parameter set to be used for the flow can be chosen. The columns three and four specify the sender and the receiver for this flow. If a connection is ready specified, the „Add“ button is used to add this connection to the test. The connections can also be deleted afterwards in case of a mistake in specification. A connection without using a parameter set will not be added. Completion of the specification is done with the „ok“ button. The test is now written to the database and will be started, if the system time of the CM Server reaches the start-up time.

If no connection is specified and the window is left with any of the menu items, the test will get the state „to be deleted“. It won't be executed and does not produce results. These tests should be deleted on the „list tests“ window.

5.1.2.3 List/Add new Hosts

Hosts (or sometimes interfaces of hosts) are identified by their IP address. The hostname which is used in the CM Toolset is not resolved via DNS.

Hosts can be used by any user, but added, edited (changing name or description) and deleted only by administrators.

The „list hosts“ item in the menu returns a list of all hosts which are stored in the database. Administrators are enabled to edit and delete hosts using the two right-hand columns.

For adding a new host, the administrator is asked for the IP address, the name and a short description for the host.

Because of data integrity, hosts can only be deleted, if they are not used in any of the tests.

5.1.2.4 List Parameters

All users can view a list of parameters (menu item „list parameters“). This displays a table with a complete list of all available parameter sets. The parameter sets can be seen as the different load generators, that are used. The list shows a number, the protocol, which is used, the parameter description and the number of aggregated flows used for this parameter set. With a click on the number a new window will come up with a detailed view of the set where all parameters are displayed sequentially.

5.1.2.5 Add Parameters

New parameter sets can be added by the administrators by using the menu item „add parameters“. The first of two forms which have to be filled out contains the following fields:

- Protocol: This specifies the used protocol for this parameter set. Currently you can choose between TCP and UDP.
- Aggregated Flows: If a test should contain some aggregated flows with the same behaviour it is useful to use a parameter set which generates more than one flow within one connection. „Aggregated Flows“ specifies the number of the same flows which are started between sender and receiver.
- Distribution of packet size: This specifies the distribution of the size of the sent packets (e.g. a constant packet size or an exponential distributed packet size).
- Distribution of sending interval: Same as distribution of packet size, but for the time interval between two sent packets.
- Number of Packets/Total Length/Duration: To specify the whole amount of data to be transmitted. Currently only the first option (number of packets) is implemented.
- Type of Service: With this option you can specify, how the type of service byte in the IP header should be set. The types of service are set regarding [RFC1349].

5.1.2.6 Edit Parameters

The menu item „edit parameters“ leads the administrator to a table of all available parameter sets, which can be used for tests. The right-hand two columns of the table are linked with the functions „delete“ and „edit and duplicate“. Like hosts, parameter sets can only be deleted, if they are not active in any test.

The „edit and duplicate” function is to derive a new parameter set from an already existing one. Choosing this function it is possible to change the values „ToS Field“, „No. of aggregated flows“, the length of the transmission, the packet size, the time between the packets are sent („interval“) and the description.

5.1.2.7 List/Add new Users

Administrators are enabled to list/add/edit and delete users of the CM Toolset. Users are specified by a unique username, a password and a short description about the user.

5.2 Active Network Probing Tool

The T-Nova toolset allows the measurement of the IP Performance parameters „One Way Delay (OWD)”, „Packet Loss (PL)” and „Delay Variation (DV)” based on the correspondent IETF RFC’s. Main goal of the development was an (as much as possible) accurate toolset that can be used for measurement and online monitoring of the IP performance parameters mentioned above. Therefore the use of an real-time operating system instead of an Windows NT solution was preferred. For the measurement of the exact OWD each measurement client is equipped with a GPS receiver (GPS = Global Positioning System). This allows the generation of timestamps with an accuracy of $\frac{1}{2} \mu\text{s}$. The overall toolset structure shows Figure 5-6.

Key features of the T-Nova toolset:

- Distributed measurement system
- Components: Measurement clients, master station, WWW-Server and WWW-Browser
- Measurement and online visualisation of OWD, DV and PL
- QoS support (DiffServ)
- Accuracy: $\sim 100 \mu\text{s}$
- WWW-based offline display

Because of its accuracy the toolset is applicable for testing of network component functions (e.g. the DiffServ queuing mechanism), (online) monitoring the actual parameter values of IP flows and also for detecting performance asymmetric in IP networks.

5.2.1 Architecture of the T-Nova toolset

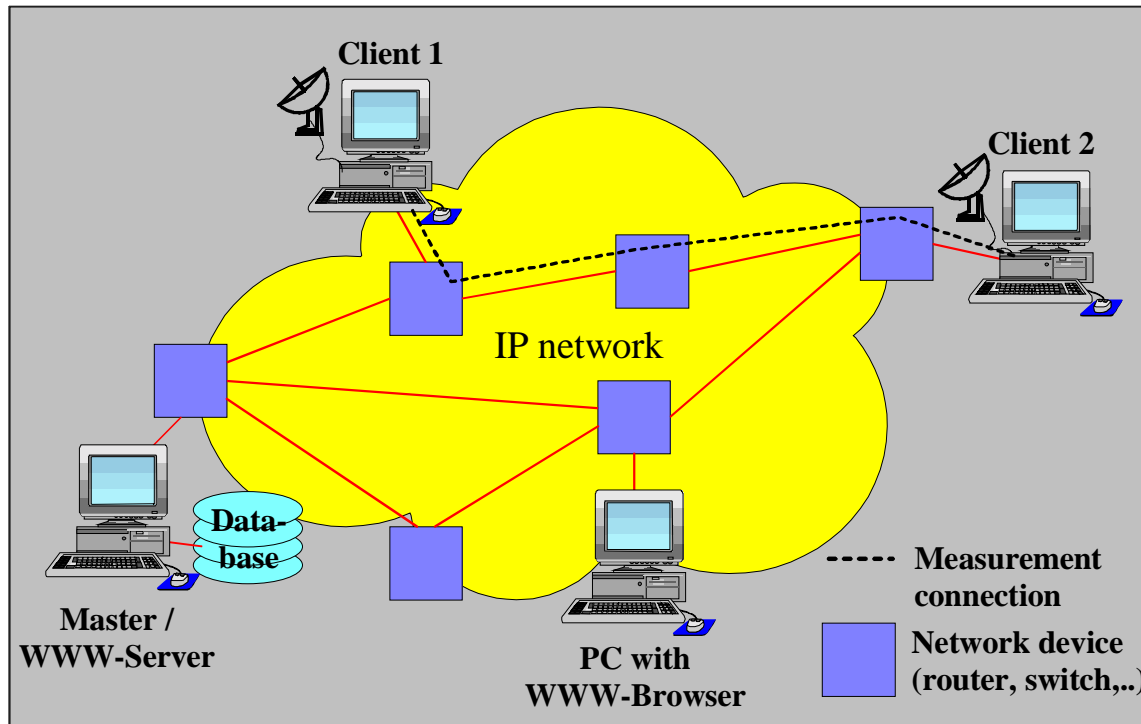


Figure 5-6: Architecture of the T-Nova toolset

The overall toolset architecture consists of a distributed system of measurement clients, a master station and a WWW-Server/-Browser (see Figure 5-6). Between the clients UDP packets (UDP - User Datagram Protocol) are sent over unidirectional measurement „connections”. Among other things this packets contain timestamps and sequence numbers. Exact measurement of the OWD requires very accurate timestamps. Therefore a GPS receiver (GPS - Global Positioning System) is installed in every client. The GPS receiver allows the generation of timestamps with a precision of $\pm \frac{1}{2} \mu\text{s}$. Measurement results are stored in a database (CSV-Format) which is accessible by the WWW-Server. A CGI-Program on the WWW-Server processes the data and generates the OWD-, PL- and DV-graphs. These graphs can be displayed by a WWW-Browser.

5.2.1.1 Client

As mentioned above the measurement is done by the clients. To achieve exact measurement results the clients consist of VME-bus based hardware (32 bit Motorola PowerPC) with the real-time operating system LynxOS. Manufacturer of the GPS-cards is Truetime (USA). The measurement software has no graphical user interface (GUI), it is completely controlled by the master station. The measurement results are only temporary buffered on the client, because accessing the hard disk influences the measurement accuracy due to the additional processor load. The clients are working passive, that means establishing the measurement connections,

transferring the results and the status to the master station are initiated by commands sent to the clients from the master station. The client tasks are:

- generating and sending of measurement packets
- receiving and buffering of measurement results
- transferring results to the master station
- error check (GPS receiver, CRC for measurement packets)
- sending status information to the master

5.2.1.2 Master

The master is responsible for the complete control of the toolset. It sends the connection parameter values to the clients, starts/stops the measurement connections, retrieves the results and status information from the clients, stores the results in the database and handles the on-line display. The software on the master station runs on every standard PC (≥ 200 MHz) with Windows 98/NT. The GUI of the program shows Figure 5-7. The GUI enables the user to configure, start and stop the system.

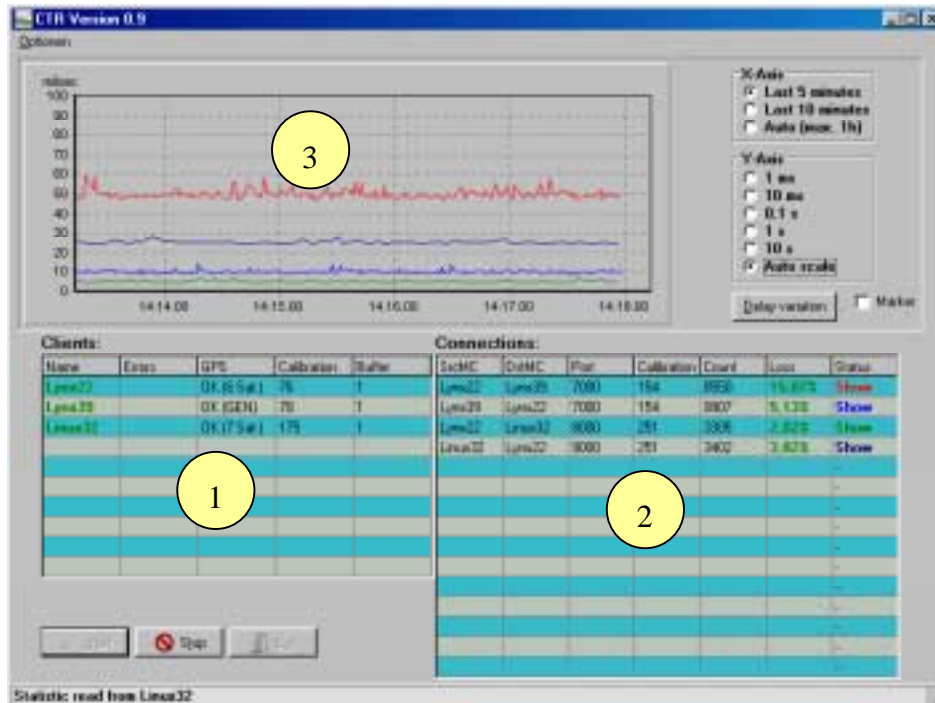


Figure 5-7: GUI of the master program

The GUI is divided in 3 different regions:

- Client region (1)
- Connection region (2)
- Display region (3)

After the start of the master program it is in the so called „configuration mode”. This mode allows the configuration of the measurement system:

- add clients to the system, delete existing clients
- add / change / delete measurement connections
- start / stop system

When the system starts (Start button) the mode changes to „Running” and the following tasks are executed:

1. Connections to the clients are established
2. Client configuration (measurement connections, GPS mode)
3. Retrieving data from the clients (status information, measurement results)
4. Storing results
5. Displaying results (OWD, DV, PL)

The tasks 3-5 are executed periodically from now on (Interval adjustable).

5.2.1.2.1 Client region (1)

Configuration mode

With the following dialog a new client can be added to the system. An meaningful name (e.g. location of the client) can be assigned to the IP address. The checkbox selects whether the GPS card receives its signal from an antenna or an NTP [RFC 1305] server via the IRIG-B protocol.

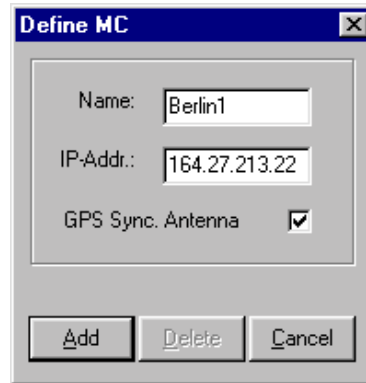


Figure 5-8: Client dialog

If the system is in the running mode the client region shows the status of the clients. The following information is shown:

- *Errorcodes* (if an error occurs in one of the clients)
- Mode and Status of the GPS receiver (No Error, No antenna connected, Receiver not Synchronised)
- Number of measurement packets in the receive buffer of the clients

5.2.1.2.2 Connection region (2)

Configuration mode

With the following dialog new measurement connections can be defined, existing altered or deleted.

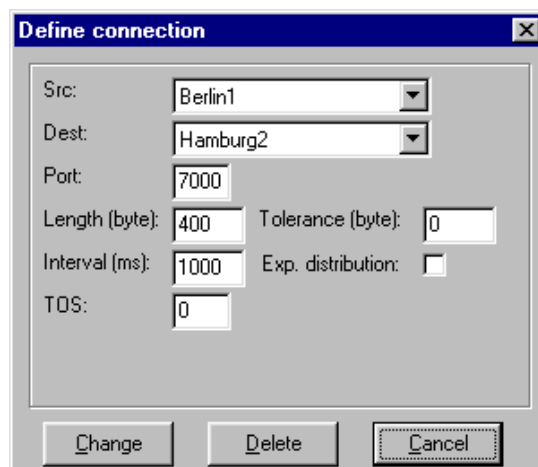


Figure 5-9: Connection dialog

Following parameters of a measurement connection can be adjusted:

Parameter name	Meaning
Src	Name of the source client (where the packets are generated)
Dest	Name of the destination client (receiver of the packets)
Port	Port number of the receiving socket
Length	Packet length
Tolerance	Tolerance of the packet length. The packet length varies between [Packet length – Tolerance] and [Packet length]. The variation is uniform distributed.
Interval	Inter-arrival-time of the sender (inter-transmittal-time of outgoing packets)
Exp. distribution	If the box is checked then the packet inter-arrival-time will be exponential otherwise constant distributed..
TOS	To allow testing of CoS (DiffServ) functions (e.g. priority queuing in CoS capable network elements) the value of the TOS field has to be adjustable. This value allows to assign the measurement packets to a specific traffic class.

Table 5-1: Connection parameters

Running mode

If the system is running the connection region shows some of the connection parameter values (Source, destination and destination port number) and the actual status of the measurement connections.

Column	Meaning
1	Name of the source client (where the packets are generated)
2	Name of the destination client (receiver of the packets)
3	Port number of the receiving socket
4	Number of packets the destination clients received up to now.
5	Actual packet loss (in %)
6	Connection status („Down“, „Up“, Show“, see below)

Table 5-2: Information in the connection region

Down-Status

The master is running in configuration mode. The connections are defined (Parameter values are adjusted) but the connection is not active. No measurement packets are sent from source- to the destination-clients.

Up-Status

The master is in the running mode. The connection parameter values are sent to the clients, the connections are active. Measurement packets are sent from source- to the destination-clients.

Show-Status

A mouse click on the sixth column of a connection changes the status from „Up” to „Show”. The measurement results of the connections marked in that way are displayed in the display region. The colour of the word „Show” will be the same as the colour of appropriate graph in the display.

5.2.1.2.3 Display region (3)

In this region the actual measurement results of the marked connections (connections with the status „Show”, see above) are displayed (online display). The button „Delay variation”/“One-way delay” toggles the display between showing the OWD (see Figure 5-10) and DV (see Figure 5-11).

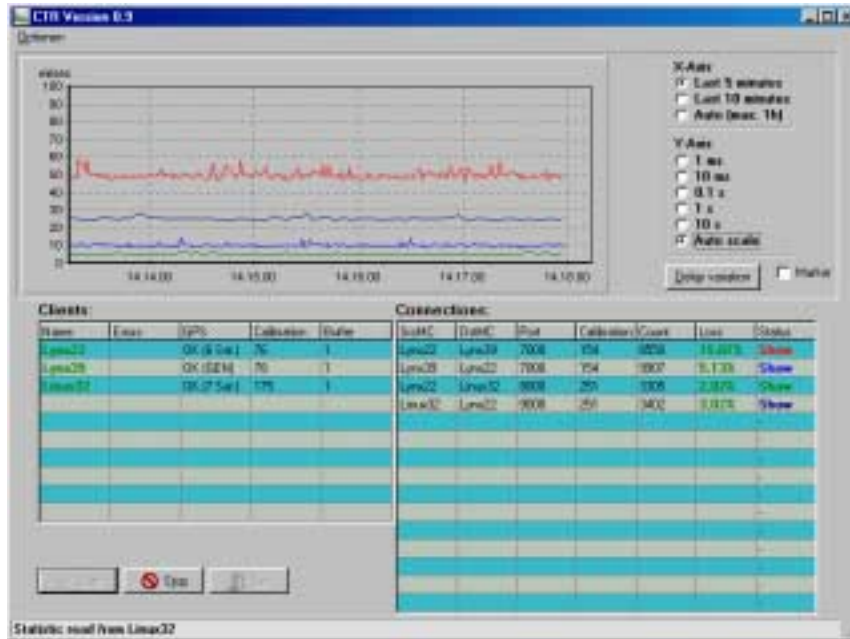


Figure 5-10: Online display of the „One Way Delay ”

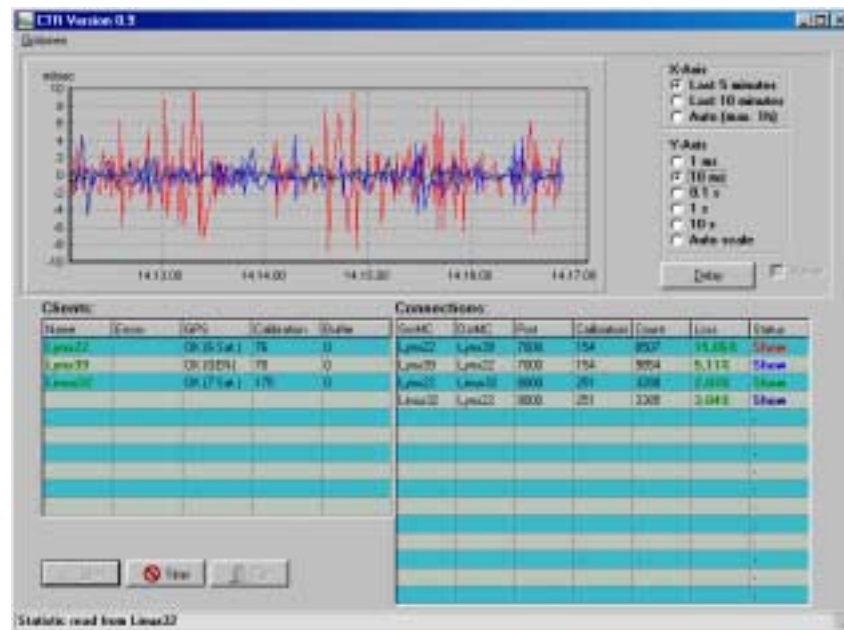


Figure 5-11: Online display of the „Delay Variation”

5.2.1.3 Offline display

Displaying the measurements data offline is achieved by a WWW-Browser, which has access to the WWW-Server mentioned above. After selecting a connection and a graph type (One Way Delay, Delay Variation or Packet Loss) a CGI-Program on the WWW-Server processes

the data and generates the appropriate OWD-, PL- or DV-graph. The graph is displayed by the WWW-Browser (e.g. see Figure 5-12).

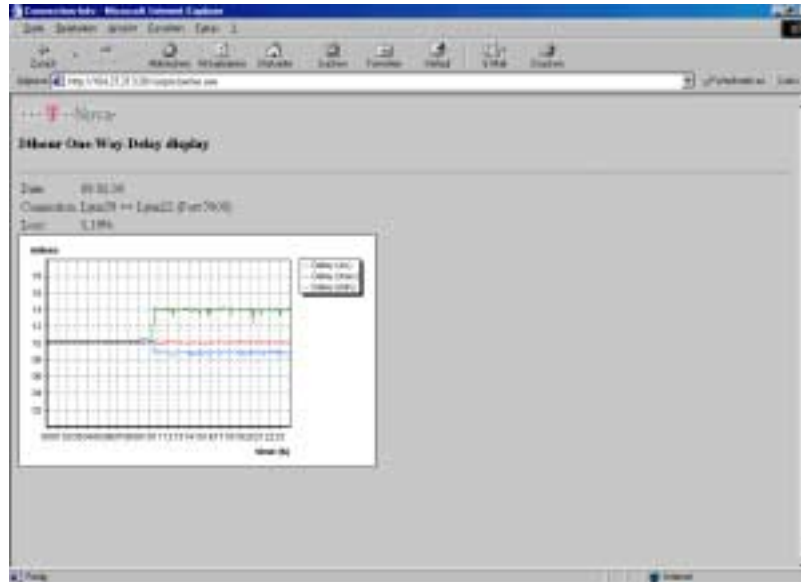


Figure 5-12: Offline display of the OneWay Delay

5.2.2 AQUILA architecture of the T-Nova toolset

The integration of the T-Nova toolset into the AQUILA measurement system requires a modification of T-Nova toolset architecture (Figure 5-13). The AQUILA measurement system, consists of:

- measurement agents (MA),
- one or more masterstations,
- Web server (Apache),
- Database (mySQL) and
- Web browser

In contrast to the old structure (see chapter 5.2.1) the configuration data **and** the measurement results are stored in a database (e.g. mySQL). The masterstation doesn't display the measurement results anymore, that's now the job of a Web-Browser. The masterstation in the new structure is only responsible for the configuration and supervision of the MA's and the handling of the measurement results (retrieving the results from the MA's, aggregation and saving in the database).

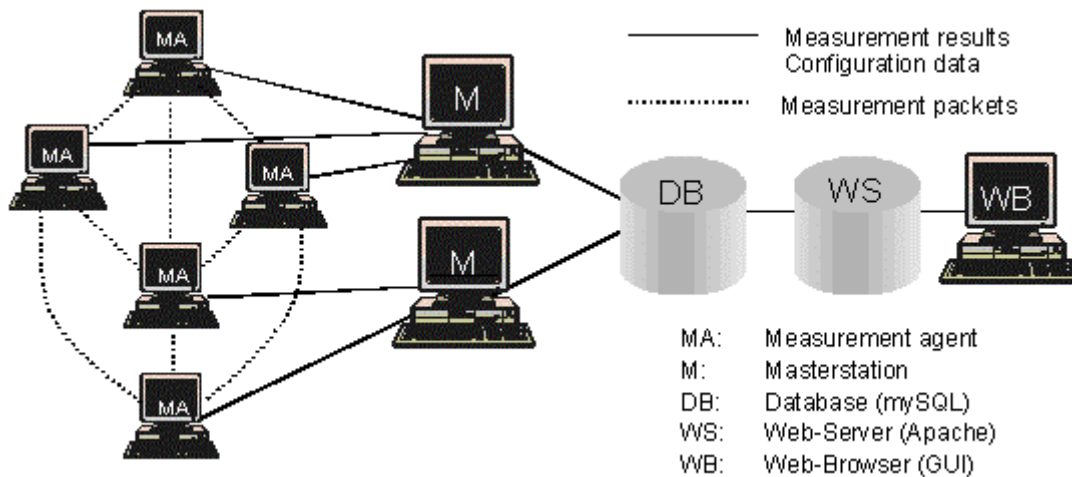


Figure 5-13: AQUILA architecture of the T-Nova toolset

The Browser extracts the results from the database (with the help of a PHP-Script in the Web-Server), converts it to an image (gnuPlot) and displays the resultgraph. The Web-Browser is also used for the system configuration (in the old architecture it was the task of the masterstation).Figure 5-14 shows an example Web-Browser screenshot of a configuration menu (prototype) for the AQUILA measurement system.

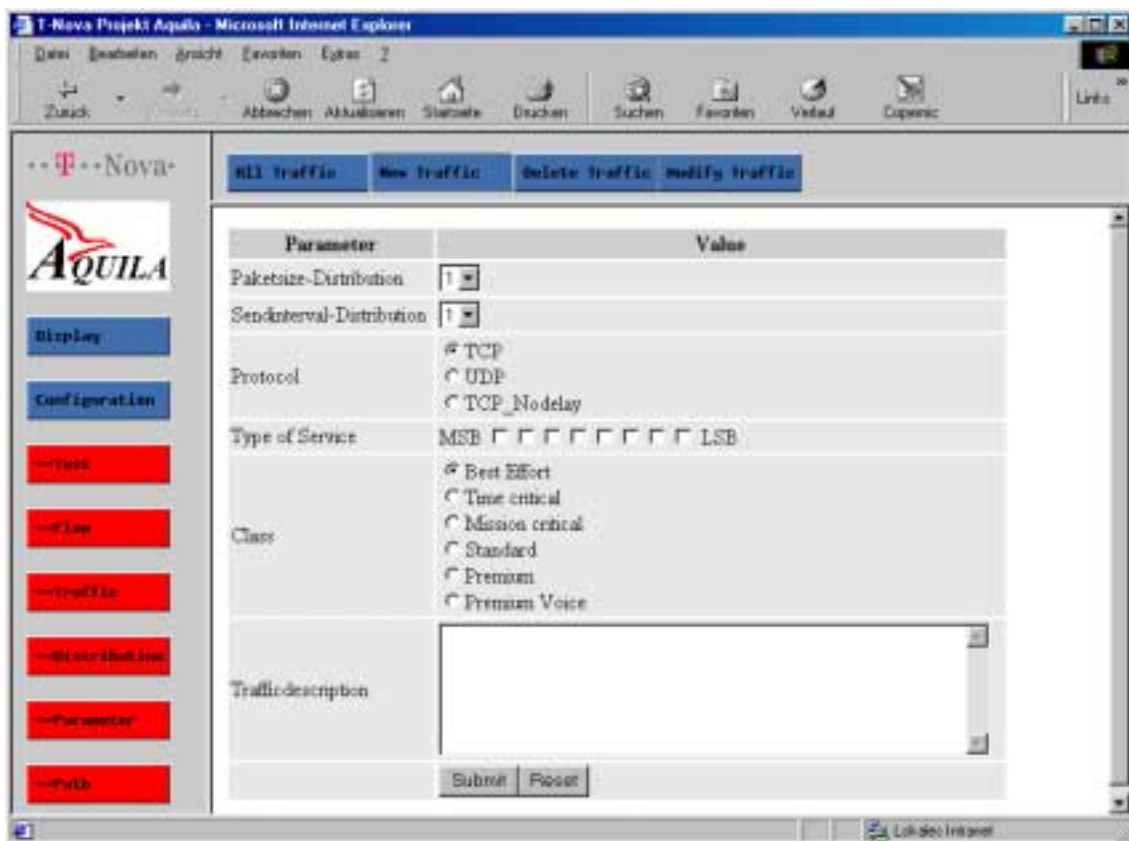


Figure 5-14: AQUILA Configuration example

5.3 Router QoS Monitoring Tool

A router has to perform various actions depending on the location in the network. A router in a core network has usually limited set of functions since it is supposed to handle fast amounts of traffic as efficiently as possible. The edge router between administrative domains, on the other hand, needs to make sure that there is no misuse of network resources and also facilitate to minimise processing load of the core routers. Network provider may also use two edge routers: one in the customer premises and other in providers own premises as an aggregation router where there are more than one customer connected to it. Since each of these routers perform different type of tasks there are different type of statistics obtainable from the router.

In the router parameters are commonly hierarchically organised. In Cisco routers some parameters are global and some are attributed to a certain interface. In the interface there are statistics for outbound traffic and for inbound traffic. In order to represent this information in a compact way this hierarchy must be understood.

Below some possible parameters are listed as an example; more detailed information on Cisco System's routers can be found in chapter 5.3.1.

- Edge router: Global
 - CPU usage (helps to determine how loaded the router is),
 - The number of packets matched with the access list entry (this could be also be interface specific if unique access list numbers are deployed)
- Edge router: Interface: Inbound
 - Statistics of the traffic conditioning functions (dropping, marking, remarking)
 - Number of packet received (can be used to calculate how loaded the router is)
- Edge router: Interface: Outbound: Class:
 - Number of packet dropped (tail or random) and matched
 - Mean queue depth
- Core router: Global
 - CPU usage (helps to determine how loaded the router is),
- Core router: Interface: Inbound
 - Number of packet received (can be used to calculate how loaded the router is)
- Core router: Interface: Outbound: Class:

- Number of packet dropped (tail or random) and matched
- Mean queue depth.

5.3.1 Statistics Retrieval

There are basically two ways to get DiffServ related statistics from Cisco routers. The first is Command Line Interface (CLI) and the second is Simple Network Management Protocol (SNMP). CLI is accessible with Telnet, SSH or console interface. SNMP requires the usage of SNMP manager software. Basically both CLI and SNMP provide the same information. Both of them are covered in this chapter.

5.3.1.1 Command Line Interface

This chapter provides information on what kind of Differentiated Services related statistics and parameters it is possible to monitor in Cisco's routers running IOS 12.0 or 12.1 using Command Line Interface (CLI). More information can be found at:

http://www.cisco.com/univercd/cc/td/doc/product/software/ios121/121cgcr/qos_r/index.htm

The outcomes of the commands are taken from Cisco 1750 platform running IOS 12.1. The following shows the relevant parts of the router's configuration. There are three Committed Access Rate (CAR) policers/markers in the input of **FastEthernet0** interface. The packets matching to access-lists 2, 3 and 4 (see the access-group number) are classified to be policed/marked by each of the CAR lists. There are three scheduling classes configured for the output of **serial10** interface (bandwidth 2 Mbps). Class **prec45** has a strict priority upto 200 kbps, while classes **prec23** and **class-default** have 700 kbps and 600 kbps respectively. In those classes Weighted Random Early Detection (WRED) is used. Precedences 2 and 3 are classified to belong to class **prec23**, while 4 and 5 belong to class **prec45**. The rest of the precedences go to **class-default**.

```
c1750#show configuration
!
version 12.1
!
class-map prec23
  match access-group 104
class-map prec45
  match access-group 105
!
policy-map policy2
  class prec23
    bandwidth 700
    random-detect
    random-detect precedence 2    8    16
    random-detect precedence 3    20   30
  class prec45
    priority 200
  class class-default
    bandwidth 600
    random-detect
!
!
interface Serial0
```



```
bandwidth 2000
ip address 194.241.226.222 255.255.255.252
service-policy output policy2
!
interface FastEthernet0
 ip address 11.11.11.1 255.255.255.0
 rate-limit input access-group 3 200000 2000 2000 conform-action set-prec-transmit 5
 exceed-action set-prec-transmit 4
 rate-limit input access-group 2 200000 2000 2000 conform-action set-prec-transmit 3
 exceed-action set-prec-transmit 2
 rate-limit input access-group 4 496000 6000 6000 conform-action set-prec-transmit 3
 exceed-action set-prec-transmit 2
!
ip route 0.0.0.0 0.0.0.0 Serial0
!
access-list 2 permit 11.11.11.4
access-list 2 permit 11.11.11.19
access-list 2 permit 11.11.11.20
access-list 3 permit 11.11.11.10
access-list 3 permit 11.11.11.9
access-list 4 permit 11.11.11.11
access-list 4 permit 11.11.11.15
access-list 4 permit 11.11.11.14
access-list 104 permit ip any any precedence immediate
access-list 104 permit ip any any precedence flash
access-list 105 permit ip any any precedence flash-override
access-list 105 permit ip any any precedence critical
!
end
```

SHOW INTERFACES RATE-LIMIT

With **show interfaces rate-limit** command it is possible to see the performance parameters for Committed Access Rate (CAR) policers/markers for each input and output interface. The command shows the token bucket parameters (rate, burst limit, extended burst limit), the number of both conformed and exceeded packets and the action that was taken for each of them (setting of precedence, dropping etc.). The counters are cumulative, so **clear counters** command has to be given in order to reset the counters.

The command can be used for example to check whether the profile for a certain customer for a certain traffic class is in balance with her actual profile for that class, i.e. not too many non-conforming (exceeded) packets etc. The information can be used as a basis for admission control or to do the required reconfigurations (such as increasing the token rate for a certain class or customer) when a new service request is being done.

```
c1750#show interfaces rate-limit
FastEthernet0
  Input
    matches: access-group 3
      params: 200000 bps, 2000 limit, 2000 extended limit
      conformed 1566 packets, 394632 bytes; action: set-prec-transmit 5
      exceeded 0 packets, 0 bytes; action: set-prec-transmit 4
      last packet: 29600ms ago, current burst: 0 bytes
      last cleared 00:04:40 ago, conformed 11000 bps, exceeded 0 bps
    matches: access-group 4
      params: 496000 bps, 2000 limit, 2000 extended limit
      conformed 13149 packets, 3313548 bytes; action: set-prec-transmit 3
      exceeded 2511 packets, 632772 bytes; action: set-prec-transmit 2
      last packet: 29600ms ago, current burst: 1860 bytes
      last cleared 00:04:40 ago, conformed 94000 bps, exceeded 18000 bps
    matches: access-group 2
      params: 200000 bps, 2000 limit, 2000 extended limit
      conformed 6806 packets, 1715112 bytes; action: set-prec-transmit 3
```



```
exceeded 8854 packets, 2231208 bytes; action: set-prec-transmit 2
last packet: 29608ms ago, current burst: 1784 bytes
last cleared 00:04:40 ago, conformed 48000 bps, exceeded 63000 bps
```

show access-lists command can be used to see how many packets have been matched to a certain access-list entry. The access-list has to an “extended access-list” (access-list number larger than 100, is able to match more complex expressions) so that the accounting is being done. The command can be used for example to check how many packets with which IP precedence have been forwarded by the router. **clear access-list counters** command has to be given in order to reset the counters.

SHOW ACCESS-LISTS

```
c1750#show access-lists 104
Extended IP access list 104
  permit ip any any precedence immediate (11365 matches)
  permit ip any any precedence flash (19875 matches)
c1750#show access-lists 105
Extended IP access list 105
  permit ip any any precedence flash-override
  permit ip any any precedence critical (3132 matches)
```

SHOW POLICY-MAP INTERFACE

It is possible to monitor the performance of Class Based Weighted Fair Queuing (CBWFQ) and Weighted Random Early Detection (WRED) in each output interface. The best way to do this is to use **show policy-map interface** command. The command shows the static parameters for each of the scheduling classes (amount or percentage of bandwidth or strict priority) and their internal WRED thresholds in number of packets. Number of matched packets in each class is shown, as is the number of discarded packets. If WRED is used within a class, the discard numbers are shown for each precedence and they are categorised by the reason of the discard (either random or tail drop). The mean queue depth for each class is also given, and it seems to vary in timescale of seconds.

The command can be used to monitor the performance of each scheduling class and drop priority level in terms of amount of traffic and discarded packets. Also queue depths can be polled in almost real time. This information can be used to do some counter-measures, such as increasing the bandwidth for a certain class if the number of discards in that class grows too high. **clear counters** command has to be given in order to reset the statistics.

```
c1750#show policy-map interface serial0
Serial0 output : policy2
  Weighted Fair Queueing
    Class prec23
      Output Queue: Conversation 265
      Bandwidth 700 (kbps) Packets Matched 31240
      mean queue depth: 0
      drops: class random tail min-th max-th mark-prob
              0 0 0 20 40 1/10
              1 0 0 22 40 1/10
              2 397 3723 8 16 1/10
              3 0 0 20 30 1/10
              4 0 0 28 40 1/10
              5 0 0 30 40 1/10
              6 0 0 32 40 1/10
              7 0 0 34 40 1/10
              rsvp 0 0 36 40 1/10
    Class prec45
```

```

Strict Priority
Output Queue: Conversation 264
  Bandwidth 200 (Kbps) Packets Matched 3132
  (pkts discards/bytes discards) 0/0
Class class-default
Output Queue: Conversation 266
  Bandwidth 600 (Kbps) Packets Matched 122035
  mean queue depth: 0
  drops: class random tail min-th max-th mark-prob
         0 3582 80463 20 40 1/10
         1 0 0 22 40 1/10
         2 0 0 24 40 1/10
         3 0 0 26 40 1/10
         4 0 0 28 40 1/10
         5 0 0 30 40 1/10
         6 0 1 32 40 1/10
         7 0 0 34 40 1/10
  
```

At least in Cisco 7500 platform the same command provides even more detailed information, such as 30 second average rate per class. The example below is taken from Cisco System's laboratory demo in Finland. Apparently no real traffic was sent by the time the command was issued, so all figures are zero.

```

hellab-7507-1#show policy-map interface atm4/0/0.2
ATM4/0/0.2
  service-policy output: IPTV
  class-map: standard (match-all)
    0 packets, 0 bytes
    30 second rate 0 bps
    match: ip precedence 5
    queue size 0, queue limit 0
    packet output 0, packet drop 0
    tail/random drop 0, no buffer drop 0, other drop 0
    bandwidth: class-based wfq, weight 13
    random-detect:
      Exp-weight-constant: 9 (1/512)
      Mean queue depth: 0
      Class Random Tail Minimum Maximum Mark Output
      drop drop threshold threshold probability packets
      0 0 0 0 0 1/10 0
      1 0 0 0 0 1/10 0
      2 0 0 0 0 1/10 0
      3 0 0 0 0 1/10 0
      4 0 0 0 0 1/10 0
      5 0 0 0 0 1/10 0
      6 0 0 0 0 1/10 0
      7 0 0 0 0 1/10 0
  class-map: gold (match-all)
    0 packets, 0 bytes
    30 second rate 0 bps
    match: ip precedence 7
    queue size 0, queue limit 0
    packet output 0, packet drop 0
    tail/random drop 0, no buffer drop 0, other drop 0
    bandwidth: class-based wfq, weight 2
    random-detect:
      Exp-weight-constant: 9 (1/512)
      Mean queue depth: 0
      Class Random Tail Minimum Maximum Mark Output
      drop drop threshold threshold probability packets
      0 0 0 0 0 1/10 0
      1 0 0 0 0 1/10 0
      2 0 0 0 0 1/10 0
  
```

```
3          0          0          0          0          1/10          0
4          0          0          0          0          1/10          0
5          0          0          0          0          1/10          0
6          0          0          0          0          1/10          0
7          0          0          0          0          1/10          0
class-map: class-default (match-any)
  0 packets, 0 bytes
  30 second rate 0 bps
  match: any
    0 packets, 0 bytes
    30 second rate 0 bps
  queue size 0, queue limit 2124
  packet output 11, packet drop 0
  tail/random drop 0, no buffer drop 0, other drop 0
  fair-queue: flow-based wfq
    per-flow queue limit 624
```

SHOW QUEUE & SHOW QUEUEING INTERFACE

show queue and **show queueing interface** commands provide basic general statistics for each of the interfaces, such as the total number of drops and the number of current conversations (flows). In theory the amount of total output drops should be equal to the sum of drops in all scheduling classes.

```
c1750#show queue serial0
Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 38493
Queueing strategy: weighted fair
Output queue: 0/1000/64/38493 (size/max total/threshold/drops)
Conversations 0/95/256 (active/max active/max total)
Reserved Conversations 2/2 (allocated/max allocated)
```

```
c1750#show queueing interface serial0
Interface Serial0 queueing strategy: fair
Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 38493
Queueing strategy: weighted fair
Output queue: 0/1000/64/38493 (size/max total/threshold/drops)
Conversations 0/95/256 (active/max active/max total)
Reserved Conversations 2/2 (allocated/max allocated)
```

SHOW INTERFACE

show interface command provides additional statistics of a certain interface. The main new information is the average 5 minute input/output rate.

```
c1750#show interface serial0
Serial0 is up, line protocol is up
Hardware is PowerQUICC Serial
Internet address is 194.241.226.222/30
MTU 1500 bytes, BW 2000 Kbit, DLY 20000 usec,
  reliability 255/255, txload 1/255, rxload 1/255
Encapsulation HDLC, loopback not set
Keepalive set (10 sec)
Last input 00:00:00, output 00:00:00, output hang never
Last clearing of "show interface" counters 5d17h
Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 38493
Queueing strategy: weighted fair
Output queue: 0/1000/64/38493 (size/max total/threshold/drops)
Conversations 0/95/256 (active/max active/max total)
Reserved Conversations 2/2 (allocated/max allocated)
5 minute input rate 1000 bits/sec, 3 packets/sec
5 minute output rate 0 bits/sec, 0 packets/sec
86607 packets input, 7851286 bytes, 0 no buffer
Received 65815 broadcasts, 0 runts, 0 giants, 0 throttles
```

```
0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
96950 packets output, 10425307 bytes, 0 underruns
0 output errors, 0 collisions, 0 interface resets
0 output buffer failures, 0 output buffers swapped out
0 carrier transitions
DCD=up DSR=up DTR=up RTS=up CTS=up
```

SHOW PROCESSES CPU

show processes cpu command provides information on the utilisation of router's processor. This is important since Quality of Service related tasks are often computationally expensive, and may result in over-utilisation of the processor.

```
c1750#show processes cpu
CPU utilization for five seconds: 0%/0%; one minute: 0%; five minutes: 0%
 PID  Runtime(ms)  Invoked  uSecs   5Sec   1Min   5Min  TTY Process
   1      16364      219827    74     0.00%  0.00%  0.00%  0 Load Meter
   2         780        294    2653   0.00%  0.33%  0.17%  6 Virtual Exec
   3    583144    112201   5197   0.00%  0.05%  0.05%  0 Check heaps
   4         0         1         0   0.00%  0.00%  0.00%  0 Chunk Manager
   5         652        367    1776   0.00%  0.00%  0.00%  0 Pool Manager
   6         0         2         0   0.00%  0.00%  0.00%  0 Timers
   7         0         2         0   0.00%  0.00%  0.00%  0 Serial Backgrou
   8    24620    22279   1105   0.00%  0.00%  0.00%  0 ARP Input
   9         0         3         0   0.00%  0.00%  0.00%  0 DDR Timers
  10         0         2         0   0.00%  0.00%  0.00%  0 Dialer event
  11         0         1         0   0.00%  0.00%  0.00%  0 Entity MIB API
  12         0         1         0   0.00%  0.00%  0.00%  0 SERIAL A'detect
  13         0         3         0   0.00%  0.00%  0.00%  0 Critical Bkgnd
  14    42372    138689   305   0.00%  0.00%  0.00%  0 Net Background
  15         12        349     34   0.00%  0.00%  0.00%  0 Logger
  16    24868    1098612   22   0.00%  0.00%  0.00%  0 TTY Background
  17     3596    1098617    3   0.00%  0.00%  0.00%  0 Per-Second Jobs
  18    17496    109991   159   0.00%  0.00%  0.00%  0 Net Input
  19     9008    219827    40   0.00%  0.00%  0.00%  0 Compute load av
  20   265092    18329  14462   0.00%  0.02%  0.00%  0 Per-minute Jobs
...
```

RSVP/IntServ Commands

In addition to these commands there are several RSVP/IntServ-related **show** commands available.

5.3.1.2 Simple Network Management Protocol

There are several MIBs available for monitoring purposes but few of them are directly Diff-Serv related.

No standard based MIBs are deployed since RFCs for DiffServ are yet to come; DiffServ MIBs are being defined in IETF's DiffServ working group [DIFFSERV-MIB] and also in rmonmib working group [RMONMIB-DSMONMIB].

Following MIBs are relevant if monitoring is to be performed using SNMP.

- CISCO-CAR-MIB
- CISCO-IP-STAT-MIB

- CISCO-WRED-MIB
- CISCO-RTTMON-MIB-120_5_T
- CISCO-QUEUE-MIB

The detailed description of Differentiated Services related MIBs that can be used for monitoring statistic can be found on Cisco Web site (<http://www.cisco.com/public/mibs/v2/>).

CISCO-CAR-MIB

<http://www.cisco.com/public/mibs/v2/CISCO-CAR-MIB.my>

CAR MIB contains configuration MIB objects and provides some information on statistics of specific CAR policy. Bellow is the statistics provided by CAR MIB for each CAR rule. Almost the same information is obtainable via CLI. With CLI some extra counters can be viewed: the time when the counters where cleared and the time when the last packet was matched with the CAR rule.

```
CcarStatTable:
  CcarStatEntry:
    ccarStatSwitchedPkts: Counter 32, "The counter of packets permitted
    by this rate limit"
    ccarStatSwitchedBytes: Counter 32, "The counter of bytes permitted
    by this interface"
    ccarStatFilteredPkts: Counter 32, "The counter of packets which ex-
    ceeded this rate limit."
    ccarStatFilteredBytes: Counter 32, "The counter of bytes which ex-
    ceeded this rate limit."
    ccarStatCurBurst: Gauge 32, "The current received burst size."
```

CISCO-IP-STAT-MIB

<http://www.cisco.com/public/mibs/v2/CISCO-IP-STAT-MIB.my>

In IP statistic MIB there is also a table that is addressing IP precedence. From the MIBs it is possible to see how many bytes and packets with specific precedence was received or transmitted per interface. With CBWFQ, however, this MIB didn't show (at least this was the case with C1750 and software version 12.1(1a)T1) all of the precedence packets, but just the ones belonging to the default class. Bellow is the structure of the IP precedence table:

```
ip accounting:
  IP Precedence Statistic Table: precedence { input | ouput } ]

  "An entry in the cipPrecedenceTable is created for each IP precedence value.
  There are 8 precedences total."

  INDEX { ifIndex, cipPrecedenceDirection, cipPrecedenceIpPrecedence }

  CipPrecedenceDirection: "The data source for the object."
    PacketSource,
  CipPrecedenceIpPrecedence: "The ip precedence value this object is
  collected upon."
    Integer32,
  CipPrecedenceSwitchedPkts: "Traffic, in packets, at the cipPreceden-
  ceIpPrecedence precedence."
    Counter32,
  CipPrecedenceSwitchedBytes: "Traffic, in bytes, at the cipPreceden-
  ceIpPrecedence precedence."
```

Counter32

CISCO-WRED-MIB

<http://www.cisco.com/public/mibs/v2/CISCO-WRED-MIB.my>

WRED MIB contains both configuration objects and objects for gathering statistics. In principle the queue average length and the current queue depth can be obtained as well as statistics of the packet drops in respect to different thresholds defined per queue. However, this MIB could not be found in software version 12.1(1a)T1 for C1750 routers or for C2600 routers (other router models were not tested). Bellow is more detailed description of the relevant objects.

```
CwredQueueEntry :
    CwredQueueAverage: Gauge32, "The computed queue average length."
    CwredQueueDepth: Gauge32, "The number of buffers/particles currently withheld in queue."

CwredStatEntry:
    WredStatSwitchedPkts: Counter32
        "The number of packets output by WRED."

    CwredStatRandomFilteredPkts: Counter32,
    "The number of packets filtered/dropped due to average queue length exceeds cwredConfigMinDepthThreshold
    and meet a defined random drop policy."

    cwredStatMaxFilteredPkts:Counter32,
    "The number of packets filtered/dropped due to average
    queue length exceeds cwredConfigMaxDepthThreshold."
```

CISCO-QUEUE-MIB

<http://www.cisco.com/public/mibs/v2/CISCO-QUEUE-MIB.my>

Queuing MIBs provide information on queue types configured as well as some detailed configuration parameters on Custom queuing. Also a statistics table per queue is available (details bellow). Queuing MIB supports only FIFO, CQ, PQ and WFQ. More information can be obtained through CLI since also show commands for CBWFQ can be utilised.

```
CQAlgorithm ::= "The type of queuing algorithm used on the interface."

    fifo(1),          -- First In First Out
    priority(2),     -- Priority Queuing
    custom(3),       -- Custom Queuing
    weightedFair(4) -- Weighted Fair Queuing
```

Sub-Queue Statistics Table:

```
CQStatsEntry :
    CQStatsQNumber:Integer32 (0..2147483647),
    "The number of the queue within the queue set"
    cQStatsDepth :Gauge32,
    "The number of messages in the sub-queue"
    cQStatsMaxDepth: Integer32,
    "The maximum number of messages permitted in the sub-queue"
    cQStatsDiscards: Counter32,
    "The number of messages discarded from this queue since restart"
```

CISCO-RTTMON-MIB-120_5_T

http://www.cisco.com/public/mibs/v2/CISCO-RTTMON-MIB-120_5_T.my

The Response Time Reporter (RTR) allows round trip delay monitoring using for example UDP and TCP packets with a certain TOS value. It captures statistics and collect error information for example statistical distributions of response times, minimum and maximum response times and number of completions and errors. SNMP traps can be sent whenever a certain predefined threshold is reached and statistics from the MIB can be read using SNMP.

5.3.1.3 Summary

Cisco routers have currently two options implemented for DiffServ related statistics retrieval: SNMP and CLI. They both contain Cisco specific features (show commands and related outputs or MIBs) and these features can vary depending on the router type. In the future it is likely that at least some of the DiffServ MIBs will be based on MIBs specified by IETF. However, presently it seems that more information is available through CLI since not all of the implemented features have related MIBs (at least CBWFQ and access lists lack MIB support).

At this time it is advisable that CLI is deployed for at least gathering information on:

- mean queue lengths (which is given in packets and can be configured by assigning different values for the exponential weight factor) per configured class per interface,
- the number of dropped packets per class per interface and if WRED is used drops for each precedence within a class on a certain interface categorised by the reason of the discard,
- the number or matched packets per class per an interface.

These values can be used to monitor the state of the network as a whole. For example to see if there are lots of discards in strict priority traffic class or if the mean queue sizes implies long delays. Additionally, the topology information can be combined with above statistics giving the possibility to evaluate the state of the individual flows in the network.

However, it is important to remember that the commands issued via telnet are usually low priority operations in the router. If there is a need to gather information even when the routers CPU is heavily loaded, interrupt intervals should be specified for low priority operations.

Both SNMP and CLI can be used for monitoring the performance of Committed Access Rate (CAR) policers/markers for each input and output interface at the edge of the network. The relevant parameters are the number of conformed packets/bytes and the number of exceeded packets/bytes. These statistics can be used for example to identify if the reservation for a certain micro flow has not been sufficient.

The Response Time Reporter (RTR) can also be used for monitoring response times inside the network. This feature is implemented in most current software releases (12.1(2)) for several router models.

5.3.2 General architecture of the monitoring tool

The purpose of the monitoring tool is to collect statistical data from routers in the test network. The data consists of several parameters, which describe how different packets are being handled in the routers. The monitoring tool uses command line interface via telnet to fetch parameters from the router.

The monitoring tool runs on a single machine, and the operating system is Linux.

The monitoring tool consists of three different executables programmed in C, and a shell script, that combines these programs. These programs are called **routerinit**, **collector** and **analyzer**.

5.3.2.1 routerinit

The first executable is called **routerinit**, it is used to make a local database, that stores static information about a specific router (interface names, IP addresses). This program is run whenever a router interface is reconfigured. The database is used to convert IP addresses to interface names in **collector**.

5.3.2.2 collector

The second executable is **collector**. The purpose of this program is to convert the list of requested parameters to router **show** commands, which are then sent to **telnet** via a pipe. The parameters collector takes are passwords, parameter number file name and IP address of the router.

The parameter number file consists of n lines with the following format:

<parameter number> <access list number> <IP address of the interface>

collector reads this parameter file line by line, and outputs the appropriate commands to the router. The output from **telnet** is then piped to the **analyzer** program. The parameter numbers are listed in Table 5-3. The parameters are taken from Cisco router output, other routers may / may not have the same parameters.

Input parameters	
0	# of conformed input packets
1	# of conformed input bytes

2	# of exceeded input packets
3	# of exceeded input bytes
4	Milliseconds since last packet
5	Size of current burst
Output precedence matches	
10	Output packets with priority 0
11	Output packets with priority 1
12	Output packets with priority 2
13	Output packets with priority 3
14	Output packets with priority 4
15	Output packets with priority 5
16	Output packets with priority 6
17	Output packets with priority 7
Output queue statistics	
20	Packets matching specified policy
21	Mean queue depth
30-37	WFQ Randomly dropped packets with priority 0-7
38	WFQ Randomly dropped RSVP packets
40-47	WFQ Tail dropped packets with priority 0-7
48	WFQ Randomly dropped RSVP packets
50	Strict Priority discarded packets
51	Strict Priority discarded bytes
Interface statistics	
60	Total input packets at interface

61	Total input bytes at interface
62	Total output packets at interface
63	Total output bytes at interface
CPU utilisation	
70	CPU average utilisation in last 5 seconds
71	CPU average utilisation in last minute
72	CPU average utilisation in last 5 minutes

Table 5-3: Parameter numbering

5.3.2.3 analyzer

The **analyzer** reads the router output from the **telnet** program, and parses the parameters needed. The data gathered is then saved to a file, or it can be written directly to the MySQL database.

5.3.2.4 Shell script

The shell script combines **collector** and **analyzer**, so the user only needs to start the script with passwords, IP address of the router and the file that contains the parameters user needs. The shell script takes care of running **collector** and **analyzer**.

5.3.3 Other notes

The time it takes to log in to the router is 0,60 seconds, and the delay between individual commands is 0,15 seconds. Thus, the minimum interval for statistical data retrieving is 0,15 seconds, when the parameters are listed in a file. Therefore, to monitor a specific parameter for 10 minutes, the parameter file supplied to **collector** has to contain $600s/0,15s = 4000$ lines. If an interval of 0,60 seconds is tolerable, then the program can be run with the same parameter file.

6 System Integration

Figure 6-1 shows the physical architecture for the 1st trial as described in [D1201].

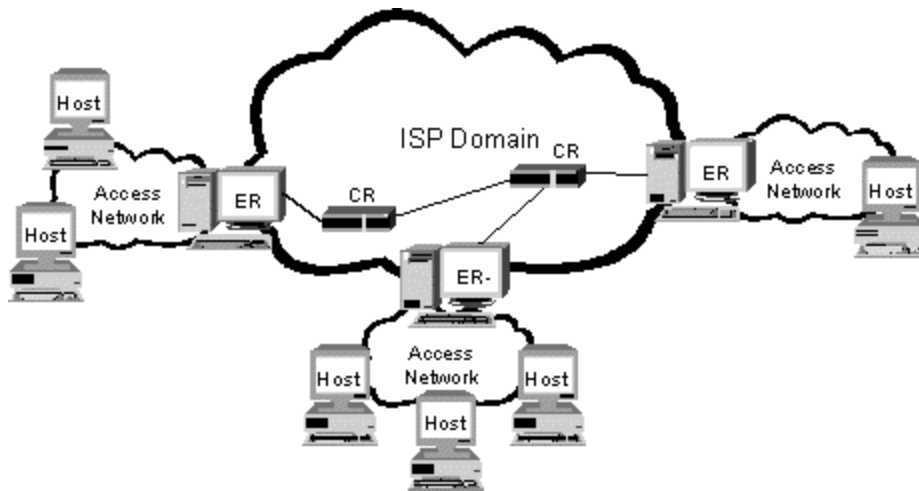


Figure 6-1: Physical architecture for the first trial

6.1 Hardware and Software Requirements

As stated in previous chapters the parts of the measurement system involved in active measurements use standard components. Their requirements are listed in the following paragraphs. It is recommended to combine the functionality of the management station, the web server and the database into one PC. In this case the following requirements should be met:

- Pentium III or AMD K7 with 700 Mhz or better
- 256MB RAM
- 20 GB Harddisk
- Network interface card: Ethernet 10 / 100 Mbit/s
- Linux kernel 2.2 or higher, e.g. SuSE 6.4
- Software requirements: see below

6.1.1 Management Station

According to the current system design the management station is a standard PC running Linux. If the management processes of the different test tools run on separate machines, the following requirements can be given:

- Pentium II with 450 Mhz or better
- 128 MB RAM
- 2 GB Harddisk
- Monitor: not required, system can be used as a black-box. For supervision purposes a monitor or telnet connection can be used.
- Network interface card: Ethernet 10 / 100 Mbit/s.
- Linux kernel 2.2 or higher, e.g. SuSE 6.4

6.1.2 Measurement Agents

According to the current system design the measurement agents are installed at standard PCs running Linux. The measurement agents of the different tools can co-exist on one machine with the following requirements:

- Pentium II with 450 Mhz or better
- 128 MB RAM
- 2 GB Harddisk
- Monitor: not required, system can be used as a black-box. For supervision purposes a monitor or telnet connection can be used.
- Network interface card: Ethernet 10 / 100 Mbit/s.
- Linux kernel 2.2 or higher, e.g. SuSE 6.4
- GPS-card / antenna for time synchronisation: Meinberg GPS 167 PC (ISA or PCI version)

6.1.3 Web Server

According to the current system design the web server is a standard PC running Linux:

- Pentium II with 450 Mhz or better
- 128 MB RAM
- 8 GB Harddisk
- Network interface card: Ethernet 10 / 100 Mbit/s.
- Linux kernel 2.2 or higher, e.g. SuSE 6.4

- Web server software with PHP4 support, e.g. apache (1.3.12 or higher).
- For graphics display: gnuplot 3.7 or higher.

6.1.4 Database

According to the current system design the database is implemented on a standard PC running Linux:

- Pentium II with 450 Mhz or better
- 128 MB RAM
- 10 GB Harddisk
- Network interface card: Ethernet 10 / 100 Mbit/s.
- Linux kernel 2.2 or higher, e.g. SuSE 6.4
- Database software: MySQL 3.22.32 or higher.

6.1.5 Routers

Routers that are part of the AQUILA first phase trial architecture are either Siemens Unisphere ERX or Cisco Systems routers. At this point router monitoring tool is only supporting Cisco routers and also this study of existing implementations of routers to support DiffServ related statistics retrieval is done only for Cisco routers. It is important to keep in mind that only IOS releases 12.0 and 12.1(2)T have been investigated and the main scope has been on the C1750, C2500, C2600 and C7200 router series.

6.1.6 Performance of load generators on different end systems

To simulate a real network in a testbed where you can measure parameters like „one-way-delay”, „jitter” and „packet loss” you have to stress the network with a defined traffic. Therefore you need load generators.

The software based load generator is controlled by the AQUILA measurement database. This load generator must generate a maximum throughput on various network connections with the adjustable parameters:

- Protocol (TCP/UDP)
- Port-No.
- DSCP / IP-ToS-Byte
- Generator function with different traffic characteristics

- Packet length

To find a suitable platform we wrote load generators for the operating systems:

- MS Windows 95
- MS Windows NT 4
- Linux kernel 2.2
- Sun Solaris 2.6 (only for reference)

In addition we have tested MS Windows NT with full-function demo products:

- ZTI LanTrafficV2
- Ganymede Chariot

6.1.6.1 Performance of different operating systems

To measure the performance of different operating systems we connected two workstations to our LAN. We started the test-tools to measure the maximum flow from Workstation 1 to Workstation 2 and then the same test in the opposite direction.

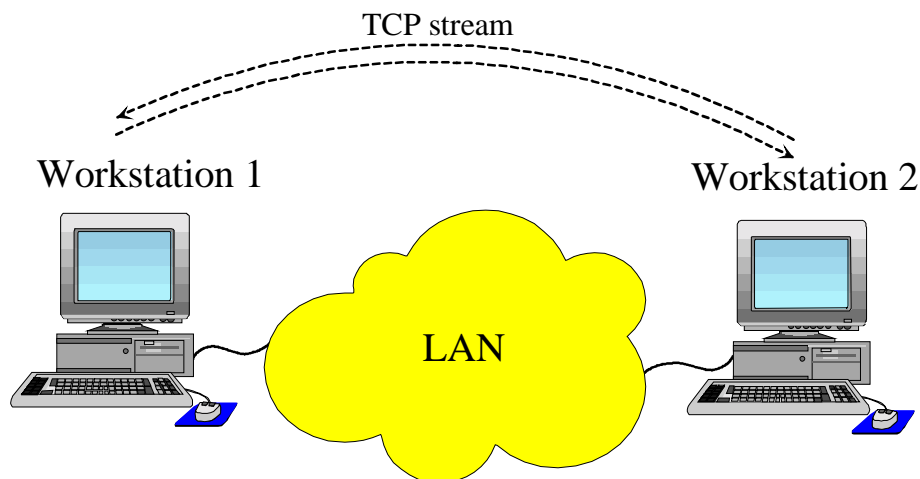


Figure 6-2: Test assembly

The following tables show the different test configurations with the measured throughput and the range of the DSCP / ToS-Byte settings.

6.1.6.1.1 Microsoft Windows

Windows NT to Windows NT

	Workstation 1	Workstation 2
operating system	WinNT-4.0	WinNT-4.0
processor	PentiumII 450MHz	PentiumIII 500MHz
network interface	100MBit/s	100MBit/s
test-tools	T-Nova E513-load generator for Windows ZTI LanTrafficV2 Ganymede Chariot	
protocol	TCP	
testbed	LAN 100Mbit/s	
max. throughput on one TCP connection	~50MBit/s	
DSCP / ToS-Byte for UDP	any value	
DSCP / ToS-Byte for TCP	any value, but only on the server side	

Table 6-1: WinNT to WinNT

Remark: with more than one TCP connection it is possible to increase the throughput up to ~85MBit/s. The *T-Nova E513-load generator for Windows* uses the WinSock version 1.1.

From the configuration *WinNT to WinNT* we expected better results than *Win95/NT to Win95*. This is the reason we have tested the commercial products *ZTI LanTrafficV2* and *Ganymede Chariot* in this configuration and not with *Win95* (see below). The throughput was for each test-tool nearly the same.

A *Windows NT* system is not suitable for the AQUILA load generator. The maximum throughput over one TCP connection is too low and the DSCP / ToS-Byte is not adjustable on the TCP client.

Windows NT to Windows 95

	Workstation 1	Workstation 2
operating system	WinNT-4.0	Win95

processor	PentiumII 450MHz	PentiumIII 500MHz
network interface	100MBit/s	100MBit/s
test-tools	T-Nova E513-load generator for Windows	
protocol	TCP	
testbed	LAN 100MBit/s	
max. throughput on one TCP connection	~41MBit/s	
DSCP / ToS-Byte for UDP	any value	
DSCP / ToS-Byte for TCP	any value, but only on the server side	

Table 6-2: WinNT to Win95

Remark: because of the low throughput we made no further tests with Windows 95. The *T-Nova E513-load generator for Windows* uses the WinSock version 1.1.

A *Windows 95* system is also not suitable for the AQUILA load generator. The maximum throughput over one TCP connection is too low and the DSCP / ToS-Byte is not adjustable on the TCP client.

6.1.6.1.2 Linux

Enable the setting of the DSCP / ToS-Byte

In a standard Linux installation, the DSCP / ToS-Byte can only set in a range of 0h – A0h for all even-numbered values. To enable the setting of any value in the range between 0 – FFh, the file `/usr/src/linux/net/ipv4/ip_sockglue.c` must be edited.

Around line 460 is in a switch-statement the line “`case IP_TOS :`”.

The following lines:

```

if (val & ~(IPTOS_TOS_MASK|IPTOS_PREC_MASK))
    return -EINVAL;
if (IPTOS_PREC(val) >= IPTOS_PREC_CRITIC_ECP &&
    !capable(CAP_NET_ADMIN))
    return -EPERM;

```

must be changed to:

```

if(val <0 || val >255)
    return -EINVAL;

```

After that a new kernel must be built and installed.

Linux to Sun

	Workstation 1	Workstation 2
operating system	Linux Kernel-2.2	Sun Solaris 2.6
processor	PentiumII 450MHz	Sparc 360MHz
network interface	100MBit/s	100MBit/s
test-tools	T-Nova E513-load generator for UNIX	
protocol	TCP	
testbed	LAN 100MBit/s	
max. throughput on one TCP connection	~95MBit/s	
DSCP / ToS-Byte for UDP / TCP	any value	

Table 6-3: Linux to Sun

Remark: the *Sun* workstation was used to get a reference value. The DSCP /ToS-Byte was not tested on the *Sun* workstation.

On *Sun* workstations the maximum throughput is approximate the full network load. So we can develop a reference load generator, but the *Sun* workstations are too expensive for the common use.

Linux to Linux

	Workstation 1	Workstation 2
operating system	Linux Kernel-2.2	Linux Kernel-2.2
processor	PentiumII 450MHz	Pentium 166MHz
network interface	100MBit/s	100MBit/s
test-tools	T-Nova E513-load generator for UNIX	
protocol	TCP	
testbed	LAN 100MBit/s	

max. throughput on one TCP connection	~80MBit/s
DSCP / ToS-Byte for UDP / TCP	any value

Table 6-4: Linux to Linux

Remark: the lower throughput of the *Linux to Linux* test compared to the *Linux to Sun* test can be explained by the low performance of Workstation 2 (166MHz).

The AQUILA load generator can be developed on *Linux*. We expect that the throughput increases with a second fast workstation up to full network load. The DSCP / ToS-Byte is adjustable for common values and the costs for the hardware and the software are very low.

6.1.6.2 Properties

In addition to the previous tests, we have compared the properties of other operating systems.

Table 6-5 gives an overview of the required properties for a suitable platform. The information is tested by our own or is taken from data sheets. Properties we could not find out yet are marked with a „-“.

Description of the columns:

- tested: we have tested the OS with a load generator
- DSCP/ToS: the DSCP/ToS-Byte is adjustable from a program
- RSVP: is RSVP supported by the OS?
- real time: required for exact measurement
- direct hardware access: is needed for a GPS-Clock
- throughput: maximum throughput we measured in our tests
- costs: the costs for a complete system (hard- and software)

A suitable platform let a user program adjust the DSCP/ToS-Byte and it supports RSVP. For exact measurement a real time system is required, a GPS-Clock must read out by direct hardware access and the maximum throughput over a network has to be nearly the full network load.

property	tested	DSCP / ToS	RSVP	real time	direct hardware access	throughput	cost
OS							
Win95	yes	only on server side	no	no	yes	~41Mbit/s	low
Win98	no	only on server side	no	no	yes	-	low
WinNT 4	yes	only on server side	no	no	no, only with a driver	~50Mbit/s	medium
Win2000	no	yes	yes	no	no, only with a driver	-	medium
RT-DOS	no	-	no	yes	yes	-	medium
Linux 2.2	yes	yes	no	no	yes	>85Mbit/s	low
RT-Linux	no	yes	no	yes	yes, but no system calls (e.g. network functions)	-	low
KURT	no	yes	no	yes, but not preemptive	yes	-	low
Solaris	yes	yes	-	no	yes	~95Mbit/s	high
LynxOS	no	yes	no	yes	yes	-	high

Table 6-5: Overview of operating systems

No operating system in this table matches all requirements. The throughput of MS Windows 9x and NT4 is too low and RSVP is not supported, but Windows 2000 with WinSock 2 should be tested. The developer kit for RT-DOS is too expensive to buy it only for testing.

Linux 2.2 has no support for RSVP and is not a real time system, but it is a cheap system with a high throughput. RT-Linux and KURT are Linux kernel patches. RT-Linux has the handicap, that no system calls are available in the real time mode and so the network access is still running in a „non real time mode”. But this is necessary for the load generator. KURT is not pre-emptive in the real time mode (no advantage to normal Linux) and still under develop.

Solaris has no advantage to Linux, but it is more expensive.

LynxOS is a full real time OS, but very expensive. We expect, that the maximum throughput is as high as Solaris.

6.1.6.3 Conclusion

The conclusion of this comparison is that we develop our load generator on a *Linux* system. The throughput over a TCP connection is high enough, the DSCP / ToS-Byte is adjustable and the costs for a system are very low.

For *Sun* workstations we expect a little better performance, but the costs for a workstation are higher. So this platform can be used as a reference tool.

The performance of *Windows* workstations (*NT + 95*) is too low for a useful load generator. Also the DSCP / ToS-Byte couldn't be set on the TCP-Client-Workstation.

6.2 Proposed Example Scenarios

The following example scenarios are proposed for the integration of the measurement tools in AQUILA network architecture of the 1st trial.

6.2.1 Simple QoS Measurement Scenario

The first measurement scenario should be to verify, whether the flows, which are defined for specific network services are mapped to the according traffic classes and whether the flows are treated in the desired way. Therefore a reservation request is made by using the EAT and an end-to-end flow with the reserved (or higher) bandwidth is started. The results produced by the end-to-end flow like throughput and packet delay can be compared with the targeted QoS parameters for this network service. Additionally the marking and dropping policy at the edge routers can be verified with the data collected by the router QoS monitoring tool.

6.2.2 Admission Control Scenario

This scenario is targeted to validate admission control in AQUILA. This can be done by a stepwise increase of the number of flows in the network. Therefore the synthetic flow load generator is used in combination with the EAT to reserve bandwidth. For each reserved flow an end-2-end flow is started and measured. The bandwidth can be reserved either via the GUI of the EAT or via its proxy (depends on implementation progress).

For example an incremental scenario using different network services could be:

Step	# of flows per network service			
	PCBR	PVBR	PMM	PMC
1	1	1	1	1
2	2	2	2	Reject
3	3	3	3	Reject
4	4	4	Reject	Reject

5	5	Reject	Reject	Reject
6	Reject	Reject	Reject	Reject

Table 6-6: Example Scenario for admission control

The result of this example would be that the network is admitting 5 PCBR-flows, 4 PVBR-flows, 3 PMM-flows and 1 PMC-flow. As this flows are generated using the synthetic flow load generator from the measurement utilities, for each flow the measurement results are available, which can be analysed (e.g. whether and at which level the target quantitative values for QoS Parameters of the network services are reached as defined). To get a sufficient load on the network, a large number of flows may be necessary. These can be generated either

- by using additional load generators, which set the ToS-Byte in the IP-Header themselves and are therefore assigned directly to the traffic classes at the routers or
- by using an aggregated flow model and reserve the maximum admitted bandwidth (e.g. 200 kbit/s for PCBR) for this flow.

The reasons why flows are possibly not satisfied can be found out by an analysis using the information collected by the router QoS monitoring tool. This supplies for example information about packet drops per traffic class or the overall load situation of the router.

All measurement results are available from the measurement database and are accessible via a GUI (for simple result analysis) or via a standard interfaces (e.g. SQL, JDBC).

6.2.3 Observing the QoS properties of traffic flows by probing

In order to measure and monitor the QoS properties of traffic flows in a DiffServ network, active measurements can be used. The basic idea is to periodically inject measurement packets to get “online” measurement of the achieved QoS properties (delay, jitter, packet loss rate). A suitable measurement system is described in chapter 5.2.2. Figure 6-3 shows a sample view of the system with measurement agents and required server entities (WWW-server, MySQL database server) and master station.

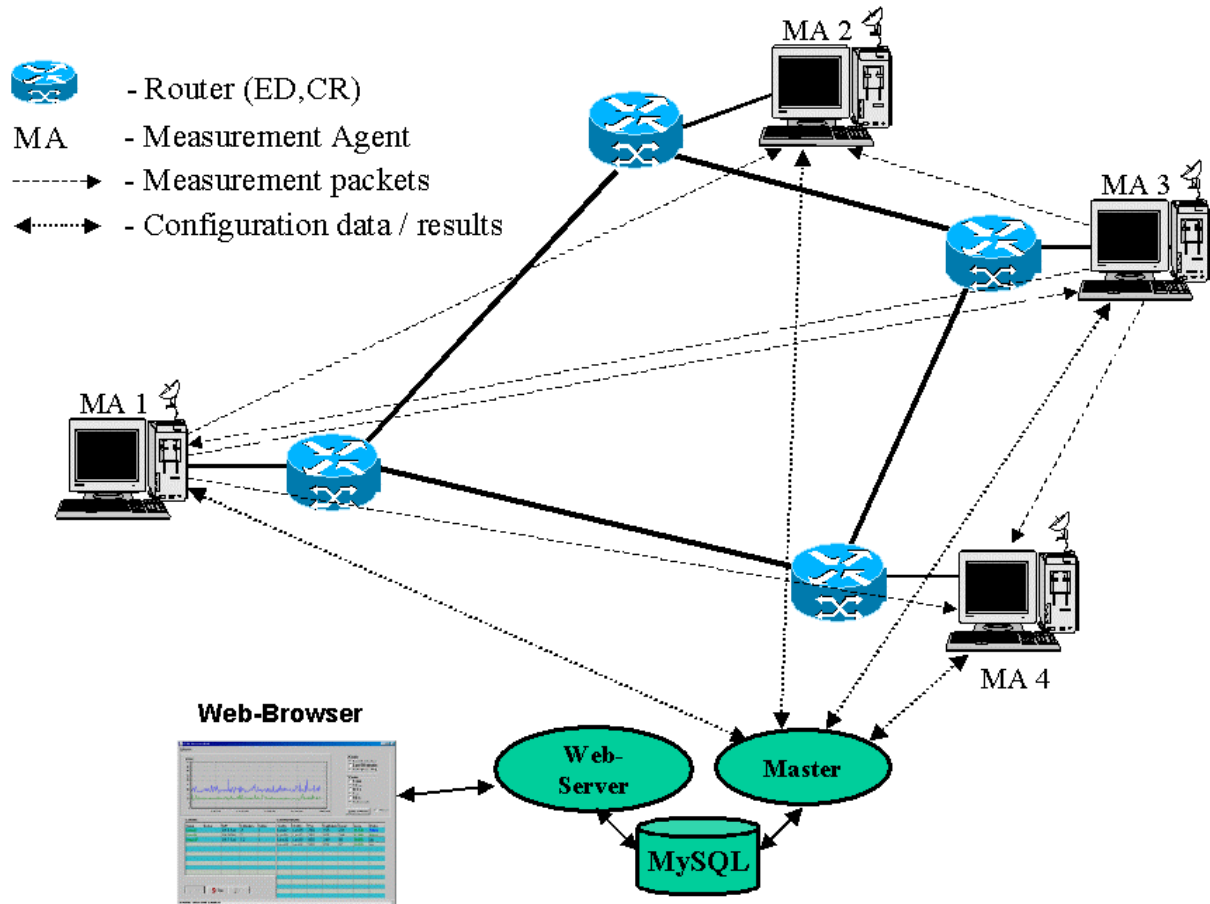


Figure 6-3: QoS monitoring

Each of the MAs is associated with an edge or core router. The MAs are fully meshed, that means:

- From one MA to every other MA in the DiffServ network a measurement flow is established (Figure 6-3 shows only the measurement flows from MA 1 and 3) .
- Every flow transports measurement packets for every traffic class. Every MA therefore has to handle $(n-1)*t_c*2$ measurement connections (n = number of MA's, t_c = number of traffic classes, 2 = one flow for each direction).

The measurement results are stored in the database in the system server. The results

- one way delay,
- jitter and
- packet loss

can be displayed by every WWW-browser which has access to the system server. In a network with an increasing number of MA's several masterstations can be integrated into the system, each responsible for up to 15 MA's.

With this architecture a whole network can constantly be monitored. The measurement agents can be placed at locations of core routers, edge devices and / or customer hosts. Depending on the actual placement of measurement agents different granularities in monitoring the network can be achieved.

7 Abbreviations

ACA	Admission Control Agent
AQUILA	Adaptive Resource Control for QoS Using an IP-based Layered Architecture
CAR	Committed Access Rate
CBWFQ	Class Based Weighted Fair Queuing
CLI	Command Line Interface
CR	Core Router
DBMS	Database Management System
DMA	Distributed Measurement Architecture
DNS	Domain Name Service
DSCP	DiffServ Code Point
DV	Delay Variation
EAT	End-User Application Toolkit
ED	Edge Device
GPS	Global Positioning System
GUI	Graphical User Interface
IETF	Internet Engineering Task Force
IP	Internet Protocol
IPDV	Instantaneous Packet Delay Variation
IPPM	IP Performance Metrics
ITU	International Telecommunication Union
JDBC	Java Database Connectivity
MA	Measurement Agent
MIB	Management Information Base
MPQM	Moving Picture Quality Metric
NIC	Network Interface Card
OWD	One-way Delay
PCBR	Premium CBR (AQUILA network service)
PHB	Per Hop Behaviour
PL	Packet Loss

PMC	Premium Mission Critical (AQUILA network service)
PMM	Premium Multimedia (AQUILA network service)
PVBR	Premium VBR (AQUILA network service)
QoS	Quality of Service
RCA	Resource Control Agent
RCL	Resource Control Layer
RFC	Request for Comment
RSVP	Resource Reservation Protocol
SLA	Service Level Agreement
SNMP	Simple Network Management Protocol
SQL	Standard Query Language
TCP	Transmission Control Protocol
TOS	Type of Service
UDP	User Datagram Protocol
VoIP	Voice over IP
WRED	Weighted Random Early Detection

8 Glossary

Active Network Probing Tool	In AQUILA: a performance monitoring tool based on active probes for collecting path performance metrics.
Active probe	A device or embedded software program that combines both synthetic source and probe functionality.
Admission Control (ACA Agent)	<p>A logical entity of the RCL. The ACA performs policy based local admission control.</p> <p>Admission control. The process of determining whether a flow can be granted the requested QoS. Admission Control is processed by the network and can be resource and/or policy based.</p> <p>Local Admission Control. Based on locally managed resources and/or policies.</p>
Application	<p>In terms of AQUILA an end-user application that uses a network (i.e. the Internet) for communication-based purposes. Such applications mainly consists of two levels: Firstly, the underlying user program, and secondly, the online service to be made available.</p> <p>Legacy Application. An end-user application which is not QoS-aware but can <i>indirectly</i> benefit from the QoS capabilities of the network.</p> <p>QoS-aware Application. In terms of AQUILA an application that benefits <i>directly</i> from the QoS capabilities of the network by using either an API of a QoS middleware or an appropriate signalling protocol such as RSVP. (Applications that use the API of the EAT middleware are called EAT-based applications in the following.)</p>
Autonomous System (AS)	A self-connected set of networks that are generally operated within the same administrative domain.
Benchmark	A measurement scenario which is specified for an entity to validate its performance
Clock's "Skew"	The frequency difference (first derivative of its offset with respect to true time) between the clock and true time at a particular moment.

Cloud	An undirected (possibly cyclic) graph whose vertices are routers and whose edges are links that connect pairs of routers. Formally, ethernets, frame relay clouds, and other links that connect more than two routers are modelled as fully-connected meshes of graph edges. Note that to connect to a cloud means to connect to a router of the cloud over a link; this link is not itself part of the cloud.
Cloud subpath	A subpath of a given path, all of whose hosts are routers of a given cloud.
Core Router	A router that is deployed at the core of an administrative domain.
Customer	An entity that purchases a specific network service. The customer acts either as an intermediate entity between the network provider and the end-user or as the end-user itself. In AQUILA a customer is equivalent to an end-user.
Distributed Measurement Architecture (DMA)	A collection of measurement entities which are distributed in the network architecture.
Drift	Real clocks exhibit some variation in skew. The second derivative of the clock's offset with respect to true time is generally non-zero. This quantity is defined as the clock's "drift".
Edge Device	A device such as a router or a gateway that is deployed at the border of an administrative domain. This can be an inter-domain border (then also called border router) or the border to the hosts. Two specialisations exist specifying whether the edge device belongs to the core (provider edge, edge router) or to the access side of a network (customer edge, access router).
Egress	The point where traffic <i>leaves</i> the network or the domain. The receiver is located at this point.
Egress traffic	Traffic exiting the network
End-user	A person or a group of persons external to the network that utilises the network to work on a task, to offer something, etc., by using so-called end-user applications.
End-user Application Toolkit (EAT)	A logical entity of the RCL. The EAT mediates between the user programs of the host and the ACA on the network.
Exchange	A special case of a link, an exchange directly connects either a host to a cloud and/or one cloud to another cloud.

Flow	In terms of AQUILA a set of packets belonging the same application session.
Guarantee	The level of probability that an end-user gets the QoS he/she requested. While hard guarantee means the probability of 100%, a soft guarantee means a lower probability
Host	A computer capable of communicating using the Internet protocols; includes "routers".
Ingress	The point where traffic <i>enters</i> the network or the domain. The sender is located at this point.
Ingress traffic	Traffic entering the network
Intrusive source	Source that modifies an existing traffic flow in some manner
Link	<p>A single link-level connection between two (or more) hosts. A network communications channel consisting of a circuit or transmission path and all related equipment between a sender and a receiver. Includes leased lines, ethernet, frame relay clouds, etc.</p> <p>Propagation time of a link. The time, in seconds, required by a single bit to travel from the output port on one Internet host across a single link to another Internet host.</p>
Measurement Agent (MA)	A logical entity of the DMA. The MA performs measurement function using active or passive probe. There is a 1-1 relation between the logical entity MA and the physical edge device.
Measurement Database	A database for storage of traffic flow measurement scenarios and results (metrics) inclusive timestamps for correlation analysis
Measurement Management Station	A logical entity to control the measurement agents.

Measurement Methodology	<p>A repeatable measurement technique used to derive one or more metrics of interest. It could be based on</p> <ul style="list-style-type: none">• Direct measurement of a performance metric using injected test traffic.• Projection of a metric from lower-level measurements.• Estimation of a constituent metric from a set of more aggregated measurements.• Estimation of a given metric at one time from a set of related metrics at other times.
Measurement Scenario	<p>One or more tests with well defined goals and performance metrics.</p>
Monitoring	<p>In AQUILA: the process of collecting statistics from passive probes.</p>
Natural Sources	<p>Those that generate traffic to accomplish some unit of work and are measured passively by a measurement device or probe.</p>
NetFlow Services	<p>NetFlow services are specific for collecting of traffic flows of MIB at Cisco routers and Catalyst 5000 series switches. Traffic flows is considered as unidirectional sequences of packets between a particular source device and destination device that share the same protocol and transport-layer information. Routers and switches identify flows by looking for the following fields within IP packets: Source IP address, Destination IP address, Source port number, Destination port number, Protocol type, Type of service (ToS); Input interface</p>
Network Provider	<p>An entity that controls a network infrastructure and offers network services. A provider can act as an access provider to prepare network access and/or as a service provider to offer network services with a specific behaviour, and perhaps to charge and account them</p>
Network Resource	<p>The capacities of a network infrastructure to be shared between several utilisation. Main resources are bandwidth of links and buffers within routers, for example.</p>

Network Service	A product that a network provider offers to its customer. In detail, it describes how customer's traffic is handled across the network as it is implemented by one or more traffic classes. Usually, there will be a set of pre-defined services but also the possibility to request for special parameters.
One-way Delay	Also known as <i>latency</i> ; refers to the interval between transmitting and receiving packets between two reference points.
One-way Delay variation	Also called <i>jitter</i> ; refers to the variation in time duration between two or more consecutively received packets taking the same route.
Packet Loss	The rate at which packets are discarded during transfer through a network; packet loss typically results from congestion.
Passive Probe	A probe, which non-intrusively listens to packets flowing across the 'wire' or monitors request/responses on a client or server, and provides a performance monitoring function based upon its observations.
Path	A sequence of the form $\langle h_0, l_1, h_1, \dots, l_n, h_n \rangle$, where $n \geq 0$, each h_i is a host, each l_i is a link between h_{i-1} and h_i , each $h_{i-1} \dots h_i$ is a router. A pair $\langle l_i, h_i \rangle$ is termed a 'hop'. In an appropriate operational configuration, the links and routers in the path facilitate network-layer communication of packets from h_0 to h_n . Note that path is a unidirectional concept.
Path Digest	A sequence of the form $\langle h_0, e_1, C_1, \dots, e_n, h_n \rangle$, where $n \geq 0$, h_0 and h_n are hosts, each $e_1 \dots e_n$ is an exchange, and each $C_1 \dots C_{n-1}$ is a cloud subpath.
Per Hop Behaviour (PHB)	The forwarding treatment given to a specific class of traffic, based on criteria defined in the DiffServ field. Routers and switches use PHBs to determine priorities for servicing various traffic flows [Star00]. There are currently two standard PHBs defined by the IETF: Expedited Forwarding (EF) [RFC2598] and Assured Forwarding (AF) [RFC2597].
Performance Metric	A carefully specified quantity defined related to the performance of some aspect of an Internet service
Performance Monitoring	The act of monitoring traffic for the purpose of evaluating a statistic of a metric related to the performance of the system. A performance monitoring system is comprised of a) traffic generators, b) measurement, c) data reduction, and d) reporting.

Performance Validation	Process of checking the expected performance metrics for a given measurement scenario
Poisson sampling	if $G(t)$ is an exponential distribution with rate λ ($G(t) = 1 - e^{-\lambda t}$), then the arrival of new samples <i>cannot</i> be predicted (the sampling is unbiased).
Policy (QoS Policy)	<p>The binding of traffic recognition and registration profiles to specific network behaviours including, though not exclusive to:</p> <p>Admittance/denial of identified traffic getting anything better than best-effort QoS.</p> <p>Simple prioritisation or specific bandwidth reservation for identified flows or aggregated flows. [Cisco99]</p>
Policy Control	The process of determining whether access to a particular resource should be granted.
Probe	is a device or embedded software program that is placed in the data flow path or on a client or server to provide a performance monitoring function.
Probing	In AQUILA: collecting QoS parameters from a defined path using active probes.
Quality of Service (QoS)	<p>An overall measurement of the service quality based on certain key parameters [Black99]. QoS can be seen on two levels: In terms of end-user applications, it is expected to get the data in a sufficient manner with minimal delay or latency, minimal variations of delay (jitter), and error freeness.</p> <p>In terms of a network, QoS is used to describe a connection, on which data are transmitted in a manner better than best-effort by using the network resources efficiently, and with minimal data loss.</p> <p>QoS Scenario. A use case where applications request for QoS in order to have a better service quality for their communication.</p> <p>Request. (QoS Request). An explicit demand for getting QoS from an infrastructure. Usually, signalling protocols such as RSVP are used for the request. However, requests could be based on APIs and CORBA as well.</p>

Reservation	<p>Part of a resource that has been dedicated for the use of a particular traffic type for a period of time through the application of policies. [Star00]</p> <p>Reservation Mode. The assignment of the requester role. Three modes can occur: sender-initiated (forward reservation), receiver-initiated (backward reservation), as well as third-party-initiated.</p> <p>Reservation Style. The amount of senders/receivers involved in a reservation. Generally, there are three types: point-to-point (p2p; - one sender, one receiver), point-to-anywhere (p2a; one sender, loads of receivers), and anywhere-to-point (a2p; loads of senders, one receiver).</p>
Resolution	<p>A clock's "resolution" is the smallest unit by which the clock's time is updated. It gives a lower bound on the clock's uncertainty.</p>
Resource Control Layer (RCL)	<p>An overlay network layer which monitors and controls the resources of the core DiffServ and access network as well as offers a QoS interface to the applications. The RCL consists of: ACAs, RCAs, and EATs.</p> <p>Resource Control Agent (RCA). A logical entity of the RCL. The RCA controls resources and distributes them to the ACAs.</p> <p>RCL Platform. Physical platform, on which one or several ACAs and/or RCAs are running. May be a separate hardware entity or integrated into a router.</p>
Route	<p>The path from host A to host B at a given time.</p> <p>Hop count of a route. The value 'n' of the route path.</p>
Router	<p>A host which facilitates network-level communication between hosts by forwarding IP packets.</p>
Router QoS Monitoring Tool	<p>In AQUILA: a tool capturing traffic statistics about traffic classes.</p>
Sample metric	<p>Metric derived from a given singleton metric by taking a number of distinct instances together. For example, we might define a sample metric of one-way delays from one host to another as an hour's worth of measurements, each made at Poisson intervals with a mean spacing of one second.</p>

Service Level Agreement (SLA)	A contract between a network provider and a customer defining provider responsibilities in terms of network services. In detail, a SLA includes QoS properties (throughput, loss rate, delays and jitter) of the network service and times of availability, method of measurement, consequences if network services aren't met or the here defined traffic levels are exceeded by the customer, and all costs involved. [D1101, Chapter 9.2, SLA]
Service Level Specification (SLS)	A set of parameters and their values which together define the service offered to a traffic stream by a DS domain. Specific term for DiffServ. [D1101, Chapter 9.2, SLA]
Session	Time during which an application uses a network with QoS. Session Characteristics. An end-user does have the possibility to individualise his/her applications in order to choose their session quality. For example: He/she can either choose between different pre-defined video qualities or directly set the parameters such as frame rate, picture size, etc. These characteristics have to be mapped into network services.
Singleton metric	A metric that is atomic. For example, a single instance of "bulk throughput capacity" from one host to another might be defined as a singleton metric, even though the instance involves measuring the timing of a number of Internet packets.
Statistical metric	A metric derived from a given sample metric by computing some statistic of the values defined by the singleton metric on the sample. For example, the mean of all the one-way delay values on the sample given above might be defined as a statistical metric.
Subpath	Within a given path, a subpath is any subsequence of the given path which is itself a path. (Thus, the first and last element of a subpath is a host.)
Synthetic flow generator	In AQUILA: Synthetic source traffic flow generator which is called also load generator.
Synthetic source	A device or an embedded software program which generates a data packet (or packets) and injects it (them) onto the path to a corresponding probe or existing server solely in support of a performance monitoring function. A synthetic source may talk intrusively to existing application servers.
Test	Specification of probes in a specific network, operating and end-system environment.

Throughput	The rate at which data is transmitted in a network.
Traffic Class	In terms of AQUILA the implementation of a network service, i.e. the network view on that product. A traffic class contains rules how to handle the traffic belonging to this class such as per-hop behaviour, rules for traffic conditioning as well as for admission control.
Traffic flow	Unidirectional sequences of packets between a particular source device and destination device which is assigned to a given traffic class.
Wire time	<p>For a given packet, the 'wire arrival time' of this packet at a host on a specific link is the first time at which any bit of the packet has appeared at the hosts observational position on the link.</p> <p>For a given packet, the 'wire exit time' of this packet at a host on a specific link is the first time at which all the bits of the packet have appeared at the hosts observational position on the link.</p>

9 References

- [BDTL96] A. Basso, I. Dalgic, F. A. Tobagi, C. J. van den Branden Lambrecht: Study of MPEG-2 Coding Performance based on a Perceptual Quality Metric. Proceedings of PCS 96, Melbourne, Australia. March 1996.
http://ltssg3.epfl.ch/pub_files/vdb/papers/pcs96.ps.gz
- [Black99] Black, Darryl P.: Building Switched Networks – Multilayer Switching, QoS, IP Multicast, Network Policy, and Service Level Agreements. Addison Wesley, 1999
- [BSU98] M. S. Borella, D. Swider, S. Uludag, and G. B. Brewster: Internet Packet Loss: Measurement and Implications for End-to-End QoS. Proceedings, International Conference on Parallel Processing, August 1998.
- [CAIDA] Cooperative Association for Internet Data Analysis (CAIDA). June 1999.
<http://www.caida.org>
- [CCMP00] J.P. Curtis, J.G. Clear, A.J. McGregor, W.M. Pearson: Measurement of Voice over IP Traffic. Proceedings of Workshop on Passive & Active Measurement, Dept. Computer Science, University of Waikato, New Zealand, April 2000. http://pam2000.cs.waikato.ac.nz/pdf_papers/pam2000c.pdf
- [CDG00] J. Clearly, S. Donnelly, I. Graham, A. McGregor, M. Pearson: Design Principles for Accurate Passive Measurement. Proceedings of Workshop on Passive & Active Measurement, Dept. Computer Science, University of Waikato, New Zealand, April 2000.
http://pam2000.cs.waikato.ac.nz/pdf_papers/P034.pdf
- [Cisco99] Cisco IOS™ Software: Quality of Service Solutions. Whitepaper,
http://www.cisco.com/warp/public/cc/cisco/mkt/ios/tech/tch/qosio_wp.htm
- [Claf99] K. Claffy: Internet measurement and data Analysis: topology, workload, performance and routing statistics. NAE 99 Workshop, 1999.
<http://www.caida.org/outreach/papers/Nae/>
- [CMP98] K. Claffy, G. Miller and K. Thompson: The Nature of the Beast: Recent Traffic Measurements from an Internet Backbone. INET'98; July 1998.
<http://www.caida.org/Papers/Inet98/index.html>
- [COLE-SSPM] R. G. Cole, R. Dietz, C. Kalbfleisch, D. Romascano: A Framework for Synthetic Sources for Performance Monitoring. June 2000.
<http://www.ietf.org/internet-drafts/draft-cole-sspm-00.txt>

- [Crash-me] web page for comparison of databases: <http://www.tcx.se/crash-me.html> or <http://www.mysql.com/crash-me-choose.htm>
- [D1101] IST-1999-10077-WP1.1-DTA-1101-PU-R/b0, AQUILA Analysis and Requirements Report
- [D1201] IST-1999-10077-WP1.2-SAG-1201-PU-O/b0, System architecture and specification for first trial
- [D1301] IST-1999-10077-WP1.3-COR-1301-PU-O/b0, Specification of traffic handling for the first trial
- [DIFFSERV-MIB] F. Baker, K. Chan, A. Smith: Management Information Base for the Differentiated Services Architecture. Internet Draft, July 2000. <http://www.ietf.org/internet-drafts/draft-ietf-diffserv-mib-04.txt>
- [DISMAN-RE-MOPSMIB] K. White: Definitions of Managed Objects for Remote Ping, Traceroute, and Lookup Operations. Internet Draft, March 2000. <http://www.ietf.org/internet-drafts/draft-ietf-disman-remops-mib-08.txt>
- [GDM98] I. D. Graham, S. F. Donnelly, S. Martin, J. Martens, and J. G. Cleary: Nonintrusive and Accurate Measurement of Unidirectional Delay and Delay Variation on the Internet. Proceedings of INET '98, July 1998.
- [GuHa00] H. Guy, R. Hatem: Statistics & Measurements: From Network Research to Network Operation. Proceedings of Workshop on Passive & Active Measurement, Dept. Computer Science, University of Waikato, New Zealand, April 2000. http://pam2000.cs.waikato.ac.nz/pdf_papers/P001.pdf
- [Horn00] M. Horneffer: Assessing Internet Performance Metrics Using Large Scale TCP SYN-based Measurements. Proceedings of Workshop on Passive & Active Measurement, Dept. Computer Science, University of Waikato, New Zealand, April 2000. http://pam2000.cs.waikato.ac.nz/pdf_papers/P010.pdf
- [HPH00] U. Hofmann, T. Pfeiffenberger, B. Hechenleitner: One-way Delay Measurements with CM Toolset. Proceedings of IEEE IPCCC 2000, February 2000.
- [IETF98] charter of the 43rd IETF Meeting in Florida 1998: 2.7.3 IP Performance Metrics. <http://www.ietf.org/proceedings/98dec/43rd-ietf-98dec-122.html>
- [IETF99] slides of the 44th IETF Meeting in Minneapolis: 2.7.3 IP Performance Metrics Minnesota 1999. <http://www.ietf.org/proceedings/99mar/slides/ippm-itu-99mar.pdf>
- [IPMP98] IP Measurement Protocol (IPMP). September 1998. <http://watt.nlanr.net/AMP/IPMP/>

- [IPPM-BTC-FRAME] M. Mathis, M. Allman: Empirical Bulk Transfer Capacity. Expired Internet Draft, October 1999.
<http://www.advanced.org/IPPM/docs/draft-ietf-ippm-btc-framework-02.txt>
- [IPPM-IPDV] C. Demichelis, P. Chimento: Instantaneous Packet Delay Variation Metric for IPPM. Internet Draft, July 2000.
<http://www.ietf.org/internet-drafts/draft-ietf-ippm-ipdv-05.txt>
- [IPPM-LOSS-PATTERN] R. Koodi, R. Ravikanth: One-way Loss Pattern Sample Metrics. Internet Draft, July 2000.
<http://www.ietf.org/internet-drafts/draft-ietf-ippm-loss-pattern-03.txt>
- [IPPM-NPMPS] V. Raisanen, G. Grotefeld: Network performance measurement for periodic streams. Internet Draft, July 2000,
<http://www.ietf.org/internet-drafts/draft-ietf-ippm-npmgs-02.txt>
- [IPPM-TRENO-BTC] M. Mathis: TReno Bulk transfer Capacity. Expired Internet Draft, February 1999. <http://www.advanced.org/IPPM/docs/draft-ietf-ippm-treno-btc-03.txt>
- [ITU380] New ITU-T Recommendation I.380 (approved text): Internet Protocol Data Communication Service - IP Packet Transfer and Availability Performance Parameters. February 1999 (I.380 will become Y.1540 in the ITU's IP Recommendations)
- [ITU861] ITU-T Recommendation P.861: Objective quality measurement of telephone-band (300-3400 Hz) speech codecs. February 1998.
- [KaZe99] S. Kalidindi, M.J. Zekausus: Surveyor: An Infrastructure for Internet Performance Measurements. June 1999.
<http://telesto.advanced.org/~kalidindi/papers/INET/inet99.html>
- [Lab99] C. Labovitz, et al.: The Internet Performance and Analysis Project (IPMA).
<http://www.merit.edu/ipma/>
- [LMH00] L. Lamont, I. Miloucheva, D. Hetzer: Performance Analysis of Streaming Audio Applications. Research Paper, Mai 2000
- [McBr00] T. McGregor, H.-W. Braun: Balancing Cost and Utility in Active Monitoring: The AMP example. INET 2000
- [MINC] Multicast-based Inference of Network-internal Characteristics (MINC)
<http://www.research.att.com/~duffield/minc/>
- [MYSQL] The MySQL web page. <http://www.mysql.com>
- [NIMI] The National Internet Measurement Infrastructure (NIMI) web page.
<http://www.psc.edu/networking/nimi/>

- [NLANR] NLANR's Active Measurement Program (AMP) web site.
<http://amp.nlanr.net>
- [OMKN00] K. Ohta, G. Mansfield, N. Kato, Y. Nemoto: Wide area fault detection by monitoring aggregated traffic. Proceedings of Workshop on Passive & Active Measurement, Dept. Computer Science, University of Waikato, New Zealand, April 2000. http://pam2000.cs.waikato.ac.nz/pdf_papers/P026.pdf
- [PAM00] V. Paxson, A.K. Adams, M. Mathis: Experiences with NIMI. Proceedings of Workshop on Passive & Active Measurement, Dept. Computer Science, University of Waikato, New Zealand, April 2000.
http://pam2000.cs.waikato.ac.nz/pdf_papers/P006.pdf
- [Remos] Remos: Resource Monitoring for network aware applications.
<http://www.cs.cmu.edu/~cmcl/remulac/remos.html>
- [RFC1305] D.L. Mills: Network Time Protocol (Version 3) Specification, Implementation and Analysis. RFC 1305, March 1992.
<http://www.ietf.org/rfc/rfc1305.txt>
- [RFC1349] P. Almquist: Type of Service in the Internet Protocol Suite, RFC1349, July 1992. <http://www.ietf.org/rfc/rfc1349.txt>
- [RFC1757] S. Waldbusser: Remote Network Monitoring Management Information Base. RFC 1757, February 1995. <http://www.ietf.org/rfc/rfc1757.txt>
- [RFC2021] S. Waldbusser: Remote Network Monitoring Management Information Base Version 2 using SMIV2. RFC 2021, January 1997.
<http://www.ietf.org/rfc/rfc2021.txt>
- [RFC2063] N. Brownlee, C. Mills, G. Ruth: Traffic Flow Measurement: Architecture. January 1997. <http://www.ietf.org/rfc/rfc2063.txt>
- [RFC2064] N. Brownlee: Traffic Flow Measurement: Meter MIB. January 1997.
<http://www.ietf.org/rfc/rfc2064.txt>
- [RFC2123] N. Brownlee: Traffic Flow Measurement: Experiences with NeTraMet. March 1997. <http://www.ietf.org/rfc/rfc2123.txt>
- [RFC2309] B. Branden et. al.: Recommendations on Queue Management and Congestion Avoidance in the Internet. RFC2309, April 1998.
<http://www.ietf.org/rfc/rfc2309.txt>
- [RFC2330] V. Paxson, G. Almes, J. Mahdavi, M. Mathis: Framework for IP Performance Metrics. RFC 2330. February 1998. <http://www.ietf.org/rfc/rfc2330.txt>

- [RFC2498] J. Mahdavi, V. Paxson: IPPM Metrics for Measuring Connectivity. RFC 2498, January 1999. <http://www.ietf.org/rfc/rfc2498.txt>
- [RFC2597] J. Heinanen et al.: Assured Forwarding PHB Group. RFC2597, June 1999. <http://www.ietf.org/rfc/rfc2597.txt>
- [RFC2598] V. Jacobson et al.: An Expedited Forwarding PHB. RFC 2598, June 1999. <http://www.ietf.org/rfc/rfc2598.txt>
- [RFC2678] J. Mahdavi and V. Paxson: IPPM Metrics for Measuring Connectivity. RFC 2678, September 1999. <http://www.ietf.org/rfc/rfc2678.txt>
- [RFC2679] G. Almes, S. Kalidindi, M. Zekauskas: A One-way Delay Metric for IPPM. RFC 2679, September 1999. <http://www.ietf.org/rfc/rfc2679.txt>
- [RFC2680] G. Almes, S. Kalidindi, M. Zekauskas: A One-way Packet Loss Metric for IPPM. RFC 2680, September 1999. <http://www.ietf.org/rfc/rfc2680.txt>
- [RFC2681] G. Almes, S. Kalidindi, M. Zekauskas: A Round-trip Delay Metric for IPPM. RFC 2681, September 1999. <http://www.ietf.org/rfc/rfc2681.txt>
- [RIPE] RIPE Network Coordination Centre, Test Traffic Project Homepage <http://www.ripe.net/test-traffic/index.html>.
- [RMONMIB-APMCAPS] A. Bierman: Application Performance Measurement Capabilities MIB. Internet Draft, March 2000. <http://www.ietf.org/internet-drafts/draft-bierman-rmonmib-apmcaps-00.txt>
- [RMONMIB-APMMIB] S. Waldbusser: Application performance measurement MIB. May 2000. <http://www.ietf.org/internet-drafts/draft-ietf-rmonmib-apm-mib-00.txt>
- [RMONMIB-DSMONMIB] A. Bierman: Remote Monitoring MIB Extensions for Differentiated Services. Internet Draft, June 2000. <http://www.ietf.org/internet-drafts/draft-ietf-rmonmib-dsmon-mib-02.txt>
- [RMONMIB-TPMMIB] Dietz, R.: Application Performance Measurement Framework Transport Performance Metrics MIB. Internet Draft, May 2000. <http://www.ietf.org/internet-drafts/draft-ietf-rmonmib-tpm-mib-00.txt>
- [Skitter] CAIDA's Skitter project web page. <http://www.caida.org/tools/measurement/skitter/>
- [SNMPCONF-PM] S. Waldbusser, J. Saperia, and T. Hongal: Policy Based Management MIB. Internet Draft, July 2000. <http://www.ietf.org/internet-drafts/draft-ietf-snmppconf-pm-02.txt>

- [SSK00] S. Seshan, M. Stemm and R. H. Katz: A Network Measurement Architecture for Adaptive Applications. Proceedings of IEEE Infocom 2000, Tel Aviv, Israel, March 2000
- [SSK97] S. Seshan, M. Stemm and R. H. Katz: SPAND: Shared Passive Network Performance Discovery, Proceedings of 1st Usenix Symposium on Internet Technologies and Systems (USITS '97), Monterey, CA, December 1997
- [SSK98] S. Seshan, M. Stemm, and R. H. Katz: Benefits of Transparent Content Negotiation in HTTP, Proceedings of IEEE Globecom '98 Internet Mini-Conference, Sydney, Australia, November 1998
- [Star00] Stardust.com: A Quality of Service glossary of terms, <http://stardust.com/qos/whitepapers/glossary.htm>
- [Stem99] M.R. Stemm: An Network Measurement Architecture for Adaptive Applications, Ph.D. Thesis, UC Berkeley, 1999
- [Surveyor] Advanced Network & Services and the Common Solutions Group (CSG)'s Surveyor project web page, June 1999. <http://www.advanced.org/surveyor/>
- [tcpdump] Network dumping tool: tcpdump. July 1998, Available via anonymous ftp at <ftp://ftp.ee.lbl.gov/tcpdump.tar.Z>
- [traceroute] Network Route Trace Utility: Traceroute. June 1997, Available via anonymous ftp at <ftp://ftp.ee.lbl.gov/traceroute.tar.Z>
- [Treno] Traceroute Reno (TReno) network measurement tool. http://www.psc.edu/networking/treno_info.html, June 1999.
- [TTA00] Y. Tobe, Y. Tamura, H. Aida: Detection of Change in One-Way Delay for Analyzing the Path Status. Proceedings of Workshop on Passive & Active Measurement, Dept. Computer Science, University of Waikato, New Zealand, April 2000. http://pam2000.cs.waikato.ac.nz/pdf_papers/P009.pdf
- [WAND] I. D. Graham, et al.: Waikato Applied Network Dynamics (WAND) Project homepage, 1998. <http://atm.cs.waikato.ac.nz/wand/>
- [YMKT99] M. Yajnik, S. Moon, J. Kurose, D. Towsley: Measurement and Modeling of the Temporal Dependence in Packet Loss. Proceedings of IEEE Infocom 1999, March 1999. ftp://gaia.cs.umass.edu/pub/Yajn99_Meas.ps.Z
- [ZPS00] Y. Zhang, V. Paxson, S. Shenker: The Stationarity of Internet Path Properties: Rounting, Loss, and Throughput. ACIRI Technical Report, May 2000. <http://www.cs.cornell.edu/yzhang/papers/station-tr00.ps.Z>