WBEM-based Inter-AS Performance Monitoring for QoS-guaranteed DiffServ-over-MPLS

Abdurakhmon Abdurakhmanov, Shahnaza Tursunova, Young-Tak Kim
Dept. of Information and Communication Engineering, Graduate School, Yeungnam University
214-1, Dae-Dong, Kyung-San-Si, Kyung-Book, 712-749, KOREA
graf_best@yahoo.com , sh_tursunova@ynu.ac.kr, ytkim@yu.ac.kr

Abstract

To guarantee the user-requested QoS and to keep the network utilization at maximum, the performance management of DiffServ-over-MPLS is essential. The performance management function monitors the network utilization and verifies the network performance compared with the agreed SLA (Service Level agreement). Constraint-based routing for guaranteed QoS provisioning must be used in the LSP establishment phase. The established LSPs must be continuously monitored by the performance management function to verify the assured performance and QoS parameters. However, in current network managements of public network operators, MOs of inter-AS traffic engineering for QoS-guaranteed DiffServ are not well standardized in public domain. As a result, it is not possible to easily collect the performance information across multiple domain networks to analyze end-to-end performance, and it is extremely difficult to find out which of the transit network domain has violated negotiation in the case of performance degradation. In this paper we propose a WBEM-based inter-autonomous (AS) system performance management architecture for QoS-guaranteed DiffServ-over-MPLS network. The WBEM based network management system (NMS) must manage the real network to configure the performance management function to monitor the network utilization and to verify the network performance compared with the agreed Service Level agreement (SLA). The proposed performance monitoring system provides end-to-end, edge-to-edge and TE link QoS performance monitoring and management. As a result, any severely degraded performance compared with the agreed performance level can be treated promptly to guarantee the agreed QoS provisioning.

1. Introduction

QoS monitoring is becoming crucial to Internet service providers (ISP) for provisioning QoS based services and service assurance and for managing network resources at both intra- and inter-domain levels. Intra-domain measurements are performed in a single/multiple AS where monitoring and measuring processes and realization are under the control of a single administrator. Inter-AS measurements are performed among multiple domains, which may not be under the same administrative authority [4].

In addition, users require network performance statistics, as network performance has a direct impact on the perceived quality of the application viewed by the users. The performance requirement of a customer's service is described in an SLA and consequently its service level specification (SLS) part.

Customers require various level of quality according to the application and want to verify required QoS and traffic parameters. That is one of the reasons why service providers must provide end-to-end QoS and traffic parameters monitoring. Moreover, to provide QoS-guaranteed service, providers must manage performance of QoS, handling all abnormal conditions in the network and fault restoration or alternative routing at service performance degradation using backup paths. However, in current network managements of public network operators, MOs of inter-AS traffic engineering for QoS-guaranteed DiffServ are not well standardized in public domain. As a result, it is not easy to collect the performance information across multiple domain networks to analyze end-to-end performance, and it is extremely difficult to find out which of the transit network domain has violated negotiation in the case of performance degradation.

In this paper we propose a WBEM-based inter-AS performance management system for QoS-guaranteed DiffServ-over-MPLS network. The proposed system provides distributed performance management for inter-AS networks and also provides end-to-end, edge-to-edge and TE link performance monitoring and management in order to verify whether the QoS performance guarantees the committed SLA/SLS. Proposed architecture detects transit AS which violates SLS negotiation in the case of end-to-end QoS degradation. Besides, we have designed managed objects (MO) for inter-AS TE and performance management respectively.

In the proposed architecture, NMS and MO are implemented based on DMTF's Web-based Enterprise Management (WBEM) and Common Information Model (CIM) respectively.

The rest of this paper is organized as follows. In Section II, related work on Web based enterprise management, common information model, Performance management in inter-AS environment and MESCAL [4] are introduced. In Section III, WBEM-based inter-AS TE architecture is explained. In section IV, implementation of WBEM-based inter-AS TE is explained. In section V, performance evaluation and analysis is given, and finally we conclude this paper in Section VI.

2. Background and Related Work

2.1 WBEM (Web-based Enterprise Management)

The WBEM [3] is a platform and resource independent Distributed Management Task Force (DMTF) standard developed to unify the management of enterprise computing, using a Common Information Model (CIM) standard - resulting in cost-effective products that interoperate as flawlessly as possible. WBEM comprises a set of standardized technologies, which includes a data model, (i.e., the CIM standard ;), an encoding specification, (i.e., xmlCIM Encoding specification), and a transport mechanism, (i.e., CIM Operation over HTTP). The xmlCIM Encoding specification is a standard for encoding CIM data and operations into extensible markup language (XML). The CIM Operation over HTTP is the definition of a standard protocol for transporting xmlCIM encoded requests and responses over HTTP, allowing implementations of CIM to interoperate in an open, standardized manner and complete the technologies that support WBEM.

The CIM specification is the language and methodology for describing management data. CIM is an object-oriented schema for modeling the objects. The CIM schema can be divided to three areas: the core model, the common model, and the extension model. First, the core model captures notions that are applicable to all areas of management. Second, the common model is an information model that captures notions that are common to a particular technology. For example, it includes the model for systems, applications, networks and devices. Last, the extension model represents technology-specific extensions of common models.

The CIM-XML encoding specification defines XML elements, written in Document Type Definition (DTD) that can be used to represent CIM classes and instances. The CIM operations over HTTP specification defines a mapping of CIM operations onto HTTP that allows implementations of CIM to interoperate in an open, standardized manner and completes the technologies that support WBEM. Therefore, in the WBEM architecture, the management information is described by the CIM schema, converted to XML, and finally embedded in an HTTP payload to transport to the target node.

2.2 CIM (Common Information Model)

The CIM is an object-oriented data model that provides a uniform data approach to define and describe all components in an enterprise computing environment [5]. Each component is described in a common and consistent manner independent of vendor and device architecture. CIM consists of a specification and a schema expressed in a MOF. The CIM specification describes the language, naming, high-level concepts and mapping techniques. The CIM schema provides the modeling descriptions and details for representing devices and the overall management environment. Together, they consistently and completely describe all aspects for managing an enterprise computing environment. Additionally, they provide a comprehensive method for adding vendor specific extensions in a CIM compliant manner.

DMTF defines a set of CIM schema to represent the management information, which can be divided into three parts:

(1) Core Model: Captures notions that are applicable to all areas of management.

(2) Common Model: Captures notions that are common to particular management areas, but independent of any particular technology or implementation.

(3) Extension Model: Represents technology specific extensions of the common models. Developers can extend the model by creating subclasses of objects. Applications can then obtain object instances in the standard model to manage different products in a heterogeneous environment.

2.3 Performance Monitoring Architecture of MESCAL model

The MESCAL [4] proposes cascaded model for performance management. In the cascaded model, a Mescal only establishes QoS peering agreements with its immediate neighboring provider/s to construct an end-to-end QoS service. In the cascaded peering model, the business relationship is between the source INP and its immediate adjacent INPs. There is no peering relationship between source INP and transit INPs.

The cascaded model is adapted as the preferred model in MESCAL, as it reflects the loosely coupled structure of Internet. In MESCAL, three solution options are defined offering loose, statistical and hard QoS guarantees. A Monitoring infrastructure should have three distinct entities in order to fulfil the stated requirements. These entities are namely Node, Network, and Service Level Monitors.

Node Monitors initiate the measurements and collects information on PHBs and QoS routes. This can be from the PHBs at the border routers for inter-domain resource provisioning/allocation purposes. In addition, it can be performance-related information per AS, i.e., between ingress border router of a domain to the ingress border router of its immediate downstream domain for specific QoS route.

Passive monitoring (offered load and throughput) for cSLS can be performed at the ingress and egress points. Passive monitoring for pSLS can be performed at the border routers (e.g., ASBR2 for offered load and ASBR3 for throughput as shown in Fig. 1).

The Network monitor at each domain can compose and deduce an end-to-end performance view by analysing PHBs and routes related measurements and measurement information received from downstream domains.

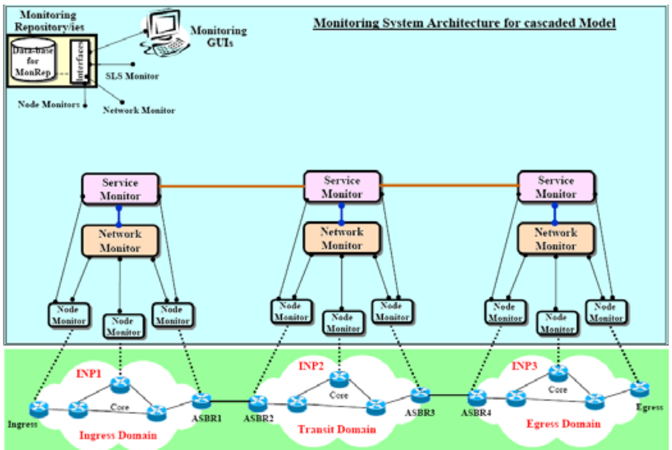

Fig. 1 Monitoring system architecture for the cascaded model

2.4 MPLS Network Management Systems

The standardizations of DiffServ-over-MPLS traffic engineering have been pursued mainly in IETF. Several MPLS network management schemes have been proposed and some NMSs are commercially available, but the functions for the DiffServ-over-MPLS are not fully supported yet.

Connectivity is the basic stuff from which the Internet is made. Therefore, metrics determining whether pairs of hosts (IP addresses) can reach each other must form the base of a measurement suite. The RFC2678 defines several metrics, some of which serve mainly as building blocks for the others. The RFC2679 and 2680 defines one-way delay metric and One-way packet loss metric for IP performance management.

RATES(Routing and Traffic Engineering Server) has been developed for MPLS traffic engineering[11]. The RATES also supports restoration of LSPs with a restoration-capable online routing algorithm. RATES, however, does not support DiffServ and does not provide performance measurement and analysis functions.

Wandl's IP/MPLSView is a tool for the network administers, performance management teams and IP/MPLS network control personnel to optimize time- and cost- savings, network bandwidth and network resources efficiently and productively [12]. It operates in a multi-layer, multi-vendor, multi-protocol environment, supporting the IP/MPLS configuration/performance management, network planning, VPN management, extensive report generation with fully web-enabled user interfaces. Wandl's IP/MPLSView supports differentiated services (DiffServ) and VPN model as an additional feature.

Sheer Networks' Broadband Operation Supervisor(BOS) [13] supports multi-layer (physical, ATM, Ethernet/VLAA, IP, MPLS, VPN) topology auto-discovery, real-time fault intelligence and root-cause isolation, GUI-based surveillance, service path tracing, service provisioning and activation, event correlation and service impact analysis, and IP-VPN service management. However, SheerBOS does not support DiffServ-over-MPLS services and traffic engineering.

3. Design of WBEM-based Inter-AS Traffic Engineering System

3.1 WBEM-based Inter-AS Traffic Engineering

The provisioning of inter-AS traffic engineering is required to support inter-AS bandwidth guarantees, inter-AS resource optimization, end-to-end performance monitoring and fast recovery across ASs. Generally the inter-AS MPLS traffic engineering need to be supported in (i) within single service provider's (ISP) administrative domains, and (ii) across multiple service providers administrative domain networks [6].

Within single autonomous system managed by a network operator, it would be easy to gather the information of link status and the available network resources with routing information. For QoS-guaranteed inter networking across multiple MPLS domain AS networks, certain information like reachability information with the specified bandwidth and QoS requirement of each domain network are required. In IP-based Internet, BGP (Border Gateway Protocol) provides basic inter-networking information, such as reachability. The detailed parameters for the DiffServ-over-MPLS traffic engineering are not supported yet.

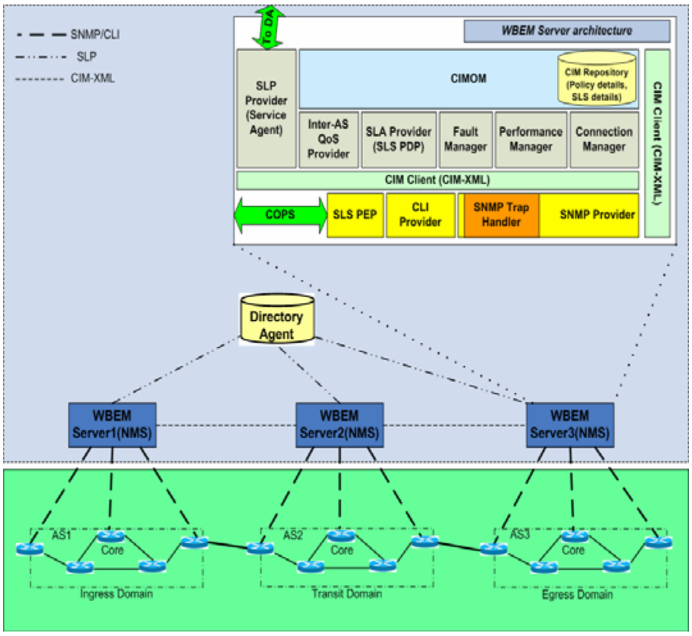

Fig. 2 WBEM based Inter-AS TE architecture

Fig. 2 depicts the functional architecture of WBEM-based Inter-AS TE. In this architecture each WBEM server (NMS) is registered in directory agent. The OpenSLP [8] server acts as a dedicated DA where all NMSs register /discover the service contact points using Service Location Protocol (SLP).

Each NMS is configured with Open Pegasus WBEM Server [8]. Current version of Open Pegasus supports SA functionality for service registration to DA. And the API for Open Pegasus client includes UA functionality such as service lookup and request. The SLP provider has SA functionality for service registering and CIM client has UA functionality for service request. During the NMS's WBEM initialization, the SA in SLP provider tries to detect DA and registers itself to the DA. The service type has been defined as "service:wbem.InterAS:pegasus". Once the service registration is successful, the UA can directly request service discovery to DA, and get information about interested service contact points.

For connection establishment, we have designed the Connection Manager. It establishes TE LSP connection within single intra domain or multiple domains.

For SLA negotiation, we have designed the SLA Provider [8] as SLS PDP and PEP functionalities. The Common Open Policy Service (COPS) protocol is used for the interaction between PEP and PDP. The Inter-AS QoS Provider is designed and implemented for the purpose of Inter-AS Traffic Engineering for QoS-guaranteed DiffServ-over-MPLS provisioning. SNMP Provider can interact with SNMP supported network element directly, using SNMP protocol. SNMP Trap Handler is included in SNMP Provider, and is used for fault detection mechanism by Fault Manager [7].

3.2 Performance Management

Fig. 3 shows the proposed WBEM-based Inter-AS Performance Management architecture. In our work we are proposing source-based model for performance management architecture. In the source based performance management systems ingress or central NMS must continuously gather performance parameters from all transit NMSs across the connection chain and analyze them to verify peer SLS agreement, and to measure end-to-end view. To avoid main disadvantage of that model, such as signaling load in large scale network, we propose distributed performance and fault management which reduce the signaling load by distributing some functions of central NMS to intermediate NMSs.

Ingress NMS, instead of gathering performance data from the path to detect any violation it setup intermediate NMSs which control ASes to analyze edge-to-edge LSP performance. Intermediate NMS in turns contacts with ingress NMS (central) only if it detects any violation in edge-to-edge LSP performance. After receiving notification from any intermediate NMS, ingress NMS calculates end-to-end violation and it is necessary rerouted LSP. In this distributed performance management concept, ingress NMS doesn't need to continuously monitor performance of end-to-end LSP, it mostly starts actively monitor network after receiving violation report (notification) from intermediate NMSs.

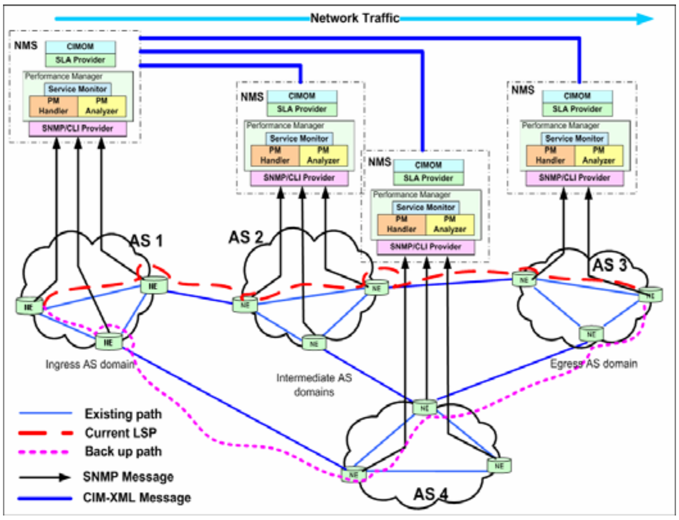

Fig. 3 WBEM-based performance management architecture

In this proposed architecture ingress NMS establishes QoS peering arguments directly with the number of downstream providers to offer an inter-domain QoS service. This is archived by making peering agreements with a chain of ASs so as to create a service within a scope beyond its boundaries.

Performance management system environment contains following components: SNMP/CLI provider, SLA provider, and performance manager. SNMP/CLI provider is used as an adapter between real network and WBEM framework. SLA provider is used to negotiate service agreement. Finally, performance manager which contains two functional blocks such as PM handler, PM analyzer and Service monitor.

PH handler is configured to perform the measurements on PHBs and QoS-enabled paths (LSP). IP engineered paths (LSPs) are used to carry aggregate user traffic belonging to several SLSs with similar performance requirements. Measurements can be specified from the PHBs at the border routers for inter-domain or edge/core routers for intra-domain resource provisioning and traffic engineering purposes. In addition, performance measurements can be carried out per AS hop i.e., between the ingress border router of a domain to the egress border router of the domain for a specific QoS route. End-to-end performance measurements between ingress and egress points of two remote domains can be carried out for specific QoS routes. Passive monitoring (offered load and throughput) for cSLS can be performed at the ingress and egress points. Passive monitoring for pSLS can be performed at the border routers.

The PM Analyzer in each domain can deduce per AS/domain performance view by analysing PHB and LSP related measurements. PHB QoS performance measurements can be used for managing inter-domain links and the link buffer space. The intra/inter-domain Traffic Engineering sub-system can use the measured performance of the various routes in order to do route management, load balancing, and dimensioning.

The Service Monitor uses the measurement data, which are collected by Network Monitors and Node Monitors, and composes the data for its domain and if SLS negotiation may be violated, it will pass the related information to the source NMS. This information includes the path level performance related measurements and the SLS specific traffic related statistics. Since each service type has specific requirements, different metrics may need to be measured for each service type.

3.3 Fast restoration procedure

In this architecture, we propose a distributed performance management, the ingress NMS is the central point which takes more responsibility for the overall service management including service monitoring of any given customer end-to-end service instance then others. The ingress NMS as the central entity establishes business relationships with all AS domains to be involved in the service delivery chain. Therefore, if customer requires end-to-end service performance results it must actively measure QoS and traffic parameters by invoking egress NMS to get necessary measurements, but if there is no such requirement from customer, ingress NMS does not continuously monitors end-to-end performance due to proposed distributed performance management approach. In our architecture, each NMS measures performance of edge-to-edge and TE link LSP, and based on the analyses it detects any degradation in edge-to-edge LSP (when it is the part of whole inter-AS LSP), it notifies ingress LSP about that, and in the case of TE link LSP degradation, it reroutes LSP (or switches the traffic to backup path). The ingress NMS in turn after receiving notification from one of peer NMSes calculates the impact of peer SLS violation of that AS to whole end-to-end LSP performance, and if the calculated end-to-end performance degradation is more than negotiated in customer SLS, it reroutes traffic to backup path. As a result the above mentioned concept allows detecting any QoS or traffic parameter degradation faster than the usual continuously end-to-end measurement, and allows detecting exact location of the AS which violated agreed SLS. Moreover, it reduces the signaling overhead.

4. Implementation of Performance Management in WBEM-based Inter-AS TE

4.1 CIM MOF-based MO design for performance management

Fig. 4 shows overall the important CIM classes and associations designed for performance management in inter-domain traffic engineering. Classes with CIM_prefix are supported by DMTF CIM, and CIM classes with QoS_prefix are newly designed MOF-based extensions for performance management in inter domain networks.

The ProtocolEndpoint class represents the interface of routers, routers as presented in Router and ASBRouter classes. MPLSTunnel and MPLSLSP classes are connected through MPLSBackupLSP and MPLSCurrentlyAssignedLSP association classes; they use many-to-many relation. MPLSTunnel is connected with NetworkPort class that defines ports on router. They are connected through MPLSTunnelPort association class and use one-to-many relation.

.

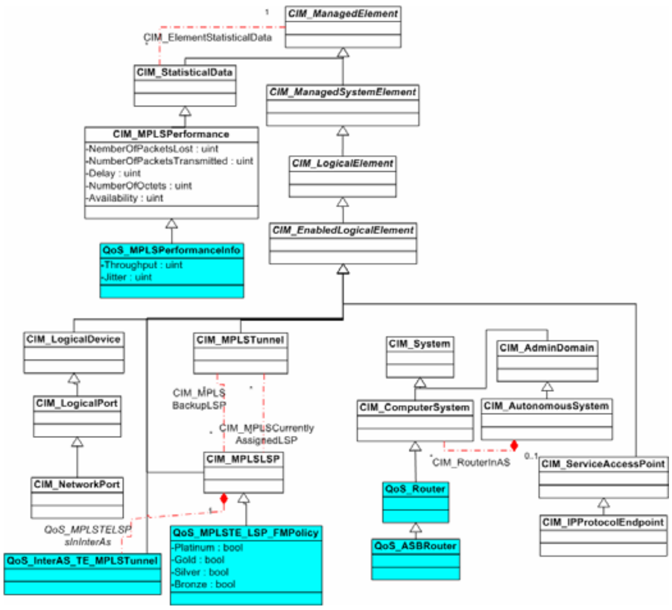

Fig. 4 DMTF CIM based MOF classes diagram for TE LSP

We extended the standard CIM models with our own extended CIM classes to optimize schema to our own purposes. We extended the standard CIM models with InterAS_TE_MPLSTunnel for inter-AS TE LSPs to differentiate from intra domain TE LSPs. It connected with MPLSTunnel class through MPLSTELSPsInInterASTunnel association class, it uses one-to-many relation, because one established inter-AS TE LSP may contain several intra-AS TE LSPs. The MPLSPerformanceInfo class inherits from MPLSPerformance class, we have extended that class to add new attribute to keep throughput (Fig. 4), in octets, outOctets and jitter values additionally to define in MPLSPerformance class. The Router class inherits from ComputerSystem class. ElementStatisticalData association class is used to connect that class with

MPLSLSP and ProtocolEndpoint classes

Each of MPLSTunnel and InterAS_TE_MPLSTunnel classes MOs may be associated with two MOs of MPLSPerformanceInfo class accordingly for LSP's ingress and egress routers. MOFs of QoS_interASNet and QoS_DiffServNet (Fig. 5) are derived from CIM_Network MOF that may contain multiple CIM_NetworkPipes with related CIM_ServiceAccessPoint. QoS_DiffServNet MOF contains several QoS_DiffServ class-types which are defined by the QoS_HostedDiffServ association. QoS_intraASDiffServTransitNet and QoS_interASDiffServNet MOFs are derived from the QoS_DiffServNet. Since inter-AS networking can be implemented by MPLS TE-LSP, the QoS_interASDiffServNet may contain several QoS_interAS_TE_MPLSTunnel which is associated this existing before CIM_MPLSTunnel MOF, which can be applied only to intraAS networking. They are defined by QoS_SupportedByInterASMPLSTunnel and QoS_SupportedByIntraASMPLSTunnel association accordingly.

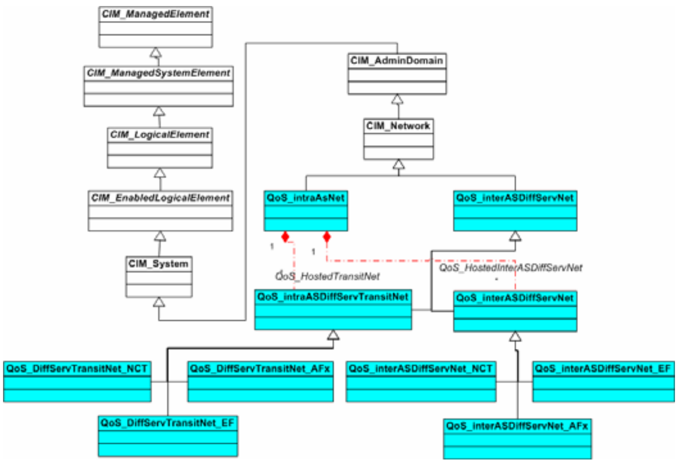

Fig. 5 DMTF CIM based MOF classes diagram for Overlay network

Fig. 6 shows OAM services for inter-AS TE. For performance management we designed InterASNet_PMSrv class derived from OAMService. The QoS_interASNet_PMSrv is used to configure the start and stop of performance monitoring operation, and the collection of results statistics. GetThoughput() method is used to obtain throughput of ingress NMS. GetDelay() and GetNumberOfPacketLoss() are used to obtain delay and number of packet loss accordingly. The fault management service is defined in QoS_InterASNet_FMSrv class, and it has the external methods for notifying the faults/recovery on managed objects. The QoS_interASNet_FMSrv is used to configure the fault & alarm notification functions for abnormal conditions on the established inter-AS TE-LSPs and also for fault restoration. The QoS_interASNet_CMSrv is used to AS connection establishment and gather traffic engineering information. Moreover QoS_OAMPolicy and QoS_OAMController classes were designed to define OAM policy for distributed fault management. All the classes are inherited from QoS_OAMService class.

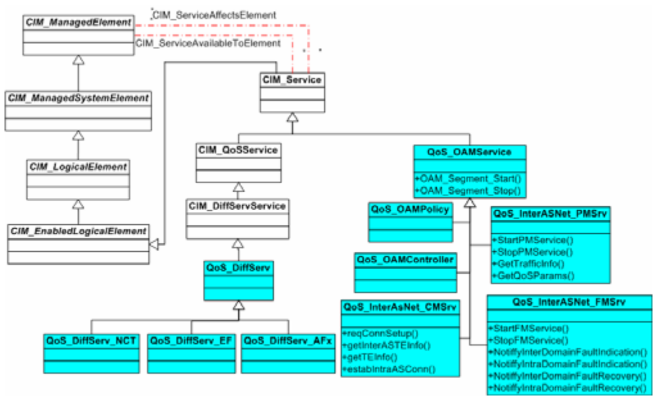

Fig. 6 DMTF CIM based MOF classes of MPLS OAM and QoS services

QoSDiffServ defines the QoS-guaranteed DiffServ class-types with related traffic parameters and QoS parameters. QoSDiffServ_NCT, QoSDiffServ_EF, QoSDiffServ _AFx are derived from the QoSDiffServ MOF, and define the details of the provided QoS-guaranteed DiffServ in each intra-AS and inter-AS domain.

4.2 Performance Monitoring

PM Handler continuously monitors established LSP's throughput and QoS parameters, save current values in MPLSPerformanceInfo MO accordingly. It retrieves active LSP's list, then associated ASBRouters ID, knowing this information it sends request to SNMP provider to obtain current parameters of those LSPs in ingress and egress routers. After receiving it saves this information such as throughput, delay, jitter, packet loss in MPLSPerformanceInfo MO. By this concept it obtains QoS parameters also.

In the case of end-to-end performance management, only ingress NMS may perform such monitoring, analyzer continuously retrieves instances of MPLSPerformanceInfo classes for interAS TE LSPs and compares traffic and performance parameters with obtained values from egress NMS. If computed loss rate is more than the agreed in SLA, it notifies fault restoration module to restore the degraded LSP.

In the case of edge-to-edge and TE link LSP performance management, analyzer retrieves active LSPs list from MPLSTunnel class MO, and then obtains related MPLSPerformanceInfo class MOs for ingress and egress routers (or ASBRouters). Since MPLSPerformanceInfo MO contains QoS and traffic parameters, analyzer compares those ingress and egress node values within one AS. If computed loss rate is more than the agreed QoS level (or delay and jitter) in SLS negotiation, it notifies fault restoration module of ingress NMS to restore degraded LSP.

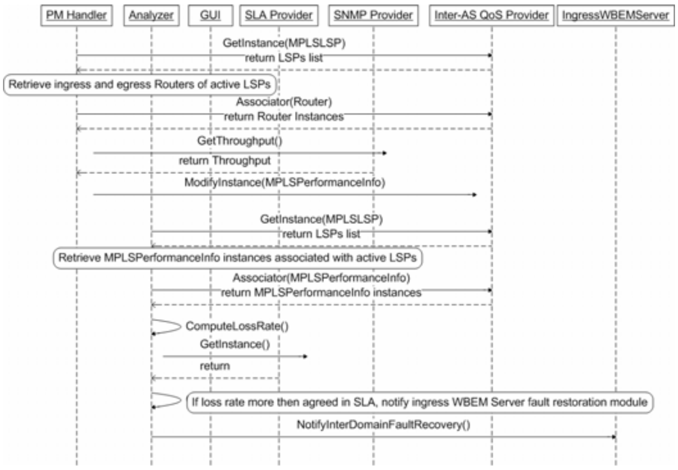

Fig. 7 Sequence diagram of edge-to-edge TE LSP performance management

4.3 Performance manager implementation

Currently performance manager (shown in Fig. 8) is implemented based on OpenPegasus platform; it contains three separated functional modules such as performance management handler, analyzer, and GUI. Performance Manager extends from CIM_MethodProvider, to handle the request for performance measurements.

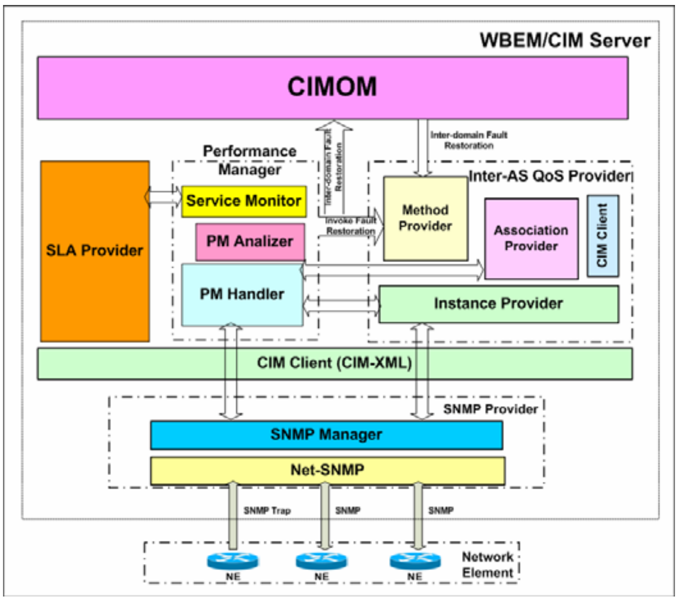

Fig. 8 Functional architecture of WBEM-based performance management

PM handler is used to connect with other providers and gather information. It uses CIM_Client interface to interact with Inter-AS QoS provider and SNMP Providers. PM handler application consist of two parallel threads: first is used to continuously get QoS and traffic parameters of TE LSPs from SNMP Provider, and store in MO of Inter-AS QoS provider, second is used to handle requests from other NMS and GUI.

PM Analyzer also has CIM Client interface to retrieve instances of SLA Provider and Inter-AS Provider for analysis. The GUI module can be implemented in C++.Net language. It was implemented in one thread which continuously get the QoS_MPLSPerformanceInfo MO instances, which includes QoS and traffic parameters such as throughput, delay, jitter, packet loss. Then it gets LSP requirements from SLA provider and compares. In the case of violation, the local or ingress NMS will be notified.

GUI module of our performance manager is implemented in C++.NET language in Windows XP platform. It visually can show the performance of each LSP with different parameters. It shows graphs for throughput, delay, packet loss and jitter. It connects with PM handler through CIM client interface.

5. Performance Evaluation and Analysis

5.1 Test Environment for inter-AS TE

For the Inter-AS TE, we implemented and used five NMSs with WBEM server/ client setup. Each NMS controls and manages whole intra-domain network. All of NMS (WBEM server) have been implemented on Linux platform. A hardware specification of each NMS is shown in Table 1.

Fig. 3 depicts the test bed for inter-AS TE with WBEM server which controls intra- domain network. It contains of three main parts: GUI, WBEM server and network nodes. The first is GUI, it is implemented in WindowsXP operation system using C++.NET language. Second is WBEM server, which contains of SLA/SLS, InterASQoS, SNMP, CLI Providers and Connection, Configuration, Performance, Fault managers. Third is real network elements like IP/IPMPLS Routers of Cisco 7204/6506 series.

Table 1 WBEM Hardware Specification

NMS ID	Hardware Specification
127	Pentium4 3GHz, 1 Gigabyte RAM, Redhat 9.0
234	Pentium4 3.2GHz, 8 Gigabyte RAM, Redhat 9.0
236	Pentium4 3.2GHz, 8 Gigabyte RAM, Redhat 9.0
223	Pentium4 3.2GHz, 8 Gigabyte RAM, Redhat 9.0
233	Pentium3 3GHz, 1 Gigabyte RAM, Redhat 9.0

5.2 Performance measure for all activities of WBEM-based Inter-AS TE

Fig. 9 shows the complete picture of time taken by each functions of WBEM architecture involved for performance management in inter-domain networks. The service discovery using SLP protocol took around 7 ms and the CSPF computation for the given topology with five NMS took around 0.252 ms. In general the method invocation such as StartPM(), StopPM() take round of 12 ms .

Local association traversals take around of 100 ms, and remote association traversals take around 120 ms. Scenarios of those traversals is shown in Table 2, Some of the performance results were improved comparing with our previous results [7, 8]. Finally connection establishment take around 320 ms.

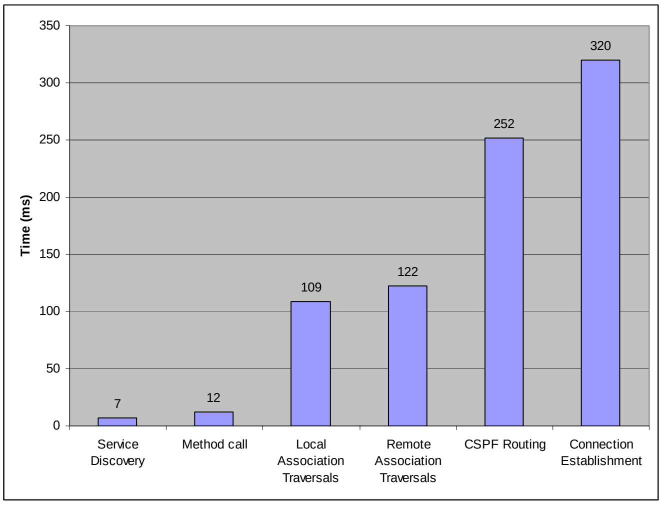

Fig. 9 Time taken by each functions of WBEM architecture

From the performance analysis, we found that the instance creation time depends on the hardware specification and method invocation does not take much time. The Provider API function for association traversals and enumerating instances also does not take much time. Enhance hardware performance and thread based parallel implementation can improve the performance. Moreover we achieved performance results better than in our previous results due to optimized MIT, it means we associated most frequently used classes to each other, where it was possible, by this way we reduced association traversals time, which is very important procedure when you must frequently access to MIT.

Table 2. Association Traversals (tests were performed with one association class)

Obtained Classes	Input classes	Association classes	Local Traversals time (ms)	Remote Traver-sals time (ms)
DiffServ	MPLSLSP	HostedDiffServ	106	121
MPLSPerfo rmanceInfo	MPLSLSP	Element-Statistical-Data	104	122
Port	Router	PortsIn Router	109	120

Tables 3 shows association traversals results without optimized MIT with two and three association classes respectively in traversal.

Table 3. Association Traversals (tests were performed with two and three association classes)

Obtained classes	Input classes	Association classes	Local Traver-sals time (ms)	Remote Traver-sals time (ms)
MPLSLSP	Port Router	PortsInRouter LSPsInPort	182	260
MPLSPerfo rm-anceInfo	MPL-SLSP Port	ElementStatistical-Data LSPsInPort	184	255
DiffServ	Router	PortsInRouter MPLSTunnelInPort HostedDiffServ	262	363

5.3 Performance evaluations on performance management

In our WBEM-based inter-AS performance management system, we use GUI shown in Fig. 10 and Fig. 11. Fig. 11 shows LSP list information. In composition of the “LSPList” window, “Port IP Address” field is the address of the selected port, and “LSR Node ID” filed is the node id of the selected node, finally “Start Monitoring” button is for the function that starts PM of selected LSP.

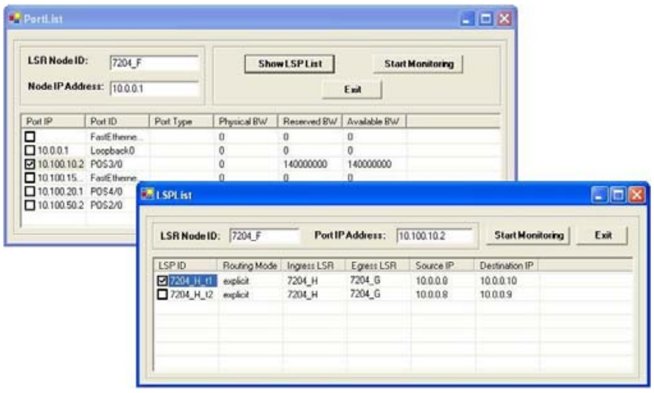

Fig. 10 LSP list GUI window

If we select any router in GUI, and choose port list command by right clicking on the selected router icon, we can see port list information that belongs to the router. If we select any port to be monitored, and push “Start Monitoring” button, performance manager of WBEM server starts to work. In order to view the performance of all LSP information of a port, we can use the “Show LSP list”, and then GUI shows LSP information shown in Fig. 10.

Fig. 11 depicts GUI window shows throughput of selected port or LSP. It shows in and out throughput of port, or whole throughput of LSP.

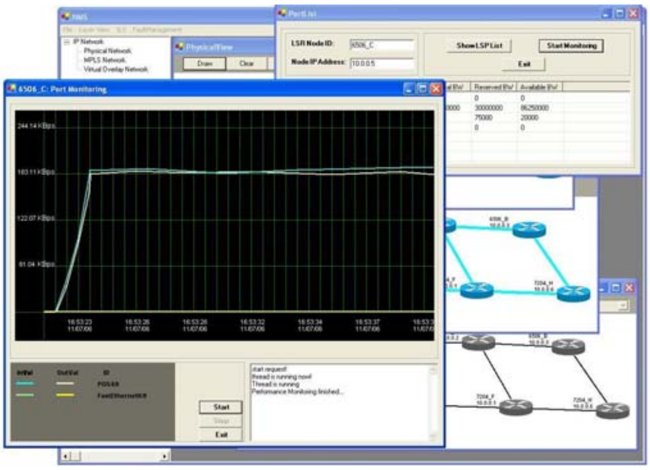

Fig. 11 Throughput monitoring diagram GUI window

5. Conclusion

In this paper, we proposed a WBEM-based inter-AS performance management architecture for QoS-guaranteed DiffServ-over-MPLS networks. In our work, to achieve more flexibility, better load balancing and other features, we proposed source-based model for performance management. To avoid disadvantages of source-based model such as signaling overhead in the case of large scale networks, we proposed performance and fault management architecture. In this architecture, even so we propose distributed performance management, the ingress NMS is the central point which takes more responsibility for the overall service management including service monitoring of any given customer end-to-end service instance than others. Hence, the proposed performance management architecture allows detecting any QoS or traffic parameters degradation faster than usual continue end-to-end measurement and exact location of the AS violation agreed SLS. Moreover, it reduces signaling load in the network.

Our proposed implementation provides performance management of QoS and traffic parameters for end-to-end, edge-to-edge and TE link.

We also designed MO for inter-AS TE and performance management with several extensions on existing experimental DMTF CIM models. We explained detailed implementation approach on WBEM OpenPegasus open source.

References

[1] Young-Tak Kim, “Inter-AS Session & Connection Management for QoS-guaranteed DiffServ Provisioning,” Proc. of International Conference on Software Engineering Research, Management & Applications (SERA 2005), Mt. Pleasant, USA, pp. 325~330.

[2] Dong-Jin Shin, Young-Tak Kim "Design and Implementation of Performance Management for the DiffServ-aware-MPLS Network," Proceedings of Conference on APNOMS 2003, Fukuoka, Japan, October 2003.

[3] Web-based Enterprise Management (WBEM), http://www.dmtf.org/standards/wbem/.

[4] MESCAL Consortium, D1.1 Specification of Business Models and a Functional Architecture for Inter-domain QoS Delivery, June 2003.

[5] DMTF CIM Schema http://www.dmtf.org/standards/cim/cim_schema_v210

[6] Shanmugham Sundaram, Abdurakhmon Abdurakhmanov, Young-Tak Kim, "WBEM-based Inter-AS Traffic Engineering for QoS-guaranteed DiffServ Provisioning," IEEE Broadband Convergence Networks (BcN2006) Workshop, Vancouver, British Columbia, Canada, 2006.

[7] Abdurakhmon Abdurakhmanov, Shahnoza Tursunova, Young-Tak Kim, “Distributed Fault Management in WBEM-based inter-AS TE for QoS guaranteed DiffServ-over-MPLS,” Proceedings of Conference on APNOMS 2006, Busan, Korea, September 2006.

[8] Jong-Cheol Seo, Young-Tak Kim, “WBEM-based SLA Management across multi-domain network for QoS-guaranteed DiffServ-over-MPLS Provisioning, ” Proceedings of Conference on APNOMS 2006, Busan, Korea, September 2006.

[9] Open Pegasus, http://www.openpegasus.org/.

[10] Chris Hobbs, A Practical Approach to WBEM/CIM Management, Auerbach Publications, 2004.

[11] Petri Aukia et al., “RATES : A Server for MPLS Traffic Engineering,” IEEE Network Magazine, Mar./Apr. 2000.

[12] Wandal IP/MPLSView, http://www.wandl.com/html/mplsview/MPLSview_new.cfm.

[13] Sheer Broadband Operating Supervisor(BOS), Sheer Networks, http://www.sheernetworks.com/solutions/overview.shtml.

Abdurakhmon Abdurakhmanov is a master course student in the Advanced Networking Technology Lab. (ANTL) of Yeungnam University (http://antl.yu.ac.kr/), Korea. He received bachelor degree in Tashkent University of Information Technologies in June, 2004 (Tashkent, Uzbekistan). Since February 2005 and so far he is a member of Network Management team of ANT Lab. He is involved in ITRC (Information Technology Research Center) Project sponsored by

MIC (Ministry of Information and Communication) of Korea. His research interests include web service based network management and QoS-guaranteed inter-AS traffic engineering. He is a member of Korean Institute of Communication Sciences (KICS).

Shahnaza Tursunova received B.E. degree in computer science and information technology from Tashkent University of Information Technologies in June 2005 (Uzbekistan). Currently she is a candidate of M.S. degree in the department of Information and Communication Engineering at Yeungnam University. Her research focus is WBEM-based inter-AS network management and QoS guaranteed multimedia service provisioning. She is a member of Korean Institute of Communication Sciences (KICS).

Young-Tak Kim is a professor in the school of electrical engineering and computer science (EECS) of Yeungnam University (http://antl.yu.ac.kr/), Korea. He graduated Yeungnam Univ. in 1984, and received M.E. and Ph.D. degree from KAIST (Korea Advanced Institute of Science and Technology) in 1986 and 1990, respectively. He joined KT (Korea Telecom) in March 1990, where he had been working as a technical staff of research & development of an ATM Metropolitan Area Network Switching System, and networking technologies of the ATM Highway for Korean Information Infrastructure. In September 1994, he transferred to Yeungnam University. In 2001, he worked as a visiting scholar in NIST (National Institute of Standard and Technology), USA, where he was involved in the development of the GMPLS-based Optical Transport Network Simulator, called GLASS (GMPLS Lightwave Agile Switching Simulator, http://dns.antd.nist.gov/glass/). His research interests include QoS-guaranteed inter-AS traffic engineering and broadband mobile Internet service provisioning on a broadband converged wired & wireless network environment. Young-Tak is a member of IEEE Communication Society (ComSoc), KICS (Korea Institute of Communication Society), KISS (Korea Information Society), KIPS (Korea Information Processing Society), and Korea Multimedia Society. He has been working as an organizing committee member of APNOMS, IEEE NOMS 2004, and IEEE BcN 2006. Since May 2005, he has been working as the Technical Program Chair of IEEE CNOM.