



Advanced architecture for INTER-domain quality of service MONitoring, modelling and visualisation



INTERMON-IST-2001-34123

INTERMON Deliverable 10
“Prototype of inter-domain data base with policy-controlled data collection ”

WORK-PACKAGE NO. / TITLE	WP 4 Prototype of inter-domain QoS monitoring and structure discovery components
DELIVERABLE TYPE	Internal, work in progress (change to public when finished)
PLANNED ISSUING DATE	2003-03-31
DISTRIBUTION	INTERMON Consortium (all INTERMON members)
DOCUMENT IDENTIFIER	IM-WP4-V105-SR-ANC-Deliverable-D10
FILENAME	im-wp4-v105-sr-anc-d10.doc
VERSION	V1.05
EDITOR/AUTHOR	Salzburg Research
CONTACT PERSON(S)	Ilka Miloucheva < ilka.miloucheva@salzburgresearch.at >

Consortium partners	Author(s)
TID	Pedro A. Aranda Gutiérrez, Cristina Soto
NEC	Sandra Tartarelli
TILAB	Maurizio Beoni
SR-ANC	Ilka Miloucheva
SAG	Ali Nassri

Change History

V100	Fist proposal for table of contents (SR-ANC)
V101	First Version (SR-ANC)
V102	Included Topology databases (TID)
V103	IPFIX Flow data base (NEC)
V104	Network performance data base (TILAB)
V105	Executive summary, updates common chapters, conclusions (SR)
V106	Network measurement data base – campaign enhancement (TILAB)

Content

EXECUTIVE SUMMARY	4
1 INTERMON POLICY CONTROLLED DATA COLLECTION	5
2 INTERMON DATA BASE	6
2.1 Design enhancements	6
2.2 INTERMON data base Implementation – MySQL	7
2.3 Data access procedures	7
3 INTERMON DATA BASE ELEMENTS IN MYSQL	7
3.1 IM User	7
3.2 Network elements	8
3.3 Traceroute topology data	8
3.3.1 Traceroute topology scenario	8
3.3.2 Traceroute call	8
3.3.3 Hop measurement	9
3.4 MRT Topology data	9
3.4.1 MRT Topology scenario	9
3.4.2 MRT Topology tables	9
3.4.2.1 Update independent table	10
3.4.2.2 Update dependent table	10
3.5 IPFIX traffic flow	11
3.6 Network measurement data	12
3.6.1 Campaign	12
3.6.2 Campaign Configuration	12
3.6.3 Node Interface resources	12
3.6.4 Global variables collection	13
3.6.5 SAA variables collection	13
3.6.6 Interface Variable collection	14
3.7 QoS parameter measurement data	15
3.7.1 QoS parameter measurement scenario	16
3.7.2 Packet	18
3.7.3 QoS Statistics	18
3.7.4 Modelling entity	20
4 CONCLUSIONS	20
5 REFERENCES	20
6 LIST OF FIGURES	21

7 LIST OF TABLES	21
8 APPENDIX	21
8.1 MYSQL Column Types	21

Executive summary

This deliverable describes implementation issues of the INTERMON data base using MySQL technology.

The INTERMON policy controlled data collection is explained from implementation point of view considering configurable functions for XML based data access.

The design of the INTERMON data base considering deliverable 2 and 5 is enhanced in order to support more efficient distributed data base concepts and partitioning for enhanced performance.

New data sources are included in the INTERMON data base:

- Network performance data (TILAB tool).
- MRT topology.
- Traceroute topology.

Other data base sources are finalised in design and implementation:

- IPFIX traffic flow.
- Active QoS measurement data.

1 INTERMON policy controlled data collection

The INTERMON architecture is based on XML access to data base sources using configurable data access functions mapping XML to physical data base presentations.

Policy based collection in INTERMON based on integrated data collection in a data base(s) by different tools for solving of some QoS analysis task.

The main data source for INTERMON architecture is the INTERMON data base designed to integrate and relate the data of different tools. The actual stage of data source integration in the INTERMON data base by different tools is shown in figure 1.

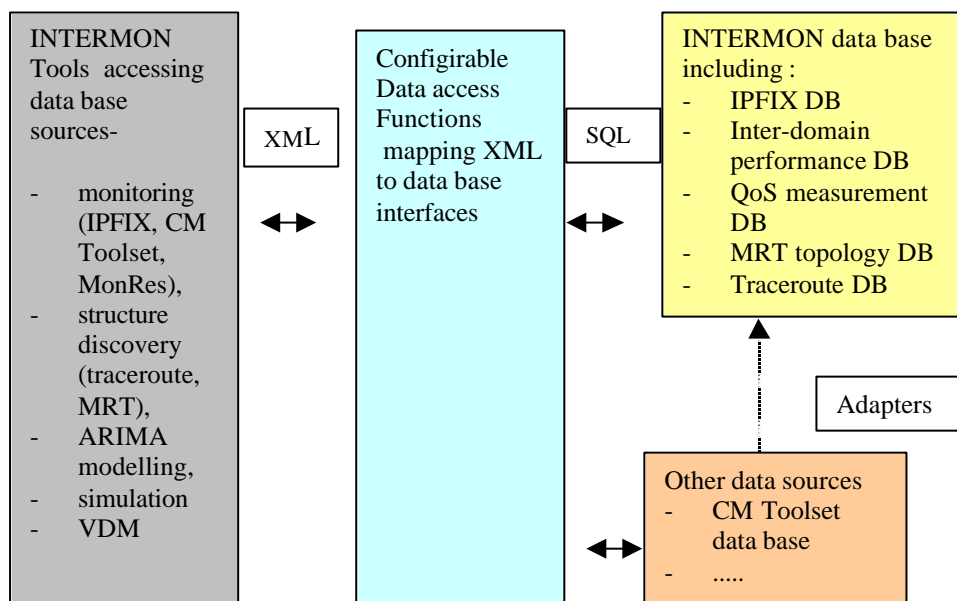


Figure 1: Data access relevant component in the INTERMON architecture

The relevant data base and access components for implementation of policy controlled data collection in INTERMON (abstracting from global/local controller which are focus of the architecture deliverable) are described in figure 1:

- INTERMON data base.
- Additional data sources and adapters (for instance CM Toolset DB).
- Configurable function to access data (XML to data base mapping).

The configurable functions to map XML to data base physical presentations are required, because currently only a few commercial data bases [XML DB] are based on XML technology and the selected MySQL technology for INTERMON data base implementation does not provide XML interface [D5].

For more flexibility and overhead reducing as well as real time operation, direct access to the additional sources using the configurable data access functions is allowed. Optionally adapters could be used to transform additional sources to INTERMON data base.

2 INTERMON Data base

2.1 Design enhancements

The INTERMON data base design of Deliverable 2 [D2] and 5 [D5] is simplified based on the knowledge on more details on the tools and applications using data base included in the INTERMON toolkit.

The INTERMON data base at this stage integrates measured data and scenarios from following tools:

- IPFIX traffic meter.
- MRT topology discovery.
- Traceroute topology discovery (CM Toolset Path Analyser).
- TILAB Network performance monitoring.
- Active QoS measurement data (CM Toolset QoS monitoring).

Figure 2 shows the structure of INTERMON data base considering the integration of data sources of different tools.

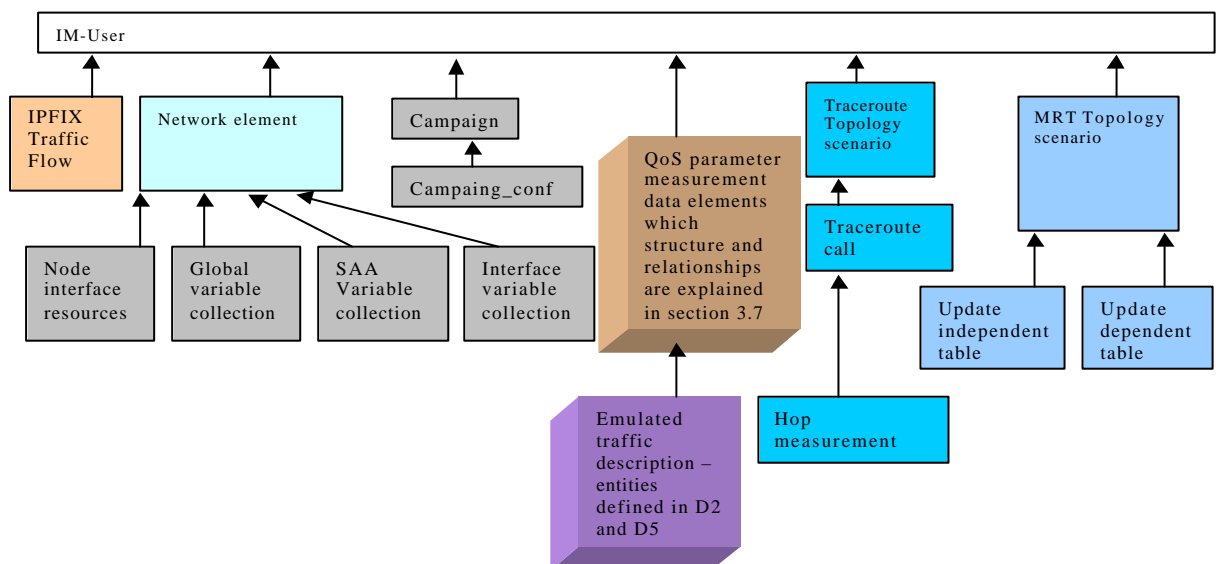


Figure 2: INTERMON data base structure

Future work is to consider simulation scenarios in the data base which will be defined at the integration stage due the progress of the simulation and modelling work.

The INTERMON data base design is simplified. It allows partitioning of the INTERMON data base in parts (IPFIX, Topology, QoS measurement, Network measurements) for enhanced performance in case of very large data volumes.

2.2 INTERMON data base Implementation – MySQL

The INTERMON data base version is implemented in MySQL and is publicly available <http://212.183.10.180/phpmyadmin/>. At the same http address also additional data base sources of the INTERMON Toolkit are contained for testing purposes, like the CM Toolset.

Considering distributed data base optimisations, MySQL supports replication, which is the process of copying and maintaining database objects in multiple databases that make up a distributed database system. Replication can improve the performance and protect the availability of applications because alternate data access options exist. Replication works fine in the last stable MySQL version.

It is master-slave replication using binary log of operations on the server side. It is possible to build star or chain type structures.

2.3 Data access procedures

The data access procedures represent interfaces from tools to data base for reading/storing data transforming SQL output in XML presentations. The XML presentations should be mapped into the MySQL column data types as listed in the appendix 4.1.

The data access procedures should be written for each particular applications and tool function accessing the data base (reading/writing) and are part of the integration deliverables D16, integration of modelling and simulation tools D11 and D12 visual data mining implementation.

3 INTERMON data base elements in MySQL

The focus is to explain the new and changed INTERMON data base elements prototypes defined in MySQL. The INTERMON prototype data base is available at web where all elements could be obtained.

3.1 IM User

Element	Type	Attribute	Key	Description
Userid	INTEGER	UNSIGNED	PK	ID of the user
Username	VARCHAR (50)			Name of users
Userpasswd	VARCHAR (50)			Password of user
Email	VARCHAR (50)			Email of user
Task	CHAR (1)			Task identification: - ISP operator - End-user
ISP	VARCHAR (50)			ISP name
User descr	VARCHAR (255)			A short description of a user

Table 1: User

3.2 Network elements

The description of the “network element” are shown in the next table:

Element	Data type	Attribute	Key	Description
NE ident	INTEGER	UNSIGNED	PK	INTERMON Network Element identifier
NE type INTERMON	INTEGER	UNSIGNED		Type: border router, core router, end system, exchange point, access router
Node Type Mon Res	Varchar(50)			MonRes node type (router, ascend, rascisco, workstation)
IP_VERSION	ENUM ('4','6')			IP version
IPV4_ADDR	INTEGER	UNSIGNED		IPv4 source address
IPV6_ADDR	VARCHAR(40)			IPv6 source address
SNMP version	VARCHAR (5)			SNMP version
SNMP community	VARCHAR (20)			SNMP community

Table 2: Network element

3.3 Traceroute topology data

3.3.1 Traceroute topology scenario

The attributes of the “traceroute topology scenario” are shown in the next table:

Element	Data type	Attributes	Key	Description
Scenario ident	INTEGER	UNSIGNED	PK	ID of traceroute scenario
Start	DATETIME			Traceroute start
Stop	DATETIME			Traceroute stop
Sending NE	INTEGER	UNSIGNED		Sending NE ident
Receiving NE	INTEGER	UNSIGNED		Receiving NE ident
Monitoring interval	INTEGER	UNSIGNED		Monitoring interval
TOS	INTEGER	UNSIGNED		Field to characterise the service class (DiffServ class)

Table 3: Traceroute topology scenario

3.3.2 Traceroute call

The attributes of the “traceroute call” are shown in the next table:

Element	Data type	Attribute	Key	Description
Call sequence number	INTEGER	UNSIGNED	PK	Sequence number of traceroute call
Start	DATETIME			Call start

Number hops	INTEGER	UNSIGNED		Number hops of traceroute call
-------------	---------	----------	--	--------------------------------

Table 4: Traceroute call

3.3.3 Hop measurement

The attributes of the hop measurement entity are shown in the next table:

Element	Data type	Attribute	Key	Description
Hop seq number	INTEGER	UNSIGNED	PK	Seq Number of Hop
NE Ident	INTEGER	UNSIGNED		NE identification
First RTT	TIME			First hop ping
Second RTT	TIME			Second hop ping
Third RTT	TIME			Third hop ping

Table 5: Hop measurement

3.4 MRT Topology data

3.4.1 MRT Topology scenario

This table holds one entry for each instance of a topology detection toolkit that is currently running:

<i>Element</i>	<i>Type</i>	<i>Attribute</i>	<i>Key</i>	<i>Description</i>
Id	INTEGER	UNSIGNED	PK	unique scenario ID
Ipaddr	INTEGER	UNSIGNED		IP-address of the Topology Detector
Name	TINYTEXT			Some instance name
Start time	DATETIME			Start time for the topology detector
Stop time	DATETIME			Stop time for the topology detector

Table 6: MRT Topology scenario

3.4.2 MRT Topology tables

We have a common part for all topology records which will be stored in one table and an update dependent part which will only appear for some records and which will be stored in a separate table. The main reason for this is the optimisation:

- Optimisation of the storage space by avoiding allocation space, which will not be used and
- Optimisation of the programming effort by uniform query structure.

3.4.2.1 Update independent table

This table holds the update independent data, i.e. data that will always be generated by the topology probe.

Element	Data type	Attribute	Key	Description
Id	INTEGER	UNSIGNED	FK	unique scenario id
Lastexport_Id	INTEGER	UNSIGNED	PK	ID of the last export data
Timestamp	TIME			timestamp of the update
Protocol	VARCHAR(10)			Protocol (BGP,BGP4MP,TABLE_DUMP)
PrefixIP	INTEGER	UNSIGNED		the network base address of the update
MaskIP	INTEGER	UNSIGNED		The network mask of the update
PeerIP	INTEGER	UNSIGNED		IP address of the peer which generated the update
PeerAS	SMALLINT	UNSIGNED		ASID of the peer which generated the update
Type	CHAR			type of record (A advertisement, B routing table, W withdrawal)
Bgpinfo_Id	INTEGER	UNSIGNED		ID of the BGP_INFO record if update type is 'A' or 'B' Or 0 if update type was 'W'

Table 7:Update independent table

3.4.2.2 Update dependent table

This table holds the update dependent BGP-4 routing information. It will only be present for network advertisements and BGP-4 table dumps.

Element	Type	Attribute	Key	Description
Id	INTEGER	UNSIGNED	FK	unique scenario id
Record_Id	INTEGER	UNSIGNED	PK	ID of the RECORDS data
Aspath	VARCHAR(1024)			AS_PATH attribute
Nexthop	INTEGER	UNSIGNED		IP address of the next hop for the network prefix

Table 8: Update dependent table

3.5 IPFIX traffic flow

Element	Data type	Attribute	Key	Description
USER_ID	TINYINT	UNSIGNED	FK	User ID
TASK_ID	TINYINT	UNSIGNED	PK	Task ID
MEAS_POINT	VARCHAR(40)		IK	Measurement point address (able to support IPv6 addresses)
IP_VERSION	ENUM ('4','6')			IP version
IPV4_SRC_ADDR	INTEGER	UNSIGNED		IPv4 source address
IPV4_DEST_ADDR	INTEGER	UNSIGNED		IPv4 destination address
IPV6_SRC_ADDR	VARCHAR(40)			IPv6 source address
IPV6_DEST_ADDR	VARCHAR(40)			IPv6 destination address
PROTO	ENUM ('1','2','4','6','17','41','58')			IP protocol identifier number
L4_SRC_PORT	SMALLINT	UNSIGNED		Layer 4 source port value (The IPFIX meter supports more complex data types but database limitations seem to impose to have only a single value specification here)
L4_DEST_PORT	SMALLINT	UNSIGNED		Layer 4 destination port value (The IPFIX meter supports more complex data types but database limitations seem to impose to have only a single value specification here)
COS	TINYINT	UNSIGNED		Class Of Service value (The IPFIX meter supports more complex data types but database limitations seem to impose to have only a single value specification here)
TIME_FIRST_PACKET_OFFSET	INTEGER	UNSIGNED		Time offset of the first packet with respect of the reference time
TIME_LAST_PACKET_OFFSET	INTEGER	UNSIGNED		Time offset of the first packet with respect of the reference time
NUM_BYTES	BIGINT	UNSIGNED		Number of bytes
NUM_PACKETS	BIGINT	UNSIGNED		Number of packets
Reference Time	DATETIME			MySQL displays DATETIME values in 'YYYY-MM-DD HH:MM:SS' format while IPFIX meter tool is able to report DATETIME values in 'YYYY-MM-DD HH:MM:SS±HH:MM' where ±HH:MM is an optional offset from UTC.

Table 9: IPFIX traffic flow table

3.6 Network measurement data

3.6.1 Campaign

The campaign is defined to get measures about a certain number of NE involved in the measures collection campaign. Campaign could be of course task but it's also measures logical grouping. The campaign identifier is a unique tag present in every collection request that different INTERMON users performs.

Element	Type	Attribute	Key	Description
CAMP_ID	INTEGER	UNSIGNED	PK	Campaign identifier
User_ID	INTEGER	UNSIGNED	FK	User Identifier
Campaign Descr	VARCHAR (50)			Description of campaign

Table 10: Campaign

3.6.2 Campaign Configuration

The campaign configuration specifies the network elements included in the campaign.

Element	Type	Attribute	Key	Description
CAMP_CONF	INTEGER	UNSIGNED	PK	Campaign configuration identifier
NE_ID	INTEGER	UNSIGNED	FK	Identifier of NE included in campaign
User_ID	INTEGER	UNSIGNED	FK	User Identifier

Table 11: Campaign configuration

3.6.3 Node Interface resources

This table describes interfaces static resources information of the managed elements.

Element	Type	Attribute	Key	Description
IF_ID	INTEGER	UNSIGNED	PK	Interface identifier
NE ident	INTEGER	UNSIGNED	FK	Network Element identifier
IF_NAME	VARCHAR (20)			Interface Name
IF_TYPE	VARCHAR (20)			Interface Type (VC_ATM, ATM, E1, Ethernet, FastEthernet, EthernetIPv6, SWLAN, FDDI, PVC_FR, PortChannel, Serial, Tunnel)
IF_STATUS	VARCHAR (20)			Interface Status

IF_BANDWIDTH	INTEGER	UNSIGNED		Interface Bandwidth
IF_DESCR	VARCHAR (150)			Interface operator description

Table 12: Node Interface resources

3.6.4 Global variables collection

In this table LC will store working state variables of the managed host collected from the tool management workstation

Element	Type	Attribute	Key	Description
GV_COL_ID	INTEGER	UNSIGNED	PK	Global Variable sample identifier
COL_TIME	DATETIME			Time of the sample
NE Ident	INTEGER	UNSIGNED	FK	
CAMPAIGN_ID	INTEGER	UNSIGNED	FK	Campaign id
AvgRTD	DOUBLE			Node RTD
Availability	DOUBLE			Node availability
AvgBusy5	DOUBLE			5 minutes windowed average of CPU usage
FreeMem	INTEGER	UNSIGNED		Free memory
DeadLanModem	DOUBLE			Nuber of Lan modem interface down
SuspectLanMondem	DOUBLE			Number of Lan Modem Interfaces in suspected state
CallCurrentAnalogIncoming	DOUBLE			Number of PSTN dial-up calls
CallCurrentDigitalIncoming	DOUBLE			Number of ISDN calls
WanAvailableChannels	INTEGER	UNSIGNED		Number of WAN available channels

Table 13: Global variables collection

3.6.5 SAA variables collection

The table contains samples related to active distribute SAA measures. Depending on performance issues the table can be normalized for each SAA measure type.

Element	Type	Attribute	Key	Description
SAA_COL_ID	INTEGER	UNSIGNED	PK	SAA sample identifier
COL_TIME	DATETIME			Time of the sample
CAMPAIGN_ID	INTEGER	UNSIGNED	FK	Campaign ID
NE Ident	INTEGER	UNSIGNED	FK	
MEAS_TYPE	VARCHAR (50)			SAA measure type

SAA_MEAS_ID	INTEGER	UNSIGNED		SAA measure configuration ID
MEAS_NAME	VARCHAR (50)			SAA measure name
SAAAvailabilityEcho	DOUBLE			
SAAMaxRTDEcho	DOUBLE			
SAAMinRTDEcho	DOUBLE			
SAAAvgRTDEcho	DOUBLE			
SAA%PosJitterSD	DOUBLE			Percentage of positive jitter Source->Destination
SAAMinPosJitterSD	DOUBLE			
SAAMaxPosJitterSD	DOUBLE			Number of ISDN calls
SAAAvgPosJitterSD	DOUBLE			
SAAVarPosJitterSD	DOUBLE			Jitter Variance S-> D
SAA%PosJitterSD	DOUBLE			Percent of positive jitter Destination ->Source
SAAMinPosJitterDS	DOUBLE			
SAAMaxPosJitterDS	DOUBLE			
SAAAvgPosJitterDS	DOUBLE			
SAAVarPosJitterDS	DOUBLE			Jitter Variance D->S
SAAAvailabilityHttpGet	DOUBLE			HTTP GET Availability
SAAMinRTDHttpGet	DOUBLE			
SAAMaxRTDHttpGet	DOUBLE			
SAAMinRTDHttpGet	DOUBLE			
SAAAvgRTDHttpGet	DOUBLE			
SAAConnectionHttpGet	DOUBLE			HTTP GET TCP connection time
SAATransactionHttpGet	DOUBLE			HTTP GET TCP total transaction time
SAAAvailabilityHttpRaw	DOUBLE			HTTP RAW Availability
SAAMinRTDHttpRaw	DOUBLE			
SAAMaxRTDHttpRaw	DOUBLE			
SAAMinRTDHttpRaw	DOUBLE			
SAAAvgRTDHttpRaw	DOUBLE			
SAAConnectionHttpRaw	DOUBLE			HTTP GET TCP connection time
SAATransactionHttpRaw	DOUBLE			HTTP GET TCP total transaction time

Table 14: SAA variables collection

3.6.6 Interface Variable collection

Element	Type	Attribute	Key	Description
IF_COL_ID	INTEGER	UNSIGNED	PK	Global Variable sample

				identifier
COL_TIME	DATETIME			Time of the sample
CAMPAIGN_ID	INTEGER	UNSIGNED	FK	Campaign id
NE Ident	INTEGER	UNSIGNED	FK	NE identifier
IF_NAME	VARCHAR(50)			Type of collected Interface
IF_TYPE	VARCHAR(30)			MonRes IF type
IfStatus	VARCHAR(10)			Interface status
If%inErrors	DOUBLE			
If%outErrors	INTEGER	UNSIGNED		
If%inDiscard	DOUBLE			
If%outDiscard	DOUBLE			
If%util_IN	DOUBLE			
If%util_OUT	DOUBLE			
IfInOctects	INTEGER	UNSIGNED		
IfOutOctects	INTEGER	UNSIGNED		
IfHCinOctects	BIGINT	UNSIGNED		
IfHCoutOctects	BIGINT	UNSIGNED		

Table 15: Interface variables collection

3.7 QoS parameter measurement data

In respect to D2, the active QoSP parameter measurement data base is simplified. Only one element describes the active probing QoS measurement scenario that includes the source and destination network element (NE) identifiers. This decision is done for optimised DB design – reducing the number of elements.

The QoS measurement scenario entity specifies the QoS parameter measurement and modelling options concerning:

- Source / Dest NE ident.
- Start and end time for measurement.
- Measurement aggregates defining the intervals for which measurement statistics are obtained.
- Measured active flow (flow ident, flow group, flow traffic state, flow characteristics options, multiplexing, etc).
- Options for statistics and modelling reports: QoS measurement options, QoS parameter statistics options, time aggregate option, histogram options, correlation options, modelling options.

The related entities to the QoS measurement scenario are shown in the next figure:

- The flow traffic model of the QoS measurement scenario contained in the emulated traffic description data base part (D2)
- Measured packets for a given aggregate of the QoS measurement scenario
- QoSP statistics calculated for a given aggregate of QoS measurement scenario
- QoSP Models describing forecasting based on QoS measurement statistics.

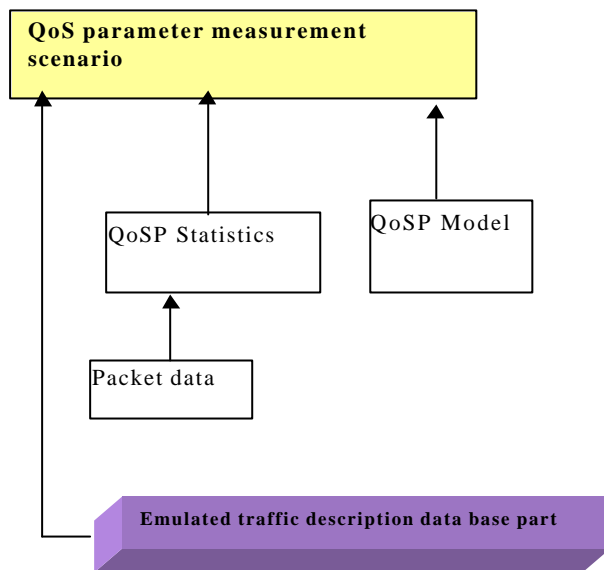


Figure 3: QoS measurement data elements and relationships

3.7.1 QoS parameter measurement scenario

Element	Data type	Attribute	Key	Description
QoS measurement scenario ident	BIGINT	UNSIGNED	PK	ID of the QoS measurement scenario including the QoS scenario ID in the first part and flow number of the scenario in the second integer part
Flow traffic id	INTEGER	UNSIGNED	FK	ID of the flow traffic (emulated traffic data base)
Flow group	INTEGER	UNSIGNED		Flow group defines traffic class
Flow state	INTEGER	UNSIGNED		Flow state
Flow option	INTEGER	UNSIGNED		1-unidirectional flow 2-bidirectional flow
Multiplex option	SMALLINT	UNSIGNED		Number flows which are multiplexed in one direction
NE source	INTEGER	UNSIGNED	FK	Source NE identification
NE dest	INTEGRE	UNSIGNED	FK	Destination NE identification
Port	SMALLINT	UNSIGNED		Port number.
QoS options	TINYINT	UNSIGNED		Defines how the ToS byte at the sender should be set, default is 0
Start measur. Time	DATETIME			Measurement start time
End measur. Time	DATETIME			Measurement ending time

Measurement frequency option	INTEGER	UNISGNED		-Regular -Irregular
Measurement frequency interval	INTEGER	UNSIGNED		Interval of execution of measurement scenarios
Measurement frequency interval unit	INTEGER	UNSIGNED		Type of measurement frequency interval <ul style="list-style-type: none"> - 1. Second - 2- minute - 3- hour - 4- day - 5- week - 6- month - 7- year
Measurement frequency number	INTEGER	UNSIGNED		Number of measurement frequencies
Measurement aggregation interval	INTEGER	UNSIGNED		The value of measurement aggregation interval
Measurement aggregation interval unit	INTEGER	UNSIGNED		Type of measurement aggregation interval <ul style="list-style-type: none"> - 1. second - 2- minute - 3- hour - 4- day - 5- week - 6- month - 7- year
Statistics aggregate Options	SMALLINT	UNSIGNED		Options for measurement statistics and modelling reports aggregation <ul style="list-style-type: none"> - 1- hour - 2 - day - 4 - week - 8 -month - 16 - year
QOSP measurement options	SMALLINT	USINGNED		Specification of precision and other options for QOSP measurement <ul style="list-style-type: none"> - 1- GPS usage
QoSP statistics options	SMALLINT	USINGNED		Options specifying the QOSP parameters measured by the active inter-domain probing tool <ul style="list-style-type: none"> - 1- end-to-end delay - 2- packet loss - 4- delay variation

				- 8- throughput Measurement of more than one parameters is specified by the sum of option flags
Reservation option	SMALLINT	UNSIGNED		Defines options for resource reservation of measurement scenario - per flow - per multiplexed flows
Bandwidth reservation	INTEGER	UNSIGNED		Bandwidth reserved for inter-domain measurement scenario

Table 16: QoS measurement scenario

3.7.2 Packet

This entity stores the sending and receiving times and the state and length of every single packet. The timestamps have a precision of micro-seconds.

The attributes of the “end-to-end packet aggregate” entity are shown in the next table:

Element	Data type	Attribute	Key	Description
Aggregate ident	INTEGER	UNSIGNED	FK	ID of the measured aggregate to which this packet belongs to
Pktnr	INTEGER	UNSIGNED	PK	The number of the packet in the aggregate
Sndtime	TIMESTAMP			Packet send time
Rcvtime	TIMESTAMP			Packet receive time
Pktstate	TINYINT	UNSIGNED		Bit oriented field for the state of the packet
Pktlength	SMALLINT	UNSIGNED		The length of the packet in bytes
Pktdescr	TINYINT	UNSIGNED		DiffServ code point of the packet at the receiver (AQUILA DB compatibility)

Table 17: Packet entity

3.7.3 QoSP Statistics

This entity describes QoSP measurement statistics for the for a given aggregate.

The attributes of the QoSP Statistics entity are shown in the next table:

Element	Data type	Attribute	Key	Description
Aggregate sequence number	INTEGER	UNSIGNED	PK	Aggregate sequence number identifier is uniquely identifying the aggregate packet set and interval for

				the measured statistics.
Aggregated packet number	INTEGER	UNSIGNED		Number of packets which are aggregated to this aggregate set
MIN_DELAY	INTEGER	UNSIGNED		Minimum delay
MEAN_DELAY	INTEGER	UNSIGNED		Mean delay
MAX_DELAY	INTEGER	UNSIGNED		Maximum delay
Lastpkt time	TIMESTAMP			The recv time of the last packet of this aggregate set
Errorstate	TINYINT	UNSIGNED		Error state of the part of the flow to which this aggregate result belong
Throughput	BIGINT	UNSIGNED		The throughput of the flow in bit/sec (according to RFC 2330 metric prefixes) for the aggregate
IPDV	BIGINT	UNSIGNED		IP delay variation in ms
Pktloss	INTEGER	UNSIGNED		The number of lost packets for the aggregate
Burstyloss	INTEGER	UNSIGNED		The number of packets which are dropped consecutive for the aggregate
Maxlossdistance	INTEGER	UNSIGNED		The number of packets, which are successive transmitted without packet loss (= distance between loss packets) for the aggregate
Numloss periods	INTEGER	UNSIGNED		The number of periods when packet loss occurs
Duration	INTEGER	UNSIGNED		The measured duration of the flow. This value is end minus start time
First packet send time	TIMESTAMP			The send time of the first packet of the aggregate
Connsetup time_s2r	INTEGER	UNSIGNED		Connection set-up time (in milli-seconds)
Reservationsetup time	INTEGER	UNSIGNED		Reservation set-up time (in milli-seconds)

Table 18: QoSP statistics entity

3.7.4 Modelling entity

Element	Data type	Attribute	Key	Description
Model_id	INTEGER	UNSIGNED	PK	Identification of model
QoSSP_ident	INTEGER	UNSIGNED		"QoSSP identifier" is identifying the QoSSP: <ul style="list-style-type: none"> - 1-end-to-end delay min - 2 -end-to-end delay mean - 3 -end-to-end delay max - 4- byte loss - 5- ipdv (IETF draft) - 6- throughput
<i>P</i>	INTEGER	UNSIGNED		order of autoregressive operator
<i>D</i>	FLOAT	FLOAT		the number of differences
<i>Q</i>	INTEGER	UNSIGNED		order of the moving average operator
<i>Period</i>	INTEGER	UNSIGNED		the period of ARIMA operators (for modeling seasonality)
<i>ar</i>	LIST	LIST		vector of autoregressive coefficients (length <i>p</i>)
<i>ma</i>	LIST	LIST		vector of moving average coefficients (length <i>q</i>)

Table 19: QoSSP Model entity

4 Conclusions

This deliverable discusses the implementation design of INTERMON data base using MySQL technology focussing on efficient distributed data base support and partitioning for enhanced performance.

The new elements in the data base including traceroute topology, MRT topology and border router network measurements are explained.

The changes in the old design considering QoS measurements were described.

At this stage, the simulation scenarios are still not included in the data base.

5 References

[XML DB] Arin Salminen, Frank Wm. Tompa, Requirements for XML Document Database Systems, 2001

[D2] [Specification of Inter-domain QoS Analysis Architecture](#), Deliverable 2, INTERMON, 30-08-2002

[D5] [Specification of inter-domain data base with policy controlled data collection](#), Deliverable 5, INTERMON, 30-09-2002

6 List of figures

FIGURE 1: DATA ACCESS RELEVANT COMPONENT IN THE INTERMON ARCHITECTURE	5
FIGURE 2: INTERMON DATA BASE STRUCTURE	6
FIGURE 3: QOS MEASUREMENT DATA ELEMENTS AND RELATIONSHIPS	16

7 List of Tables

TABLE 1: USER	7
TABLE 2: NETWORK ELEMENT	8
TABLE 3: TRACEROUTE TOPOLOGY SCENARIO	8
TABLE 4: TRACEROUTE CALL	9
TABLE 5: HOP MEASUREMENT	9
TABLE 6: MRT TOPOLOGY SCENARIO	9
TABLE 7: UPDATE INDEPENDENT TABLE	10
TABLE 8: UPDATE DEPENDENT TABLE	10
TABLE 9: IPFIX TRAFFIC FLOW TABLE	12
TABLE 10: CAMPAIGN	12
TABLE 11: CAMPAIGN CONFIGURATION	12
TABLE 12: NODE INTERFACE RESOURCES	13
TABLE 13: GLOBAL VARIABLES COLLECTION	13
TABLE 14: SAA VARIABLES COLLECTION	14
TABLE 15: INTERFACE VARIABLES COLLECTION	15
TABLE 16: QOS MEASUREMENT SCENARIO	18
TABLE 17: "PACKET DATA" ENTITY	18
TABLE 18: "QOSP STATISTICS" ENTITY	19
TABLE 19: QOSP MODEL ENTITY	20

8 Appendix

8.1 MYSQL Column Types

MySQL supports a number of column types, which may be grouped into three categories: numeric types, date and time types, and string (character) types. The column types supported by MySQL are listed below:

M	Indicates the maximum display size. The maximum legal display size is 255.
D	Applies to floating-point types and indicates the number of digits following the decimal point. The maximum possible value is 30, but should be no greater than M-2. Square brackets ('[' and `']') indicate parts of type specifiers that are optional. If ZEROFILL is specified for a column, MySQL will automatically add the UNSIGNED attribute to the column.
TINYINT[(M)] [UNSIGNED] [ZEROFILL]	A very small integer. The signed range is -128 to 127. The unsigned range is 0 to 255.
BIT	Synonyms for TINYINT(1).

BOOL	
SMALLINT[(M)] [UNSIGNED] [ZEROFILL]	A small integer. The signed range is -32768 to 32767. The unsigned range is 0 to 65535.
MEDIUMINT[(M)] [UNSIGNED] [ZEROFILL]	A medium-size integer. The signed range is -8388608 to 8388607. The unsigned range is 0 to 16777215.
INT[(M)] [UNSIGNED] [ZEROFILL]	A normal-size integer. The signed range is -2147483648 to 2147483647. The unsigned range is 0 to 4294967295.
INTEGER[(M)] [UNSIGNED] [ZEROFILL]	Synonym for INT.
BIGINT[(M)] [UNSIGNED] [ZEROFILL]	<p>A large integer. The signed range is -9223372036854775808 to 9223372036854775807. The unsigned range is 0 to 18446744073709551615. Some things you should be aware of with respect to BIGINT columns:</p> <ul style="list-style-type: none"> • All arithmetic is done using signed BIGINT or DOUBLE values, so you shouldn't use unsigned big integers larger than 9223372036854775807 (63 bits) except with bit functions! If you do that, some of the last digits in the result may be wrong because of rounding errors when converting the BIGINT to a DOUBLE. MySQL 4.0 can handle BIGINT in the following cases: <ul style="list-style-type: none"> • Use integers to store big unsigned values in a BIGINT column. • In MIN (big_int_column) and MAX (big_int_column). • When using operators (+, -, *, etc.) where both operands are integers. • An exact integer value in a BIGINT column could be stored as a string. In this case, MySQL will perform a string-to-number conversion that involves no intermediate double representation. • '-', '+', and '*' will use BIGINT arithmetic when both arguments are integer values! This means that if you multiply two big integers (or results from functions that return integers) you may get unexpected results when the result is larger than 9223372036854775807.
FLOAT(precision) [UNSIGNED] [ZEROFILL]	A floating-point number. Precision can be <=24 for a single-precision floating-point number and between 25 and 53 for a double-precision floating-point number. These types are like the FLOAT and DOUBLE types described immediately below. FLOAT(X) has the same range as the corresponding FLOAT and DOUBLE types, but the display size and number of decimals are undefined. In MySQL Version 3.23, this is a true floating-point value. In earlier MySQL versions, FLOAT(precision) always has 2 decimals. Note that using FLOAT may give you some unexpected problems as all calculations in MySQL are done with double precision. This syntax is provided for ODBC compatibility
FLOAT[(M,D)] [UNSIGNED] [ZEROFILL]	A small (single-precision) floating-point number. Allowable values are -3.402823466E+38 to -1.175494351E-38, 0, and 1.175494351E-38 to 3.402823466E+38. If UNSIGNED is specified, negative values are disallowed. The M is the display width and D is the number of decimals. FLOAT without arguments or FLOAT(X) where X <= 24 stands for a single-precision floating-point number.

DOUBLE[(M,D)] [UNSIGNED] [ZEROFILL]	A normal-size (double-precision) floating-point number. Allowable values are -1.7976931348623157E+308 to -2.2250738585072014E-308, 0, and 2.2250738585072014E-308 to 1.7976931348623157E+308. If UNSIGNED is specified, negative values are disallowed. The M is the display width and D is the number of decimals. DOUBLE without arguments or FLOAT(X) where 25 <= X <= 53 stands for a double-precision floating-point number.
DOUBLE PRECISION[(M, D)] [UNSIGNED] [ZEROFILL] REAL[(M,D)] [UNSIGNED] [ZEROFILL]	Synonyms for DOUBLE.
DECIMAL[(M[,D])] [UNSIGNED] [ZEROFILL]	An unpacked floating-point number. Behaves like a CHAR column: ``unpacked'' means the number is stored as a string, using one character for each digit of the value. The decimal point and, for negative numbers, the '-' sign, are not counted in M (but space for these is reserved). If D is 0, values will have no decimal point or fractional part. The maximum range of DECIMAL values is the same as for DOUBLE, but the actual range for a given DECIMAL column may be constrained by the choice of M and D. If UNSIGNED is specified, negative values are disallowed. If D is omitted, the default is 0. If M is omitted, the default is 10. Prior to MySQL Version 3.23, the M argument must include the space needed for the sign and the decimal point.
DEC[(M[,D])] [UNSIGNED] [ZEROFILL] NUMERIC[(M[,D])] [UNSIGNED] [ZEROFILL]	Synonyms for DECIMAL.
DATE	A date. The supported range is '1000-01-01' to '9999-12-31'. MySQL displays DATE values in 'YYYY-MM-DD' format, but allows to assign values to DATE columns using either strings or numbers.
DATETIME	A date and time combination. The supported range is '1000-01-01 00:00:00' to '9999-12-31 23:59:59'. MySQL displays DATETIME values in 'YYYY-MM-DD HH:MM:SS' format, but allows you to assign values to DATETIME columns using either strings or numbers.
TIMESTAMP[(M)]	A timestamp. The range is '1970-01-01 00:00:00' to sometime in the year 2037. MySQL displays TIMESTAMP values in YYYYMMDDHHMMSS, YYMMDDHHMMSS, YYYYMMDD, or YYMMDD format, depending on whether M is 14 (or missing), 12, 8, or 6, but allows you to assign values to TIMESTAMP columns using either strings or numbers. A TIMESTAMP column is useful for recording the date and time of an INSERT or UPDATE operation because it is automatically set to the date and time of the most recent operation if you don't give it a value yourself. You can also set it to the current date and time by assigning it a NULL value. The M argument affects only how a TIMESTAMP column is displayed; its values always are stored using 4 bytes each. Note that TIMESTAMP(M) columns where M is 8 or 14 are reported to be numbers while other TIMESTAMP(M) columns are reported to be strings. This is just to ensure that one can reliably dump and restore the table with these types!
TIME	A time. The range is '-838:59:59' to '838:59:59'. MySQL displays TIME values in 'HH:MM:SS' format, but allows you to assign values to TIME

	columns using either strings or numbers.
YEAR[(2 4)]	A year in 2- or 4-digit format (default is 4-digit). The allowable values are 1901 to 2155, 0000 in the 4-digit year format, and 1970-2069 if you use the 2-digit format (70-69). MySQL displays YEAR values in YYYY format, but allows you to assign values to YEAR columns using either strings or numbers. (The YEAR type is unavailable prior to MySQL Version 3.22.)
[NATIONAL] CHAR(M) [BINARY]	A fixed-length string that is always right-padded with spaces to the specified length when stored. The range of M is 0 to 255 characters (1 to 255 prior to MySQL Version 3.23). Trailing spaces are removed when the value is retrieved. CHAR values are sorted and compared in case insensitive fashion according to the default character set unless the BINARY keyword is given. NATIONAL CHAR (or its equivalent short form, NCHAR) is the ANSI SQL way to define that a CHAR column should use the default CHARACTER set. This is the default in MySQL. CHAR is a shorthand for CHARACTER. MySQL allows to create a column of type CHAR(0).
CHAR	This is a synonym for CHAR(1).
[NATIONAL] VARCHAR(M) [BINARY]	A variable-length string. Note: trailing spaces are removed when the value is stored (this differs from the ANSI SQL specification). The range of M is 0 to 255 characters (1 to 255 prior to MySQL Version 4.0.2). VARCHAR values are sorted and compared in case-insensitive fashion unless the BINARY keyword is given. VARCHAR is a shorthand for CHARACTER VARYING.
TINYBLOB TINYTEXT	A BLOB or TEXT column with a maximum length of 255 (2 ⁸ - 1) characters.
BLOB TEXT	A BLOB or TEXT column with a maximum length of 65535 (2 ¹⁶ - 1) characters.
MEDIUMBLOB MEDIUMTEXT	A BLOB or TEXT column with a maximum length of 16777215 (2 ²⁴ - 1) characters.
LOB LONGTEXT	A BLOB or TEXT column with a maximum length of 4294967295 (2 ³² - 1) characters. Note that because the server/client protocol and MyISAM tables has currently a limit of 16M per communication packet / table row, you can't yet use this the whole range of this type.
ENUM('value1','value2',...)	An enumeration. A string object that can have only one value, chosen from the list of values 'value1', 'value2', ..., NULL or the special "" error value. An ENUM can have a maximum of 65535 distinct values.
SET('value1','value2',...)	A set. A string object that can have zero or more values, each of which must be chosen from the list of values 'value1', 'value2', ... A SET can have a maximum of 64 members.