MESCAL

Management of End-to-end Quality of Service
Across the Internet at Large

IST-2001-37961

D2.1: Implementation plan and high level engineering design of simulation models and testbed prototypes for inter-domain QoS delivery

Document Identifier: MESCAL/WP2/UCL/D2.1/final	
Deliverable Type: Report **Deliverable Nature:** Public	**Contractual Date:** 31 August 2003 **Actual Date:** 30 January 2004

Editor:	David Griffin, Jason Spencer, University College London
Authors:	*FTR&D:* Mohamed Boucadair, Pierrick Morand *TRT:* Hamid Asgari, Richard Egan *UCL:* Jonas Griem, David Griffin, Jason Spencer *UniS:* Paris Flegkas, Stelios Georgoulas, Kin (Roy) Hon Ho, Michael Howarth, George Pavlou, Panos Trimintzios, Ning Wang *Algo:* Takis Damilatis, Panos Georgatsos
Abstract:	The main purpose of this deliverable is to present the overall engineering design and the implementation plan of the MESCAL simulation models and testbed prototypes for inter-domain QoS delivery. The tasks required for the detailed implementation design and development of the models and prototypes are identified, dimensioned in terms of resources and projected in time in the form of system releases. The implementation plan is a refinement of the WP2 plans compiled at the time of proposal. The refinements were based on the actual progress of the project over its initial theoretical work and on the specifications of the MESCAL model and functional architecture as well as the specific algorithms and protocols as documented in deliverables D1.1 and D1.2. This document also surveys existing network elements, system prototypes and simulation environments in the light of MESCAL requirements and documents the decisions taken on which tools and technologies will form the basis of the MESCAL development work.
Keywords:	Implementation plan, Inter-domain QoS delivery, Testbed, Simulation

France Telecom Research and Development	FTR&D	Co-ordinator	France
Thales Research and Technology	TRT	Principal Contractor	UK
University College London	UCL	Principal Contractor	UK
The University of Surrey	UniS	Principal Contractor	UK
Algonet SA	Algo	Principal Contractor	Greece

Project funded by the European Community under the
"Information Society Technology" Programme (1998-2002)

Executive Summary

This deliverable discusses issues relating to the implementation and high level engineering design of an experimentation testbed and simulation framework to prototype inter-domain QoS delivery. The implementation plan has been compiled in collaboration with WP3 so that the schedule of delivered simulation models and testbed prototypes is synchronised with the initial experimentation plans of the project.

The aspects of the MESCAL system to be developed for experiments to be undertaken in a simulated network environment include: the off-line TE algorithms, including inter-domain binding, selection and resource optimisation algorithms and IP-based intra-domain TE including a network resource scheduler and resource optimisation algorithms for calculating intra-domain link weights per QoS class. qBGP protocols and route selection algorithms will be developed in J-Sim in addition to the testbed development discussed below. The qBGP simulation models will facilitate scalability and stability tests that would otherwise be difficult to exercise in a limited testbed. It is intended that the off-line TE algorithms will be integrated with the qBGP simulation models in the later stages of experimentation. The pSLS management components, including SLS ordering and SLS order handling; together with dynamic pSLS invocation will initially be implemented to operate as stand-alone prototypes. The stand-alone prototypes will validate the pSLS negotiation protocols and associated logic and they will subsequently be integrated, in a loose fashion, with both the simulator- and testbed-based experiments. SLS invocation handling as well as algorithms and protocols for multicast QoS will be implemented to work with packet-based simulations using the ns-2 simulator.

The MESCAL testbed implementation plan details the steps and indicates the milestones for the development of a "proof of concept"-oriented testbed. The testbed will be located in the FTR&D premises. Further information about the configuration and the set up of this testbed will be provided by WP3 in due course. Three phases of implementation have been identified, as discussed below:

The objectives of the first phase are to deploy an operational testbed including several ASs exchanging inter-domain routing information between them and implement the meta-QoS-classes within separate autonomous systems. The ZebOS Suite will be the base routing stack activated on the Linux routers.

The objectives of the second phase are to: implement a QoS-inferred BGP protocol (qBGP); use qBGP as an inter-domain routing protocol in the testbed deployed in phase 1; and modify the forwarding mechanism by introducing meta-QoS-class-planes. Dynamic inter-domain traffic engineering should take into account the QoS information and routing selection process should be QoS-based. The ZebOS Suite and especially its BGP stack will be modified. In the end of this phase, we obtain a prototype of a full loose solution option.

The objectives of the third phase are to: use the output of the off-line intra-domain TE; implement PCS and the PCS Communication protocol; and validate the interaction between the PCS process and qBGP. Within this phase, we will implement the MESCAL hard solution option above the loose solution option. The establishment of the inter-domain LSPs will be initiated by PCS. Cisco routers will be tested and integrated in this phase when required IOS features are available.

This document also reviews the available network elements for the testbed, including routing hardware, routing software and simulation packages. Also discussed are ancillary components such as topology generators and traffic generators. The components of a MESCAL enabled network management system are discussed with an examination of implementation technologies, management interfaces and protocols. Further, software that could be brought in from other projects for assisting in various aspects of an implementation is examined, considering contributions from TEQUILA and various policy management protocols.

Cisco uses its proprietary operating system (IOS). Cisco IOS Software provides a wide range of functionality - from routing, connectivity, security, and network management to technically advanced services that enable to deploy applications. Cisco commercial routers and IOS features have been

reviewed in D2.1. This includes IOS features for Diffserv model implementation, MPLS traffic engineering capability, BGP support, and support of various signalling protocols.

Linux routing software implementations are fully reviewed in D2.1. Zebra, Quagga, ZebOS, GateD, Multithreaded Routing Toolkit and XORP are compared against the criteria of their support for intra-domain unicast, intra-domain multicast, inter-domain, management interfaces, software development environment and cost to the project. Based on its capabilities and support, ZebOS routing software Suite (version 3.5 or later) is selected to be the base routing stack for the Linux routers within the MESCAL project.

The simulators of most interest to MESCAL were compared with regards to their support for BGP, DiffServ, RSVP, MPLS, IP Multicast and IPv6.

The NS simulator, which was used extensively in the TEQUILA project, recently acquired BGP support. However, the NS BGP module lacks some advanced features, its functionality has not yet been extensively validated and the NS packet forwarding mechanism needs to be modified in order to use the routing tables constructed by this BGP module. While the functionality provided by the OPNET packages is clearly sufficient, these products had to be discarded for reasons of expenditure limitations. J-Sim and SSFNET are the most appropriate tools to undertake the MESCAL experimentation involving BGP. Both these two simulators already include implementations of the basic functionality required by MESCAL, thus the implementation effort can focus exclusively on the additional functionality that will be defined in WP1.

SSFNET has the implementation of BGP4 based on RFC 1771. It has recently received attention by researchers on BGP simulations (e.g. stability and convergence studies of BGP). SSFNET is free for research purpose and source code - especially its complete BGP module - is available for MESCAL in order to implement required qBGP extensions. SSFNET has greater processing requirements than NS-2, but smaller memory demands. J-Sim supports most of the mechanisms that are considered by MESCAL such as DiffServ and MPLS. Hence, these mechanisms can be used together with BGP (or qBGP), to perform the required simulations of MESCAL approaches. J-Sim requires significantly less memory than SSFNET, thus allowing larger simulations. For example, it can run simulations with an AS-level topology of a size close to the magnitude as the Internet (nearly 3000 AS and 10000 BGP sessions). A trade-off of memory requirement versus execution speed, exists between J-Sim and SSFNET. J-Sim is seen to be more appropriate for MESCAL, if large-scale simulations will be required.

Table of Contents

List of Figures

List of Tables

1 INTRODUCTION

This document is produced as part of work package 2, "*System Design and Implementation*". The work package is responsible for implementing the MESCAL solutions defined by WP1 to allow experiments on the algorithms and protocols in WP3.

Chapter 2 presents the MESCAL implementation plan. Development activities in the project are driven by the requirements of the experimentation tasks in WP3. The experimentation tasks are identified which in turn leads to the identification of those aspects of the MESCAL functional model that are to be developed. The implementation tasks are dimensioned in terms of: the effort required for software design and development; the time schedule for the system releases; and, the allocation of tasks to the project partners. Chapters 3 and 4 give an overview of the testbed-based and simulation-based implementation activities of the project. Chapter 5 reviews and selects the technologies, protocols, equipment and software to be used during the implementation phase of the project.

2 IMPLEMENTATION PLAN

2.1 Categories of experiments

Experimentation is an essential aspect of MESCAL work to the end of fulfilling overall project objectives. Experimentation activities are distinguished according to:

- the particular environment where they will be undertaken, and
- the specific objectives they aim at achieving.

Experimentation activities will be carried out both in realistic and simulated network environment, as appropriate to the aspect of the MESCAL work under test and the experimentation objectives. Specifically, experiments will be undertaken:

- In a testbed comprised of Linux-based and Cisco routers. The project testbed is provided by FTR&D and in located in Caen, France.
- In simulators, which, depending on the aspect of the network environment they simulate, can be distinguished into:
    - (Dynamic) network operation simulation engines, simulating the dynamics of network behaviour at such a level of abstraction as appropriate to the purpose of the experiment e.g. at a packet, flow or protocol or control-plane activity levels.
    - (Static) network environment simulation tools, simulating the static aspects of the network environment i.e. the context in which the network is to operate; such aspects include network topology, number of supported QoS-classes, established service agreements, aggregate QoS traffic demands and QoS traffic generation patterns.

Evidently, the type of the experimentation environment affects the nature of the releases coming out from the WP2 implementation activities. In this respect, we distinguish two major types for the nature of the WP2 implementation releases:

- Prototype releases, where the developed aspect of the MESCAL solution is developed to apply in general engineering environment, according to the selected implementation technologies. This type of releases is for experimenting in testbeds and in network environment simulation tools.
- Simulation engine-specific releases, where the developed aspect of the MESCAL solution is developed specifically for experimentation in the chosen network operation simulation environment. In this case, the released component is built, programming-wise, around the abstractions, tools and facilities offered by the particular network simulation engine chosen by the project (J-Sim and ns-2); as such, cannot execute in a general engineering context, as a prototype release would.

As for their objectives, experimentation activities can fall under the following commonly recognised categories:

- functional validation experiments, aiming at assessing feasibility of implementation and validity of specifications, and
- performance assessment experiments, aiming at assessing the behaviour of the aspect under test in a variety of network operation and environment set-ups and conditions. Behaviour is assessed in terms of scalability, stability, sensitivity and yielded benefits/incurred cost; as such, corresponding experiments will be carried out.

Obviously, experimentation objectives are restricted by the capabilities of the experimentation environment. As such, performance assessment experiments can only be performed in a simulated network environment, static or dynamic; not, in a fixed testbed environment. And, functional validity experiments better be carried out in a testbed environment for exhibiting the validity of the functionality under test from network operation perspectives in a realistic network environment, where network operational credentials prevail those that can possibly be acquired in a simulated network environment.

From WP2 perspectives, given that implementation activities are experimentation driven, experimentation focus poses the requirement that, in addition to MESCAL functional aspects, appropriate tools need to be implemented as required for fulfilling experimentation objectives.

Summarising, WP2 will have to produce:

- prototype releases targeting at functional validation experiments in the testbeds as well as appropriate interface tools facilitating integration to the network environment
- prototypes releases targeting at performance assessment experiments in a simulated network environment as well as appropriate tools for generating the required testing conditions
- releases specific to the engine selected by the project to simulate network operation, targeting at assessing the behaviour of the functional aspect under test in a dynamic network environment, as well as appropriate tools for controlling test execution.

Conclusively, WP2 implementation activities have a clear orientation in nature and target, which is substantiated in a concrete time plan presented in the following sections.

2.2 Categories of implementation tasks

Deliverable D1.2 describes the overall MESCAL functional architecture and specifies algorithms and protocols for each of the main functional components. Figure 1 highlights those aspects of functional architecture that are subject to experimentation and therefore implementation within the project.

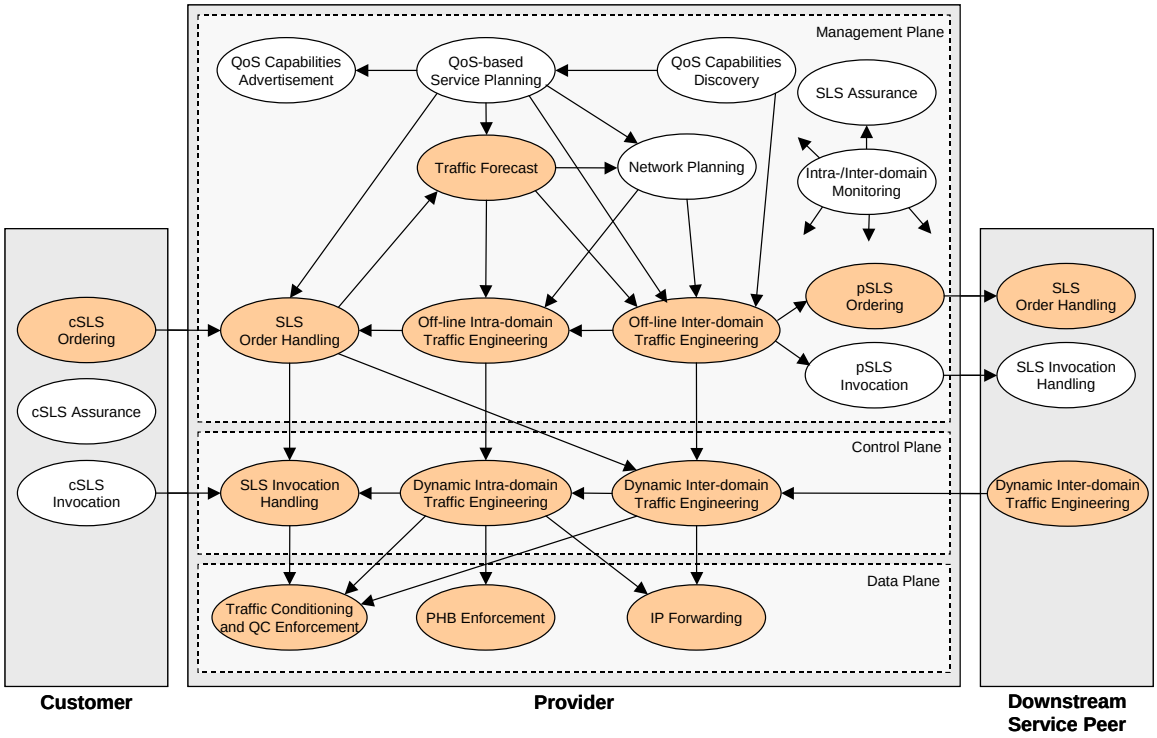

Figure 1 MESCAL functional architecture, highlighting the components to be implemented

Each of the functional components is further decomposed into the engineering components to be implemented in order to test the algorithms and protocols specified by the project. The implementation tasks may be classified as follows:

- Development of the *testbed network*: tasks related to the deployment of the QoS capabilities required in the control and data plane of Linux routers within the project testbed.
- Development of the *simulated network*: tasks related to the development of the required control and data plane QoS capabilities in the simulators.
- Development of the MESCAL *management plane*: the off-line traffic engineering and SLS management algorithms and protocols. These components are to be tested in two modes: to assess their validity and performance with regard to the static aspects of network operation without being integrated with control/data planes of either the testbed or simulated network; and, subsequently, selected groups of functions will be integrated with the testbed and/or network simulators to assess their validity in dynamic network environments.

In addition to the engineering counterparts of the functional model, a set of adaptors are required to interface the management plane components with the testbed and/or simulators and, furthermore, a set of experimentation tools, such as traffic generators and results analysis applications are required to assist the testing activities of WP3.

Given the experimentation objectives highlighted in the previous section and the implementation categories introduced above we have identified the following implementation tasks for WP2.

Traffic Engineering implementation tasks:

- Off-line inter-domain TE
  - Binding selection
  - Binding activation
  - Inter-domain resource optimisation
  - Off-line multicast TE
  - Path computation algorithm
  - PCS interactions
- Off-line intra-domain TE
  - Network reconfiguration scheduler
  - Intra-domain resource optimisation
  - Off-line multicast TE
- Dynamic inter-domain TE
  - qRIB
  - qBGP interactions
  - Route selection
- Dynamic intra-domain TE (IGP extensions)
- Network repository

Data plane implementation tasks:

- Traffic conditioning
- PHB enforcement
- IP forwarding/qFIB

SLS management implementation tasks:

- cSLS ordering
- pSLS ordering
- SLS order handling
- SLS repository
- SLS-based demand derivation tools (traffic forecast)
- SLS invocation handling

Adaptors to testbed/simulation environment

- SLS management -> testbed
- Off-line inter-domain TE -> testbed
- SLS management -> Off-line intra- and inter-domain TE (traffic forecast)
- Off-line inter-domain TE -> Off-line intra-domain TE (traffic forecast)
- Off-line inter-domain TE -> Dynamic inter-domain TE (qBGP)
- Multicast TE -> network simulator

Experimentation tools

- Traffic generation tools
- Bulk c/pSLS order emulator
- Network topology generator
- Results analysis tools

A software engineering design for all of the identified components will be specified in I2.2, *Detailed implementation design of components for inter-domain, QoS-based service management, network design, dynamic resource management and routing,* due in month 16 of the project.

2.3 Implementation plan

2.3.1 Effort plan

As estimation of the effort required for the detailed engineering design, software coding and stand-alone testing of each component is presented in Table 1.

	Implementation tasks, effort (PMs)			
	testbed network	simulated network	management plane	Total
Traffic Engineering				
Off-line inter-domain TE				***16***
Binding selection	-	-	1.5	1.5
Binding activation	-	-	1.5	1.5
Inter-domain resource optimisation	-	-	4	4
Off-line multicast TE	-	-	2	2
Path computation algorithm	-	-	2	2
PCS interactions	-	-	5	5
Off-line intra-domain TE				***6***
Network reconfiguration scheduler	-	-	1	1
Intra-domain resource optimisation	-	-	3	3
Off-line multicast TE	-	-	2	2
Dynamic inter-domain TE				***15***
qRIB	4	0.5	-	4.5
qBGP interactions	5	2.5	-	7.5
Route selection	2	1	-	3
Dynamic intra-domain TE (IGP extensions)	-	0.5	-	0.5
Network repository	-	-	0.5	0.5
Data plane				**5**
Traffic conditioning	1	-	-	1
PHB enforcement	1	-	-	1
IP forwarding/qFIB	3	-	-	3
SLS management				**16**
cSLS ordering	-	-	0.5	0.5
pSLS ordering	-	-	4	4
SLS order handling	-	-	6	6
SLS repository	-	-	1	1
SLS-based demand derivation tools (traffic forecast)	-	-	1.5	1.5
SLS invocation handling	-	3	-	3
Adaptors to testbed/simulation environment				**10.5**
SLS management -> testbed	2	-	-	2
Off-line inter-domain TE -> testbed	2	-	-	2
SLS management -> Off-line intra- and inter-domain TE (traffic forecast)	-	-	2	2
Off-line inter-domain TE -> Off-line intra-domain TE (traffic forecast)	-	-	1	1
Off-line inter-domain TE -> Dynamic inter-domain TE (qBGP)	2	0.5	-	2.5
Multicast TE -> network simulator	-	1	-	1
Experimentation tools				**8.2**
Traffic generation tools	1.2	-	-	1.2
Bulk c/pSLS order emulator	-	-	2	2
Network topology generator	-	1	-	1
Results analysis tools	1.5	0.5	2	4
Total effort	***24.7***	***10.5***	***42.5***	***77.7***

Table 1 Implementation tasks and estimation of development effort

It should be noted that some of the implementation tasks are already underway and effort has already been consumed. The figures presented in the table above refer only to the remaining effort required, as of the end of January 2004.

2.3.2 Implementation schedule

The overall project Gantt chart is shown in Figure 2. This shows the relationships between WP2 and WP3 and the overall schedule in time.

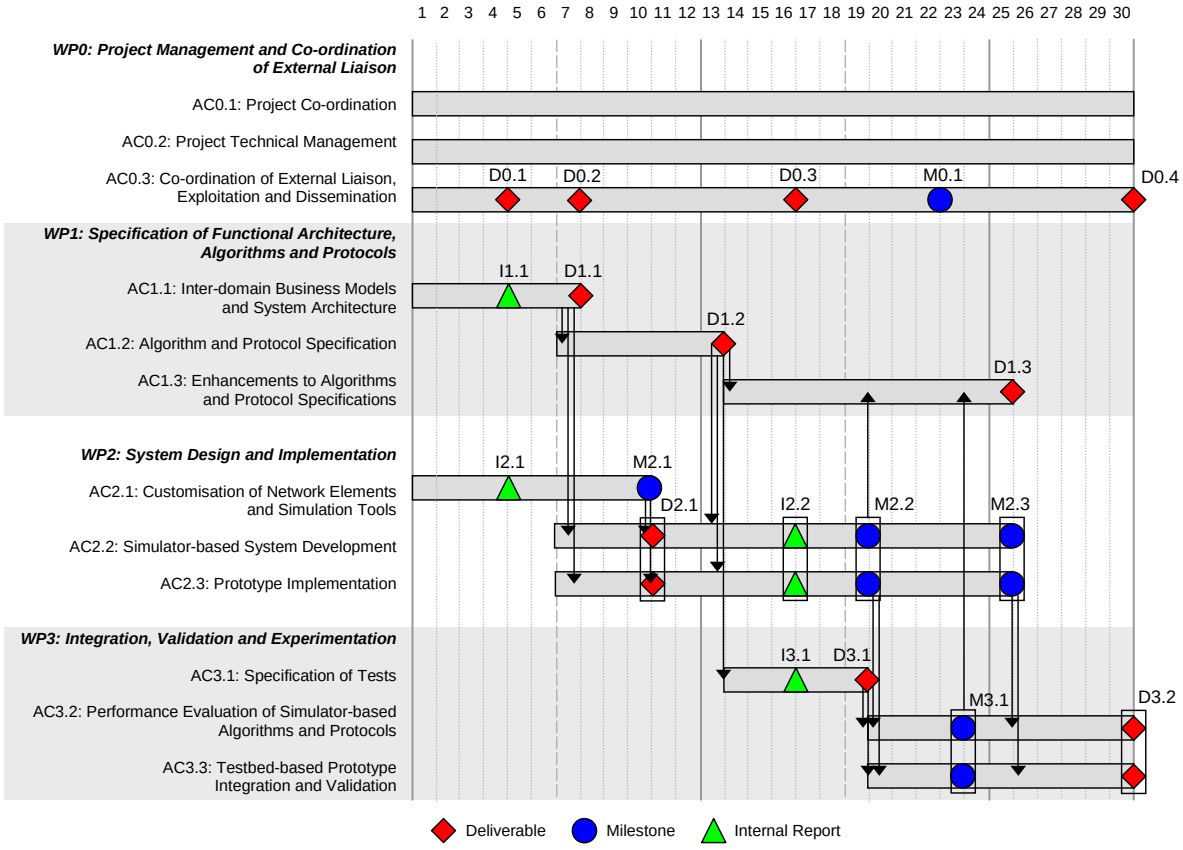

Figure 2 Project Gantt chart: overall schedule of work

An indicative milestone, M2.2, was placed at month 19 (May 2005) to show that WP2 software releases would be made before the end of the activities of the workpackage. This was included to allow experimentation to begin on initial aspects of the MESCAL solutions and to enable feedback from this preliminary experimentation work to influence algorithm and protocol revisions in WP1 and the implementation of refined components in WP2. The following section refines this initial view by identifying a more detailed set of milestones for software releases, according to experimentation plans.

2.3.2.1 Milestones

WP3 is currently defining the experimentation plans of the project and has identified the following sets of tests together with the dates they should commence. The precise schedule of the tests is to be specified in detail in the experimentation plan in deliverable D3.1.

Testbed-based tests (refer to section 3 for an overview of the objectives and scope of each phase):

- *phase 1, initial validate the deployment of the loose solution option:* April 2004
- *phase 2, validation of the q-BGP:* September 2004
- *phase 3, validation of the path computation approach for the hard service option:* November 2004
- *phase 2bis (optional):* from February 2005
    - SLS Management configuration of testbed
    - Off-line Inter-domain TE configuration of testbed
- Solution option 1, 2 and 3 validity tests will start in February 2005.

Simulator-based tests (refer to section 4 for an overview of the objectives and scope of each phase and a more detailed schedule of evaluation tests per subcomponent):

A. static network simulation environments:

- *Off-line Inter-domain TE*: Binding Activation evaluation from April 2004, Binding Selection evaluation from July 2004, Resource Optimisation from August 2004, and integrated tests from October 2004.
- *Off-line Intra-domain TE*: Resource Optimisation evaluation from July 2004, Resource reconfiguration scheduling from September 2004, integrated tests from January 2005.
- *Off-line Intra- and Inter-domain TE interworking*: January 2005
- *SrNP (pSLS Ordering/Order Handling):* September 2004

B. dynamic network simulation environments:

- *qBGP*: August 2004
- *SLS invocation handling*: intra-domain tests from April 2004, inter-domain tests from July 2004.
- *Multicast*: phase one from April 2004, intra-domain evaluation from August 2004 and inter-domain evaluation from December 2004.
- *Off-line Inter-domain TE and qBGP interworking*: January 2005
- *Off-line Intra-domain TE and qIGP interworking:* January 2005

In order to achieve these milestones the implementation tasks have been scheduled to complete by the dates shown in Table 2. It should be noted that many of the implementation activities are due to have several phases. The table specifies the dates of the initial releases of the components. Refer to sections 3 and 4 for more details and a detailed gantt chart of the implementation tasks is due to be produced in I2.2, in month 16.

2.3.3 Allocation of tasks to partners

The final part of the implementation plan is the distribution of development activities between the project partners. This is shown in Table 3 together with the effort allocated to each partner for each task.

	End of development (*)		
	testbed network	simulated network	management plane
Traffic Engineering			
Off-line inter-domain TE			
Binding selection	-	-	Jun-04
Binding activation	-	-	Apr-04
Inter-domain resource optimisation	-	-	Jul-04
Off-line multicast TE	-	-	Nov-04
Path computation algorithm	-	-	Nov-04
PCS interactions	-	-	Nov-04
Off-line intra-domain TE			
Network reconfiguration scheduler	-	-	Aug-04
Intra-domain resource optimisation	-	-	Jun-04
Off-line multicast TE	-	-	Jul-04
Dynamic inter-domain TE			
qRIB	Sep-04	Jul-04	-
qBGP interactions	Sep-04	Jul-04	-
Route selection	Sep-04	Jul-04	-
Dynamic intra-domain TE (IGP extensions)	-	Nov-04	-
Network repository	-	-	Jul-04
Data plane			
Traffic conditioning	Sep-04	-	-
PHB enforcement	Sep-04	-	-
IP forwarding/qFIB	Sep-04	-	-
SLS management			
cSLS ordering	-	-	Jun-04
pSLS ordering	-	-	Sep-04
SLS order handling	-	-	Sep-04
SLS repository	-	-	Sep-04
SLS-based demand derivation tools (traffic forecast)	-	-	Sep-04
SLS invocation handling	-	Mar-04	-
Adaptors to testbed/simulation environment			
SLS management -> testbed	Nov-04	-	-
Off-line inter-domain TE -> testbed	Nov-04	-	-
SLS management -> Off-line intra- and inter-domain TE (traffic forecast)	-	-	Sep-04
Off-line inter-domain TE -> Off-line intra-domain TE (traffic forecast)	-	-	Nov-04
Off-line inter-domain TE -> Dynamic inter-domain TE (qBGP)	Nov-04	Nov-04	-
Multicast TE -> network simulator	-	Oct-04	-
Experimentation tools			
Traffic generation tools	Sep-04	-	-
Bulk c/pSLS order emulator	-	-	Nov-04
Network topology generator	-	Jun-04	-
Results analysis tools	Sep-04	Sep-04	Sep-04

Table 2 Schedule of implementation tasks

	FTRD	TRT	UCL	UniS	Algo	*Total*
Traffic Engineering						
Off-line inter-domain TE						
Binding selection	-	-	-	1.5	-	*1.5*
Binding activation	-	-	-	1.5	-	*1.5*
Inter-domain resource optimisation	-	-	-	4	-	*4*
Off-line multicast TE	-	-	-	2	-	*2*
Path computation algorithm	2	-	-	-	-	*2*
PCS interactions	5	-	-	-	-	*5*
Off-line intra-domain TE						
Network reconfiguration scheduler	-	-	1	-	-	*1*
Intra-domain resource optimisation	-	-	3	-	-	*3*
Off-line multicast TE	-	-	-	2	-	*2*
Dynamic inter-domain TE						
qRIB	3	1	0.5	-	-	*4.5*
qBGP interactions	5	-	2.5	-	-	*7.5*
Route selection	2	-	1	-	-	*3*
Dynamic intra-domain TE (IGP extensions)	-	-	0.5	-	-	*0.5*
Network repository	-	-	0.5	-	-	*0.5*
Data plane						
Traffic conditioning	-	1	-	-	-	*1*
PHB enforcement	-	1	-	-	-	*1*
IP forwarding/qFIB	2	1	-	-	-	*3*
SLS management						
cSLS ordering	-	-	-	-	0.5	*0.5*
pSLS ordering	-	-	-	-	4	*4*
SLS order handling	-	-	-	-	6	*6*
SLS repository	-	-	-	-	1	*1*
SLS-based demand derivation tools (traffic forecast)	-	-	-	-	1.5	*1.5*
SLS invocation handling	-	-	-	3	-	*3*
Adaptors to testbed/simulation environment						
SLS management -> testbed	-	2	-	-	-	*2*
Off-line inter-domain TE -> testbed	-	2	-	-	-	*2*
SLS management -> Off-line intra- and inter-domain TE (traffic forecast)	-	-	-	-	2	*2*
Off-line inter-domain TE -> Off-line intra-domain TE (traffic forecast)	-	-	-	-	1	*1*
Off-line inter-domain TE -> Dynamic inter-domain TE (qBGP)	-	2	0.5	-	-	*2.5*
Multicast TE -> network simulator	-	-	-	1	-	*1*
Experimentation tools						
Traffic generation tools	-	1.2	-	-	-	*1.2*
Bulk c/pSLS order emulator	-	-	-	-	2	*2*
Network topology generator	-	-	1	-	-	*1*
Results analysis tools	-	1.5	0.5	-	2	*4*
Total effort	***19***	***12.7***	***11***	***15***	***20***	***77.7***

Table 3 Allocation of implementation tasks to project partners

3 OVERVIEW OF TESTBED-BASED IMPLEMENTATION ACTIVITIES

One of the major objectives of the MESCAL project is to develop and evaluate the concepts, the algorithms and the architecture that have been specified by the project for deploying an end-to-end QoS delivery chain. These specifications are/will be the subject of several MESCAL deliverables, especially [D1.1], [D1.2] and [D1.3].

In order to achieve this goal and to better control implementation risks, the MESCAL testbed implementation plan introduces progressive implementation phases and milestones for the development of a "proof of concept"-oriented testbed.

Three phases have been defined. Each phase has been hereafter described in terms of:

- Objectives;
- Functional blocks it will impact;
- Milestone resulting of its completion.

MESCAL testbed will not implement the whole MESCAL functional system, but will focus on some key aspects of the MESCAL specifications. Some functional blocks will be implemented completely and others partially. Some blocks will be used without modifying the native behaviour of existent implementations. In this respect, all functional blocks impacted by the implementation phase have been assigned a colour reflecting the importance of the modifications to make. Three levels of modification have been introduced as described below:

- Green: no modifications will be made to the current behaviour of the functional block for the related phase.
- Yellow: the output of this block is mandatory but will not be implemented. Only emulation or partial implementation will be made for the related phase.
- Red: all necessary modifications will be made to the functional block for the related phase.

The testbed will be located in the FTR&D premises. Further information about the configuration and the set up of this testbed will be provided by WP3 in internal report I3.1, *Definition of testbed infrastructure for prototype integration and validation*, due in month 16 and in deliverable D3.1, *Specification of test campaigns and experimentation plans for performance analysis and prototype validation*, due in month 19.

3.1.1 Phase 1

3.1.1.1 Goals

The objective of this first phase is to validate the deployment of the loose solution option in a very simple context. This will allow to validate the meta-QoS-class approach, the architecture of the testbed, and to qualify tools and functions (shaping, forwarding, routing, DSCP marking) embedded in Linux routers.

In order to achieve this goal it will be necessary to:

- Deploy an experimental testbed including several ASs exchanging inter-domain routing information between them. This step consists mainly in:
  - Building an experimental testbed made of several autonomous systems.
  - Running and configuring BGP (Border Gateway Protocol) between ASs
  - Running and configuring intra-domain routing protocols (OSPF for instance)
- Implement the meta-QoS-classes within separate autonomous systems. This consists in:

- Defining QoS classes (l-QC) for each AS configured in the previous step
- Classify these l-QCs with regard the meta-QoS-class paradigm

This phase will not alter or create any of the MESCAL functional blocks. In particular, no QoS information will be carried in the BGP UPDATE messages.

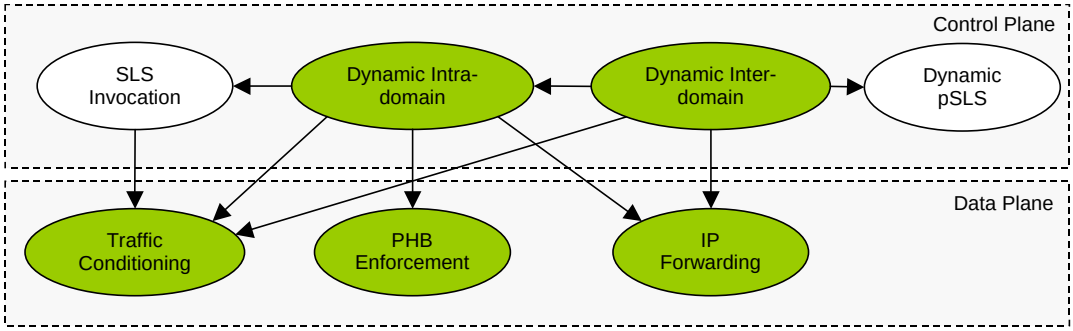

Figure 3 Overview of the functional blocks involved in phase 1

As shown in the figure above, the functional blocks involved within this phase are:

- Dynamic intra-domain: within the context of MESCAL we will use the OSPF protocol as an intra-domain routing protocol with TE extensions [OSPF-TE].
- Dynamic inter-domain: the BGP-v4 will be used as an inter-domain routing protocol.
- Traffic conditioning and PHB Enforcement:

The ZeboS Routing Suite (version 5.3 or latter) will be the base routing stack activated on the Linux routers. Linux Network control Traffic will be used to configure QoS functions in Linux routers.

3.1.1.2 Milestone

Experimentation for this phase will begin in April 2004.

3.1.2 Phase 2

3.1.2.1 Goals

The objectives of this phase are to validate the q-BGP (both q-eBGP and q-iBGP) specification together with the IP forwarding for the loose solution option and to make available a demonstration platform on which phase 3 will rely on. This phase will be the basis of q-BGP functional test campaigns that will aim to validate the q-BGP specifications and check if the MESCAL requirements were met. In addition, interoperability tests will also been achieved within this phase, especially to check if a non q-BGP-enabled router will treat a q-BGP message and vice versa. For this aim, we will make use of traffic generators and embed in the platform non q-BGP-enabled routers.

In order to achieve this goal the following will be achieved:

- Implement a QoS-inferred BGP protocol (q-BGP). This will mainly consists in:
  - Implementing the "QOS_NLRI" and the "QoS Service Capability" attributes.
  - Modifying the BGP route selection process in order to take into account the QoS performance characteristics.
- Modify the forwarding functional block so that IP forwarding can be made on a per DSCP-plane basis, each DSCP plane representing a particular Meta-QoS-Class.

- Activate q-BGP as an inter-domain routing protocol in the testbed deployed within the phase 1. This step will consist mainly in running and configuring the above-developed q-BGP in the testbed.

The Figure 4 shows the functional blocks that will be involved in this second phase.

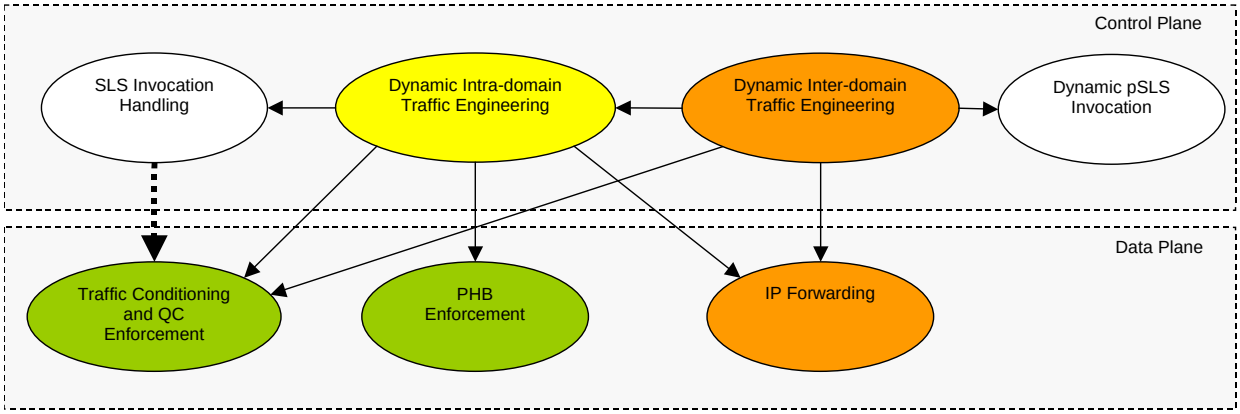

Figure 4 Overview of the functional blocks involved in phase 2

The dynamic inter-domain traffic engineering should take into account the QoS information. The routing selection process should be QoS-based.

For these developments we will make use of the ZeboS Routing Suite (version 5.3 or latter). This suite, once modified, will provide enhanced inter-domain routing capabilities and will be activated on top of Linux routers. Developments of the enhanced IP forwarding functional block will rely on Linux kernel (the latest stable version is 2.6.1).

This phase aims also to validate if the control plane can enforce decisions made by the offline TE blocks. The communication at the interface between the two layers will be achieved manually.

Completion of this phase will lead to the availability of an experimental loose solution option implementation, which in itself will allow validating some of the key MESCAL concepts.

3.1.2.2 Milestones

Implementation for this phase is due to be completed by September 2004.

3.1.3 Phase 3

3.1.3.1 Goals

The objective of this phase is to validate the path computation approach for the hard service option with the aim of building end-to-end inter-domain LSPs. More particularly, this phase will allow validating the interaction between the path computation process and q-BGP and also the pertinence of using QoS information learned by this way.

In order to achieve this goal, MESCAL will:

- Implement PCS: this step focuses on the implementation of a centralized entity that will be developed to build end-to-end LSP.
- Implement the PCS Communication Protocol (PCP): this protocol is the communication protocol used between PCS and will be developed in Java or C/C++.

The Figure 5 shows the functional blocks that will be involved in this third phase.

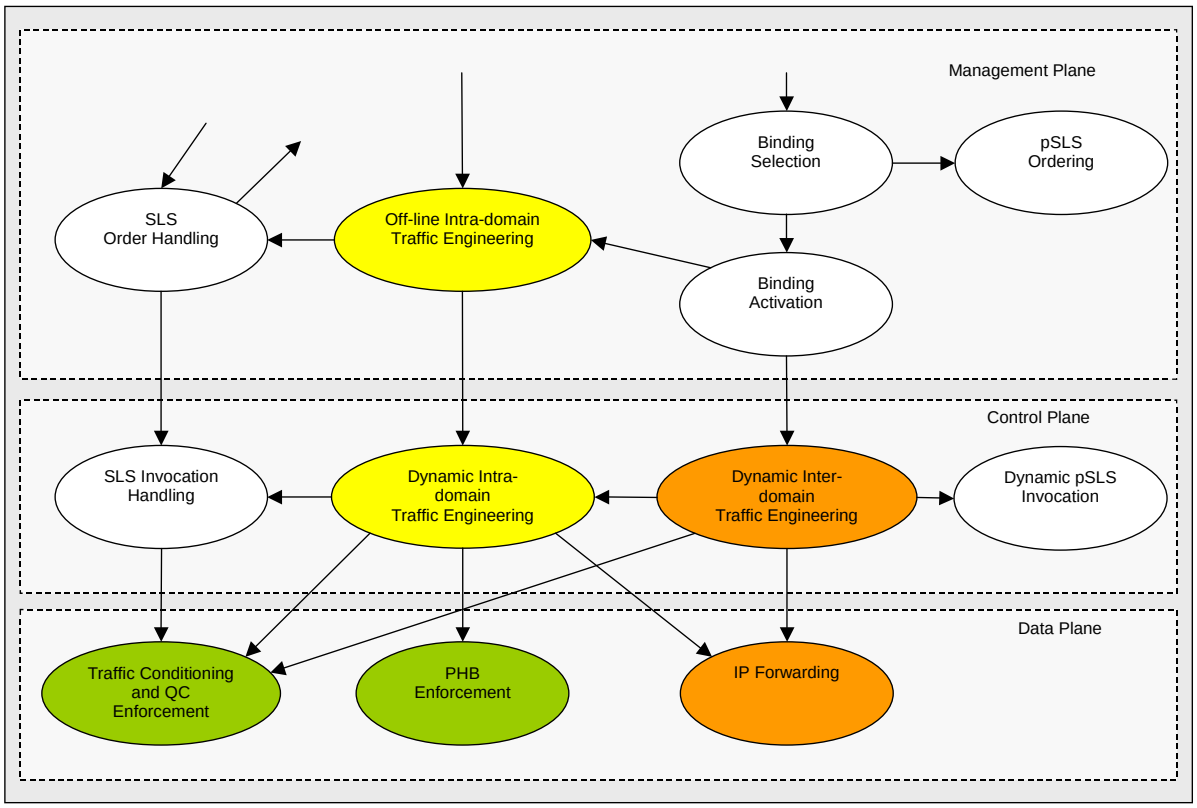

Figure 5 Overview of the functional blocks involved in phase 3

Within this phase, we will implement the hard solution option above the loose solution option (except some enhanced management functions). The establishment of inter-domain LSPs will be initiated by PCSs. Cisco routers will be integrated in this phase when required IOS features are available.

During this phase, we will make use of RSVP-TE that will be used for requesting the creation of LSPs. Intra-domain routing (like OSPF-TE) will be used to find available routes when creation of loose LSP is requested by the PCS.

3.1.3.2 Milestones

Implementation for this phase is due to be completed by November 2004.

4 OVERVIEW OF SIMULATION-BASED IMPLEMENTATION ACTIVITIES

4.1 SLS Invocation Handling (Admission Control)

4.1.1.1 Objectives

SLS Invocation Handling is an online component that is responsible for controlling the amount of traffic injected into the network so that conformant users achieve predefined performance objectives, as these are specified in the SLSs. The term 'users' can correspond to either individual customers, e.g. home users, universities, organizations, or to entire ISP domains. In the former case, cSLSs describe the required performance guarantees, whereas in the latter case, pSLSs describe the required traffic treatment.

4.1.1.2 Design

In order to evaluate SLS invocation handling algorithms, packet-level simulations are required for a variety of real-time and elastic applications and for aggregated streams composed by individual applications. The network simulator (ns-2) will be used for all simulations, since it is a free available packet-level simulator that supports a variety of applications and can be extended to support additional applications if needed. The simulations will first focus on validating algorithms for intra-domain SLS invocation handling and will then focus on the inter-domain case.

4.1.1.3 Interworking

All components needed for simulating the developed SLS invocation handling algorithms will be implemented as standalone oTCL scripts (oTCL is the interface language of ns-2) and additional functions –if needed- will be implemented in C++ code as extensions to the existing ns-2 C++ code.

4.1.1.4 Time Plan

The time plan includes both intra-domain and inter-domain phases.

Phase	Start date	End date
Intra-domain SLS Invocation Handling: customisation / development	November 2003	March 2004
Intra-domain SLS Invocation Handling: evaluation	April 2004	July 2004
Inter-domain SLS Invocation Handling: customisation / development	March 2004	June 2004
Inter-domain SLS Invocation Handling: evaluation	July 2004	October 2004

4.1.2 Offline Inter-domain TE: Binding Selection

4.1.2.1 Objectives

Binding Selection is an offline component that is responsible for selecting e-QC bindings and deciding which pSLSs need to be ordered, modified and ceased in order to meet the inter-domain QoS requirements of the predicted traffic. A principal interface of Binding Selection is with Inter-domain Resource Optimisation (Section 4.1.4). The Binding Selection simulator will implement the algorithms defined in [D1.2]. Since Binding Selection is conducted based on flows (as defined in the

external traffic matrix, eTM), packet-level simulation is not required. MESCAL will consequently develop its own simulation software based on C/C++/Java.

4.1.2.2 Design

The simulator will contain the following modules:

- Pre-processor scenario generator: inter-domain traffic generator (e-TM); network topology generator; e-QC, l-QC and o-QC scenario generator (this module will be a programmable mix of hardcoded values and algorithmic generators for volume testing);
- QC Mapping algorithms: generate e-QC lists based on l-QC and o-QC inputs as described in [D1.2] Section 5.2.1.
- Binding Selection algorithms: perform preliminary sorting and reduction of l-QC and o-QC options, using algorithms defined in [D1.2] Section 5.2.2, in particular by calling Inter-domain Resource Optimisation.

The simulation software will be written in C, C++ or Java.

4.1.2.3 Interworking

Input interfaces

The Binding Selection simulation will take input data, as follows:

- From QoS Capabilities Discovery function block: a list of o-QCs offered by peer domains;
- From QoS-based Service Planning function block: a list of e-QCs that the domain is required to offer (as defined by its business related activities and objectives); and a list of l-QCs that the domain will support internally;
- From TF: the external traffic matrix (eTM).

Since no simulations or implementations of these function blocks are planned, the "scenario generator" module will generate these input data.

Interfaces to Inter-domain Resource Optimisation and Binding Activation

Inter-domain Resource Optimisation will be called to map the inter-domain eTM to the inter-domain traffic matrix.

The output from the Binding Selection simulator will be a set of bindings, in flat file format. The format of these bindings will be such that the file can be used as input to the Binding Activation simulation.

4.1.2.4 Time plan

The Binding Selection simulator software will be released as a single Phase. A second Phase is included to reflect any design amendments that may be produced following the release of [D1.3].

Phase	Start date	End date
Binding Selection Phase 1: development	February 2004	June 2004
Binding Selection Phase 1: evaluation	July 2004	September 2004
Binding Selection Phase 2: D1.3 development updates	September 2004	October 2004
Binding Selection Phase 2: D1.3 evaluation	November 2004	January 2005

4.1.3 Offline Inter-domain TE: Binding Activation

4.1.3.1 Objectives

Binding Activation is an offline component that is responsible for mapping the predicted external (inter-domain) traffic matrix to the inter-domain network resources, satisfying QoS requirements while aiming at optimising the use of network resources across AS boundaries. A principal interface of Binding Activation is with Inter-domain Resource Optimisation (Section 4.1.4). This simulator will implement the algorithms defined in [D1.2]. Binding Activation is done at flow-level; hence its simulation only requires flow-level measurements. This implies that packet-level simulations can be eliminated. To validate and evaluate algorithms for Binding Activation, MESCAL will consequently develop its own simulator without relying on any existing software packages.

4.1.3.2 Design

The simulation software includes the following modules:

- Inter-domain traffic generator (eTM) (this will use the same software as Binding Selection);
- Network topology generator (may use existing tools) (this again will use the same software as Binding Selection);
- Retrieval of Binding Selection results;
- Implementation of binding activation algorithms.

The simulator will be implemented by means of programming languages such as C/C++ or Java.

4.1.3.3 Interworking

Interface 1

Inter-domain traffic and network topologies are generated as output files, using the same software as Binding Selection. These output files are then taken by the Binding Activation algorithm module, together with the Binding Selection results, as input to perform inter-domain traffic engineering.

Interface 2

The Inter-domain Resource Optimisation algorithm module provides an interface to accept a function call from the Binding Activation algorithm module to start inter-domain routing and traffic engineering. The Binding Activation algorithm module also provides an interface to accept results from the Inter-domain Resource Optimisation algorithm module.

4.1.3.4 Time Plan

The Binding Activation simulator software will be released as a single Phase. A second Phase is included to reflect any design amendments that may be produced following the release of [D1.3].

Phase	Start date	End date
Binding Activation: Phase 1 development	November 2003	April 2004
Binding Activation: Phase 1 evaluation	April 2004	July 2004
Binding Activation: Phase 2 D1.3 development	July 2004	September 2004
Binding Activation: Phase 2 D1.3 evaluation	October 2004	December 2004

4.1.4 Offline Inter-domain TE: Resource Optimisation

4.1.4.1 Objectives

Inter-domain Resource Optimisation is an offline component that is responsible for computing an optimal inter-domain traffic engineering solution; it achieves this based on predicted inter-domain traffic (held in the eTM) and inter-/intra-domain resources. The Inter-domain Resource Optimisation simulator will implement the algorithms defined in [D1.2]. In common with the Binding Selection and Binding Activation simulators, the algorithms are based on traffic flows and so packet-level simulation is not required. MESCAL will consequently implement its own software for this simulator.

4.1.4.2 Design

The simulation software includes the following modules:

- Input and validation from Binding Selection / Binding Activation;
- Implementation of Resource Optimisation algorithms. In D1.2, a number of algorithms are defined (random assignment, brute force, genetic algorithm, and custom heuristic), and separate software modules will be written for each algorithms;
- Stub intra-domain traffic engineering algorithm (see below).

4.1.4.3 Interworking

Inter-domain Resource Optimisation will be called by both the Binding Selection and Binding Activation modules, and will return the Inter-domain cost and configuration information.

In a full implementation of MESCAL, Inter-domain Resource Optimisation would interface with Offline Intra-domain Traffic Engineering. For the simulation however we do not intend to simulate the intra-domain functionality. Consequently, we will build a stub software component that will assign intra-domain traffic flows using a simple algorithm such as CSPF and perform a simple (not optimised) calculation of the intra-domain cost Φ.

4.1.4.4 Time plan

The Inter-domain Resource Optimisation simulator software will be released as a single Phase, but with code for each algorithm staggered. A second Phase is included to reflect any design amendments that may be produced following the release of [D1.3].

Phase	Start date	End date
Resource Optimisation: Phase 1 development	March 2004	July 2004
Resource Optimisation: Phase 1 evaluation	August 2004	October 2004
Resource Optimisation: Phase 2: D1.3 development	October 2004	December 2004
Resource Optimisation: Phase 2: D1.3 evaluation	January 2005	March 2005

4.1.5 Offline Intra-domain TE: Resource Optimisation

4.1.5.1 Objectives

Resource Optimisation is the Intra-domain TE component responsible for computing OSPF link weights based on traffic forecast demand matrices. The simulations should serve as a proof of concept as well as giving insight into the following questions,

- Algorithm stability and convergence time
- Algorithm scalability: dependence on network size and number of DSCP routing planes

In addition the simulations should allow quantification of cost functions responsible for algorithm convergence, QoS constraints and intra/inter-domain optimisations. (For further details on test requirements refer to relevant sections in D1.2)

4.1.5.2 Design

The simulator will contain the following modules:

- Scenario generator: intra-domain traffic generator (i-TM); network topology generator
- Resource Optimisation algorithm [D1.2] will be implemented in the several steps
    - Initial algorithm comprising all essential features
    - Advanced features such as cost functions for convergence and termination and Techniques for few weight changes

The simulation software will be written in C, C++ or Java.

4.1.5.3 Interworking

The Resource Optimisation block has only one interface. It is controlled by the Network Reconfiguration Scheduler. This interface is used to signal requests for resource optimisation and passes a handle to all necessary information, such as an iTM, which can then be retrieved from a database. The return value to a resource optimisation request is a handle to the location of the resulting link weight matrix.

4.1.5.4 Time plan

Resource Optimisation will be implemented in two phases, the initial version of the algorithms as specified in D1.2 and a second phase reflecting any evolution of the algorithms arising from a more in depth understanding gained during the first phase.

Phase	Start date	End date
Phase 1: Development	February 2004	June 2004
Phase 1: Evaluation	July 2004	September 2004
Phase 2: D1.3 Development Enhancements	September 2004	November 2004
Phase 2: D1.3 Evaluation of Enhancements	December 2004	March 2005

4.1.6 Offline Intra-domain TE: Network Reconfiguration Scheduler

4.1.6.1 Objectives

The Network Reconfiguration Scheduler is the control system component for Intra-domain TE. The simulator will act as supporting measure to the simulation of Resource Optimisation. Network Reconfiguration Scheduler will be responsible for implementing link weights into the network

simulator. It will also contain the scheduler described in D1.2. It is responsible for implementing pre-computed scenarios for network re-optimisation, when it detects (or is notified) that an optimisation is required. Simulations should therefore give insight into the feasibility of the scheduler regarding

- Correct identification of the network state
- Correct selection of pre-computed weight settings
- Disruptiveness of few weight changes compared to a less optimal network

(For further details on test requirements refer to relevant sections in D1.2)

4.1.6.2 Design

The simulation of the Network Reconfiguration Scheduler consist of the following components.

- Front-end processes required for Resource Optimisation such as handling requests for computation
- Scheduler functionality including
  - Periodic network optimisation
  - Disaster recovery schedules e.g. link failures, etc.
- Retrieving weight settings for scenarios and implementing them via Dynamic Intra-domain TE

The simulator will be implemented by means of programming languages such as C/C++ or Java.

4.1.6.3 Interworking

Network Reconfiguration Scheduler is the front-end process to the Intra-domain TE. It therefore has several interfaces.

- Resource Optimisation, initiate Network Optimisation and store handles to results
- Inter-domain Resource Optimisation provides the location to iTM' hypothetical traffic demand matrices for "what-if" scenarios, Network Reconfiguration Scheduler returns handles to iRAM and cost function value
- Network Reconfiguration Scheduler provides the iRAM to the SLS Order Handling
- Network Reconfiguration Scheduler configures Dynamic Intra-domain TE inside the network layer to configure weight settings, Equal Cost Multi Path (ECMP) settings, etc. It receives alerts for events such as link failures.

Since the Traffic Forecast component will not be implemented, iTM and iTM' traffic matrices will have to be generated using the traffic generators planned in section 4.1.5.2 or as part of the Inter-domain TE components.

4.1.6.4 Time Plan

Network Reconfiguration Scheduler will be implemented in two phases, the initial version of the algorithms as specified in D1.2 and a second phase reflecting any evolution of the algorithms arising from a more in depth understanding gained during the first phase. It is expected that the component will be lagging the development of Resource Optimisation, as it is effectively only the front-end to Resource Optimisation and therefore cannot be evaluated by itself.

Phase	Start date	End date
Phase 1 development	June 2003	August 2004
Phase 1 evaluation	September 2004	October 2004
Phase 2 D1.3 development	October 2004	December 2004
Phase 2 D1.3 evaluation	January 2005	March 2005

4.1.7 Multicast

4.1.7.1 Objectives

This objective of this simulation part is to study the empirical performance of the proposed algorithms for QoS multicast routing as well as Intra- and Inter-domain multicast traffic engineering.

4.1.7.2 Design

The simulation task of multicast can be divided into two parts, namely packet level simulation and network level simulation.

The packet level simulation will investigate the impact on multicast packets of different policies for PHB enforcement. The construction of per-PHB trees or hybrid trees results in different end-to-end QoS performance (delay, delay variation, loss probability etc.). This is because, in the former approach packets with different QC requirements are treated with a unified PHB in separate trees, while the latter approach allows multicast packets with lower QC requirements to be treated with higher PHB classes where there exists common path for the traffic coming from the same group. The objective of this simulation is to evaluate the end-to-end QoS difference between the two approaches, and typically to investigate the fairness issue among packet treatments with different PHBs. Hence packet-level based simulators, such as ns-2, will be used in this part of simulation.

The network level based simulation will target intra- and inter-domain multicast traffic engineering aspects. Constraint-based routing is a major solution for achieving efficient traffic engineering, and typically optimised multicast routing can be formulated into the Steiner tree problem, which is NP-complete. Hence most of the further multicast TE approaches (typically routing algorithms) will be evolved from this graph theory problem. Taking this fact into account, the relevant simulation analysis can be achieved through standalone programming. On the other hand, some existing tools can support this type of simulation very well, such as Stanford Graph Base (SGB), and it will be considered as a potential tool for the simulation part at the network level.

4.1.7.3 Interworking

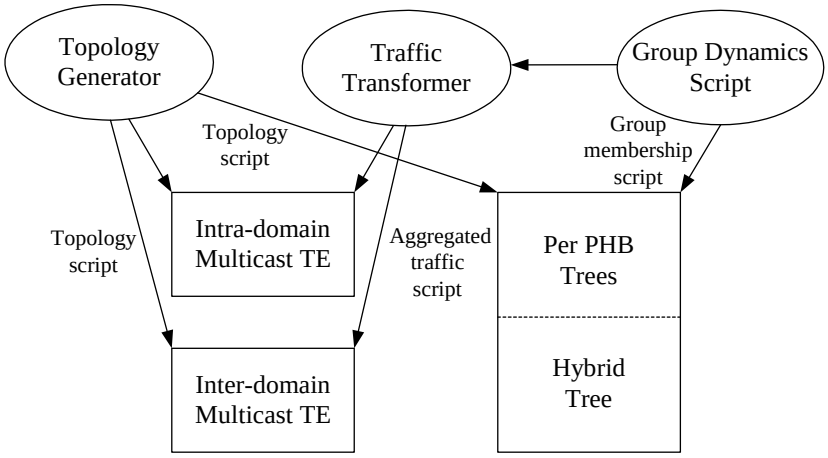

Figure 6. Interworking of multicast simulations

4.1.7.4 Time Plan

Phase	Start date	End date
Multicast Phase 1: End-to-end QoS comparison between hybrid tree vs. per PHB tree (using ns-2) – development	December 2003	March 2004
Multicast Phase 1: ns-2 - evaluation	April 2004	June 2004
Multicast Phase 2: Intra-domain Multicast TE (standalone) - development	May2004	July 2004
Multicast Phase 2: Intra-domain - evaluation	August 2004	October 2004
Multicast Phase 3: Inter-domain Multicast TE (standalone) – development	September 2004	November 2004
Multicast Phase 3: Inter-domain - evaluation	December 2004	February 2005

4.1.8 qBGP simulation

4.1.8.1 Objectives

qBGP is the extension of the current inter-domain routing protocol, BGP, to convey QoS capability between domains. However, adding QoS extensions to BGP may cause instability and scalability problems. Therefore, qBGP will need to be carefully designed to minimize the occurrence or the adverse effects of these problems.

It is necessary to understand the potential impact of new features and extensions as they are added to the BGP. Thus, having sound simulators to evaluate qBGP is crucial. Currently, there are a number of simulators supporting BGP implementation but some of them have not been extensive used and evaluated. Among a number of potential choices, discussed in the simulator section, SSFNET seems the most appropriate simulator for qBGP evaluation. The effort needed to implement qBGP on SSFNET is estimated to be the same compared to the other simulators, while additionally SSFNET has a an existing full BGP implementation, which is well verified and widely used.

4.1.8.2 Design

The qBGP simulation will include the following modules:

- qBGP implementation (BGP extended in accordance with [D1.2]);
- Module for generation of AS-level topologies and inter-domain traffic;
- Module to generate dynamic behaviours, e.g. change of available bandwidth to invoke new BGP advertisements, etc.

4.1.8.3 Inter-working

AS-level topologies and inter-domain traffic are generated as files. These files are then used by the qBGP implementation module as the basis to start the simulations.

The qBGP implementation provides an interface to dynamically accept qBGP advertisements generated by the dynamic behaviours generation module; these advertisements are then propagated by qBGP.

4.1.8.4 Time Plan

Phase	Start date	End date
qBGP: development	May 2004	July 2004
qBGP: evaluation	August 2004	October 2004

5 TECHNOLOGIES, PROTOCOLS, EQUIPMENT AND SOFTWARE

This chapter reviews the software and hardware required for the implementation and simulation tasks. Firstly, the network elements required in a testbed are reviewed, initially with an examination of dedicated routing hardware such as that available from Cisco. The implementation of DiffServ based end-to-end QoS is examined with a description of the available classification and metering techniques as well as queuing disciplines and traffic conditioning for the creation of required PHB (per-hop-behaviour).

For the implementation of a testbed supporting Solution Option 3 MPLS support is examined and QoS related enhancements. For the implementation of Solution Option 1 and 2 the implementation of BGP is examined. Section 5.1.3 there is an examination of software to provide sophisticated routing functionality on top of a general-purpose operating systems like Solaris, Linux or FreeBSD. A range of available packages are considered and compared with an emphasis on functionality and availability.

Section 5 continues with a review of available software for the simulator based experiments. A range of platforms which are specifically for the simulation of networks are considered alongside a few general purpose simulators. To complement the network simulation and experimentation scenarios additional components are also required such as traffic generators and topology generators and these are described next in this section.

Section 5.4 examines the management of network devices through a text based command line interface, SNMP as well as XML. Finally in Section 5.5 various software which could be brought in from previous projects and implementations is described. This software includes components of the TEQUILA project pertaining to network dimensioning, resource management and service management among others. Further the usefulness of PONDER, DIAMETER, LDAP, SNMP and RADIUS is considered to implement policy management, network management and authentication.

5.1 Network Elements and Selection Criteria

Firstly, this section specifies the selection criteria for selection of Network Elements for project experimentation. Secondly, it reviews the routing software that may be used in experimental NEs for use in the MESCAL networking environment.

5.1.1 Selection Criteria

Experimental/Commercial Network Elements will be used to experiment the MESCAL functionality. A number of NEs and their features will be examined against a set of selection criteria set below. It should be stated that we will mainly focus on the edge functionalities rather than on the ones in the core of the network.

5.1.1.1 DiffServ Capabilities

DiffServ approach is to be used within the domains that will construct an inter-domain networking environment for implementing the MESCAL. Thus, DiffServ capabilities such as traffic classification, traffic conditioning, scheduling, etc. in boundary routers are essential features to be taken into account.

5.1.1.1.1 Traffic classification and conditioning

- Classification: support of Behaviour-Aggregate (BA) classification, Multi-Field (MF) classification.
- Metering: support of Meter types (like Average Rate Meter, Token Bucket Meter)
- Re/Marking: Support of process allowing the setting of DSCP in a packet based on some defined rules.

- Shaping support of a process allowing delaying of packets conforming to some defined traffic profile.

5.1.1.1.2 Per Hop Behaviours

Network Elements must support the EF, AF and other PHBs. This should be realised by providing mechanisms for packet scheduling and congestion management at both edge and core routers.

5.1.1.1.3 Queuing/Scheduling and Congestion Management Mechanisms

Different types of queuing/scheduling and buffer and congestion management mechanism supported by NEs need to be examined for the proper implementation of various PHBs. These can include Strict Priority Queuing (SPQ), Class Based Queuing (CBQ), Weighted Fair Queuing (WFQ), and Class Based Weighted Fair Queuing (CBWFQ), and variations of RED algorithm.

5.1.1.2 Traffic Engineering Support

NEs must provide the building blocks for supporting the traffic engineering mechanisms to be developed in MESCAL. For example, the NEs should be capable of providing the MPLS functionality if Inter-AS MPLS tunnels need to be established. The same holds for the functionality that is required for implementing the pure IP-TE.

5.1.1.3 Networking

The supported routing protocols need to be considered. These routing protocols include OSPF, MOSPF, CBT, BGP, MBGP, PIM PIM-SIM, etc.

5.1.1.4 QoS Signalling protocols

Support of QoS signalling and reservation mechanisms such as RSVP, RSVP-TE, etc. need to be considered.

5.1.1.5 Configuration

The structure and the completeness of the configuration and management interface and the supported protocols specify the level of control provided to the NE's operator and thus are very important aspects.

- Static vs. Dynamic configuration: examine the level of automation of the configuration operations
- Configuration protocols: a listing of supported management protocols/tools such as COPS, CLI commands, or any other tool to ease the NE configuration burden
- Configuration interface: a description of the supported COPS client-type, etc
- Configuration data: the structure of the configuration data to be used in order to configure NEs such as PIB and MIB

5.1.1.6 Management

The management interfaces supported by the network equipment e.g. command line interfaces, protocols like SNMP, etc. are an important aspect to be considered.

5.1.1.7 Interoperability

- This is required if different commercial and experimental NEs are used in the same networking environment where they need to inter-operate with each other.

5.1.1.8 Usability/Ease of use

This criterion is mandatory if modifications/enhancement to the source code is required.

- Source code for experimental NEs: The possibilities of using Freeware (open source codes, or freely available non-open source codes) and those source codes requiring licences need to be examined.
- Source code for commercial routers: The possibilities of limited modifications/enhancements to the functions supported by NEs operating system by the vendor need to be considered. These may be needed in order to support and experiment the functionality by required MESCAL.
- Documentation: Detailed and well-explained documentation is essential to facilitate the use/enhancement any source code related to the functions supported by NEs.
- Modularity: When modification/enhancement of NE source code is required, the code modularity is an essential feature in order to facilitate the building of new functionalities and the enhancement of the existing ones.
- Ease of use: this criterion is essential if no prior knowledge of the product is available.

5.1.1.9 Security

This criterion is critical for the confidentiality and integrity of configuration information and also to ensure authorised access to the NEs.

5.1.1.10 Performance

A description of the performance of the NE to execute specific functionality is to be clearly provided. Additional information about stability is also to be taken into account. This issue is essential to predict the behaviour of the NE in (un)usual cases.

5.1.2 Review of Cisco Commercial Routers

This section of the document presents different aspects and features of Cisco IOS that are related to the realisation of MESCAL project.

5.1.2.1 Implementing DiffServ for End-to-End Quality of Service

Cisco IOS QoS software supports three types of service models: best-effort, integrated (IntServ), and differentiated services (DiffServ). DiffServ is a service model devised to offer services with different QoS requirements. The network tries to deliver service based on the QoS specified in each packet using the 6-bit Differentiated Services Code Point (DSCP) in IP packets. The network uses the QoS specification to classify, mark, shape, police traffic, and perform intelligent queuing. Typically, DiffServ is appropriate for aggregate flows because it performs a relatively coarse level of traffic classification.

Cisco IOS QoS includes the following features that support differentiated services:

- Access lists, which performs packet classification through a subset of 5-tuples, IP precedence, QoS group settings, DiffServ Code Points, etc.
- Committed access rate (CAR), which performs metering and policing of traffic, providing bandwidth management.
- Intelligent congestion management and queuing schemes such as Weighted Random Early Detection (WRED) and Weighted Fair Queuing (WFQ) and their equivalent features.
- Low Latency Queuing (LLQ) brings strict priority queuing to Class-Based Weighted Fair Queuing (CBWFQ). Strict priority queuing allows delay-sensitive data, such as voice, to be

dequeued and sent first (before packets in other queues are dequeued), giving delay-sensitive data preferential treatment over other traffic.

- Generic Traffic Shaping (GTS) shapes traffic by reducing outbound traffic flow to avoid congestion.
- Modular QoS Command-Line Interface (MQC) so that you can specify a traffic class independently of QoS policies.

Currently, the DiffServ model defines the use of the DSCP and the four PHBs. The PHBs simply describe the forwarding behaviour of a DS-compliant node. The model does not specify how these PHBs may be implemented. A variety of queuing, policing, metering, and shaping techniques may be used to affect the desired traffic conditioning and PHB. Figure 7 shows the different mechanisms embedded in Cisco IOS.

Figure 7 Various mechanisms embedded in Cisco IOS.

5.1.2.1.1 IOS Features for DiffServ Model Implementation

5.1.2.1.1.1 Traffic Classification and Conditioning

The four traffic-conditioning components are meters, markers, and shapers/droppers. A classifier is necessary for conditioning. *Table 4* shows Cisco routers compliance with traffic conditioning components.

Traffic Conditioning Components	Cisco Mechanism
Classification	ACL/Traffic Class
Metering	CAR/ Traffic Policer/CB Policer
Marking	CAR/ Traffic Policer/CB Marker
Traffic Shaping	GTS (WFQ)/CB Shaper
Dropping	CAR/Traffic Policer/CB Policer

Table 4 Traffic conditioning components of Cisco routers.

These mechanisms are discussed in the following sections.

5.1.2.1.1.2 Classification

Packet classification can be performed in several ways, e.g. physical ingress interface, Layer 2 or Layer 3 address, layer 4 Port number, and application layer, or URL for Web traffic.

Access Control Lists (ACLs) are Cisco's mechanism for classifying the packet entering the router. An IP access list can be defined based on a variety of traffic characteristics, including the type of traffic (Layer 4 protocols and their Port numbers), its source, its destination, and its IP precedence settings. Access Control Lists (ACLs) are used for classifying packets entering the router. Standard IP ACLs use source IP addresses for matching operations. Extended IP ACLs use source and destination addresses for matching operations and optional protocol type and port numbers for finer granularity of control.

- Extended ACLs provide the network with a highly granular filtering mechanism to define edge policies including:
  - Packet classification into classes of service
  - Network resource management including bandwidth allocation and congestion management associated with these classes.
- Traffic classification can be also based on several criteria, including ACLs (IP addresses, ports, etc.), MAC address, IP Precedence value, DSCP values, other predefined traffic classification criteria, input interfaces, and so on.

Emerging multimedia and Web-based applications require deeper packet inspection and cannot be identified by static Layer 4 Port number. Cisco has the following two QoS capabilities in Cisco IOS software that will enable deeper packet inspection.

Network-Based Application Recognition (NBAR) is a classification engine that can recognise a wide range of protocols and applications, including Web-based and client/server applications that dynamically assign Layer 4 port numbers. Once the application is recognised, specific differentiated services for that particular application can be invoked.

Context-Based Access Control (CBAC) is a form of packet filtering that examines not only network and transport-layer information, but also examines application-layer protocol information (such as FTP) to learn about and inspect the state of TCP or UDP sessions. This mechanism dynamically creates and deletes temporary openings in the firewall by temporarily modifying access lists to change packet-filtering criteria. CBAC maintains state information in its own data structures and uses that information to create the temporary entries. State information is used to make intelligent permit/deny decisions. When a session closes, its temporary entry is deleted, and the opening in the firewall is closed.

5.1.2.1.1.3 Traffic Conditioning

In the simplest form of traffic conditioning (and providing PHB for AF classes in the core of a DS-Domain), packets are metered and different actions are taken, depending on whether the packet in question conforms, violates, or exceeds the configured average rate, committed burst (Bc), or excess burst (Be). A packet can simply be transmitted, dropped, or remarked with a different DSCP value (moving it into a lower AF class, or changing its drop precedence value), depending on the configured policy.

Cisco IOS QoS offers two kinds of traffic regulation mechanisms: the rate-limiting feature for policing traffic, and Generic Traffic Shaping (GTS) for shaping traffic. Both policing and shaping mechanisms use the traffic descriptors for a packet, indicated by the packet's classification.

Policers and shapers usually identify traffic descriptors violations in an identical manner. They usually differ, however, in the way they respond to violations:

- A policer typically drops traffic. Rate-limiting policer will either drop the packet or rewrite its IP Precedence.
- A GTS shaper typically delays excess traffic using a buffer, or queuing mechanism, to hold packets and shape the flow when the data rate of the source is higher than expected.

5.1.2.1.1.4 Metering

Committed Access Rate (CAR) is an IP-layer traffic conditioner available in Cisco routers. CAR both enforces traffic contracts and marks packets according to the traffic contract. CAR enforces the bandwidth contracts using token bucket policers, which permit burstiness in a short timeframe, while limiting rates in a longer timeframe.

CAR marks the packets by setting the ToS field (IP precedence) or user-defined QoS group and limits the input or output transmission rate of traffic flows according to an ACL. CAR sets the transmission rate, burst size, exceeded burst size, conformation priority, and exceeded priority. Each interface can have multiple CAR policies corresponding to different types of traffic. The CAR feature uses token bucket filters to measure traffic load and limit sources to bandwidth allocations while accommodating the inherently bursty nature of IP traffic. Token bucket parameters include the ***committed bit rate***, the ***normal burst size*** in bytes (to handle temporary bursts over the rate limit without penalty) and the ***excess burst size*** in bytes. Tokens are added to the bucket at the committed rate and the number of tokens in the bucket is limited by the normal burst size. When traffic arrives at the bucket, if sufficient tokens are available then the traffic is said to *conform* and the corresponding number of tokens is removed from the bucket. Traffic exceeding the normal burst limit but falling within the extended burst limit is handled via a RED-like managed discard policy designed to avoid tail-drop behaviour, to provide a gradual effect for the rate limit, and to allow the traffic source to slow down before suffering repeated packet discards. Arriving packets which fail to conform to the token bucket specification (either because they exceed the excess burst limit or because they fell between the normal burst limit and the excess burst limit and were probabilistically discarded) are handled via the specified *exceed* action.

CAR policies may be utilised at either the ingress or egress of the network. CAR thresholds may be applied by access port, by IP address or by application flow. CAR utilises Extended ACLs to define policies including bandwidth utilisation thresholds under which packet priority is modified or packets are dropped.

Rate policies can be independent i.e. each rate policy deals with a different type of traffic. Alternatively, rate policies can be cascading i.e. a packet may be compared to multiple different rate policies in succession. When there are multiple rate policies, the router examines each policy in the order entered until the packet matches. If no match is found, the default action is to transmit.

The CAR rate limit mechanism is configured with a (bit rate, burst size, excess burst size) parameter list. It is regulated by a token bucket mechanism. Therefore, if a packet arrives at CAR and there are sufficient tokens available, the packet is said to conform. The usual conform action is transmit. If there are insufficient tokens, the packet is said to exceed. The usual exceed action is dropping. CAR can be applied to an entire interface or multiple rate limits, based on access lists, or may be defined on a single interface. Rate limits may be cascaded to create complex rate limits.

5.1.2.1.1.5 Marking

Using ACL and CAR, packets are marked by setting the IP precedence of ToS field in the IP header to allow routers on the end-to-end path to identify and handle packets with the appropriate mechanisms.

5.1.2.1.1.6 Traffic Shaping/Dropping

Sometimes it is better to buffer packets, rather than simply drop them in the case of congestion - especially for UDP-based applications. This can be done generally, by configuring an average-rate, committed burst (Bc), and excess burst (Be) (just as in configuring CAR). However, the biggest difference between CAR and GTS is that packets are buffered in case of congestion and the flow is shaped.

Traffic shaping or traffic limiting is done through policies, which are defined in ACLs. Traffic shaping policies or traffic limiting policies can be created on a router's interface to manage how much of the interface's bandwidth should be allocated to a specific traffic flow. Traffic shaping limits packet bursts, which might congest a network by buffering the packets and sending them onto the network in

a regulated manner. Traffic shaping sets a bandwidth limit for specific types of traffic. With traffic shaping policies, flows are affected even during times of little congestion.

The Cisco's Generic Traffic Shaping (GTS) mechanism is configured with a (bit rate, burst size, excess burst size, queue size) parameter list. It is regulated by a token bucket mechanism. Therefore, if a packet arrives at GTS and there are sufficient tokens available, the packet is transmitted to the output queue of the interface. If there are insufficient tokens available, the packet queues until tokens become available. GTS can be applied to an entire interface so that all traffic leaving the interface is shaped. Alternatively, a number of GTS shapers may operate on an interface, each shaper using an access list to identify traffic for shaping.

Shaping differs from limiting in that it attempts to throttle traffic when it reaches the rate limits. The router buffers some of the traffic bursts. Only when the buffer fills are packets dropped. Whereas, limiting policies do not drop packets until rate limits are reached, then drop all packets that exceed the rate limit.

GTS shapes traffic by reducing outbound traffic flow to avoid congestion. The GTS mechanism is configured with a (committed information rate (CIR), committed burst size (Bc), excess burst size (Be), queue size) parameter list. It is regulated by a token bucket mechanism. Therefore, if a packet arrives at GTS and there are sufficient tokens available, the packet is transmitted to the output queue. If there are insufficient tokens available, the packet queues until tokens become available.

When traffic shaping is enabled, the bit rate of the interface will not exceed the mean rate over any multiple integral of the interval (Bc/CIR). In other words, during every interval, a maximum of burst size can be sent. Within the interval, however, the bit rate may be faster than the mean rate at any given time.

When the Be size equals 0, the interface sends no more than the burst size every interval, achieving an average rate no higher than the mean rate. However, when the Be size is greater than 0, the interface can send as many as Bc plus Be bits in a burst, if in a previous time period the maximum amount was not sent. Whenever less than the burst size is sent during an interval, the remaining number of bits, up to the Excess Burst size, can be used to send more than the burst size in a later interval.

Generally for GTS, the shaping queue is a weighted fair queue. GTS can be applied to an entire interface so that all traffic leaving the interface is shaped. Alternatively, a number of GTS shapers may operate on an interface, each shaper using an access list to identify traffic for shaping.

As GTS shapes traffic, it changes the packet inter-arrival times. GTS must not be used for real-time traffic.

5.1.2.1.1.7 Percentage-Based Policing and Shaping

This feature is released in IOS 12.2(13)T and provides the ability to configure traffic policing and traffic shaping based on a percentage of bandwidth available on the interface. Configuring traffic policing and traffic shaping in this manner enables customers to use the same policy map for multiple interfaces with differing amounts of bandwidth.

5.1.2.1.2 Per Hop Behaviour (PHB)

Cisco does not explicitly offer EF and AF PHBs, but instead provides a toolbox of traffic management mechanisms from which a network operator can construct PHBs. Cisco provides mechanisms of packet scheduling and discard policy applied at each router's output queue, which construct PHB. The IP precedence value of a packet tells the router which PHB to apply to each packet.

PHBs are enforced at the edge and core routers, depending on the packet marking with the appropriate DSCP. In Cisco IOS software, EF can be implemented using LLQ (Low Latency Queuing). AFxy PHBs can be implemented using a combination of CBWFQ (Class Based Weighted Fair Queuing), and WRED (Weighted Random Early Detect)/CAR.

Table 5 shows the queuing mechanisms supported in the Cisco routers.

Scheduling/Queuing Mechanism	Classification Method	Queuing Strategy
FIFO	–	Single Queue per I/F
PQ	ACL	High, medium, normal, low priority Queues
FBWFQ	A weighted Queue (Q) per traffic micro flow	A weighted Queue per traffic flow
CBWFQ	ACL, etc. (defining up to 64 traffic classes)	Single Queue per traffic class
LLQ	ACL, etc. (defining traffic classes)	A single strict high PQ
DRR	IP precedence Mapping	Multiple (8) Queues with RR mechanism
MDRR	IP precedence Mapping (IP precedence 7 for LLHP Queue)	A high PQ with multiple (7) Queues with RR mechanism
WRED	Combines capabilities of RED with IP Precedence/DSCP	–
Flow WRED[1]	Combines capabilities of RED with traffic micro flows	–

Table 5 Queuing mechanisms supported in the Cisco routers.

5.1.2.1.3 Queuing/Scheduling and Congestion Management Mechanisms

Scheduling or queuing mechanisms supported by Cisco include FIFO, PQ, FBWFQ, CBWFQ, PQ-CBWFQ, DRR, MDRR, WRED, and flow WRED. Queuing mechanisms are part of an interface's characteristics. Queuing techniques only affect traffic when an interface is congested, or in the case of WRED, when traffic exceeds a certain threshold.

5.1.2.1.3.1 Priority Queuing (PQ)

When PQ is configured in an interface, the Cisco IOS creates four output queues to which it attributes High, Medium, Normal, and Low priority. The queue scheduling algorithm services each queue in priority order and will only service a queue if there is no data in any higher priority queue. Each queue operates in FIFO mode. Assigning data to a particular queue is performed by means of access lists and priority lists.

5.1.2.1.3.2 Flow-Based Weighted Fair Queuing (FBWFQ)

Fair queuing provides a queue for each traffic flow and services these in such a way as to ensure that no queue gets more than its fair share of available bandwidth during congestion. Access to the link is arbitrated by computing a completion time for each packet in each queue, based on its length and sending each in completion time order. Under conditions of congestion, this mechanism ensures that each flow gets an even share of the available bandwidth and that large packets do not unduly delay small packets. Fair queuing, in fact, favours small packets.

[1] Flow-WRED maintains state about active micro flows (up to 16) that have packets in the output queues. Flow-based WRED determines which flows monopolise resources and it more heavily penalises these flows. It maintains a count of the number of active flows that exist through an output interface. Given the number of active flows and the output queue size, it determines the number of buffers available per flow. To allow for some burstiness, flow-based WRED scales the number of buffers available per flow by a configured factor and allows each active flow to have a certain number of packets in the output queue. This scaling factor is common to all flows. The outcome of the scaled number of buffers becomes the per-flow limit. When a flow exceeds the per-flow limit, the probability that a packet from that flow will be dropped increases.

Cisco's WFQ takes FQ mechanism a step further and places a weight on each flow. This allows the share of the bandwidth to be biased in favour of high priority flows over low priority flows. The weighting mechanism is based on the IP precedence bits carried in the IP header. The actual bandwidth, that each flow gets, will depend on how many flows of what weight are present at the time. Let "a_i" denote the IP precedence value of flow "i" and "n" denote the number of active traffic flows. Theoretically, each traffic flow gets the following share of link bandwidth based on its IP precedence value:

$$\text{Bandwidth share of flow i} = \frac{a_i + 1}{\sum_{j=1}^{n} (a_j + 1)} * \text{Link bit rate}$$

Distributed WFQ is a special high-speed version of WFQ designed initially for Cisco 7500 VIP (Versatile Interface Processor) processors.

5.1.2.1.3.3 Low Latency Queuing (LLQ) for the EF PHB

Delay sensitive traffic, such as VoIP, needs to be given strict priority, along the packet path. To make this happen, the LLQ can be used at each hop. To ensure that excess voice traffic does not interfere with traffic of other classes, this priority queue can be policed.

The low latency queuing (LLQ) feature brings strict priority queuing to CBWFQ. Low Latency Queuing allows the operation of a single priority queue within which individual classes of traffic can be placed. To enqueue class traffic to the priority queue, the user configures the **priority** command for the class after specifying the named class within a policy map. Within a policy map, you can give one or more classes priority status. When multiple classes within a single policy map are configured as priority classes, all traffic from these classes is enqueued to the same, single, priority queue.

When user specifies the **priority** command for a traffic class, it takes a bandwidth argument that gives guaranteed maximum bandwidth in kbps for packets belonging to the class under worst-case congestion scenarios and burstiness in bytes. If excess bandwidth is available, the priority class will be allowed to utilise the bandwidth. If no excess bandwidth is available, the priority traffic will be constrained to the configured rate via packet drops. Each individual class that is configured for priority behaviour will have its traffic constrained to its individual rate. When a class is constrained to its individual rate, the traffic is permitted a certain amount of burstiness. The *bytes* parameter in the **priority** command specifies, in bytes, the amount of traffic allowed to pass through the token bucket as a one-time burst in excess of the token bucket drop parameters.

Supported Platforms for LLQ are Cisco 7500/RSP series. The LLQ feature was introduced 12.0(5)XE5. The *bytes* argument was introduced for the **priority** command for Release 12.1(2)E.

WRED **random-detect** command cannot be used with the **priority** command. In addition, because policing is used to drop packets and queue limit is not imposed, the **queue-limit** command cannot be used with the **priority** command.

When congestion occurs, traffic destined for the priority queue is metered to ensure that the bandwidth allocation configured for the class to which the traffic belongs is not exceeded.

Priority traffic metering has the following qualities:

- It is much like the rate limiting feature of committed access rate (CAR), except that priority traffic metering is only performed under congestion conditions. When the device is not congested, the priority class traffic is allowed to exceed its allocated bandwidth. When the device is congested, the priority class traffic above the allocated bandwidth is discarded.
- It is performed on a per-packet basis, and tokens are replenished as packets are sent. If not enough tokens are available to send the packet, it is dropped.
- It restrains priority traffic to its allocated bandwidth to ensure that non-priority traffic, such as routing packets and other data, is not starved.

With metering, the classes are policed and rate-limited individually. That is, although a single policy map might contain four priority classes, all of which are enqueued in a single priority queue, they are treated as separate flows with separate bandwidth allocations and constraints.

5.1.2.1.3.3.1 LLQ with Priority Percentage Support

This feature is released in IOS 12.2(2)T and allows configuring bandwidth as a percentage within low latency queuing (LLQ). Specifically, it is possible to designate a percentage of the bandwidth to be allocated to an entity (such as a physical interface) to which a policy map is attached. Traffic associated with the policy map will then be given priority treatment. This feature also allows for specification of the bandwidth percentage to be allocated to non-priority traffic classes. This feature modifies two existing commands, bandwidth and priority.

5.1.2.1.3.4 Class-Based Weighted Fair Queuing (CBWFQ) for the AF PHB

CBWQ is used to configure classes of queued traffic and provide bandwidth guarantees to those classes. CBWFQ allows the user to specify the exact amount of bandwidth to be allocated for a specific class of traffic. Taking into account available bandwidth on the interface, up to 64 classes can be configured and the bandwidth distribution among them can be controlled.

For CBWFQ, traffic classes are defined based on match criteria including protocols, access control lists, and input interfaces. A queue is reserved for each class. CBWFQ specifies the amount of bandwidth in kbps to be assigned to each class and the queue-limit for the class. The bandwidth assigned to a class is the minimum bandwidth delivered to the class during congestion.

Tail drop is used for CBWFQ classes unless the user explicitly configures policy for a class to use WRED.

CBWFQ allows the exact amount of bandwidth to be specified for a specific class of traffic. Once a class has been defined according to its match criteria using ACL, its characteristics, such as bandwidth, weight and maximum packet limit can be defined. For CBWFQ, each class is associated with a separate queue. A specific minimum amount of guaranteed bandwidth can be allocated to the class as a percentage of the link or in kbps. Unutilised Bandwidth can be shared by other classes in proportion to their assigned weights.

Class Based Weighted Fair Queuing using the MQC allows for carving out bandwidth among the various classes defined. Bandwidth may be allocated to each class on an absolute basis (specified in Kbps), or as a percentage of the interface bandwidth (to which this policy will be applied). Within an AF class, packets can be dropped based on the drop precedence scheme using Weighted Random Early Detect (WRED).

5.1.2.1.3.5 Weighted Random early Detection (WRED)

WRED sets the minimum and maximum queue depth thresholds and drop probabilities for each class of traffic (based of value of IP precedence). When a packet arrives to the queue, the following events occur:

The average queue size is calculated. If the average is less than the minimum queue threshold, the arriving packet is queued. If the average is between the minimum queue threshold and the maximum threshold, the packet is either dropped or queued, depending on the packet drop probability. If the average queue size is greater than the maximum threshold, the packet is automatically dropped.

Average queue size = (old_average * (1-1/2^n)) + (current_queue_size * 1/2^n) where n is the exponential weight factor, a user-configurable value.

The packet drop probability is based on the minimum threshold, maximum threshold, and mark probability denominator. When the average queue depth is above the minimum threshold, WRED starts dropping packets. The rate of packet dropping increases linearly, as the average queue size increases, until the average queue size reaches the maximum threshold.

WRED (IOS release 12.1(5) T) can utilise all six bits of the DSCP field, to make a decision on packet drop probability. Up to three-drop precedence levels are defined with the AF PHB, using the two drop-precedence bits in the DSCP. WRED min and max drop-thresholds can now be configured based on the DSCP field. Thus, the AF PHB defined in RFC-2597 can be implemented using policing/shaping, and WRED. WRED will use the DSCP value to calculate the drop probability.

5.1.2.1.3.6 Deficit Round Robin (DRR)

Packets are classified based on ToS field (IP precedence) and inserted in the appropriate queues. Active queues are serviced in round-robin order. Each queue has assigned to it a configurable value called a service quantum measured in bytes. A service quantum provides a measure of how much traffic should be handled from the queue in each round. Packets from that queue are serviced until their cumulative length (byte count) exceeds the service quantum. A deficit counter, which is a memory mechanism designed to enhance fairness and packet size independence, is used as a credit mechanism. The deficit counter value is added to the service quantum to determine the measure of service available for each queue during each round.

5.1.2.1.3.7 Modified Deficit Round Robin (MDRR)

Because it is difficult to scale up WFQ for high-speed transport links running at STM1, STM4, and higher rates, MDRR, a variant of DRR, is implemented for the Cisco 12xxx GSR routers to support delay-sensitive voice traffic. MDRR features a high-priority queue for real-time traffic that is treated differently from the other queues associated with service classes. Except for the high-priority queue, MDRR services all queues in round-robin fashion. MDRR can be used in either of the following two modes: strict priority or alternate priority.

In strict priority mode, if the high-priority queue contains packets, it is serviced first until all of its packets are sent and the queue is empty. Then the other queues are serviced in round-robin fashion according to the DRR algorithm.

In the alternate mode, the service alternates between the high-priority queue and the other queues. Packets are serviced from the high-priority queue and then from an active queue that is selected from among the other queues in round-robin fashion. This process of servicing the high-priority and the next queue in the list of other queues is repeated until the queues are empty.

5.1.2.1.3.8 Flow Random Early Detection

This technique provides a mechanism to penalise the flows that do not respond to Weighted Random Early Detection (WRED) drops. This feature is provided as an extension to the existing WRED functionality and can be turned on after WRED is turned on.

Flow-WRED ensures that no single flow can hog all the buffer resources at the output interface queue. With just WRED, this can occur in the presence of traffic sources that do not back off during congestion. Flow-WRED maintains minimal information about the buffer occupancy per flow. Whenever a flow exceeds its share of the output interface buffer resource, the packets of the flow are penalised because the probability of their drop (by WRED) is increased.

5.1.2.1.3.9 Class-Based QoS: Marking, Policing, Shaping, Statistics

5.1.2.1.3.9.1 Class-Based Marking

This feature enables the user to mark packets' DSCP/IP-precedence values. It also allows association of a local Quality of Service (QoS) group value with a packet. Within each class, based on ACL matching or local QoS group, the user can set the outgoing packet's new DSCP/IP-precedence value. This feature appeared in *IOS 12.1(3) T*.

Associating a packet with a local QoS group allows users to associate a group ID with a packet. The group ID can be used to classify packets into QoS groups based on prefix, autonomous system, and

community string. This QoS group marking can only be used to classify traffic within a router, and cannot be used to mark packets leaving the router.

5.1.2.1.3.9.2 Class Based Policer for DiffServ AF PHB

This policer conforms to RFC-2697, and allows the user to implement the AF PHB, defined in RFC-2597. This policer allows the user to police traffic within a class according to specified parameters. The Class Based Policer allows the user to fully implement the DiffServ AF PHB. The policer is called a Single-Rate Three-Colour Marker (srTCM), since the AF Drop-Precedence bits can be set based on the Bc, and Be. Arriving packets within an AF class can be marked with different drop-precedence bits, depending on whether they (a) fall in less than or equal to the Bc (conform), (b) fall in between Bc and Be (exceed), or (c) fall in greater than Be (violate). In addition to colouring the packets with the drop-precedence bits, other actions such as marking the packet to another class (completely different DSCP), transmitting, or dropping the packets may be performed. This feature first appeared in IOS release 12.1(5) T.

Policing a class with the conform, exceed, and violate actions (tricolour marker) allows the user to simulate WRED-like behaviour for flows within the class, and reduce the chances of a multitude of TCP/UDP sessions being dropped in the presence of congestion.

5.1.2.1.3.9.3 Class-based Shaping

Class-based shaping allows controlling the traffic that leaves an interface in order to match its transmission to the speed of the remote, target interface and to ensure that the traffic conforms to policies contracted for it. Traffic adhering to a particular profile can be shaped to meet downstream requirements, thereby eliminating bottlenecks in topologies with data-rate mismatches.

Using the Class-Based Shaping feature, one can do the following:

- Configure Generic Traffic Shaping (GTS) on a traffic class
- Specify average-rate or peak-rate traffic shaping
- Configure class-based weighted fair queuing (CBWFQ) inside GTS
- Enable class-based shaping on any interface that supports GTS

Using the Class-Based Shaping feature *(IOS 12.1(3) T),* GTS can be configured on a class. In order to do so, the traffic classes must be defined based on matching criteria including protocols, ACLs, and input interfaces. Along with access control lists, NBAR/DSCP can also be used to specify traffic for shaping. Then, traffic shaping can be applied to each defined class.

Two types of traffic shaping can be specified: average rate shaping and peak rate shaping. Average rate shaping limits the transmission rate to the CIR. Peak rate shaping configures the router to send more traffic than the CIR. To determine the peak rate, the router uses the following formula: *Peak rate = CIR * (1+Be/Bc).*

5.1.2.1.3.9.4 Class Based QoS MIB

The IOS 12.1(5) T release introduced the Class Based QoS MIB. The MIB, a database of relevant QoS counters, serves as the basis for monitoring the QoS features using SNMP.

Using SNMP, the Class Based QoS allows the user to monitor traffic shaping, traffic policing, CBWFQ, WRED, and NBAR (Network-Based Application Recognition) protocol-discovery, and generic traffic counters. As an example, the user can obtain incoming packet and byte counts for a particular class, and conformed packet counters, exceeded packet counters and byte counters for another class configured to be policed. The Class-Based QoS MIB provides read access to class-based QoS configurations. This MIB also provides QoS statistics information based on the Modular QoS Command Line Interface (CLI), including information regarding class map and policy map parameters.

This Class-Based QoS MIB is actually two MIBs: CISCO-CLASS-BASED-QOS-MIB and CISCO-CLASS-BASED-QOS-CAPABILITY-MIB.

5.1.2.1.4 IPv6 Quality of Service

This feature, supported in IOS 12.2(13)T, enables the applicability of all the Differentiated Services (DiffServ) QoS features to IPv6 packets. Specific QoS features include packet classification, traffic shaping, traffic policing, packet marking, and Drop based on Weighted Random Early Detect (WRED) on all applicable interfaces.

5.1.2.1.5 Provisioning and Configuration Tools

5.1.2.1.5.1 The Modular QoS CLI (MQC)

The MQC is a provisioning mechanism in Cisco IOS software, which allows for separation of packet classification (class-maps), from policies (policy-maps) applied on the defined classes, from the application of those policies on interfaces and sub-interfaces (service-policy). The MQC forms the basis for provisioning DiffServ, and all the QoS mechanisms are parts and features of the class-maps (classification) or policy-maps (policing, shaping, queuing, congestion avoidance, packet marking, Layer2 CoS marking, etc.).

Packets entering a DiffServ Domain can be metered, marked, shaped or policed to implement traffic policies (as defined by the administrative authority). In Cisco IOS software, classifying, and marking is done using the MQC's class-maps. Metering is done using a token bucket algorithm, shaping is done using Generic Traffic Shaping (GTS) and policing is done using class-based Policing/Committed Access Rate (CAR). On the value added side, Cisco also provides for the Per-Class Accounting MIB, wherein statistics for each class (regardless of congestion) can be collected for management purposes.

5.1.2.1.5.2 Cisco Quality of Service Device Manager

QoS Device Manager (QDM) is a web-based Java application that enables users to configure and monitor advanced IP-based Quality of Service (QoS) functionality within Cisco routers using a graphical user interface (GUI). This includes configuration of traffic classes, packet marking, QoS enforcement mechanisms (queues and schedulers), and QoS monitoring at class, policy or interface level. QDM 2.0 is available as a separate product download and is free of charge.

5.1.2.1.5.3 Quality of Service Policy Manager (QPM)

CiscoWorks QoS Policy Manager (QPM) is a centralised tool that provides a platform for defining, applying, and monitoring QoS policy on a system-wide basis for Cisco devices, including routers and switches. QPM enables the user to baseline profile network traffic, create QoS policies at an abstract level, control the deployment of policies, and then monitor QoS to verify intended results. QPM provides a web-based intuitive user interface to define QoS policies, and translates those policies into the device's command line interface (CLI) commands.

QPM includes the following management applications: QoS Analysis, Policy Configuration, QoS Configuration for IP Telephony, Device Management, Deployment, and Additional Administration Applications. Refer to the following document for more information:

http://www.cisco.com/en/US/partner/products/sw/cscowork/ps2064/products_user_guide_chapter09186a00800e0a00.html

5.1.2.2 MPLS Traffic Engineering

Cisco's MPLS products are available for both the Edge and Core routers of a MPLS network. Cisco 3600 Series, 7500 Series, and GSRs 12000 are MPLS-Compliant Cisco products. Cisco offers three main MPLS functions and services:

- Basic MPLS, which provides conventional destination based least cost routing of IP packets

- Traffic Engineered MPLS tunnels, where selected IP traffic is routed either via a pre-configured route (explicit path) or via a dynamic path route that has been determined by constraint based routing.
- MPLS VPNs, using the “Peer model”, where Cisco supports VPNs over dynamic MPLS or over TE tunnels.

Cisco MPLS traffic engineering approach determines the routes for traffic flows across a network based on the resources the traffic flow requires and the resources available in the network. It employs "constraint-based routing", in which the path for a traffic flow is the shortest path that meets the resource requirements (constraints) of the traffic flow. Cisco MPLS-TE automatically establishes and maintains LSPs across the network by using RSVP. The path that an LSP uses is determined by the LSP resource requirements and network resources, such as bandwidth. Available resources are flooded by means of extensions to a link-state based IGP such as IS-IS and OSPF. Traffic engineering tunnels are calculated at the LSP head, based on a fit between required and available resources. The IGP automatically routes the traffic onto these LSPs. In Cisco MPLS-TE and based on the bandwidth requirements, there is a need to allocate bandwidth to tunnels in order to create traffic-engineered paths for the LSPs. If no bandwidth is allocated to the tunnels and if tunnel establishments are requested, Cisco MPLS-TE will create infinite number of tunnels using the shortest path. In Cisco MPLS-TE, the tunnels can be created dynamically. The bandwidth allocated to the tunnels is based on the requests.

In Cisco, tunnel explicit routes can also be manually configured. In Cisco MPLS-TE approach, in order to have differentiated services, the PHBs along the tunnel routes must be configured manually or through some external applications. The manual configurations of LSPs and PHBs are cumbersome even for very small networks.

The following subsections highlight the IOS features related to MPLS-TE support in Cisco.

5.1.2.2.1 Inter-AS MPLS-TE Support

Cisco is currently working towards MPLS-TE support between ASs for service providers. Cisco co-authored an Internet draft [Zhang, 2003] that presents a set of requirements and applicable deployment cases. The draft first discusses issues in current BGP-based inter-AS traffic engineering practices in supporting QoS-enabled paths across multi-ASs and then deals with the requirements for extending the current MPLS TE mechanism beyond BGP AS boundaries. A set of application scenarios is also presented as potential deployment cases of inter-AS MPLS TE. A list of detailed requirements along with the evaluation criteria for a technical solution is also provided.

5.1.2.2.2 Inter-AS MPLS VPN Support

The Inter-Autonomous Systems MPLS VPN Support feature (released in IOS 12.2(1)) on the Cisco 3620 and 3660 series platforms provides a seamless integration of autonomous systems and service providers. Separate autonomous systems from different service providers can communicate by exchanging IPv4 network layer reachability information (NLRI) in the form of VPN-IPv4 addresses. The autonomous systems' border edge routers use eBGP to exchange that information. Then, an IGP distributes the network layer information for VPN-IPv4 prefixes throughout each VPN and each autonomous system. Further information can be obtained from the following document:

http://www.cisco.com/univercd/cc/td/doc/product/software/ios121/121newft/121t/121t5/interas.htm

5.1.2.2.3 Inter-AS IPv4 BGP Label Distribution for MPLS VPN

This feature enables to set up a VPN service provider network so that the autonomous system boundary routers (ASBRs) exchange IPv4 routes with MPLS labels of the provider edge (PE) routers. Route reflectors (RRs) exchange VPNv4 routes, using multihop multiprotocol eBGP. This configuration saves the ASBRs from having to store all the VPNv4 routes. Using the route reflectors to store the VPNv4 routes and forward them to the PE routers results in improved scalability.

This feature was originally introduced in Cisco IOS Release 12.0(21)ST and then the feature was integrated in IOS Release 12.2(13)T. Refer to the following document for additional information:

http://www.cisco.com/univercd/cc/td/doc/product/software/ios122/122newft/122t/122t13/ftiasl13.htm

5.1.2.2.4 Automatic Bandwidth Adjustment for MPLS-TE Tunnels

Traffic engineering automatic bandwidth adjustment provides the means to automatically adjust the bandwidth allocation for traffic engineering tunnels based on their measured traffic load. This feature is released in IOS 12.2(4)T.

Traffic engineering auto-bandwidth samples the average output rate for each tunnel marked for automatic bandwidth adjustment. For each marked tunnel, it periodically (for example, once per day) adjusts the tunnel's allocated bandwidth to be the largest sample for the tunnel since the last adjustment.

The frequency with which tunnel bandwidth is adjusted and the allowable range of adjustments is configurable on a per-tunnel basis. In addition, the sampling interval and the interval over which to average tunnel traffic to obtain the average output rate is user-configurable on a per-tunnel basis. Additional information can be obtained from the following document:

http://www.cisco.com/univercd/cc/td/doc/product/software/ios122/122newft/122t/122t4/ftbwadjm.htm

5.1.2.2.5 IP Explicit Address Exclusion for MPLS-TE

This feature was released in IOS 12.2(13)T. The MPLS traffic engineering IP explicit address exclusion feature provides a means to exclude a link or node from the path for an MPLS traffic engineering LSP. If the exclude-address for an MPLS traffic engineering LSP identifies a flooded link, the constraint-based shortest path first (CSPF) routing algorithm does not consider that link when computing paths for the LSP. If the exclude-address specifies a flooded MPLS traffic engineering router ID, the CSPF routing algorithm does not allow paths for the LSP to traverse the node identified by the router ID. Further information can be obtained from the following document:

http://www.cisco.com/univercd/cc/td/doc/product/software/ios122/122newft/122t/122t4/ftaddexc.htm

5.1.2.2.6 MPLS QoS Enhancements

This feature was released in IOS 12.2(13)T. When a customer transmits IP packets from one site to another, the IP precedence field (the first three bits in the header of an IP packet) specifies the quality of service (QoS), such as latency or the percent of bandwidth allowed for a particular class of service. The service provider might want to set an MPLS packet's QoS to a different value.

QoS transparency allows the service provider to set the MPLS experimental field instead of overwriting the value in the customer's IP precedence field. The IP header remains available for the customer's use; the IP packet QoS is not changed as the packet travels through the MPLS network. Refer to the following document for further information:

http://www.cisco.com/univercd/cc/td/doc/product/software/ios121/121newft/121t/121t5/mct1214t.htm

5.1.2.2.7 MPLS Egress NetFlow Accounting

This ***mpls netflow egress*** command was integrated into IOS Release 12.1(5)T. The MPLS egress NetFlow accounting feature allows the user to capture the MPLS VPN IP flows that are travelling from one site of a VPN to another site of the same VPN through the service provider backbone.

To capture the flow of traffic going to site 2 of VPN-1 from any remote VPN-1 sites, the user enables MPLS egress NetFlow accounting on link PE2-CE2 of provider edge router PE2. The flows are stored in a global flow cache maintained by the router. The PE routers can export the captured flows to the collector devices in the provider network.

The MPLS Egress NetFlow Accounting feature in IOS 12.2(13)T allows to capture IP flow information for packets undergoing MPLS label disposition; that is, packets that arrive on a router as MPLS and are transmitted as IP. Refer to the following document for further information:

http://www.cisco.com/univercd/cc/td/doc/product/software/ios121/121newft/121t/121t5/egress.htm

5.1.2.3 Border Gateway Protocol

BGP performs routing between different Autonomous Systems. The primary function of a BGP system is to exchange network reachability information with other BGP systems. BGP-4 provides a set of mechanisms for supporting classless inter-domain routing. BGP-4 also introduces mechanisms that allow aggregation of routes, including aggregation of AS paths.

The following subsections highlight the IOS features related to inter-domain routing support in Cisco.

5.1.2.3.1 QoS Policy Propagation via BGP

The QoS policy propagation via BGP feature allows the user to classify packets based on access lists, BGP community lists, and BGP AS paths. The supported classification policies include IP precedence setting and the ability to tag the packet with a QoS class identifier internal to the router. After a packet has been classified, the user can use other QoS features such as CAR and WRED to specify and enforce business policies to fit to the business model.

BGP provides a scalable means of utilising attributes, such as commodity values, to propagate destination-based packet classification policy throughout a large network. Propagation takes place in BGP routing updates. Packet classification policy can be propagated by BGP without the writing and deploying of complex access lists at each of a large number of routers. BGP ensures that return traffic to customers is handled as premium traffic by the network.

The QoS policy propagation via BGP feature was introduced in Cisco IOS Release 11.1(17)CC.

5.1.2.3.2 BGP Peer Groups

BGP neighbours who share the same outbound policies can be grouped together in what is called a BGP peer group. Instead of configuring each neighbour with the same policy individually, Peer group allows to group the policies, which can be applied to individual peers, thus allowing efficient update calculation along with simplified configuration.

5.1.2.3.3 BGP Link Bandwidth

BGP Link Bandwidth feature is used to advertise the bandwidth of an autonomous system exit link as an extended community. The BGP Link Bandwidth feature is supported by the iBGP and eBGP multipath features. The link bandwidth extended community indicates the preference of an autonomous system exit link in terms of bandwidth. The link bandwidth extended community attribute may be propagated to all iBGP peers and used with the BGP multipath features to configure unequal cost load balancing. When a router receives a route from a directly connected external neighbour and advertises this route to iBGP neighbours, the router may advertise the bandwidth of that link. This feature was originally introduced in Cisco IOS Release 12.2(2)T. The IOS 12.2(11)T release is porting the feature into the Cisco AS5300, Cisco AS5400, and Cisco 5800 platforms. Refer to the following document for additional information:

http://www.cisco.com/univercd/cc/td/doc/product/software/ios122/122newft/122t/122t11/ft11b_lb.htm

5.1.2.3.4 BGP Multipath Load Sharing for Both eBGP and iBGP in an MPLS-VPN

The BGP Multipath Load Sharing for eBGP and iBGP feature allows the user to configure multipath load balancing with both eBGP) and iBGP paths in BGP networks that are configured to use MPLS VPNs. This feature provides improved load balancing deployment and service offering capabilities and is useful for multi-homed autonomous systems and Provider Edge routers that import both eBGP and iBGP paths from multihomed and stub networks. This feature was originally introduced in Cisco

IOS Release 12.2(4)T. The IOS 12.2(11)T release is porting the feature into the Cisco AS5300, Cisco AS5400, and Cisco AS5800 platforms. Refer to the following document for additional information:

http://www.cisco.com/univercd/cc/td/doc/product/software/ios122/122newft/122t/122t11/ft11bmpl.htm .

5.1.2.3.5 iBGP Multipath Load Sharing

When a BGP speaker router with no local policy configured receives multiple NLRIs from the internal BGP for the same destination, the router will choose one iBGP path as the best path. The best path is then installed in the IP routing table of the router. The iBGP Multipath Load Sharing feature enables the BGP speaker router to select multiple iBGP paths as the best paths to a destination. The best paths or multipaths are then installed in the IP routing table of the router.

This feature is supported in IOS 12.2(11)T. Refer to the following document for further information:

http://www.cisco.com/univercd/cc/td/doc/product/software/ios122/122newft/122t/122t11/ft11bmls.htm

5.1.2.3.6 BGP Prefix-Based Outbound Route Filtering

The BGP Prefix-Based Outbound Route Filtering feature uses Border Gateway Protocol (BGP) outbound route filter (ORF) send and receive capabilities to minimise the number of BGP updates that are sent between peer routers. The configuration of this feature can help reduce the amount of resources required for generating and processing routing updates by filtering out unwanted routing updates at the source. This feature was originally introduced in Cisco IOS Release 12.2(4)T. The IOS 12.2(11)T release is porting the feature into the Cisco AS5300, Cisco AS5400, and Cisco AS5800 platforms. Refer to the following document for additional information:

http://www.cisco.com/univercd/cc/td/doc/product/software/ios122/122newft/122t/122t11/ft11borf.htm

5.1.2.3.7 BGP Conditional Route Injection

Cisco IOS software provides several methods in which the user can originate a prefix into the BGP. The existing methods include using the network or aggregate-address commands and redistribution. These methods assume the existence of more specific routing information (matching the route to be originated) in either the routing table or the BGP table. The BGP Conditional Route Injection feature enables the user to originate a prefix into BGP without the corresponding match. The routes are injected into the BGP table only if certain conditions are met. The most common condition is the existence of a less-specific prefix.

This feature was originally introduced in Cisco IOS Release 12.2(4)T. The IOS 12.2(11)T release is porting the feature into the Cisco AS5300, Cisco AS5350, Cisco AS5400, Cisco AS5800, and Cisco AS5850 platforms. Refer to the following document for additional information:

http://www.cisco.com/univercd/cc/td/doc/product/software/ios122/122newft/122t/122t11/ft11bpri.htm

5.1.2.3.8 BGP Hide Local-Autonomous System

The BGP Hide Local-Autonomous System feature introduces the no-prepend keyword to the neighbour local-as command. The use of the no-prepend keyword allows a network operator to configure a BGP speaker to not prepend the local AS number to any routes that are received from external peers. This feature can be used to help transparently change the AS number of a BGP network and ensure that routes can be propagated throughout the AS, while the AS number transition is incomplete. Because the local autonomous is not prepended to these routes, external routes will not be rejected by internal peers during the transition from one autonomous system number to another.

This feature was originally introduced in Cisco IOS Release 12.2(8)T. The IOS 12.2(11)T release is porting the feature into the Cisco AS5300, Cisco AS5350, and Cisco AS5400 platforms. Refer to the following document for additional information:

http://www.cisco.com/univercd/cc/td/doc/product/software/ios122/122newft/122t/122t11/ft11bhla.htm

5.1.2.3.9 BGP Policy Accounting

BGP policy accounting measures and classifies IP traffic that is sent to or received from different peers. Policy accounting is enabled on an input interface and counters based on parameters such as community list, autonomous system number or autonomous system path, are assigned to identify the IP traffic. Using the BGP table-map command, prefixes added to the routing table are classified by BGP attribute, autonomous system number or autonomous system path. A Cisco IOS policy-based classifier maps the traffic into one of eight possible buckets, representing different traffic classes. Packet and byte counters are incremented per input interface.

Using BGP policy accounting, the service provider can account for traffic according to the route it traverses. This feature is supported in IOS 12.2(13)T. Refer to the following document for additional information:

http://www.cisco.com/univercd/cc/td/doc/product/software/ios122/122newft/122t/122t13/ft_bgppa.htm

5.1.2.3.10 BGP 4 MIB Support for per-Peer Received Routes

BGP 4 MIB Support for per-Peer Received Routes introduces a new table in the CISCO-BGP4-MIB that provides the capability to query using SNMP for routes that are learned from individual BGP peers. This feature is supported in IOS 12.2(13)T. Refer to the following document for additional information:

http://www.cisco.com/univercd/cc/td/doc/product/software/ios122/122newft/122t/122t13/ftbgpmib.htm

5.1.2.4 Signalling

5.1.2.4.1 RSVP

RSVP allows end-systems to request QoS guarantees from the network. RSVP works in conjunction with weighted fair queuing (WFQ) or Random Early Detection (RED). This conjunction of reservation setting with packet queuing uses two key concepts: end-to-end flows with RSVP and router-to-router conversations with WFQ:

- RSVP flow—This is a stream that operates "multi-destination simplex", because data travels across it in only one direction: from the origin to the targets. Flows travel from a set of senders to a set of receivers. The flows can be merged or left unmerged and the method of merging them varies according to the attributes of the application using the flow.
- WFQ Conversation—This is the traffic for a single transport layer session or network layer flow that crosses a given interface. This conversation is identified from the source and destination address, protocol type, port number or other attributes in the relevant communications layer.

RSVP allows for hosts to send packets to a subset of all hosts (multicasting). RSVP assumes that resource reservation applies primarily to multicast applications. Although the primary target for RSVP is multimedia traffic, a clear interest exists for the reservation of bandwidth for unicast traffic (such as NFS and VPN management).

The use RSVP in the network must be planned carefully. At a minimum, RSVP must reflect the assessment of bandwidth needs on router interfaces. By default, the amount reservable by a single flow can be the entire reservable bandwidth and 75 percent of the bandwidth available on an interface is reservable.

5.1.2.4.2 RSVP Aggregation

In the near future, Cisco IOS software will support full RSVP aggregation, allowing reservation through a DS-Domain, and mapping of the reservation to a DSCP and PHB. The reservations will be "fat pipes" that change very rarely. This aggregated reservation overcomes the problems of maintaining thousands of RSVP soft states in the routers and flooding of refresh messages.

5.1.2.4.3 RSVP-TE

RSVP is used to support MPLS traffic engineering. MPLS traffic engineering automatically establishes and maintains LSPs across the backbone by using RSVP. RSVP with traffic engineering extensions operates at each LSP hop and is used to signal, exchange labels and maintain LSPs based on the calculated path. RSVP is used to establish and maintain LSP tunnels based on the calculated path using PATH and RSVP Reservation (RESV) messages. The RSVP protocol specification has been extended so that the RESV messages also distribute label information.

5.1.2.4.4 LDP

Label Distribution Protocol (LDP) is an IETF standard protocol between MPLS-enabled routers to negotiate the labels used to forward packets. LDP is used to support MPLS forwarding along normally routed paths. LDP defines the standard method for hop-by-hop or dynamic label distribution in an MPLS network by assigning labels to routes that have been chosen by the underlying IGP routing protocol. LDP is incorporated in Cisco IOS. Cisco pioneered MPLS with pre-standard Tag Switching and Tag Distribution Protocol several years ago.

When a router is configured for both TE and LDP, the router receives different labels from both LDP and RSVP for a given prefix. The labels from LDP and RSVP do not need to be the same in all situations. The router will install an LDP label in the forwarding table if the prefix is learned through an LDP interface, and it will install the RSVP label in the forwarding table if the prefix is learned over a TE tunnel interface.

5.1.2.4.5 SIP

Session Initiation Protocol (SIP) is a protocol developed by the IETF MMUSIC Working Group as an alternative to H.323. SIP features are compliant with IETF RFC 2543, published in March 1999. SIP equips platforms to signal the set-up of voice and multimedia calls over IP networks.

Cisco has products supporting SIP such as: Voice Gateways, Call Agents, Softswitches and signalling gateways, IP phones, SIP proxy server, Firewall, NAT. Cisco provides SIP Global long distance network solutions.

5.1.2.5 Multicast

In order to support IP multicast services across ISP network boundaries, sophisticated protocols such as Protocol Independent Multicast Sparse Mode (PIM-SM), Multiprotocol Border Gateway Protocol (MBGP), and Multicast Source Discovery Protocol (MSDP) are available in Cisco IOS software that provide solutions for implementing native inter-domain multicast service. The following sections explain the Cisco's support for these protocols.

5.1.2.6 PIM

PIM can leverage whichever unicast routing protocols are used to populate the unicast routing table, including EIGRP, OSPF, BGP and static routes. PIM uses this unicast routing information to perform the multicast forwarding function; therefore it is IP protocol independent. Although PIM is called a multicast routing protocol, it actually uses the unicast routing table to perform the Reverse Path Forwarding (RPF) check function instead of building up a completely independent multicast routing table. Unlike other routing protocols, PIM does not send and receive multicast routing updates between routers.

5.1.2.7 PIM-SM

PIM-SM is the multicast forwarding protocol used in these intra-domain multicast scenarios. PIM-SM uses a pull model to deliver multicast traffic. Only network segments with active receivers that have explicitly requested the data will be receiving the traffic. PIM-SM uses a shared tree to distribute information about active sources. Depending on the configuration options, the traffic can remain on

the shared tree or switch over to an optimised source distribution tree. The latter is the default behaviour for PIM-SM on Cisco routers. The traffic starts to flow down the shared tree and then routers along the path determine if there is a better path to the source. If a more direct path exists, the last hop router (router closest to the receiver) sends a join message toward the source and then reroutes the traffic along this path.

Because PIM-SM uses shared trees (at least initially), it requires the use of a rendezvous point (RP). The RP must be administratively configured in the network. Sources register with the RP and then data is forwarded down the shared tree to the receivers. If the shared tree is not an optimal path between the source and the receiver, the routers dynamically create a source tree and stop traffic from flowing down the shared tree. This behaviour is the default in Cisco IOS software.

PIM-SM scales well to a network of any size, including those with WAN links. The explicit join mechanism will prevent unwanted traffic from flooding the WAN links.

To successfully deploy inter-domain multicast among the four ISPs, each ISP should use the following protocols:

- MBGP for inter-domain routing.
- MSDP for inter-domain source discovery.

MBGP and MSDP connect PIM-SM domains. MBGP is a policy-based inter-domain routing protocol for choosing best paths through an IP inter-network. MSDP enables RPs from different domains to exchange information about active sources.

5.1.2.8 MBGP

MBGP provides a method for providers to distinguish which route prefixes they will use for performing multicast RPF checks. The RPF check is the fundamental mechanism that routers use to determine the paths that multicast forwarding trees will follow and to successfully deliver multicast content from sources to receivers.

MBGP is described in RFC 2283, *Multiprotocol Extensions for BGP-4*. Because MBGP is an extension of BGP, it contains all the administrative machinery that providers and customers desire in their inter-domain routing environment, including all the inter-AS tools to filter and control routing (e.g., route maps). Therefore, any network utilising internal BGP (iBGP) or external BGP (eBGP) can use MBGP to apply the multiple policy control knobs, familiar in BGP, to specify routing (and thereby forwarding) policy for multicast.

Two path attributes, MP_REACH_NLRI and MP_UNREACH_NLRI, were introduced in BGP4. These new attributes create a simple way to carry two sets of routing information—one for unicast routing and one for multicast routing. The routes associated with multicast routing are used for RPF checking at the inter-domain borders.

5.1.2.9 MSDP

In the PIM-SM model, multicast sources and receivers must register with their local RP. Actually, the router closest to the sources or receivers registers with the RP, but the key point to note is that the RP knows about all the sources and receivers for any particular group. RPs in other domains have no way of knowing about sources located in other domains. MSDP is an elegant way to solve this problem.

MSDP is a mechanism that allows RPs to share information about active sources. RPs know about the receivers in their local domain. When RPs in remote domains hear about the active sources, they can pass on that information to their local receivers and multicast data can then be forwarded between the domains. A useful feature of MSDP is that it allows each domain to maintain an independent RP that does not rely on other domains, but it does enable RPs to forward traffic between domains. PIM-SM is used to forward the traffic between the multicast domains.

The RP in each domain establishes an MSDP peering session using a TCP connection with the RPs in other domains or with border routers leading to the other domains. When the RP learns about a new

multicast source within its own domain (through the normal PIM register mechanism), the RP encapsulates the first data packet in a Source-Active (SA) message and sends the SA to all MSDP peers. The SA is forwarded by each receiving peer using a modified RPF check, until the SA reaches every MSDP router in the interconnected networks—theoretically the entire multicast internet. If the receiving MSDP peer is an RP, and the RP has a (*, G) entry for the group in the SA (there is an interested receiver), the RP creates (S, G) state for the source and joins the shortest path tree for the source. The encapsulated data is decapsulated and forwarded down the shared tree of that RP. When the packet is received by the last hop router of the receiver, the last hop router also may join the shortest path tree to the source. The MSDP speaker periodically sends SAs that include all sources within the own domain of the RP.

5.1.2.10 Applicability to MESCAL

Cisco products include software, hardware (network elements) and documentation as supporting written materials. “Software” means the routers’ Internetworking Operating System (IOS) and other related programs provided in machine-readable object code format. Different IOS versions can support different functionality at the routers. Cisco provides a range of low, medium, and high performance routers and various physical interfaces. Cisco routers are the types of commercially available network elements that MESCAL solutions will have to manage in the real world environment. The main advantage of using commercial routers is that their functionality has already been tested rigorously. The main disadvantage is that it is not possible to modify their functionality, other than what is configurable.

Cisco routers implement a number of advanced networking functions that suit the requirements of the MESCAL system. Specifically, Cisco routers support and implement the following functions that are relevant to the MESCAL requirements:

- Diffserv architecture in implementing QoS-enabled differentiated services
- Building blocks for implementing traffic classification and conditioning mechanisms, scheduling and queuing mechanisms
- MPLS-based Traffic Engineering
- A range of Intra-domain routing protocols
- Inter-domain routing protocol i.e., BGP
- A number of standardised signalling protocols. These protocols may be required at the MESCAL service layer.
- Multicast protocols

It should be noted that Cisco is currently addressing and working towards Inter-domain Traffic Engineering using MPLS technology.

MESCAL collaborates closely with Cisco as its sponsoring partner through agreement to use Cisco products. The benefits of this close collaboration between MESCAL consortium and Cisco can be listed as below:

MESCAL can use software features of Cisco products that are in Beta test period. During this period, MESCAL can get licences to use and access to the software which may be suitable for the validation and testing of features of its proposed solution and project experimentation.

During a Beta test period MESCAL can run test suites and other test programs as defined by the project. This will have mutual benefit to MESCAL and Cisco. For MESCAL, the MESCAL functionality is verified with commercial networking equipment and for Cisco, the features of the product will be tested and any necessary adjustments will be reported.

Cisco can provide preferential technical assistance to MESCAL to test the advanced features for inter-domain QoS delivery.

5.1.3 Routing software

In this section, we will list the existing implementations of the routing protocols.

5.1.3.1 GNU Zebra Routing software

5.1.3.1.1 Overview

Zebra is software distributed under GNU Generic Public License, that manages TCP/IP based routing protocols. It supports several routing protocols. The particularity of Zebra implementation is that it has a process for each protocol. Zebra uses multithread technology under multithread supported UNIX kernels. However it can be run under non-multithread supported UNIX kernels. Each module can be upgraded independently of the others.

5.1.3.1.2 Features

5.1.3.1.2.1 Supported OS

The supported Operating Systems are:

- GNU/Linux 2.0.X and 2.2.X
- FreeBSD 2.2.8
- FreeBSD 3.1
- FreeBSD 4.X
- NetBSD 1.4
- OpenBSD 2.4

5.1.3.1.2.2 Supported Routing Protocols

The routing protocols supported by Zebra are:

- RIP (v1, v2)
- OSPF (v2, v3)
- BGP-4

5.1.3.1.2.3 Supported RFCs

The supported RFCs by Zebra are:

- RFC2453: RIP
- RFC2080: RIPng
- RFC2328: OSPF
- RFC2460, RFC2373, RFC2463 and RFC 2464: IPv6 Suite
- RFC2236: IGMP
- RFC1812: Router Requirements implementation
- RFC1771: BGP-4

5.1.3.2 ZebOS

The ZebOS Server Routing Suite can transform any Linux, Solaris, or FreeBSD server into an advanced routing platform through its array of IPv4 and IPv6 RIP, BGP and OSPF routing, PIM-SM multicast routing, and MPLS-VPN switching protocols. The ZebOS Server Routing Suite extends state-of-the-art routing and switching to firewall, web or database servers to add dynamic routing or high availability functionality.

5.1.3.2.1 ZebOS features

- Implementation of IPv4 and IPv6 routing standards
- IETF-compliant IPv4 and IPv6 RIP, OSPF and BGP protocols
- Command line interface management
- SNMP management with RIP, OSPF, BGP, PIM-SM and IP forwarding MIBs
- Detailed logging of systems events and errors
- Multiprocess architecture with support for dynamic soft reconfiguration
- Route table management, redistribution and conversion
- Equal cost multipath support delivers intelligent load balancing
- Point-to-point, point-to-multipoint, broadcast and multicast support
- Integration of LDP and MPLS-VPN switching and PIM-SM multicast for Linux
- Independent modules that may be installed, configured, and upgraded separately

5.1.3.2.1.1 IETF Standards and drafts Compliance

- RFC 1058 Routing Information Protocol (RIP)
- RFC 1724 Routing Information Protocol (RIP) MIB
- RFC 2082 RIP-2 MD5 Authentication
- RFC 2453 Routing Information Protocol version 2 (RIPv2)
- RFC 2328 Open Shortest Path First version 2 (OSPFv2)
- RFC 1850 Open Shortest Path First version 2 (OSPFv2) MIB
- RFC 1370 Applicability Statement for OSPF
- RFC 1587 OSPF Not So Stubby Area (NSSA) Option
- RFC 1765 OSPF Database Overflow
- RFC 2370 OSPF Opaque Link State Advertisement (LSA) Option
- RFC 1657 Definitions of Managed Objects for BGP-4 and SMIv2
- RFC 1771 Border Gateway Protocol version 4 (BGP-4)
- RFC 1772 Applications of BGP-4 in the Internet
- RFC 1997 BGP Communities Attribute
- RFC 2439 Route Flap Damping
- RFC 2547 BGP VPNs
- RFC 2842 Capabilities Advertisement with BGP-4
- RFC 2858 Multiprotocol Extensions to BGP-4
- RFC 2918 Route Refresh Capability for BGP-4
- RFC 3031 MPLS Architecture (for Linux only)
- RFC 3032 MPLS Label Stack Encoding (for Linux only)
- RFC 3036 LDP Specification (for Linux only)

5.1.3.2.1.2 Supported OS

- Red Hat 7.x -Linux 2.4.x (Linux 2.4.18 for MPLS-VPN and PIM-SM)
- Solaris 2.8
- FreeBSD 4.4 -4.6

5.1.3.3 GateD Suite of Routing Protocols

The first GateD code was developed at Cornell University in the 1980s. By 1987, an early version of GateD was used in NSFnet backbone. By 1995, Cornell University passed the software over to the University of Michigan and the non-profit Merit GateD Consortium for

further development, investment and evolution. There was an early version of the code that was open to use by anyone, and it was freely downloaded. But, by 2000, GateD was moved to a commercial version of the code by NextHop.
The architecture of the GateD product is as follow:

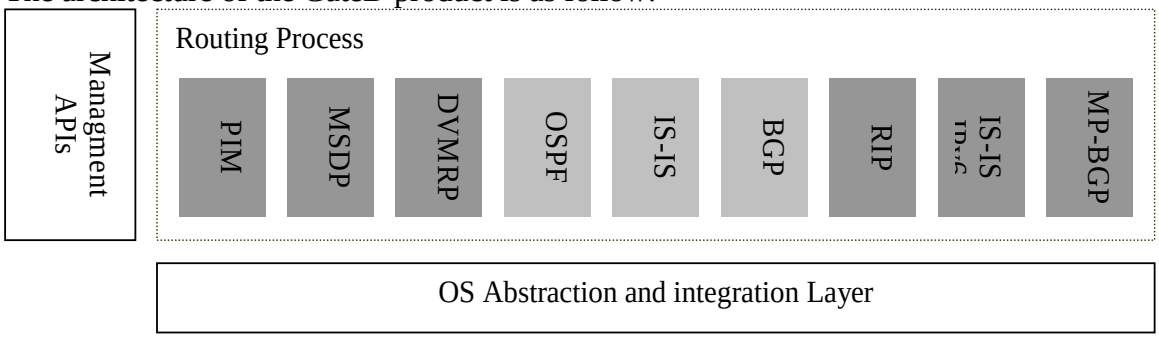

Figure 8 GateD architecture

5.1.3.3.1 Supported OS

- NetBSD
- Linux
- VxWorks
- BSD/
- FreeBSD
- Solaris
- Tru64

5.1.3.3.2 IETF Standards and drafts Compliance

- RIP MD5 Authentication (RFC 2082)
- RIPv1 (RFC 1058)
- RIPv2 (RFC 2453)
- RIP Tag Setting & Filtering
- Router Discovery (RFC 1256)
- SMUX (RFC 1227)
- ICMP Redirects (RFC 792)
- Supported MIBs
- RIPv2 MIB (RFC 1724)
- Supported Other RFCs
- Assigned Numbers (RFC 1700)
- CIDR (RFC 1519)
- MD5 Message Digest Algorithm (RFC 1321)
- Weighted Route Damping (RFC 2439)
- Stand Alone Utilities
- GDC (GateD command-line controller)
- RIPQuery (RIP command-line query tool)
- Additional Features
- IP Forwarding MIB (RFC 2096)
- RT MIB (RFC 2465)
- Checksum Generation & Verification
- Cooperative Multitasking
- Interface Management
- Interrupt Driven Timers

- Memory Management

5.1.3.3.3 Supported Routing Protocols

We list below the routing protocols supported by the GateD suite:

- IS-IS
- PIM
- BGP
- RIP
- OSPF
- DVMRP
- MSDP
- MP-BGP
- IS-IS extensions for IPv6

5.1.3.3.4 GateD IPv6 Family of Protocols

The IPv6 family of protocols is: RIP –next generation, MP-BGP extensions for IPv6 and IS-IS extensions for IPV6.

5.1.3.3.4.1 Supported Protocol RFCs and Drafts

- ICMPv6 (RFC 2463)
- IPv6 Aggregate-able Global Unicast Address Format (RFC 2374)
- Neighbour Discovery for IPv6 (RFC 2461)
- IPv6 MIB (RFC 2465)
- RIPng Specifications
- RIPng (RFC 2080)
- rip6query (RIPng command-line query tool)
- MP-BGP Specifications
- Multi-protocol/IPv6 (RFC 2858)
- IS-IS for IPv6 Specifications
- Routing IPv6 with IS-IS (draft-ietf-isis-ipv6-00)

5.1.3.3.5 BGP

We list below the IETF specifications compliant with the GateD product

- BGP4 (draft-ietf-idr-bgp4-17)
- Multi-Protocol Extensions for BGP-4 (RFC 2545)
- BGP4 MIB (draft-ietf-idr-bgp4-mib-08)
- AS Path Confederations (RFC 3065)
- Capabilities Advertisement (RFC 3392)
- Communities (RFC 1997)
- Dynamic Capability Negotiation (draft-ietf-idr-dynamic-cap-02)
- Extended Communities (draft-ietf-idr-ext-communities-05)
- Graceful Restart (draft-ietf-idr-restart-05)
- Route Flap Damping (RFC 2439)
- Route Reflection (RFC 2796)
- Route Refresh (RFC 2918)
- Carrying Label Information in BGP-4 (RFC 3107)

5.1.3.3.6 OSPF

Below, we list the supported RFCs and drafts:

- OSPFv2 (RFC 2328)
- OSPFv2 MIB draft-ietf-ospf-mib-update-05

- Hitless Restart (draft-ietf-ospf-hitless-restart-03)
- Alternate ABR (draft-ietf-ospf-abr-alt-04)
- draft-ietf-ospf-nssa-update-11
- Opaque LSAs (RFC 2370)
- Traffic Engineering Extensions (draft-katz-yeung-ospf-traffic-06)
- OSPFmon (OSPF Command-Line Monitor)
- Constrained SPF (CSPF)
- Fast SPF
- Multiple instance support
- Virtual Links
- On-the-fly Reconfiguration
- Inter and Intra-Area Routing
- Type 1 and 2 External Routing
- Broadcast, p2p, nbma, p2mp Interfaces
- MD5 Authentication
- ECMP

5.1.3.3.7 Multicast

5.1.3.3.7.1 Shared Functionality

Supported Protocol RFCs and Drafts

- IGMPv2 (RFC 2236)
- Supported MIBs
- IGMP MIB (RFC 2933)
- Multicast Routing MIB for IPv4 (RFC 2932)
- Supported the RFCs
- IP in IP Encapsulation (RFC 2003)
- MTrace (draft-ietf-idmr-traceroute-ipm-05)
- Additional Features
- Static group joins

5.1.3.3.7.2 DVMRP Specifications

Supported Protocol RFCs and Drafts

- DVMRPv3 (draft-ietf-idmr-dvmrp-v3-10)

Supported MIBs

- DVMRP MIB (draft-ietf-idmr-dvmrp-mib-11)

5.1.3.3.7.3 PIM Specifications

Supported Drafts

- PIM-SMv2 (draft-ietf-pim-sm-v2-new-05)
- BSR as described in draft-ietf-pim-sm-v2-new-02

Supported MIBs

- PIM MIB forIPv4 (RFC 2934)

5.1.3.3.7.4 MSDP Specifications

Supported Drafts

- MSDP (draft-ietf-msdp-spec-12)
- MSDP (draft-ietf-msdp-spec-6)

Supported MIBs

- MSDP MIB (draft-ietf-msdp-mib-06)

5.1.3.4 Applicability to MESCAL

To summarise the section on routing software:

For the Zebra suite:

- The Zebra suite is a GNU and then:
- No license is needed
- It is an open source
- The required routing protocols are included
- The configuration is done via CLI
- But this routing suite contains some bugs that are solved in the commercial version ZeboS.

For ZebOS:

- The ZeboS suite is commercial software and then license is needed. Nevertheless, this software could be used in the MESCAL project as far as:
- The required routing protocols are included
- Support of almost IETF standards
- The configuration is done via CLI
- The Zebra bugs are solved
- There is a support service
- Documentation is available

For GateD:

This software supports most of the IETF standards, and all required routing protocols are included in this suite.

5.1.4 Network Element Selection

5.1.4.1 Cisco routers

Applicability of Cisco routers to MESCAL is addressed in section 5.1.2.10. MESCAL may use Cisco routers as the commercial network elements in the development and testing solution option three of the project.

The main reasons for using Cisco are:

- ‘Cisco systems’ is the sponsor of MESCAL project and provides Cisco equipment and experience. Some MESCAL partners have experience in using Cisco equipment.

- Cisco equipment includes capabilities/features that suit MESCAL functional requirements and MESCAL project can exploit. Cisco systems is actively involved in Inter-AS MPLS activities.
- Cisco routers are widely deployed and they represent a more realistic environment for the MESCAL Functional Model to operate in.

Summarising, Cisco network elements are suitable for some of the MESCAL development and testing. As they are commercially available, there is advantage that their main functionality has already been tested. Close collaboration with Cisco, make it possible to modify some of the IOS functionality to suit MESCAL needs.

5.1.4.2 Linux-based routers

The QoS support (including traffic classifiers, the traffic conditioning (TC) components, and the queuing components) in the Linux kernel provides the framework for the implementation of differentiated services [RFC-2475]. Looking at the state-of-the art capture in routing software implementations for the Linux (section 2.4.4, I2.1), several elements contribute to the applicability of a Linux-based NE to the MESCAL-system. First of all the number of supported networking technologies is quite big. Linux not only provide low-level functionality, some functionality is also provided at the higher layers: support for SNMP, several routing solutions, RSVP-signalling support and MPLS. Due to the availability of source code, combined with a large community for supporting and debugging it, the Linux-based NE's are quite suitable to be a valuable tool within the development of the MESCAL-system. In addition, some of the MESCAL functional entities require the features that may not be available in commercial routers. These features can be developed and tested in Linux-based routers that provide a flexible environment. Linux give us great flexibility on how to implement these features according to the special requirements of the MESCAL functional model.

5.1.4.3 Routing Software

Cisco uses its propitiatory operating system (IOS). Cisco IOS Software provides a wide range of functionality - from routing, connectivity, security, and network management to technically advanced services that enable to deploy applications. Cisco commercial routers and IOS features have been reviewed in D1.2. This includes IOS features for Diffserv model implementation, MPLS traffic engineering capability, BGP support, and support of various signalling protocols.

Three Linux routing software implementations are fully addressed in section 5.1.2. The following section provides a comparison of the routing software implementations.

5.1.4.3.1 Comparison of Routing Software Implementations

Table 6 compares the features of different implementations of Linux based routing software suites.

Name	*Intra-domain Unicast*	*Intra-domain Multicast*	*Inter-domain*	*Other Protocols supported*	*Management Interface*	*Software Development*	*Cost*	*Reference*
Zebra	RIP OSPF	IGMP	BGP-4	IPv6 Suite	CLI	Minimal active development[2] Open Source, Modular	Free GNU Licence	www.zebra.org
Quagga	RIP OSPF	No	BGP-4+		CLI (vty), vtysh	Based on Zebra Open Source, Modular	Free GNU Licence	www.quagga.net
ZebOS	RIP OSPF	PIM-SM DVMRP	BGP-4	IPv6 Suite MPLS MPLS VPN Equal cost multi-path support RSVP-TE LDP	CLI	Source code licence available	Commercial Ipinfusion	www.ipinfusion.com
GateD	RIP IS-IS OSPF	IGMP PIM-SM PIM-SSM PIM-DM DVMRP	BGP-4 MBGP MSDP	IPv6 Suite MPLS RSVP-TE LDP	CLI, XML Routing API (see section 5.4.3.1)	Source code licence available	Commercial Nexthop Research Licence available	www.nexthop.com
Multithreaded Routing Toolkit	RIP	PIM-DM DVMRP	BGP4+		CLI	No longer under active development (since 2000) Open Source	Free, University of Michigan copyright	www.mrtd.net
XORP				Not yet mature				www.xorp.org

Table 6: Comparison of routing software suites.

[2] This is due to a code fork into Quagga and since the original maintainer is now employed by Ipinfusion.

Table 7 provides some information on the protocols that can be used in Linux environment.

Name	***Protocol type***	***Protocols supported***	***Software Development***	***Cost***	***Reference***
RSVP-TE	Signalling	MPLS	Open Source	Free, Developed on TEQUILA project	http://dsmpls.atlantis.rug.ac.be/
MPLS	Routing	Integrates with Zebra	Open Source	Free GNU Licence	http://sourceforge.net/projects/mpls-linux/
PIMD	Intra-domain Multicast (PIM-SM)		Open Source	Free GNU Licence	ftp://ftp.inr.ac.ru/ip-routing/pim/

Table 7: The routing software that can be used in a Linux enviroment.

Based on its capabilities and support, ***Zebos routing software Suite*** (version 3.5 or later) is selected to be the base routing stack for the Linux routers within the MESCAL project.

5.2 Selection of Simulation Tools

5.2.1 Introduction

Simulation is an important tool for the validation of the impending MESCAL architecture. While results from a large scale test bed with commercial routers or Linux routers would clearly provide more insight into the newly designed architecture, a full deployment is unfeasible from a cost as well as man power point of view. An implementation of MESCAL architecture would require multiple domains with potentially thousands of routers. Simulation is therefore the preferred option for proof of concept in areas such as scalability and overall impact on the network.

This chapter depicts currently available simulation environments that could potentially be of value for MESCAL simulation. This includes packet level tools such as ns-2, OPNET, J-SIM, etc. as well as higher level emulation tools such as MRT that could potentially emulate inter-AS traffic. Also included in the evaluation are some more general simulation environments such as MATLAB/SIMULINK, etc. The section concludes with the selection of some simulation environments.

A final section reviews some topology generators, which may be helpful in creating large scale - Internet like - topologies in order to provide for most realistic simulation. Creating an Internet like topology is of significance to MESCAL, since it is expected to be deployed in this environment at large scale.

5.2.2 Multithreaded Routing Toolkit (MRT)

MRT is the product of a partnership between the University of Michigan and Merit Network [MRT]. The MRT project is researching new routing software architectures, protocols and tools. Software developed to date includes multi-protocol IPv4/IPv6 routing daemons and routing analysis/simulation tools. MRT software is in use providing stress testing of commercial routers, collecting and analysing Internet routing traffic for researchers and serving as routing software connecting networks to the Internet and the backbone.

MRT can run on WindowsNT/2000, BSD and SunOS/Solaris platforms.

MRT software binaries and source code are freely available and may be freely modified and redistributed as long as the University of Michigan copyrights notice is included in the redistribution.

5.2.2.1 General Features

The latest version of MRT (version 2.2.2a released on August 2000) ships with BGP4+/BGP/RIPng/RIP2/DVMRP/PIM-DM and OSPF routing software. It also includes Cisco Systems configuration interface and data collection/processing and routing simulation utilities. MRT uses novel approaches to routing architecture design and incorporates features such as **parallel lightweight processes, multiple processor support and shared memory.** The object-oriented, modular design of the software encourages the rapid addition and prototyping of experimental routing protocol and inter-domain policy algorithms. All MRT programs can be invoked from the command line or from the Unix boot/startup script. Many MRT programs can be combined together using Unix shell-like pipe features. Once running, most MRT-based tools will begin to listen for user telnet connections on specified TCP ports. MRT-based programs may be configured by editing a configuration file or by invoking the configuration utility from the interactive user telnet interface and they all support logging. Most MRT-based tools share a common subset of user management and configuration commands. The command language used by MRT shares many similarities with the language used on Cisco Systems routers.

Although MRT has been designed with multi-threaded, multi-processor architectures in mind, the software will run in emulation mode on non-thread capable operating systems.

MRT can be used to:

- Serve as the backbone routing software for IPv6 or IPv4 network connections.
- Simultaneously handle tasks such as routing policy communication, routing policy calculation and maintenance of a RIB and distribute these tasks over multiple processors or multiple machines.
- Generate and analyse route flap statistics.
- Generate real-time graphical maps of Internet routing.
- Capture a BGP peering session and monitor it in real time.
- Record and replay sequences of events such as routing failures.

5.2.2.2 Toolkit Architecture

MRT provides a powerful collection of routing protocol **libraries** and **services** using an object **(module)**-oriented environment.

5.2.2.2.1 Libraries

The MRT libraries fall into two main categories:

- Lower-level services and support routines (timer, interface, socket routines etc)
- Protocol modules (BGP, routing table support etc)

5.2.2.2.1.1 Service Libraries

The lower-level service libraries provide routines common to most routing protocol implementations. For example, most protocols have periodic processes such as sending out KeepAlive packets or timing out routing table entries. The MRT timer library includes routines for multiplexing the Unix signal timer over multiple protocols. Similarly, most routing implementations have a need for receiving packets and buffering outbound packets. The MRT select library provides routines for handling network I/O. Other libraries, including the trace and UII libraries, provide routines to facilitate logging of trace information and the User Interactive Interface.

5.2.2.2.1.2 Routing Libraries

The MRT libraries also include high-level interfaces into routing protocols. These interfaces include access to BGP and other routing communications, as well as access to the kernel's routing table and interface management. For example, initiating a BGP peering session with a remote router can be as simple as linking in the MRT BGP library and calling *Add_BGP_Peer (hostname, AS)*.

5.2.2.2.2 Modules

MRT provides an object-oriented, multi-threaded programming environment. Under the MRT architecture, functional entities such as routing protocols and other application-level services are modelled as **modules,** or objects. Each module is associated with its own thread of control. On threads-capable multiprocessor machines, modules run on top of native kernel threads. On operating systems lacking threads support, MRT modules run on top of emulated threads.

Modules maintain control of their own private resources, which may include buffers, file descriptors and network sockets. Modules provide public methods or call routines for accessing the data and services supplied by the module. For example, a BGP module may provide public routines for initiating a peering session, trapping BGP packets and providing event notification.

Each MRT object maintains a queue of pending events (tasks). Other objects and services may schedule events by calling one of the object's public methods. For example, the BGP module might call *RIP_route_change_notify ()* to alert the RIP module that some external BGP routes have changed and RIP should begin announcing these new routes.

5.2.2.2.3 Services

The MRT architecture also includes specialized threads of control called **services.** Services generally perform specialized, narrowly focused tasks such as I/O multiplexing or timer notification. These threads only schedule events with other modules and do not maintain event queues nor conduct any event scheduling or processing of their own. Examples of services include a timer and I/O monitor service.

The current implementation of MRT includes services like **timer** and **select**.

Since most UNIX implementations only associate a single timer per process, the MRT timer thread is used to multiplex the single process timer over multiple threads. Objects like BGP and RIP schedule alarms by calling the timer scheduler method and providing both an internal and an absolute time and a callback method. The timer thread maintains a sorted queue of time-base events, which associate pending alarms with objects and their callback methods. The timer service also includes mechanisms for one-time alarms and adding jitter to timer intervals.

The select service performs synchronous I/O multiplexing on behalf of MRT modules. Since modules already block while waiting for the scheduling of new events, the objects require another mechanism for learning of pending I/O. The select thread monitors object socket descriptors for pending read, write and exception events. Objects register an interest in socket by calling a select service method. Once the select service detects an I/O event, the service invokes the callback method of an object and stops monitoring the socket. After an object completes processing of a socket, it notifies the select thread to once again begin monitoring.

5.2.2.3 Toolkit Internals

The routing and network performance tools available in MRT are briefly described in the following subsections.

5.2.2.3.1 MRTd

MRTd is a multi-threaded routing daemon with support for BGP4, RIPng, BGP4+, multiple routing information bases (RIBs) (route server), RIP1/2, OSPF, PIM-DM and DVMRP. MRTd reads Cisco Systems-like router configuration files and supports a Cisco Systems router-like interactive telnet interface. It supports most of the Cisco Systems routing policy commands, including access lists, as-path access lists and route maps. MRTd first reads its configuration file to configure routing protocols, route peerings and routing policy. After reading the configuration file it scans the kernel for existing routes, scans the kernel interface list, initiates routing protocol communications and begins listening for user telnet connections. MRTd can log both routing table dumps and binary traces of all BGP events in a format parseable by other MRT tools.

5.2.2.3.2 BGPsim

BGPsim simulates complex BGP4+ routing environments with possibly high levels of routing instability/change. BGPsim includes a perl program to generate ASCII descriptions of BGP traffic and supports an interactive interface. By default, BGPsim looks for a configuration file. It does not include mandatory attributes by default and attributes such as next hop and origin have to be explicitly defined in the configuration file. Also, BGPsim does not prepend its own AS by default.

5.2.2.3.3 SBGP

SBGP is a simple BGP4+ speaker and listener. SBGP does not apply policy to routers nor does it maintain a RIB of routes it has previously learned. Rather, SBGP provides a mechanism for monitoring routing information sent from a peer and for injecting routing information into a peering session. As arguments, SBGP takes the local AS number followed by the IP address and the AS number of the BGP4 peer. Multiple peer IP addresses and AS pairs may be specified. In all cases the

remote peer must be configured to accept a BGP4 peering session from the machine on which SBGP is running.

5.2.2.3.4 Route_BtoA

Route_BtoA converts binary MRT messages to ASCII. By default, the program writes human-readable ASCII descriptions of MRT message streams or files to standard out. Binary MRT messages may be generated by programs such as SBGP and MRTd for monitoring, research and statistics collection purposes. Route_BtoA also supports the generation of machine-readable output. This mode generates output that is easily parsed by awk or perl scripts for calculation of statistics.

5.2.2.3.5 Route_AtoB

Route_AtoB converts ASCII descriptions of MRT messages to binary. By default, the program writes binary MRT message streams or files to standard out.

5.2.2.3.6 Data Distiller

Data Distiller generates summary reports on the number of routing messages seen at one or more peering points. Data are gathered by passive (i.e. listening only) participation in the BGP peering session. Data distiller periodically updates its state from the MRTd raw data files and then creates reports. Any exceptional events such as server disconnection are logged. Data Distiller uses a telnet interface and supports standard MRT commands for remote monitoring and administration. Its configuration can be done either via a configuration file or over a telnet connection

5.2.2.4 Applicability to MESCAL

MRT could potentially be used to evaluate the MESCAL inter-domain traffic engineering approach by emulating the intra-domain traffic engineering system. MRT could be used to configure routers so as to emulate the behaviour of individual autonomous systems (ASs). However, the fact that there has been no additional development since August 2000, the lack of sufficient documentation (e.g. a tutorial that was supposed to be released was never actually delivered) and the relative low activity of the related mailing list (containing a total of 723 messages since 1996 and last updated on October 2002) should be seriously taken into consideration.

5.2.3 QoS Routing Simulator (QRS)

QRS is developed on the core of Maryland Routing Simulator (MaRS). The development work is being carried out at the Networking Laboratory of Helsinki University of Technology. The aim of QRS is to study the QoS related issues (especially QoS routing) in QoS based IP networks [QRS].

QRS is implemented in **C** for use on Unix platforms. It is a public simulator with free source codes for arbitrary modifications and redistributions.

5.2.3.1 Features

QRS is designed to provide a flexible platform for the evaluation and comparison of QoS routing algorithms in IP networks. QRS allows users to define an arbitrary network configuration, control its simulation, log the values of selected parameters and save, load and modify the network configuration. QRS only has a command line interface. The user configures the network by compiling the configuration file and looks up measurements by checking the recording files. The recording files depict the network configuration and the values (final and throughout the duration of the simulation) of all parameters and can be used to analyse network statistics.

The latest version of QRS (December 2002) introduces the **DiffServ (EF, AF, BE)** and **MPLS** concept**,** though these features are neither well documented nor supported, e.g. for MPLS, Label Distribution Protocol (LDP) is not implemented.

5.2.3.2 Simulator internals

QRS considers the target system as consisting of a physical network, a routing algorithm, a signalling, a resource management and a workload component. Also a performance monitor component is included. These components are described briefly in the following subsections. If these components are not enough for some specific purposes, the user can define additional ones.

5.2.3.2.1 Physical Network

The physical network consists of **link** and **node** components. A node component models the 'physical' aspects of a store-and-forward entity. It is characterized by parameters such as **buffer space, processing speed, failure and repair distribution.** A link component models a transmission channel between two nodes. In QRS a mechanism of **CBQ** is also implemented in link component. Therefore a link component is characterised by means of **bandwidth, propagation delay, failure and repair distributions and CBQ parameters.**

5.2.3.2.2 Routing

The Routing component maintains at each node, routing information that allows the node to route packets to their destinations. In QRS, **only** the intra-domain routing protocol **QOSPF** is provided, which calculates routes based on two metrics, i.e. bandwidth and cost (lowest cost and widest bandwidth algorithms). Both algorithms are based on Dijkstra algorithm. Link cost, as in MaRS, can be **hop count, utilization, delay and hop-normalized delay**. In any case, the shortest path routing algorithm runs to compute the shortest path for BE traffic, i.e. default routing table.

5.2.3.2.3 Workload

The Workload is defined in terms of source-sink pairs. Four types of source-sink pairs are currently supported: **Realtime Traffic, FTP, Telnet** and **Simple Traffic**. Realtime traffic has QoS requirements and submits QoS requests. Also each Realtime source-sink pair has a **Flow index** number, which can also correspond to the **label** used in **MPLS** and a **Class type**, which can correspond to the **DiffServ** traffic classes **(EF, AF, BE)**.

5.2.3.2.4 Signalling

Signalling is used to setup feasible paths (hop by hop) for Realtime traffic. In QRS, a simplified version of **RSVP** is provided and parameters such as **refreshing time** are user definable.

5.2.3.2.5 Resource Management

Resource management is used to reserve resources for paths. It decides if a path can be established at a node by simply following the rule:

If required bandwidth <= link available bandwidth

Accept

otherwise

Reject

5.2.3.2.6 Performance Monitor

Performance monitor provides two classes of performance measures: periodically updated (fixed length interval) and event-updated (triggered by an event).

5.2.3.3 Applicability to MESCAL

This simulator is unlikely to be used in MESCAL. Its greatest drawback is the lack of support for inter-domain routing protocols (**BGP)**. Also the **DiffServ** and **MPLS** support are not complete and many extensions to these modules should be implemented and integrated.

5.2.4 QUALNET

Qualnet is a commercial product of SNT (Scalable Network Technologies). Unfortunately not much information could be retrieved about it. It is based on Parsec (Parallel Simulation Environment for Complex Systems), which means that it can take advantage of **multiple processor architectures**. It comes with a Graphical User Interface (GUI), supports multicast and numerous other protocols (**BGP** included), **MPLS** and the **DiffServ** architecture, which could possibly make it suitable to meet the MESCAL project needs in terms of simulation [QUALNET].

5.2.5 Scalable Simulation Framework Network Model (SSFNET)

The project of S3 is developed by Rutgers University, Dartmouth College, Georgia Tech, and Boston University. It focuses on a large-scale network modelling and simulation, and contributes a simulator-independent network-modelling framework called Scalable Simulation Framework (SSF). SSF is a public domain standard for discrete event simulation of large complex systems. The design of SSF is object-oriented and it consists of five base classes:

- *Entity* is a container class for state variables.
- *inChannel* and *outChannel* are used for event exchange between endpoints.
- *Event* is the base class for information exchange.
- *Process* describes dynamic behaviour.

These five classes form a process and event oriented simulation. Currently, there are several independent implementations of SSF API:

- Raceway SSF, which is written in **Java** and is developed by Renesys Corp. It is commercial, but free for educational licenses.
- SSF (DaSSF), which is written in C++ and is developed by Dartmouth College. It is free software and distributed with open source.

However, **only** Raceway SSF is developed for communications network simulations. Therefore, MESCAL will consider simulation model built upon Raceway SSF only.

SSF Network Models (SSFNET) is built on top of Raceway SSF [SSFNET]. It has a collection of Java-based SSF modules for Internet protocols modelling and simulation of large, multi-protocol, multi-domain IP networks. It is designed at and above the IP packet-level granularity, but ignores the link layer implementation detail, e.g. links bandwidth and transmission delay. Link layer and physical layer modelling can be provided in separate packages.

SSFNET supports a number of major Internet Protocols and validation suites which include **IP, TCP, UDP, OSPFv2, BGP4** and also contains classes for network elements including host, routers, links, LANs, sockets. In addition, it implements global Internet topology construction and automated IP address generation/assignment. Network topologies are modelled using a self-configuring feature, which means that each class instance can query a configuration database to configure itself, which may be locally resident or available over the Web. The database called scalable configuration database package comes with the simulator and it is used to configure and describe a complete network model. The language within the package to create the network configuration files is in the **Domain Model Language (DML)** format. In SSFNET, classes used to construct any Internet model are based on two packages: SSF.OS and SSF.NET. The former is used to model all the host and operating system components especially protocols. Examples of some protocol packages are shown in Table 8:

Package	Description
SSF.OS.IP	Ipv4 package
SSF.OS.OSPF	Implementation of RFC2328 – OSPF version 2
SSF.OS.BGP4	Implementation of RFC 1771 – BGP4
SSF.OS.TCP(UDP)	TCP(UDP) package

Table 8 Examples of SSF.OS package

On the other hand, the latter package is used for modelling network connectivity, creating node and link configurations. Examples of these are creating host, router, queues, network interfaces, etc.

The architecture of SSFNET (Figure 9) is therefore composed by three components. SSF provides a standard for discrete-event simulation of large scale and complex systems. Raceway implements SSF by Java, which called Raceway SSF. Built on top of it, SSFNET has a collection of Java-based SSF packages (SSF.OS and SSF.NET) for network protocols and entities for simulation. DML is used to configure the network model and configuration. SSFNET can be installed in Windows or UNIX and the system requirement for running SSFNET is Sun's JDK version 1.2 or later can be used.

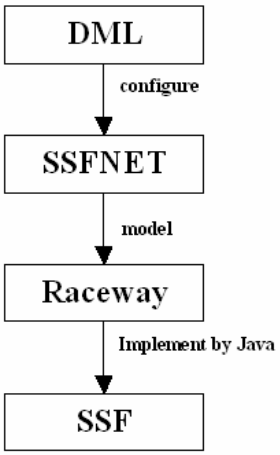

Figure 9 SSFNET architecture

5.2.5.1 Features

There are a number of features of SSFNET that are briefly described in the following subsections:

5.2.5.1.1 Fully integrated network environment

SSFNET has included many detailed network components and provides interoperability among these components.

5.2.5.1.2 Scalability

SSFNET is designed for large scale and complex network simulation. It allows arbitrarily simulation sizes vary by model and hardware.

5.2.5.1.3 Configurability

All components have multiple configurable attributes.

5.2.5.1.4 Repeatability and Reproducibility

It supports management of parallel random number streams and has strongest random number generators, statistics package from the CERN Colt package.

5.2.5.1.5 Monitoring

- SSFNET supports efficient multi-point network monitoring infrastructure for network analysis. It has a utility to monitor distributed traffic and host data at many hosts and routers by using probes. The monitor not only collects streaming and sampled data, but also everything in an SSFNET Internet model is accessible to be measured, including

- End-to-End application data
- Internal state of protocol sessions
- IP packet dumps on network interfaces and links
- Router flows and routes updates
- Queue lengths

This monitoring facility is provided by a package SSF.Util.Streams together with a probe package SSF.OS.ProbeSession.

5.2.5.1.6 Plotting

SSFNET has a package (SSF.Util.Plot) to provide a simple graph plotting which is targeted at visualizing time-series of collected data. Two data types that currently can be handled are the number of active IP flows and queue statistics. The package can generate plot files in a format that can also be recognized by standard ptplot or MATLAB.

5.2.5.1.7 Modularity

The design of SSFNET is modular (object-oriented), easily extensible and has readable code.

5.2.5.1.8 BGP support

5.2.5.1.8.1 Implementation

SSFNET fully supports BGP and it has an implementation based on RFC1771 “A Border Gateway Protocol 4”. The implementation supports the following details/modules:

- **IBGP/EBGP** This implementation simulates both Internal BGP and External BGP. Route reflection--an extension for IBGP--has also been implemented.
- **RIB** Each EGP router contains a Routing Information Base (RIB), is implemented with a type of binary tree called a radix tree that contains the routing information maintained by that router. As described in section 3.2 of RFC 1771, the RIB contains three types of information, each of which has been implemented in compliance with the specification.
  - **Adj-RIBs-In.** The unedited routing information sent by neighbouring routers.

    Loc-RIB. The actual routing information the router uses, developed from Adj-RIBs-In.

    Adj-RIBs-Out. The information the router chooses to send to neighbouring routers.
- **Messages** BGP routers exchange information using four types of messages (Open, Update, Notification, and KeepAlive) that contain the same fields as given in the specification, with a few exceptions.
- **Error Handling** Error handling is used for catching syntactical errors in messages.

- **Update Message Handling** The handling of Update messages (the type which carries route information) is fully implemented.
- **Path Attributes** All standard BGP path attributes (e.g. AS Path, MED) have been implemented, though some have not been extensively tested.
- **Timers** All five timers used by BGP are implemented, with time interval defaults as suggested in Appendix section 6.4 of RFC 1771:

  ```
  ConnectRetry
  Hold
  KeepAlive
  MinASOriginationInterval
  MinRouteAdvertisementInterval
  ```

  These timers are used to cause state transitions and each value is stored in units of seconds.
- **Jitter** RFC 1771 requires that three of the timers be jittered (MinASOriginationInterval, KeepAlive, and MinRouteAdvertisementInterval).
- **Finite State Machine** BGP Finite State Machine (FSM) highlights the major events in the process sent to peer in the state transitions. The key states in the FSM include:

  ```
  Idle
  Connect
  Active
  OpenSent
  OpenConfirm
  Established
  ```
- **Route Reflection** Route reflection, which is a BGP extension, is fully implemented.
- **Policy (Route Filtering)** This implementation also supports a policy configuration (route filtering) scheme that follows the suggestions in RFC 1772 http://www.cs.dartmouth.edu/~beej/bgp/java/BGP4/doc/ - ref3#ref3.
- **Validation** The implementation contains evaluation and testing modules to attempt to verify its correctness.

5.2.5.1.8.2 Configuration

This section briefly describes how BGP is defined in SSFNET using DML. Each individual BGP session is defined separately within a ProtocolSession attribute. This attribute has a sub-attributes name with a value of “BGP” and a value of SSF.OS.BGP4.BGPSessions. This specifies BGP is used in the session. A simple BGP protocol configuration in DML looks like

```
ProtocolSession [
  name bgp
  use SSF.OS.BGP4.BGPSession
]
```

There are currently two modes of configuration:

Full Manual Configuration This option requires all BGP attributes be fully defined in DML for each individual BGP instance.

Auto-configuration This option allows automatic configuration of attributes for each BGP instance.

To create a scenario for BGP, in addition to the above configuration, several definitions are also necessary. This includes the BGP timers that were mentioned in the previous section, and the definition of external BGP neighbours. To describe BGP neighbours, AS number, network interface addresses, hold timers, keep alive timers, etc are defined. Incoming and outgoing filtering can also be used to accept or reject routes to or from a specific BGP neighbour. A simple example of BGP configuration (connecting between current AS1 and neighbour AS2) looks like

```
ProtocolSession [
      name bgp
      use  SSF.OS.BGP4.BGPSession
      connretry_time 300
      min_as_orig_time 15
      neighbor [
        as 2  address 1(1)  use_return_address 1(1)
        hold_time 90  keep_alive_time 30  mrai 30
        infilter [ _extends .filters.permit_all ]
        outfilter [ _extends .filters.permit_all ]
      ]
]
```

5.2.5.2 Publications and support

A number of conference papers have published some simulation work using SSFNET. The number and the quality of conference can reflect the degree of SSFNET usability:

- Mohit Lad, Xiaoliang Zhao, Beichuan Zhang, Dan Massey, Lixia Zhang, "Analysis of BGP Update Burst during Slammer Attack", in *Proceedings of the 5th International Workshop on Distributed Computing*, Dec 2003.
- Hema Tahilramani Kaur, Shiv Kalyanaraman, Andreas Weiss, Shifalika Kanwar, Ayesha Gandhi, "BANANAS: An Evolutionary Framework for Explicit and Multipath Routing in the Internet", *ACM SIGCOMM Future Directions on Network Architectures (FDNA) Workshop, Karlsruhe*, Germany, August 2003
- Xiaoliang Zhao, Dan Massey, Dan Pei, Lixia Zhang, "A Study on the Routing Convergence of Latin American Networks", *ACM SIGCOMM Latin America Networking Workshop 2003*.
- D. Pei, X. Zhao, L. Wang, D. Massey, A. Mankin, S.F. Wu, and L. Zhang, "Improving BGP Convergence Through Consistency Assertions", *IEEE INFOCOM 2002*.
- M. Liljenstam, Y. Yuan, B.J. Premore and D. Nicol, "A Mixed Abstraction Level Simulation Model of Large-Scale Internet Worm Infestations", in *Proceedings of the Tenth IEEE/ACM International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS 2002)*, October 2002
- X. Zhao, D. Pei, L. Wang, D. Massey, A. Mankin, S.F. Wu, L. Zhang, "Detection of Invalid Routing Announcements in the Internet", *IEEE International Conference on Dependable Systems and Networks (DSN 2002)*, June 2002.
- Z. Mao, R. Govindan, G. Varghese, and R. Katz, "Route Flap Damping Exacerbates Internet Routing Convergence", *ACM SIGCOMM 2002*.
- T. G. Griffin and B.J. Premore, "An Experimental Analysis of BGP Convergence Time", in *Proceedings of the 9th International Conference on Network Protocols (ICNP 2001)*, November 2001.

There are also a number of websites that document and discuss SSFNET related issues:

1. SSFNET Q&A
   http://www.cs.dartmouth.edu/~beej/ssfnet/faq/fom-serve/cache/1.html
2. Detail of SSFNET Implementation of BGP (including source codes and examples)

   http://www.cs.dartmouth.edu/~beej/bgp/java/BGP4/doc/
3. SSFNET network topologies and models (gallery of baseline networks)

   http://www.ssfnet.org/Exchange/gallery/

4. SSFNET mailing list
   https://list.eecs.harvard.edu/mailman/listinfo/ssfnet/

5.2.5.3 Applicability to MESCAL

There are a number of issues, which need to be considered as regards applicability to MESCAL.

First of all, SSFNET has a full implementation of **BGP** and it has been widely used for simulating BGP behaviours. This is essential to MESCAL as it considers using this protocol as the basic means of inter-domain operation to support inter-domain traffic engineering and other forms of inter-domain policy negotiation. With the BGP implementation, it is possible to implement qBGP on SSFNET by extending BGP modules such as UPDATE message and path selection. On the other hand, however currently, **SSFNET does not officially support DiffServ, MPLS, RSVP** and **Multicast (these all are under MESCAL consideration).**

Regardless, some tentative work are undertaking to extend or support SSFNET, which may be considered by MESCAL simulator selection:

1. MPLS implementation for SSFNET. This implementation is the extension of a similar Java implementation of MPLS provided by NIST GLASS project based on SSF framework. Detail of the implementation can be found at http://www.net.uni-sb.de/~tkraemer/dipl/mpls.html.

5.2.6 J-Sim

J-Sim (formerly known as JavaSim) [J-SIM] is developed at Ohio State University implementing a component-based compositional network simulation environment and has been developed entirely in Java. The basic units of J-Sim are autonomous components so that it can be plugged into a software system, even during execution. On top of this autonomous component architecture, J-Sim has a generalized network model that defines the generic network components. These components can be used as base classes to accommodate or implement new protocols or algorithms across various layers. With the general network model, J-Sim is able to accommodate other network architectures, such as IETF **DiffServ** architecture, mobile ad hoc environment and the WDM-based optical network architecture. In addition, J-Sim supports script languages such as Perl, **Tcl** or Python to configure components at runtime and also it is a dual-language environment to integrate components using **Tcl/Java**. That facilitates simple and fast prototyping of simulation scenarios and diagnosis. J-Sim also has a **GUI** and can be installed in Windows or UNIX and the system requirements for running J-Sim is that a Java Virtual Machine (JVM) and Sun's JDK version 1.4 or later is used.

5.2.6.1 Features

There are a number of features of J-Sim, which are described in the following subsections:

5.2.6.1.1 Loosely coupled, autonomous component programming model

J-Sim is built on loosely coupled components architecture. Components have the capability to handle data in independent execution context, which enables them to be designed, implemented and tested separate from the rest of the system. Also, since they are loosely coupled or independent, components can be reused in other systems.

5.2.6.1.2 Dynamic thread execution framework for real-time process-driven simulation

In J-Sim, execution is implemented by Java threads. The thread scheduler is the Java Virtual Machine (JVM), which schedules the thread execution. With this mechanism, events are executed and simulation runs at real time (**parallel** simulation engine), thus preserves the real systems behaviour and enhances the fidelity of the simulation.

5.2.6.1.3 Implementation of a suite of Internet Integrated/ Differentiated/ Best Effort Services protocols

J-Sim supports other network architectures such as DiffServ architecture, mobile ad hoc environment and WDM-based optical network architecture.

5.2.6.1.4 A dual-language environment that allows auto-configuration and on-line monitoring

J-Sim provides a dual-language environment that creates components. A script language is used to integrate components at run time and to provide dynamic control. This facilitates fast prototyping of customized simulation scenarios, on-line monitoring and data collection.

5.2.6.1.5 BGP support

5.2.6.1.5.1 Implementation

A BGP model is recently available in J-Sim. The model follows the BGP implementation by B. J. Presmore for SSFNET which has been extensively validated and tested, plus a number of extensions:

- **Routing policies**: This extension concerns the complete support of routing policies that was not possible in SSFNET. For instance, build a filter that could match or changed required attributes such as *Community*. J-Sim BGP has provided a class to simplify this routing policies task by using a small script:
  - *Community "1"| deny*
  - which defines a policy that denies each route that contains the Community "1".
- **Redistribution Communities**: The BGP model supports the *Extended Communities* attribute, which has been defined in [EX-COMM]. The use of this attribute serves to avoid the several problems of using *Community* attribute to build scalable traffic engineering configurations.
- **Tie-breaking ID**: This extension is based on the assumption that many BGP decisions do not rely on AS-Path length but rather on tie-breaking rules. It introduces a *random* tie-breaking ID for each router to attach the ID to every advertised route. In contrast, in SSFNET, the tie-breaking ID is based on the ID of the route that advertises the route. However, this method does not correspond to a realistic IP address allocation and the distribution of router Ids thus becomes significant on the BGP route decision.

5.2.6.1.5.2 Configuration

This section briefly describes how BGP is defined in J-Sim using TCL. The basic BGP configuration can be done in six parts.

- **Topology setup**: The first part creates the topology including nodes, routers and links using J-Sim basic network components, for example we create 12 nodes (n0, n1, n2, n3, n256, n512, etc) and a number of links:

```
java::field infonet.javasim.bgp4.BGPSession log_enabled true
puts "build topology"
set LINK [java::new drcl.inet.Link]
set ADJ_MATRIX [java::new {int[][]} 12 { { 10 8 6 1 2 3 7 } { 5 0 11 6 2
9 3 } { 0 11 1 9 4 3 7 } { 10 5 8 0 1 2 4 } { 2 3 } { 1 3 } { 0 1 } { 0
2 } { 0 3 } { 1 2 } { 0 3 } { 1 2 }}]
set IDS [java::new {long[]} 12 { 0 1 2 3 256 512 768 1024 1280 1536 1792
2048 }]
java::call drcl.inet.InetUtil createTopology [! .] "n" "n" $ADJ_MATRIX
$IDS $LINK
```

- **TCP Attachment**: Each node created in the first part is then built with the second part: associate BGPSession class with the TCP layer (TCP Full):

```
puts "Setup nodes"
set NODE_BUILDER [java::new drcl.inet.NodeBuilder]
$NODE_BUILDER build [! n*] {
tcp drcl.inet.socket.TCP_full
   bgp -/tcp infonet.javasim.bgp4.BGPSession
}
```

- **Static routes setup**: The third part installs static routes between each pair of nodes which will establish a BGP session:

```
puts "Setup static routes"
java::call infonet.javasim.util.NetUtil setupBRoute [! n0] [! n1792]
java::call infonet.javasim.util.NetUtil setupBRoute [! n0] [! n1280]
java::call infonet.javasim.util.NetUtil setupBRoute [! n0] [! n768]
java::call infonet.javasim.util.NetUtil setupBRoute [! n0] [! n1]
java::call infonet.javasim.util.NetUtil setupBRoute [! n0] [! n2]
java::call infonet.javasim.util.NetUtil setupBRoute [! n0] [! n3]
java::call infonet.javasim.util.NetUtil setupBRoute [! n0] [! n1024]
```

  - Similar configuration is done for the rest of routers.

- **Configuration**: The fourth part configures each router with an AS number and a router ID, and configures each peering. J-Sim provides a number of method to configure each BGP router and each peering

  - `config (AS number, Tie-breaking ID)`: This method sets up a BGP router. It takes two arguments, the AS number and a Tie-breaking ID of the router.

  - `addPeer (IP address, AS number)`: This method adds a new peer to this BGP router. It takes two arguments, the IP address of the peer and its AS number. If the AS number is the same as this router, an iBGP session will be established, otherwise an eBGP session.

  - `setPeerInFilter (IP address, Filter)`: This method establishes an input filter for routes received from the peer that has a given IP address.

  - `setPeerOutFilter (IP address, Filter)`: This method establishes an output filter for routes to be advertised to the peer that has a given IP address.

  - The following example shows a basic configuration of this step, assuming that all routers have no filtering on any incoming and outgoing routes.

```
puts "Setup BGP"
! n0/bgp config 1 0         //n0 has AS number 1 with tie-breaking ID 0
! n0/bgp addPeer 1792 8     //n0 has a peer AS 8 with IP 1792
                            (represented by a ID)
! n0/bgp addPeer 1280 6
! n0/bgp addPeer 768 4
! n0/bgp addPeer 1 1        //setup an iBGP with n1
! n0/bgp addPeer 2 1        //setup an iBGP with n2
! n0/bgp addPeer 3 1        //setup an iBGP with n3
! n0/bgp addPeer 1024 5
```

- Similar configuration is done on the other routers.

- **Debugging and Trace**: The fifth part configures debugging, trace and event filtering files for each router.

```
puts "Setting up log files"
mkdir drcl.comp.io.FileComponent BGP_DEBUG_FILE
! BGP_DEBUG_FILE open "./bgp.debug"
setflag EventFiltering true [! BGP_DEBUG_FILE]
mkdir drcl.comp.io.FileComponent BGP_TRACE_FILE
! BGP_TRACE_FILE open "./bgp.trace"
setflag EventFiltering true [! BGP_TRACE_FILE]
mkdir drcl.comp.io.FileComponent BGP_FSM_FILE
! BGP_FSM_FILE open "./bgp.fsm"
setflag EventFiltering true [! BGP_FSM_FILE]
connect -c n0/bgp/msg@ -to BGP_TRACE_FILE/in@
connect -c n0/bgp/dbg@ -to BGP_DEBUG_FILE/in@
connect -c n0/bgp/fsm@ -to BGP_FSM_FILE/in@
```

- **Running simulation**: The sixth part starts the simulation, initialises BGP routers and installs various events such as the dump of Loc-Rib at a specific time interval.

```
puts "Run simulation"
set sim [attach_simulator .]
rt . stop
puts "Init BGP"
! n0/bgp init  //This method initialises the BGP router 0 and starts
               //the establishment of BGP sessions with the configured
               //peers.
! n1/bgp init
! n2/bgp init
! n3/bgp init
! n256/bgp init
! n512/bgp init
! n768/bgp init
! n1024/bgp init
! n1280/bgp init
! n1536/bgp init
! n1792/bgp init
! n2048/bgp init
puts "Resume simulation"
rt . resumeTo 500
script { exit } -at 499 -on $sim
// setup Loc-RIB dump at the 250 seconds for each router
script { ! n0/bgp dumpLocRIB } -at 250 -on $sim
script { ! n1/bgp dumpLocRIB } -at 250 -on $sim
script { ! n2/bgp dumpLocRIB } -at 250 -on $sim
script { ! n3/bgp dumpLocRIB } -at 250 -on $sim
script { ! n256/bgp dumpLocRIB } -at 250 -on $sim
script { ! n512/bgp dumpLocRIB } -at 250 -on $sim
script { ! n768/bgp dumpLocRIB } -at 250 -on $sim
script { ! n1024/bgp dumpLocRIB } -at 250 -on $sim
script { ! n1280/bgp dumpLocRIB } -at 250 -on $sim
script { ! n1536/bgp dumpLocRIB } -at 250 -on $sim
script { ! n1792/bgp dumpLocRIB } -at 250 -on $sim
script { ! n2048/bgp dumpLocRIB } -at 250 -on $sim
```

5.2.6.2 Applicability to MESCAL

There are a number of issues to bear in mind as regards J-Sim's applicability to MESCAL. First of all, it provides components both in the **DiffServ** architecture (such as marker at edge routes and buffer

management at core routers) and integrated service architecture (such as RSVP signalling). Secondly, it supports QoS-driven protocols (such as QoS extension to OSPFv2 and CBT). Thirdly, J-Sim has an implemented MPLS package made available recently. Finally, a BGP module is available for J-Sim, which was ported from part of the SSFNET BGP module and then enhanced with additional functionalities. However, this module has not yet been adequately validated. To find out more detail about J-Sim, readers are referred to the following websites:

1. J-Sim webpage www.j-sim.org
2. Add-on packages contributed to J-Sim http://www.j-sim.org/contribute.html
3. J-Sim mailing list http://mail.cs.uiuc.edu/mailman/listinfo/j-sim-announce

5.2.7 Network Simulator (NS)

The NS network simulator, from U.C. Berkeley/LBNL, is a discrete event simulator targeted at networking research, which provides substantial support for simulation of TCP, routing, and multicast protocols. The simulator is written in C++ and uses OTcl as a command and configuration interface. NS has the advantage that it provides numerous frameworks as the basis for extending its core capabilities.

In NS arbitrary network topologies, composed of routers, links and shared media can be defined. A rich set of protocols such as TCP and UDP is available and various types of applications can be simulated. Among them are FTP, Telnet, and HTTP, which use TCP as the underlying transport protocol, and applications requiring a constant bit rate (CBR) traffic pattern, which use the UDP transport protocol. Multiple queuing and scheduling policies can be configured such as drop-tail, random early detect, priority and fair queuing. The simulator is event-driven and runs in non-real-time fashion. Packet losses are simulated by buffer overflows in routers, which is also the dominant way packets get lost in the Internet. There is also support for error models other than losses through buffer overflow, but these are not used within the scope of our simulations. The routing model can be static or dynamic. The link state (i.e. Dijkstra's SPF) and distance vector (i.e. Bellman-Ford) routing algorithms are supported [NS].

5.2.7.1 Simulator internals

5.2.7.1.1 The Network Topology

The network topology is specified by listing the network nodes and edges in a topology file. Various configuration parameters for the specification of the size of the network and the connectivity can be set.

5.2.7.1.2 The Network Model

The network model represents the interconnection of network elements, hereafter called nodes. It consists of nodes and links as shown in Figure 10.

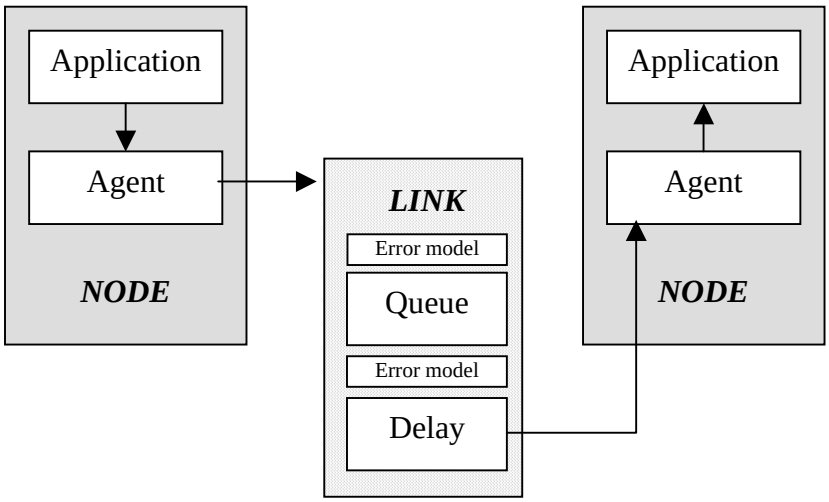

Figure 10 The NS network model.

Single or multiple traffic generators, including statistical generators and other typical generators such as FTP and Telnet, can be attached to any node. In addition, network and transport protocol behaviour are simulated by attaching the appropriate agents to the relevant nodes. Examples of routing agents include the ones supporting static routing (Dijkstra's SPF algorithm), dynamic routing (Distance Vector algorithm), or multicast routing. Examples of transport agents include the TCP agent and the UDP agent.

Links are modelled either as simplex- or duplex-links with a predefined capacity, delay, and queuing discipline. In addition, links can be torn down or restored at any point in time during the simulation, simulating link failures.

5.2.7.2 Features

5.2.7.2.1 Trace and Monitoring Support

There are a number of ways for collecting output or trace data on a simulation. Generally, trace data is either displayed directly during the execution of the simulation or, more commonly, stored in a file to be post-processed and analysed. NS supports two types of monitoring capabilities. The first, called *traces*, record each individual packet as it arrives, departs, or is dropped at a link or queue. Trace objects are configured into a simulation as nodes in the network topology, usually with a TCL channel object hooked to them, representing the destination of collected data. The other type of objects, called *monitors*, record counts of various interesting quantities such as packet and byte arrivals, departures, etc. Monitors can monitor counts associated with all packets or on a per-flow basis.

5.2.7.2.2 The Network Animator

Trace files generated during simulations contain topology information as well as packet traces and can then be used for post-processing and analysis of the simulation results. The Network Animator (NAM) is a Tcl/TK based animation tool for viewing NS trace files as well as real world packet traces. It supports topology layout, packet level animation, and various data inspection tools.

An example of NAM's graphical user interface is shown in Figure 11.

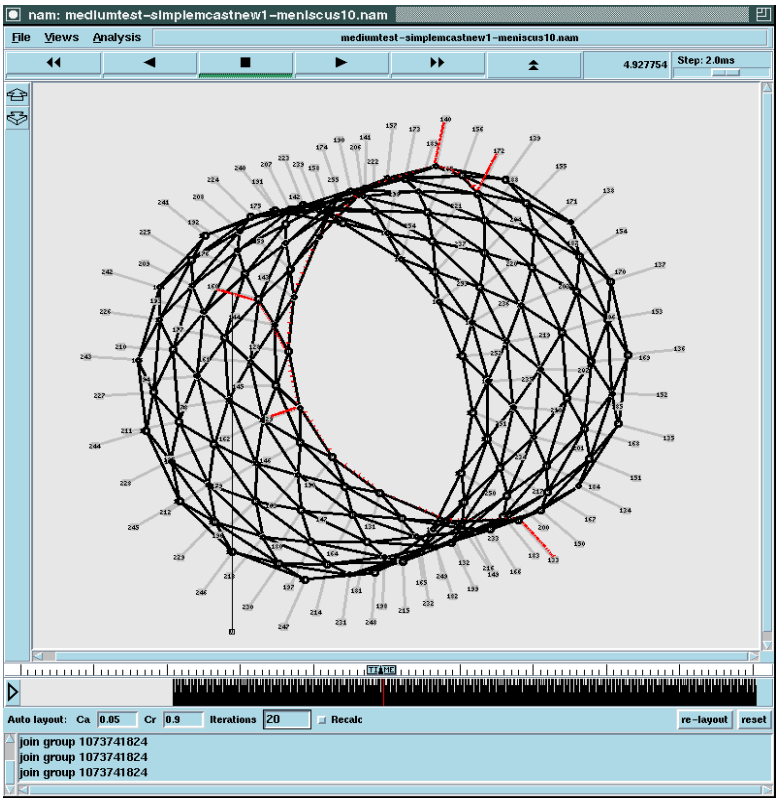

Figure 11 Example network animation with NAM.

5.2.7.2.3 Emulation

Another facility of NS is ***emulation,*** which is the ability to introduce the simulator into a live network. Special objects within the simulator are capable of introducing life traffic into the simulator and injecting traffic from the simulator into the live network.

5.2.7.3 Applicability to MESCAL

NS provides substantial support for simulation of TCP, routing, and multicast protocols. New protocols can be added to NS by specifying new agents in C++ and exposing the relevant parameters to the TCL interpreter. In particular, OTCL is considerably advantageous for quick prototyping purposes. **DiffServ** and **MPLS** are also supported. Recently, a **BGP** module, based on Zebra version 0.92a BGP daemon was implemented in C++ and added to NS. However, this module doesn't support all features found in Zebra, such as IPv6 BGP and MPLS VPN BGP extensions. Additionally, the NS packet forwarding mechanism needs to be modified in order to use the routing tables constructed by this BGP module. Furthermore, this BGP module has not yet been exhaustively validated. All these facts need to be taken into account if NS is to be used for simulations in MESCAL.

Finally, NS is widely used in the networking research community and has found large acceptance as a tool to experiment new ideas, protocols and distributed algorithms.

5.2.8 OPNET

5.2.8.1 Introduction

There are several products based on the original OPNET by MIL3 (now OPNET Technologies Inc, [OPNET]). These are IT Guru, SP Guru, Modeler, WDM Guru and NetBiz. Since the products address different type of organizations, they differ in flexibility compared to user-friendliness and some special purpose tools. The common features of the simulator packages will be described before the discussion of each of the relevant products. All OPNET simulator packages are built on a discrete

event simulation tool based on C. They feature an object-oriented modelling approach and graphical editors to mirror the structure of actual networks and network components. A standard library of models containing network protocols, technologies and applications is included with all products. Extension protocols such as MPLS, IPv6 and Multicast are available at extra cost.

5.2.8.2 Protocol, Application and Network Device Models

These include Multi-Tier Applications, Voice, HTTP, TCP, IP, OSPF, BGP, EIGRP, RIP, RSVP, Frame Relay, FDDI, Ethernet, ATM, 802.11 Wireless LANs, etc. Also available as specialized models are MPLS, PNNI, DOCSIS, UMTS, IP Multicast, and Circuit Switch, provided as finite state machines (FSM).

The Standard Model Library includes many vendor specific and generic device models including routers, switches, workstations and packet generators. Aggregate traffic can be formed into LANs or "Cloud" nodes. It is possible to create new device models using the "Device Creator".

5.2.8.3 OPNET Modules

OPNET can be extended with a number of Modules, such as the High-Level Architecture (HLA) Module.

HLA supports building and running a federation of many simulators, each modelling some aspect of a composite system. The OPNET HLA Module enables OPNET simulations to model some portion (or the whole) of the communications aspects of the HLA federation models. The OPNET-HLA interface provides the various simulators (federates), the necessary mechanisms to share a common object representation (for persistent elements), to exchange messages (interactions) and to maintain appropriate time synchronization.

Further Modules include:

- ACE Decode Module
- Application Characterization Environment (ACE)
- Expert Service Prediction
- Flow Analysis
- Multi-Vendor Import
- NetDoctor
- Wireless
- Terrain Modelling

5.2.8.4 BGP Features

Model implementation based on RFC 1771-1774, RFC 1997-1998, RFC 2796, RFC 3065 implements network reachability information exchange between BGP speakers in different ASs to model inter-AS routing. AS number assignment can be manual or automatic

- Explicit modelling of both IBGP and EBGP
- TCP interface to communicate over IP
- Each BGP router supports neighbour-specific configuration
- Ability to configure route maps to alter or filter routes based on the local administrative policies
- Configurable parameters for setting ConnectRetry, HoldTimer, and KeepAlive timers
- Simulation efficiency mode to suppress protocol activity after routing tables have stabilized
- Routing table export/import to/ from CSV files

- BGP Confederations support
- BGP Route reflector support. (Configuration of multiple route reflectors in a cluster and also the disabling of client to client reflection are supported)
- Use of Communities in route maps is supported.

5.2.8.5 OPNET Modeler

For network research and development. Used primarily to design and study network technologies, ranging from communications protocols to network equipment and systems. Modeler is the only package to supply the model library and library extensions with open source code.

Features of OPNET Modeler include:

- Integrated Debugger to validate simulation behaviour or track problems.
- Tools to display simulation results. Plotting and analysing time series, histograms, probability functions, parametric curves and confidence intervals. Support to export to spreadsheets.
- Hybrid simulations improve performance by combining discrete-event simulation with analytical modelling.
- Runtime environment to deliver proprietary protocol and device models to end-users, running simulations and working at the network level only.
- Data can be imported from text files, XML and popular tools from HP, Concord, Network Associates' Sniffer, NetScout, Infovista and others.
- Animation of model behaviour, either during or after simulation.
- APIs for program-driven construction or inspection of all models and result files. Existing code libraries can be integrated into simulations. Source code provided for all standard models.
- Hierarchical network models, complex network topologies can be managed with unlimited subnetwork nesting.
- Windows NT, Windows 2000 and UNIX supported.

The OPNET Modeler network view is arranged in hierarchical layers that directly depict the structure of networks, equipment and protocols. Each layer is edited and controlled with a dedicated editor.

5.2.8.5.1 The Network Layer

The Network Editor graphically represents the topology of a communications network. Networks consist of node and link objects, configurable via dialog boxes. Objects of node and link models can be created or selected from the OPNET library. The Network Editor provides geographical context, with physical characteristics reflected appropriately in the network simulation.

5.2.8.5.2 The Node Layer

The Node Editor captures the architecture of a network device or system by depicting the flow of data between functional elements, called “modules”. Each module can generate, send and receive packets from other modules to perform its function within the node. Modules typically represent applications, protocol layers, algorithms and physical resources, such as buffers, ports and buses. Modules are assigned process models (developed in the Process Editor) to achieve any required behaviour.

5.2.8.5.3 The Process Layer

The Process Editor uses a finite state machine approach to support specification, at any level of detail, of protocols, resources, applications, algorithms and queuing policies. States and transitions graphically define the progression of a process in response to events. Each state of a process model contains C/C++ code, supported by a library of functions designed for protocol programming. Each

FSM can define private state variables and can make calls to code in user-provided libraries. FSMs are dynamic and can be spawned (by other FSMs) during simulation in response to specific events. Dynamic FSMs simplify specification of protocols that manage a scalable number of resources or session, such as TCP or ATM. The user can develop entirely new process models or use the models in OPNET Technologies' Model Library as a starting point.

5.2.8.6 OPNET SP Guru

SP Guru is a network management tool for service providers. Used primarily to troubleshoot, validate, plan and design service provider networks. OPNET Service Provider Guru focuses on Layer 2/3 networks, including routers, switches, protocols, servers, and traffic demands. SP Guru allows for troubleshooting, operational validation, planning and engineering of service provider networks.

5.2.8.6.1 Generating Models

SP Guru provides mechanisms to automatically build a model of the network. It includes facilities to capture topology and configuration information, as well as traffic and link utilization data. SP Guru can recreate the behaviour of an existing service provider network or predict routing, link utilization and service levels for a wide range of scenarios.

- Virtual networks can be created from device configuration files, including Cisco and Juniper routers and Lucent switches. Configurations are set automatically on each device for connections, interfaces, link speed, routing protocols and other detailed information that determine network behaviour. SP Guru determines network topology based on configuration information.
- Alternatively, discovery capability from network management platforms can be utilised by importing topology data from platforms such as HP OpenView.
- The network can be loaded with traffic matrix information and link utilization data directly from a variety of traffic data collection and performance management tools, including:
  - Network Associates Sniffer Analyzer, Distributed Sniffer, and Sniffer Pro
  - NetScout Systems' NetScout Manager and nGenius
  - Concord Network Health/ eHealth-Network
  - Cisco Netflow Collector
  - InfoVista VistaMart
  - Lucent NavisCore
  - MRTG
  - Cflowd
- Optionally, integrated network topology, configuration and traffic/utilization information can be imported from OPNET VNE Server, which is available separately.
- An import function allows importing network data from standard formats (XML, ASCII).

5.2.8.6.2 Troubleshooting and Validation of Network Changes

SP Guru allows service level management by reducing the frequency and duration of problems that influence services. The Virtual Network Environment provides a buffer between network engineering and production networks.

- SP Guru's NetDoctor allows identification of problems caused by network configuration errors. Testing the network model against established rule suites allows identifying configuration errors and warnings.
- Automated reachability analysis.

- NetDoctor's open scripting interface enables the creation of custom rules that codify local configuration policies.
- Planned network changes can be "tested" in the Virtual Network. SP Guru predicts link utilizations and end-to-end QoS measures after configuration changes or capacity upgrades. It automatically reports unrouteable flows and over-utilized links.

5.2.8.6.3 Network Dimensioning

SP Guru enables planning and engineering to support network growth, consolidation and new service deployment with predictions of network behaviour.

- Impact of deploying new technologies, protocols or services, including RFC 2547-compliant MPLS-based VPNs.
- "Testing" of the sensitivity of proposed network designs to assumptions about growth, service mix, etc.
- Comparison of network behaviour of alternative technology migration scenarios.
- BGP peering studies
- Failure Analyses to determine which traffic flows will be most affected by outages and where resulting bottlenecks are most likely to occur.
- Network failure analysis, such as the effect of MPLS Fast Reroute.

Features include:

- Flow Analysis determines the path of each logical link (e.g., PVC, LSP) and traffic demand and predicts the resulting impact on links and nodes as well as end-to-end delay.
- Discrete-event simulation of transient phenomena, such as failure situations. Hybrid Simulation of QoS for targeted demands and services in large-scale networks.
- Support for many protocols and technologies, including OSPF, IS-IS, BGP4, ATM/PNNI, MPLS, UMTS, DOCSIS, IPv6, IP Multicast, vendor-specific device models.
- Survivable MPLS Traffic Engineering for explicit routes that minimize maximum link utilizations under normal conditions and secondary routes that will survive link and node failures.

5.2.8.7 OPNET IT Guru

Network management for enterprise IT managers to diagnose application performance problems, validate changes to server and router configurations and plan for growth and high availability. Common uses are network design, capacity planning, technology evaluation, application deployment and for managing service level agreements.

5.2.8.7.1 Generating Models

Import part or the entire network, replicating all devices individually or aggregating at the segment or subnet level. IT Guru can also create a network model from router configuration files, automatically setting configurations on each router.

A network model can be loaded with 'traffic matrix' information representing traffic flows between end-systems. Import 'background traffic', to create a baseline load on the network infrastructure.

IT Guru can recreate the behaviour of the existing network or predict performance of network changes.

5.2.8.7.2 Device Configuration

IT Guru diagnoses the source of performance and configuration problems.

IT Guru tests the network model against rule suites in order to identify errors or warnings that result from misconfigurations or protocol conflicts. Definition of custom rules using an open scripting interface.

5.2.8.7.3 Validation of Network Changes

IT Guru's Virtual Network Environment provides a buffer between the network manager and the operational network by validating changes prior to implementation.

- NetDoctor module catches errors introduced by modifications to device configuration files.
- Flow Analysis module predicts link utilizations after configuration or topological changes are made.
- Protocol models verify if protocol tuning achieves the desired effect.
- Hybrid Simulation feature accurately estimates benefits of QoS policy modifications.
- Server models predict performance benefits of changes to server configurations (CPUs, disk interfaces), location and replication strategies.

5.2.8.7.4 Network Dimensioning

IT Guru combines discrete event simulation, analytical modelling and rules-based analysis.

- Simulation of projected traffic growth.
- Failure analysis to determine risk areas and planning of redundant routes.
- Service-level agreements (SLAs) for any metric, such as application response time.

5.2.8.8 OPNET NetBiz

Netbiz provides network equipment manufacturers and service providers a platform for automating network design, provisioning, proposals, and analysis. Netbiz is used to tune network designs and plan network evolution in the face of traffic growth, outages and the introduction of new technologies.

Features Include:

- Network Optimization
- Network Visualization
- Network Analysis
- Network Simulation
- Database Integration
- Custom Rules
- Custom Reporting

5.2.8.9 Applicability to MESCAL

Choice of OPNET Product

Several OPNET “flavours” are available as introduced. The most appropriate choice is Modeler, since source code is available for all library models. This is particularly interesting in the case of BGP, which MESCAL intends to extend within the project. If full simulations of the MESCAL architecture are to be implemented, extensive additional changes to protocols may be required.

OPNET SP Guru and IT Guru may be considered for less extensible, albeit rapidly implementable simulations. SP Guru mechanisms, such as topology discovery of existing networks and more special purpose data-evaluation tools, may speed up the simulation phase. However, SP Guru is targeted at

implementation and development of network topologies, not research of network technology, as required by MESCAL and extensibility of the simulator is therefore limited.

OPNET NetBiz is primarily targeted at providing a platform for automating network design, provisioning, proposals, and analysis. Business issues are not a primary concern of MESCAL so use is limited.

Advantages

OPNET is one of the most powerful simulator tools available with a complete set of required inter-domain protocols as well as some router device models. Should the complete simulation of the MESCAL architecture become necessary, OPNET may be most suitable choice.

Disadvantages

OPNET is an expensive tool, additional libraries may be required in addition to the simulator package. An OPNET licence has to be purchased individually by all partners who wish to use it.

5.2.9 General Simulation Environments

Apart from specialised network simulation environments, there are several more generic packages. These could be of use to MESCAL should the more specialised network simulators prove too complex or expensive for the simulations required or should the necessary inter-domain functionality not be available. Simulation environments such as MATLAB may provide a powerful ground on which to base simulations of more fundamental form. MATLAB is introduced in some detail, followed by a list of generic simulation platforms together with a short description and web links.

5.2.9.1 MATLAB

5.2.9.1.1 Overview

MATLAB is an integrated technical computing environment that combines numeric computation, advanced graphics and visualisation, and a high-level programming language. [MATH]

MATLAB includes hundreds of functions for:

- Data analysis and visualisation
- Numeric and symbolic computation
- Engineering and scientific graphics
- Modelling, simulation and prototyping
- Programming, application development and GUI design

MATLAB is used in a variety of application areas including signal and image processing, control system design, financial engineering and medical research. The open architecture makes it easy to use MATLAB and companion products to explore data and create custom tools that provide early insights and competitive advantages.

5.2.9.1.2 Add-on Products

MATLAB features a family of application-specific toolboxes that contain comprehensive collections of functions for solving particular classes of problems in areas such as signal processing, image processing, control system design, neural networks and more. Other products extend the capabilities of MATLAB to include data acquisition, report generation and the creation of standalone C/C++ code from MATLAB language programs, among others.

5.2.9.1.3 Extensible and Portable Design

MATLAB is easily extensible and it allows for creating applications. Many numerical problems can be solved in a fraction of the time that it would take to write a program in a language such as Fortran, C, or C++. External software can also be linked to and use data from MATLAB. MATLAB code,

called M-files, and data files, called MAT-files, are platform independent, so sharing ideas and designs across platforms is seamless.

More information on MATLAB can be found at http://www.mathworks.com/.

5.2.9.2 Further Simulators

Listed below are further simulation environments. None of them was investigated thoroughly, but are listed here for possible further use if required.

Name: Simulink [MATH]

URL: http://www.mathworks.com/products/simulink/

Description: Simulink is an interactive tool for modelling, simulating and analysing dynamic, multi-domain systems. Models are built using block diagrams to simulate the system's behaviour, evaluate its performance and refine the design. Simulink integrates with MATLAB, thereby providing immediate access to a range of analysis and design tools.

Name: The REAL Network Simulator [REAL]

URL: http://minnie.tuhs.org/REAL/

Description: REAL is a simulator for studying the dynamic behaviour of flow and congestion control schemes in packet switched data networks. It provides users with the means to build network models and tools to observe their behaviour. Source code is provided so that users can modify the simulator for their own purposes. The simulator is supplied with network topology, protocols, workload, and control parameters. It produces output statistics such as the number of packets sent by each source of data, the queuing delay at each queuing point, the number of dropped and retransmitted packets and more.

Name: Maisie Programming Language [MAISIE]

URL: http://may.cs.ucla.edu/projects/maisie/

Description: Maisie is a C-based simulation language purpose built for sequential and parallel execution of discrete-event simulation models.

Name: OMNeT++ - Object-Oriented Discrete Event Simulator [OMNET]

URL: http://whale.hit.bme.hu/omnetpp/

Description: OMNeT++ is an object-oriented modular discrete event simulator. The simulator can be used for modelling: communication protocols, computer networks and traffic modelling, multi-processors and distributed systems, etc. OMNeT++ supports animation and interactive execution.

Name: Traffic and Queuing Software [ERLANG]

URL: http://www.erlang-software.com/

Description: Erlang software supporting Erlang B, Erlang C, Poisson, Delay, Engset, and Binomial models. Can be used for Voice or Data Traffic Engineering and Call Centre design.

5.2.10 Investigation into AS topology generators

5.2.10.1 The AS topology

The topology of ASs in the Internet doesn't follow a global design as there is no single global entity governing its connectivity. The connectivity isn't purely random and the ASs can differ in purpose such as transit ASs and stub ASs. Stub ASs are those that have many end-users and usually cover the geographic space of their users (regional) while transit ASs are those that will only carry traffic between other ASs, usually over larger geographic distances (national). The transit ASs are usually further grouped into those which span multiple countries (owned by long-haul ISPs) and those spanning single countries (owned by national ISPs). Business relationships and policies [Minoli] limit the direct connection of Tier 3 ASs (regional ISPs) to Tier 1 ASs (long-haul ISPs) and technology limitations prevent the direct connection of ASs that are geographically far away. A three tier model is therefore often considered as a loose design for the Internet Ass (see MESCAL deliverable 1.4 [D1.4]).

While there is no explicit global design, there is, however, evidence for emergent traits. Faloutsos et al. [Faloutsos] showed a number of traits in three instances of the Internet AS topology: for four connectivity metrics (such as out-degree (the number of outgoing links) distribution) they followed a series of power-laws. This has led to a number of power-law generating models which are being proposed as models of the AS topology.

Lastly, real AS topology data over a number of years is available for public download [NLANR][routeviews][pch] but this data doesn't always form a complete map. This would however allow simulation on real, not estimated or modelled topologies.

5.2.10.2 Topology models

There are various models of what the AS topology looks like and they can be divided into two groups: those that are structure based, which consider the tiered architecture, and those that are graph based, which use graph theory methods to aim for a power-law complaint topology.

5.2.10.2.1 Structure based

These are some of the older models of AS topology and include models such as Transit-Stub [Ellen] and the Tiers [Calvert] model. These have been shown not to comply with the emergent power-laws [Medina][Tang].

5.2.10.2.2 Graph based

These usually create topologies that are compliant with ideal distributions of various metrics; for example the simplest random model that connects all nodes randomly with a uniform probability results in a bell shaped degree (number of links per node) distribution. Other models consider network evolution aspects such as network growth [Barabasi] (which leads to an exponential degree distribution) or a geographic distance cost function such as one proposed by Waxman [Waxman]. The more suitable models should however generate power-law compliant topologies. These include [Palmer][Mitzen] and the best known one, the Albert-Barabasi model [Barabasi].

5.2.10.3 Topology generators

Of the proposed models a range of them have been implemented in software:

5.2.10.3.1 GT-ITM

Is a freely available UNIX/Linux based package [gtitm] which implements purely random networks, the Waxman model, including variations of it, and hierarchical models, as well as transit-stub models. GT-ITM hasn't been updated since late 1996.

5.2.10.3.2 BRITE

The **B**oston university **R**epresentative **I**nternet **T**opology g**E**nerator [Medina][BRITE] is an attempt to build a framework for topology generators and includes implementations of a number of models. Among the models is the Waxman model but also the Albert-Barabasi model. BRITE has a wide range of features and options to generate AS level, as well as router level topologies. It is also capable of creating flat or hierarchical topologies. Additionally it has features pertaining to geographic node distribution as well as a wide range of topology input and output modules. BRITE was written to be extensible to allow for the creation of new models and portable (there are Java and C++ versions) so it can be run on a variety of platforms. Currently at version 2.0 BRITE is constantly in development and has mailing lists for support.

5.2.10.3.3 Inet

Inet [inet] was designed to generate AS topologies rather than be a universal topology generator. It uses a model different to Albert-Barabasi to generate power-law approximate topologies. This generator only provides a single model and does not seem to be very extensible but does have a number of configuration options to skew the distribution and therefore the result isn't always a power-law compliant topology. Inet-3.0 improves upon Inet-2.2's two main weaknesses by creating topologies with more accurate degree distributions and minimum vertex covers as compared to Internet topologies. Inet-3.0 also better approximates the actual Internet AS topology than does Inet-2.2. Inet-3.0's topologies still do not well represent the Internet in terms of maximum clique size and clustering coefficient.

5.2.10.3.4 Tiers

This model [Calvert] implements a three-tier hierarchy of LAN, MAN and WAN. The connectivity schemes used in each are different, such as star topologies for the LAN and the use of redundant links in the WAN. In the MAN and WAN tiers, geographic separation is considered when linking nodes. This generator was developed by November 1996 and has seen little development since then, but, more importantly, it doesn't produce power-law compliant topologies [Medina].

5.2.10.4 Applicability to MESCAL

Since the publication of the power-laws [Faloutsos], there has been a shift away from approximations of the expected structure to power-law compliant models, which are rather more representative [Tang] of the node level topology and the AS level topology. Inet and BRITE are the only generators developed specifically to model AS domains and the only ones able to produce topologies approaching the power-laws. No current topology generator will capture all the features of the Internet AS topology, even the power-laws are approximations based on partial measurements. Inet, but specifically BRITE, looks as if it will generate topologies most representative of today's view of the AS topology. The Albert-Barabasi model, in particular, should be enough to provide realistic models of Internet AS topology. If changes are needed, BRITE is quite flexible and could be modified.

What none of the topology generators currently do, however, is to provide realistic capacity values on links: an aspect required when simulating the use of network resources. As a control scenario, real network data could always be used; while not necessarily complete, it could still give a good idea of the real topology.

5.2.11 Simulators Comparison and Selection

The following provides a summary of the reviewed simulator packages[3].

While the functionality provided by the OPNET packages is clearly sufficient, these products had to be discarded for reasons of expenditure limitations.

[3] No selection is made on topology generators in this section.

The NS simulator, which was used extensively for the TEQUILA project, recently acquired BGP support. However, the NS BGP module lacks some advanced features, its functionality has not yet been extensively validated and the NS packet forwarding mechanism needs to be modified in order to use the routing tables constructed by this BGP module. Therefore, NS can only be considered for packet-level simulations that don't require any BGP functionalities.

The MRT package may prove useful for AS emulation, the apparent lack of development and support are a large drawback.

The simulators of most interest to MESCAL are compared in *Table 9*. The remaining packages – especially the more generic tools such as MATLAB - may have applications in smaller simulations that could become necessary in due course of the project.

From the table, and excluding OPNET and NS for the reasons already discussed, J-Sim and SSFNET are the most appropriate tools to undertake the MESCAL experimentation involving BGP. Both these two simulators already include implementations of the basic functionality required by MESCAL, thus the implementation effort can focus exclusively on the additional functionality that will be defined in WP1. In the following the arguments will be summarised for the final selection between the two simulators according to the functionality and experimentation requirements of the project.

	BGP	**DiffServ**	**RSVP**	**MPLS**	**IP Multicast**	**IPv6**
NS	X+	X	X	X	X	N
OPNET	X	X*	X	X	X	X
J-Sim	X+	X	X	X	X	
SSFNET	X			X+		

** See http://cairo.cs.uiuc.edu/qosrouting/hpr.html for DiffServ module*
+ Tentative
N Only support Mobile IPv6

Table 9 Simulator capabilities comparison

SSFNET has the implementation of BGP4 based on RFC 1771. It has recently received attention by researchers on BGP simulations (e.g. stability and convergence studies of BGP). A number of papers published in some major conferences have presented their simulation results using SSFNET. Hence, it seems that SSFNET is progressively accepted and used as a legacy BGP simulator. SSFNET is free for research purpose and source code - especially its complete BGP module - is available for MESCAL in order to implement required qBGP extensions. SSFNET has greater processing requirements than NS-2, but smaller memory demands. This is particularly important when performing large-scale simulations, such as those that MESCAL will be considering. However, SSFNET does not support other mechanisms that are considered by MESCAL such as DiffServ, MPLS, Multicast, etc. Nevertheless, SSFNET can still be considered by MESCAL if the implementation is not based on all these mechanisms. For example, DiffServ implementation can be avoided if we only consider the flow level measurements, for which SSFNET uses NetFlow, for measuring and filtering IP flows. In this case, SSFNET is sufficient for MESCAL.

J-Sim has recently released a BGP module developed by the University of Namur, in Belgium, within the context of the IST Project ATRIUM. The J-Sim BGP implementation is a truncated version of the SSFNET BGP implementation. Hence, at least, their basic functionalities are similar. In addition, J-Sim BGP is enhanced [Quotin02], to support the redistribution community attributes [2], which are used to build scalable traffic engineering configurations. However, some other functions from SSFNET have not been completely ported into or extensively tested in J-Sim BGP. For instance, route reflectors have not yet been ported, but this is something the developers plan to do in the future. Since J-Sim BGP is extended from SSFNET, the amount of effort/work for a qBGP implementation in both of them is estimated to be approximately the same.

J-Sim supports most of the mechanisms that are considered by MESCAL such as DiffServ, MPLS. Hence, these mechanisms can be used together with BGP (or qBGP), to perform simulations for

MESCAL approaches, such as qBGP/MPLS based on DiffServ. Currently only very few publications, all are written by the developer, base their results on J-Sim BGP or MPLS. Thus, the validity of these modules has not been yet widely acknowledged, which is a risk for the simulation development.

On the other hand, J-Sim requires significantly less memory than SSFNET, thus allowing larger simulations. For example, it can run simulations with an AS-level topology of a size close to the magnitude as the Internet (nearly 3000 AS and 10000 BGP sessions). However, J-Sim may be a little slower than SSFNET. A trade-off of memory requirement versus execution speed, exists between J-Sim and SSFNET. J-Sim will be more appropriate for MESCAL, if large-scale simulations will be required. The simulation performance analysis of NS, J-Sim and SSFNET can be found in [Eval02] and [Nicol02].

5.3 Traffic generators

5.3.1 Why and where?

Within the context of the MESCAL and from a routing perspective, the use of traffic generators could be subdivided into two cases:

- The first use consists in generating traffic in order to test if the traffic is well routed. In particular if it is sent to the appropriate destination, via the correct inter-domain path with the specified QoS guarantees. For this purpose we will use a SmartBits traffic generator.
- The second use consists in generating pure BGP messages: one of the uses of these messages is to test the capability of the modified inter-domain routing process to handle these messages and quantify its performance. For this purpose we will use QA Robot.

5.3.2 Availability

The aforementioned tools are operated under FTRD licenses, however results of activities carried within these tools will be made available to the MESCAL Consortium.

5.4 Management Interfaces

Configuration of networks' devices has become a critical requirement for operators in today's highly interoperable networks. Operators from large to small have used vendor specific mechanisms or developed their own mechanism to transfer configuration data to and from a device, and for examining device state information which may impact the configuration. Each of these mechanisms may be different in various aspects, such as configuration data exchange, user authentication, session establishment, and error responses.

5.4.1 Command-line interface (CLI)

The CLI commands allow the user to configure, manage, and troubleshoot the routers (e.g., Linux and Cisco). Linux-based ZebOS Server Routing Suite (SRS) platforms use industry-standard command line interface. The user can access to the ZebOS Server Routing Suite (SRS) CLI through a standard Telnet or SSH session. It is also possible to manage ZebOS SRS through the VTYSH daemon. HTML web user interface for administration is also provided in Zebos. The majority of configuration and monitoring of Cisco routers is via the proprietary CLI. The Cisco IOS CLI is divided into different modes. Two main modes are the user mode and the privileged EXEC mode. In user mode, only a limited subset of the commands are available such as *show* commands, which show the current configuration status, and *clear* commands, which clear counters or interfaces. To have access to all commands, a password protected privileged EXEC mode is used. From this mode, any privileged EXEC command or enter global configuration mode can be used. Using the configuration modes,

changes to the running configuration can be made. Standard MIB access is also provided via CLI for some features.

5.4.2 Configuration Management with SNMP

The ‘snmpconf’ working group is chartered to create a Best Current Practice document that outlines the most effective methods for using the SNMP Framework to accomplish configuration management. The scope of the work includes recommendations for device specific as well as network-wide (Policy) configuration. The RFC 3512 is an informational RFC to provide guidelines on how to best use the existing Internet Standard Management Framework to perform configuration management. This information is relevant to vendors that build network elements, management application developers, and those that acquire and deploy this technology in their networks.

The group is also chartered to write any MIB modules necessary to facilitate configuration management. The Internet draft [Hazewinkel 03] describes a MIB module that provides a conceptual layer between high-level "network-wide" policy definitions that effect configuration of the Diffserv subsystem and the instance-specific information that would include such details as the parameters for all the queues associated with each interface in a system. This essentially provides an interface for configuring differentiated services at a conceptually higher layer than that of the Differentiated Services MIB. The Internet draft [Waldbusser 03] defines a portion of the MIB for use with network management protocols in IP-based internets. In particular, this MIB defines objects that enable policy-based monitoring and management of SNMP infrastructures as well as a scripting language and a script execution environment.

Zebos provides SNMP management support. It supports SNMP management through IETF compliant SNMP MIBs.

Cisco uses SNMP for network management purposes, OSPF configuration management, reloading the router configuration, and so on.

- All Cisco IOS Software releases to date include SNMPv1.
- Releases from IOS Software 11.2(6)F and later have SNMPv2C support.
- Releases from IOS Software 12.0(3)T and higher have SNMPv3 support.

Some SNMP platforms can directly share data with the CiscoWorks2000 server using Common Information Model/eXtensible Markup Language (CIM/XML) methods. CIM is a common data model of an implementation-neutral schema for describing overall management information in a network/enterprise environment. CIM is comprised of a specification and a schema. The specification defines the details for integration with other management models such as SNMP MIBs or Desktop Management Task Force Management Information Files (DMTF MIFs), while the schema provides the actual model descriptions.

5.4.3 XML-based Network Configuration

The IETF ‘netconf’ Working Group is chartered to produce a protocol suitable for network configuration, with several characteristics including: to be extensible enough that vendors will provide access to all configuration data on the device using a single protocol, to use a textual data representation, that can be easily manipulated using non-specialised text manipulation tools, to have a programmatic interface, and so on.

The working group has taken the XMLCONF Configuration Protocol [Enns-xml 03] as a starting point. An Internet draft [Enns-netconf 03] is produced that defines a mechanism through which a network device can be managed. Netconf uses a remote procedure call (RPC) paradigm to define a formal API for the network device. The Netconf protocol use XML for data encoding purposes, because XML is a widely deployed standard, which is supported by a large number of applications. A client encodes an RPC in XML and sends it to a server and the server responds with a reply encoded in XML.

Most of the XML-based network configuration activities are work in progress and is at pre-standard stage. There are some products explained in the following sections that use XML-based interfaces.

5.4.3.1 XML based interfaces to network devices

The use of a standards-based XML-based API improves interoperability and integration of diverse systems by providing a common language through which network management applications and routing platforms can communicate. An API can also dramatically reduce the amount of integration work necessary to make a complex multi-vendor network manageable.

Cisco systems:

- NetFlow services capture a rich set of traffic statistics exported from routers and switches while they perform their switching functions. CNS NetFlow Collection Engine is a platform that provides fast and scalable data collection from multiple export devices exporting NetFlow data records. A XML interface (CNS/ XML) is used as a message-based application interface that allows for messaging from, and remote manageability of, the CNS NetFlow Collection Engine application. The CNS/XML interface is to send and receive configuration/control requests and responses, and unsolicited event notifications.
- CiscoWorks Data Extracting Engine is a utility that allows customers to extract data for devices managed by CiscoWorks. It extracts detailed device inventory and running configuration information in XML format from an inventory database and configuration archive.
- IP phones and call manager speak in XML format.

NextHop Technologies: NextHop Technologies' GateD family of products is a vendor-neutral, control-plane solution which provides layer 3 IP routing protocols, MPLS, virtual routing, and virtual private networking. NextHop Technologies GateD 10.0 routing software is the first IP routing source code product with an XML Routing API. This API enables vendors to quickly and easily build customised command line or web interfaces and provide users with a readily accessible script interface for managing any network device. For more information visit: www.nexthop.com

Juniper Networks: JUNOScript is an XML-based Network Management API for managing devices. XML-based JUNOScript API provides a standard integration layer between management applications and the platforms that they manage. It allows access to both operational and configuration data using a simple RPC mechanism. Sessions can be established using a variety of connection-oriented access methods. The JUNOScript API is an alternative to the existing SNMP and CLI scripting methods. Visit http://www.juniper.net/ for more information.

5.5 Brought in software

5.5.1 Selection Criteria

The background objects reviewed in this chapter are software components relevant to MESCAL work, which are either commercially available or have been specified, developed and implemented in the context of other R&D projects, especially in the context of the IST TEQUILA project. Depending on well-defined criteria, the envisaged use of a background object to MESCAL is seen along the following two dimensions:

- it can be used to facilitate the specification and design of a MESCAL component,
- its implementation can be used as such, or affordably modified, to be incorporated in the MESCAL system.

The criteria against which the presented background object will be selected for facilitating MESCAL system design and realisation, are listed in the following.

Functional suitability: The criterion of functional suitability examines how appropriate the functionality of a background object is in the MESCAL functional model.

Availability: The availability criterion entails both the physical availability (if the under consideration object exists physically and it is in an operating state) and the constraints that may restrict its distribution (since the background objects come from other project consortia with various types of restrictions).

Ease of migration: The ease of migration criterion entails the following aspects:

Available documentation: for ensuring adequate level of information regarding the functionality/interface specification/run time environment of the component, according to the specific MESCAL needs.

Operating platform: the operating platform (the platform that the object operates on) must be compatible with and/or affordably ported to the platform(s) selected for the MESCAL systems.

Technology and development platform: the technology and development platform and tools used for implementing the object must be compatible with and/or affordably ported to the platform(s) selected for the MESCAL systems.

Modifications and support: The effort (in PMs) required for carrying out the necessary modifications (either by object developer or by the project team) for incorporating the functionality and interfaces of the background object to the MESCAL system. The effort should be assessed according to the MESCAL implementation plans. The available support from the developers of the object to facilitate bug-fixing, clarification of any obscurities should also be taken into account.

5.5.2 SoA Review

5.5.2.1 TEQUILA Service Negotiation Protocol (SrNP)

5.5.2.1.1 Overview

To automate the process of service subscription, TEQUILA has specified a protocol for service negotiations, the *Service Negotiation Protocol (SrNP)*. SrNP is an application layer protocol, which provides the necessary messages and procedures for enabling the process of negotiations between two parties towards establishing, modifying and terminating service contracts. It is session-oriented and adopts a client-server, dialogue-based (half-duplex) approach for implementing the required interactions. It offers a number of negotiation primitives: counter-offers, 'last word'/‘take it or leave it’ and ‘please wait’. The following aspects contribute to protocol openness and applicability. SrNP is not specific to the SLS context or format; it is general enough to apply for negotiating any document. Furthermore, it clearly distinguishes between the negotiation primitives it offers and the logic according to which negotiations are conducted on behalf of each negotiating party; in essence, SrNP provides the necessary services based on which application-specific negotiation logics could be built. SrNP has been carefully designed so as not to duplicate but to complement the functionality of existing QoS signalling or reservation or session establishment protocols. Finally, SrNP is independent of the underlying transport protocols. SrNP has been implemented in C++ on the TCP/IP stack, and in Java on the HTTP stack.

5.5.2.1.2 Applicability to MESCAL

One of the aspects of the MESCAL work is the negotiation and subscription of SLSs between customers and providers (cSLSs) and between providers (pSLSs). SrNP may be used for providing the basis for these negotiations. Its suitability emerges from its very features. It focuses on SLS negotiations it is agnostic to the SLS context and format (so it can support any cSLSs and pSLSs

specifications). Furthermore stable, well-tested and well-documented implementations can be made directly available from TEQUILA partners participating in MESCAL.

5.5.2.2 TEQUILA Service Subscription Management (SSM)

5.5.2.2.1 Overview

Service Subscription Management (SSM) is a module of the TEQUILA system responsible for handling service subscription requests. It entails functions for configuring/activating the service according to agreed subscription requirements as well as admission control functions. SSM includes the following aspects:

Modelling of QoS-based connectivity services

Based on the proposed TEQUILA SLS and service subscription templates and utilising the expressive power of XML schemas, a number of QoS-based connectivity services have been modelled e.g. multi-service fixed/aggregate VPN, dial-up Internet, server-access, peer-to-peer services. To increase its practical usefulness and applicability, the proposed modelling approach distinguishes between customer- and network-level views of QoS-based connectivity services. The network-level view represents a service from the perspectives of the network i.e. as understood by the capabilities of the network, and its information model directly corresponds to the TEQUILA SLS template; as such, all QoS services have the same network-level view. The customer-level view represents a service in a way suitable to the customers, where suitability is meant from the point of view of comprehension and easiness of the subscription form fill-in process. The proposed modelling approach specifies means for defining customer-level views, that is, QoS services based on the TEQUILA SLS template, and for automatically mapping them to the appropriate network-level view using XSLT transformations.

Subscription validation and translation

Validation is required for ensuring the correctness (in terms of syntax and content) of the submitted subscription request, and for ensuring uniqueness of customer/user identification and their flow specifications amongst subscriptions. Translation is required for extracting the information required for configuring/activating the service in the network. The translation process involves the analysis of the subscription request into its constituent SLSs –connectivity legs-, their mapping from customer to network parlance (cf. customer- and network-level views above), from customer sites to network edges, from quality levels as understood by the customers to OAs supported by the network and from customer flow address groups to extended IP access lists. All these functions are executed automatically (implemented as PL/SQL processes).

Subscription admission logic

The objective of the subscription admission logic is to determine whether requested service subscriptions could be accepted, with the purpose not to eventually overload the network, while maximizing subscribed traffic. Subscriptions are accepted as long as the network could satisfactorily accommodate the demand of the already established subscriptions and of the requested one; otherwise negotiations are initiated or the subscription is rejected. Satisfactory accommodation is deduced on the basis of estimates of the availability of the engineered network to accommodate QoS traffic, produced by the off-line TE functions, and a policy parameter, called satisfaction level, which determines how stringently the network availability estimates would be interpreted.

Subscription Negotiation Server

The Subscription Negotiation Server drives the subscription process on behalf of the provider. It receives subscription requests from customers and handles the negotiations. The protocol used for the communication between the customers and the negotiation server is the aforementioned SrNP –the negotiation server implements the server side of SrNP. The negotiation server is able to serve multiple subscription requests at the same time (a separate thread handles each request). The subscription validation and translation and the subscription admission logic processes, mentioned above,

implement the logic behind the subscription negotiations, being invoked by the negotiation server for every incoming subscription request.

5.5.2.2.2 Applicability to MESCAL

The TEQUILA SLS template can be used by MESCAL for modelling cSLSs and pSLSs, with appropriate enhancements to capture inter-domain and multicast service aspects.

The TEQUILA service modelling approach, subscription validation and translation processes and the subscription admission logic could be used for their underlying concepts and notions. The gained know-how could be directly fed to the MESCAL inter-domain service architecture; handling of service requests between providers and between customers and providers. Appropriate enhancements should be made to adjust them to the MESCAL inter-domain service and traffic engineering architecture.

5.5.2.3 TEQUILA Network Dimensioning (ND)

5.5.2.3.1 Overview

Network Dimensioning (ND) is a centralised module of the TEQUILA system, responsible for dimensioning the network on the basis of anticipated QoS traffic demand. The network is dimensioned in terms of:

- QoS-route constraints, which in an MPLS environment result to LSPs, whereas in an OSPF-based routing environment to link weights per OA,
- PHB configuration parameters, in terms of link scheduling and buffer parameters as appropriate for the PHB, and
- network (edge-to-edge) availability estimates to accommodate QoS traffic.

ND includes the following aspects:

Traffic aggregation

Anticipated QoS traffic demand is derived by aggregating the maximum subscribed demand of existing and forecasted subscriptions –note that subscriptions clearly specify their QoS requirements and topological scope, which have been mapped to OAs and network edges respectively, through the subscription validation and translation functions (see previous section). Aggregation is done on the basis of commonly accepted parameters, namely multiplexing factors and aggregation weights per distinguished service class. The notion of a service class is introduced to cope with user diversity in service usage -the services of a service class bare similar usage patterns. The determination of service classes and their aggregation parameters fall into the realm of traffic analysis, which is outside the scope of our work; these are assumed as input.

Mapping of QoS traffic to network resources

Anticipated QoS traffic demand is mapped into network physical resources so that certain network-wide optimisation criteria are satisfied, while QoS requirements and link capacity constraints are met. To this end, an appropriate mathematical optimisation problem is formulated and solved. The optimisation criteria express criteria for network-wide load balancing; different such criteria may be used depending on a relative parameter, which is left to be set by policies. QoS requirements are (statically) translated into hop-count or additive metric constraints in determining edge-to-edge routes. The optimisation procedure is based on the Gradient Projection method, which is commonly used in solving optimum routing problems. It is an iterative procedure and the number of iterations affects the number of alternative routes to be found; as such, the maximum number of iterations is left to be set by policies. The outcome of this procedure consists of: sets of appropriate QoS routes from edge to edge, and expected load per PHB per link as yielded by optimally distributing, i.e. through the determined routes, the anticipated QoS traffic demand between network edges.

Constrained Steiner-tree algorithm

At each iteration of the network optimisation procedure outlined previously, algorithms for calculating optimum constrained (Steiner) trees need to apply to determine appropriate QoS routes between the network edges. The reason for considering trees, not paths, lies in the fact that anticipated QoS traffic is in general of hose type (point to multi-point), not necessarily of pipe type (point-to-point). The cost of the trees against which optimality is sought for, reflects the load to be routed down the tree. The constraint in tree finding reflects the QoS requirements of the traffic to be routed through the tree. It is expressed in the form of hop-counts from the tree source to the leaves or in the form of an upper bound with respect to an additive metric alternative to the one used for calculating the tree cost. In addition to well-known graph-theory algorithms, TEQUILA has designed new algorithms for finding optimum constrained trees. The rationale behind this effort was to improve the known trade-off between algorithm processing requirements and yielded optimality.

Network availability estimates

Having mapped anticipated QoS traffic demand into the physical network resources, the availability of the network to accommodate QoS traffic is determined. Network availability is determined edge-to-edge and per supported OA and is based on the topology of the corresponding QoS routes found by the network optimisation algorithm. These estimates are mainly required by the service admission control functions so that to base their decisions on the availability of the engineered network to accommodate QoS traffic. Several network availability indications are calculated depending on whether the anticipated QoS traffic demand exceeds the one for, which the network was optimised- and on whether link congestion amongst competing QoS traffic is resolved. The algorithms used are based on well-known algorithms for calculating maximum flows (minimum cut) in directed networks. In parallel to calculating network-wide (edge-to-edge) QoS availability, PHB load availability is also calculated. These PHB load estimates are obviously in accordance with the network-wide availability estimates, as they are produced based on the QoS routes determined in the previous step. They are used by the dynamic TE functions for setting accordingly the link scheduling parameters required by the implementation of the PHBs.

LSP/OSPF and PHB configuration

The results produced by the above-described functionality, QoS routes, several indications of PHB loads and edge-to-edge network availability for QoS traffic, are independent of the underlying network technology. Therefore, they need to be mapped to the appropriate entities/information supported/required by the underlying TE capabilities. Specifically: PHB load indications are mapped to link scheduling parameters as appropriate per PHB. Considering an MPLS-capable IP network, the determined QoS routes are mapped to LSPs. Furthermore, the IP addresses of the subscribed flows are assigned to one of the alternative LSPs established between the edges corresponding to the topological scope of the subscriptions. This is done following a bin-packing type of logic, taking into account the bandwidth allocated to LSPs and the rate requirements of the subscribed flows, and in a way to eliminate duplicate or overlapping, in terms of IP address space, assignments. These results are fed to the appropriate lower-level distributed, dynamic TE and service admission control functions, and eventually through them to the routers, through automated means, depending on the routers' interface capabilities. TEQUILA has considered Cisco and Linux-based routers.

Web-based interface

A web-based interface has been developed for viewing the produced network dimensioning results and related statistics. The interface provides for a number of queries to enable viewing of the results in a variety of ways, per (groups of) edges, interfaces, OAs or conditions regarding their values.

5.5.2.3.2 Applicability to MESCAL

The principles underlying the TEQUILA ND design, leading from customer service subscriptions to network resource dimensioning and configuration, and the know-how gained could be fed to the design of the inter domain traffic engineering of MESCAL. The component-based design of the

Network Dimensioning system facilitates modifications and the incorporation of its aspects to the MESCAL system.

The ND component can be used ‘as is’ in MESCAL for the purpose of intra-domain traffic engineering and implementing QoS provisioning within an autonomous system.

5.5.2.3.3 TEQUILA Router Service Invocation Management (RSIM)

5.5.2.3.4 Overview

The Router Service Invocation Management (RSIM) component is part of the Service Management subsystem, responsible for performing authorisation and admission control at service invocation epochs and for activating the admitted service invocations. The term invocation implies both implicit and explicit service invocations. In implicit service invocations users may directly inject traffic into the network, e.g. for VPN services, whereas in explicit service invocations a signalling protocol interaction is required with the SP domain before traffic is actually injected into the network. RSIM consists of the following aspects:

QoS Signalling

Signalling has two distinct aspects: the customer-to-network and the network-to-network communication. The customer initiates and terminates a service invocation interacting with an ingress node. The ingress node performs admission control based on local information and in turn interacts with the egress nodes to ensure resource availability at the end access interfaces. The signalling protocol is based on RSVP. The invocation data carried by the invocation messages are based on the proposed TEQUILA SLS template. The RSVP machinery is exploited for communication between peers and the invocation signalling messages are encapsulated into the POLICY_DATA object of RSVP messages. In Linux-based routers the RSVP Daemon and the RSVP API have been appropriately enhanced. In Cisco routers, deployment using the COPS for RSVP outsourcing native capabilities has been studied.

Authentication and Authorisation

Service invocations or terminations are processed by the admission control functions residing at network edges. Authentication and administrative authorisation checks are first performed for ensuring that the requested invocation corresponds to an established subscription and is in line with the agreed subscription profile. Once these checks pass successfully, resource-based authorisation is performed for checking that the agreed QoS requirements of the invoked SLSs can be delivered by the network, given its actual state. Finally, once the invocation is admitted, it is activated in the network. To activate an admitted invocation request, the invoked SLSs are mapped into appropriate traffic conditioning blocks enforced at the service's ingress router(s). Authentication and administrative authorisation entail the execution of queries on subscription data held in appropriate repositories. Recourse-based authorisation is based on probabilistic criteria for deciding whether to admit requested invocations, entailing simple checks on locally available data, which are determined by the invocation admission functions discussed in the following.

Invocation admission

Invocation admission functions aim at meeting in an optimum way the following diverse objectives: The objectives of the resource-based admission control logic are: to maximise the use of network resources; to prevent QoS deterioration; to effectively resolve eventual network congestion and to provide fair treatment to services network wide. To this end, admission control regulates the traffic entering the network. This task encompasses two different aspects: (1) control the number and the type of the active services and (2) control the volume of the traffic injected by the active services. Hence, admission control operates at both the invocation (call) and the rate (packet) levels.

Admission logic is based on a feedback model and is decomposed into the dynamic admission management and the admission strategy enforcement functions. The dynamic admission management function, driven by monitoring events, deduces the appropriate admission control strategy to apply, based on estimates of the engineered network to accommodate QoS traffic between its edges, which

have been produced by the off-line TE functions (5.5.2.3). The monitoring events are in the form of edge-to-edge network status alarms and threshold crossing events on aggregated traffic, injected into the network by the local edge. Network status is expressed as a binary flag, indicating whether QoS requirements are (close to be) violated or not. An admission control strategy specifies the service invocation admission probabilities and the service rate to be offered to the active services. The admission strategy enforcement function realises the determined strategy. This is done by interacting with the customer, to handle explicit service invocation requests, and with the routers, to configure the appropriate traffic conditioning settings that determine the admitted service rate for the active services.

5.5.2.3.5 Applicability to MESCAL

QoS Signalling, authentication/authorization and invocation admission are all necessary aspects of the intra-domain aspects of the MESCAL system. As such, the TEQUILA results could be used with the appropriate enhancements to make them fit to the inter-domain requirements of MESCAL. Such enhancements should be made along the following dimensions: QoS signalling may need to traverse different autonomous systems, authentication/authorization must be aligned with the MESCAL SLS template (both cSLSs and pSLSs) and invocation admission logic must take into account the inter-domain traffic engineering, monitoring and pSLSs agreement requirements of MESCAL.

5.5.2.4 TEQUILA Dynamic Routing Management (DRtM)

5.5.2.4.1 Overview

Following an MPLS-based explicit-path routing scheme, whereby routes for QoS traffic are calculated by off-line TE functions in the form of edge-to-edge LSPs, the TEQUILA system incorporates QoS-based dynamic routing functions with the purpose to drive the network in load-balanced states as far as the load in each LSP is concerned. In essence, the TEQUILA dynamic routing functionality tries to exploit the advantages of multi-path routing with the purpose to increase network goodput (QoS traffic throughput).

The TEQUILA dynamic routing functionality sees into dynamically distributing incoming load to the determined LSPs, trying to avoid cases where some LSPs are critically overloaded whereas others are not. This is achieved by dynamically assigning/reassigning flows to alternative LSPs; and results in re-writing the LSP forwarding tables maintained in the routers. The IP addresses of the flows to be assigned to LSPs are drawn from the established subscriptions.

Dynamic routing functionality is distributed at the network edges and is triggered whenever new subscriptions are accepted and by LSP related events received by the network monitoring services: LSP performance deterioration (reactive operation) and LSP load threshold crossings events (proactive operation). On subscription acceptance events, the flows of the accepted subscriptions are assigned on per SLS basis to one of the alternative LSPs, based on a bin-packing type of logic, taking into account the bandwidth allocated to LSPs by the off-line TE functions and the rate requirements of the subscribed SLS flows. SLS flows are assigned to LSPs in a way to eliminate duplicate or overlapping, in terms of IP address space, entries in the resulting LSP forwarding tables. On network triggers, the flows currently assigned to the LSP, which the trigger referred to, are reassigned amongst the alternative LSPs, if such exist. Flow reassignment is based on a bin-packing type of logic, taking into account the rate requirements of the subscribed flows and the spare bandwidth of the LSPs, with respect to their current load and the bandwidth allocated by the off-line TE functions. To avoid out-of-sequence packet delivery, reassignment of currently active flows is prohibited or taken as last action, depending on the QoS requirements of the flows.

5.5.2.4.2 Applicability to MESCAL

The principles underlying the TEQUILA dynamic routing management functionality and the know-how gained could be fed to the design of the (intra- and inter-) traffic engineering solutions of MESCAL. The component can be used ‘as is’ in MESCAL as part of the intra-domain traffic engineering solution for QoS provisioning within an autonomous system.

5.5.2.5 TEQUILA Dynamic Resource Manager (DRsM)

5.5.2.5.1 Overview

Dynamic ReSource Management (DRsM) aims at ensuring that link capacity is appropriately distributed between the PHBs defined on a link. This is achieved by dynamically configuring buffer and scheduling parameters associated with the interface the link is attached to, according to Network Dimensioning directives, constraints and rules. DRsM has distributed functionality, with an instance attached to each router. DRsM gets estimates of the *required* resources for each PHB at each network interface, from Network Dimensioning, together with a margin on the *required* bandwidth, in terms of upper and lower bounds. Within the limits of this margin, DRsM is allowed to dynamically manage the resource reservations (i.e., the *effective* resources required to cope with e.g., unexpected SLS invocations). DRsM manages the resources according to its algorithm, which considers actual, experienced load as compared to the required resources. Compared to Network Dimensioning, DRsM operates on a relatively short time-scale (e.g., seconds or minutes).

The experiments undertaken in TEQUILA show that DRsM improves QoS, in terms of reduced packet losses, in all traffic scenarios and under all aggregate average load levels tested (refer to TEQUILA D3.4 for details of experiments performed). It has been shown that the DRsM algorithm, managing PHB scheduling parameters according to actual load conditions, improves network performance in terms of packet losses when compared to static link bandwidth allocation.

5.5.2.5.2 Applicability to MESCAL

The DRsM algorithms developed in TEQUILA are applicable to dynamic intra-domain TE schemes for delivering QoS across individual domains, and as such are within the scope of MESCAL, even though *intra*-domain QoS is not the main focus of the project. MESCAL does not intend to develop new intra-domain TE algorithms apart from those additional aspects required for interworking with the inter-domain TE algorithms and for dynamically managing intra-domain resources involved in end-to-end QoS-based services. For this reason the TEQUILA implementations could be reused in any prototype deployments in the MESCAL testbed. This is dependent on the types of experiments to be undertaken in the project, however, and these have not been frozen at the time of this report.

While DRsM in TEQUILA was responsible for managing intra-domain resources there is the potential for a similar function to apply to inter-domain resources between service peers: the pSLSs. In this case a DRsM-like component could be responsible for distributing link capacity between a number of pSLSs established on an inter-domain link. This assumes that there is a dynamic pSLS invocation scheme agreed between ASs which would allow existing pSLSs to be modified (within certain constraints). In such a scenario there needs to be an agreement between service peers on any modification to pSLS capacities which could be implemented via a signalling protocol, for example. In this sense such algorithms would differ from the intra-domain TEQUILA DRsM which was able to determine and configure link bandwidth distribution autonomously.

If an inter-domain dynamic resource management mechanism is adopted in MESCAL then the principles underlying the TEQUILA DRsM functionality and the know-how gained could be fed to the design of MESCAL solutions.

5.5.2.6 TEQUILA Monitoring

5.5.2.6.1 Overview

Quality of service (QoS) delivery in IP networks is an important area for service providers, pointing to new business opportunities for premium quality traffic. QoS-based value-added services in IP networks necessitate the use of traffic engineering. Traffic engineering relies on measurement data to be used for network provisioning, dynamic resource allocation, route management, and in-service performance verification of value-added IP services. Monitoring systems are becoming increasingly important for providing quantified QoS-based services. As the network attempts to offer several

service types by employing traffic-engineering mechanisms, service monitoring is important for providing end-to-end quality of service assurance. In this context, the role of monitoring systems goes beyond diagnostic monitoring in current networks by becoming an important tool for supporting the network operation and providing service auditing for both traditional and value-added services. A monitoring system should be scalable in terms of network size, speed and number of customers subscribed to value-added services. Scalability is thus a big challenge for monitoring systems and necessitates suitable system architectures for achieving it in order to provide real-time monitoring information. TEQUILA proposed novel principles for designing and assessing monitoring systems and implemented a scalable Intra-domain *Monitoring system* for QoS delivery in IP Differentiated Services networks. The proposed novel principles for designing such a *Monitoring system* are as follows:

Defining monitoring process granularity

Distributing data collection system at node level

Minimising measurement transmission overhead

Using aggregate performance measurements in combination with per-SLS traffic measurements

Reducing and controlling the amount of synthetic traffic

The TEQUILA *Monitoring system* is distributed throughout the network, and has both per-node components and centralised components. The Generic Adaptation Layer (GAL) is used as per-node component. The other components used are: Monitoring agents as per-node components, Node Monitoring as per-node component, Network Monitoring, SLS Monitoring, and Monitoring GUI as centralised components. It also includes the Monitoring Repository which contains configuration information, the Network Topology Repository that contains details of the network topology, including physical routers, interfaces and links, a Monitoring GUI, CORBA Name Server, and CORBA Notification Server used to de-couple monitoring clients from the sources of monitoring events. Node Monitor is responsible for node related measurements and there exists one per network element (router). Node Monitor is able to perform active measurements between itself and any other Node Monitor, at path or hop level, as well as passive monitoring of the router it is attached to. Network Monitor is responsible for network-wide processing of measurement data. SLS Monitor is responsible for customer related service monitoring. In-service verification of traffic and performance characteristics per service type is required for monitoring the continuity and quality of services offered to customers, auditing the services, and preparing reports. Monitoring Repository consists of two major parts for data cataloguing, a "data store" for storing large amounts of measurement data and an "information store" for storing smaller amounts of configuration information. Monitoring GUI is used for displaying the measurement results. Components communicate between themselves and with clients using CORBA RMI. The CORBA Notification service is used to deliver monitoring results to clients. Components access an Oracle database to retrieve configuration information, network topology information, and to store monitoring results.

TEQUILA *Monitoring system* supports both MPLS and IP-based DiffServ-capable networks. The *Monitoring system* is distributed in order to guarantee quick response times and to minimise necessary management traffic and ensures to maintain stability as the network size increases. It is capable of providing both threshold-crossing events and periodic statistical events. Objects that can be monitored are LSPs, PHBs, and IP routes. Monitoring activity is classified as either passive, or active. Passive metrics are as follows: LSP offered load, LSP throughput, PHB bandwidth used, PHB packet discards. Active metrics are as follows: PHB, LSP, and IP route one-way packet delay (OWD) and one-way packet loss (OWL). Active monitoring is further classified as either edge-to-edge, per-hop, or hop-by-hop. The edge-to-edge approach considers an entire path through the domain and involves injecting synthetic traffic at a domain ingress node and receiving it at a domain egress node. The per-hop method considers a single hop within the domain. The hop-by-hop approach is an extension of the per-hop method in which the results of several per-hop measurements are aggregated to derive an edge-to-edge result. The motivation behind the hop-by-hop approach was to reduce the quantity of synthetic traffic injected to the network when compared to the edge-to-edge approach.

TEQUILA *Monitoring system* demonstrated the ability of a *Monitoring* system in providing measurements in relatively short time-scales at the path level granularity in order to assist various management and control functions. It contributed towards operationally optimised traffic-engineered DiffServ networks that can support large number of customers.

5.5.2.6.2 Applicability to MESCAL

It is possible to use the TEQUILA *Monitoring system* for both intra- and inter-domain networks in the testbed environments. But there are some limitations. TEQUILA *Monitoring system* depends heavily on the GAL functionality. Within the TEQUILA system, the scope of GAL extends beyond monitoring and it has interactions with several other components. In TEQUILA, the GAL is required to communicate with Cisco/Linux routers. Within the scope of the TEQUILA *Monitoring system*, the GAL provides two functions. The first is to provide access to router statistics for passive monitoring. The second is to install traffic classifiers in the network elements to direct active monitoring synthetic traffic to the required LSP / PHB. Without the GAL it is not possible to perform passive monitoring, because requests for router statistics go via the GAL. Active monitoring of OWD and OWL is possible for an IP route, for a single class of traffic (i.e. DSCP = 0). Monitoring of OWD and OWL for a LSP, or for a PHB is not possible using the current Monitoring Agents, because this requires the installation of a traffic classifier in the network element to mark synthetic packets with the appropriate DSCP. In the TEQUILA system this is performed by the GAL. Performing OWD monitoring for an LSP using the hop-by-hop approach relies on both knowledge of the path of the LSP and the ability to install traffic classifiers to set the synthetic traffic DSCP to direct traffic to the required PHB, hence this is not possible without the GAL. To support this functionality (i.e. monitoring OWD and OWL for BE traffic over an IP route) a subset of the information stored in the Network Repository is required.

5.5.2.7 TEQUILA Generic Adaptation Layer (GAL)

It is not yet clear that we are going to use GAL like component in MESCAL.

5.5.2.7.1 Overview

The Generic Adaptation Layer (GAL) component has been developed to enable the validation of the algorithms and protocols in physical testbed prototypes. It decouples the TEQUILA algorithms from the specific implementation of the data plane functionality of each underlying router type. The GAL mediates between the TEQUILA system components and the routers and provides an abstract interface exporting the IP DiffServ and MPLS capabilities of the routers in a generic way. Drivers have been developed for Cisco and Linux-based routers, which have been validated in corresponding testbeds. The supported data plane functionality spans the areas of traffic conditioning, MPLS routing and forwarding, PHB enforcement.

The interface exported is designed to emulate the COPS-PR specifications. The TEQUILA system acts as a COPS-PR PDP, downloading configuration information to and requesting monitoring information from the GAL. The GAL has the role of COPS-PR PEP, reporting on the success of the downloaded configuration and providing the required statistics. The syntax of the GAL Decision and Report messages for the supported abstractions (objects) is described in an XML schema. GAL implements a CORBA interface receiving Decision and returning Report XML documents.

The Cisco driver interacts with the router using IOS 12.2 CLI commands via a telnet session. Traffic classification and policing in the input interfaces are implemented via Access Lists and Rate Limits native mechanisms. The Class Based Weighted Fair Queuing scheduling mechanism is used at the output interfaces, within which the Low Latency Queuing scheduling mechanism implements priority queues (EF PHB). LSPs are configured as MPLS traffic engineering tunnels routed through explicit paths. For forwarding, flows were mapped to LSPs originally using the Policy-Based Routing mechanism. However, during functional validation tests, it has been proven that this mechanism did not work in combination with setting the MPLS EXP field in forwarded flows. The case has been reported to Cisco, which confirmed the bug and raised CSCdx84421 for this problem. The GAL Cisco

driver was modified to work with static IP routes at the expense of some flexibility in flow identification as only the packet destination IP address is taken into account.

The Linux kernel supports through Traffic Control (TC) a sophisticated system for classifying, sharing, prioritising and limiting incoming and outgoing traffic. The problem with the native Linux functionality with respect to Traffic Control is that multiple Linux kernel related mechanisms must be used to install, remove and verify the TC state in the kernel. In order to facilitate the development of the Linux GAL driver, a TC API was developed to support the ingress traffic conditioning and PHB configuration part of the GAL functionality. In contrast with TC and PHB configuration, Linux does not offer MPLS support natively and support had to be implemented. The MPLS routing part of the Linux GAL driver uses a slightly adapted and embedded version of the DSMPLS open source project to establish and map traffic onto LSPs. Finally the Linux driver also served as a basis for the IFT GAL driver.

5.5.2.7.2 Applicability to MESCAL

A Generic Adaptation Layer component may need to be employed by MESCAL. As in TEQUILA, it will enable the validation of the algorithms and protocols prototypes in testbeds. It will decouple the MESCAL algorithms from the specific implementation of the data plane functionality of each underlying router type. The TEQUILA GAL must be enhanced accordingly to support the configuration and monitoring requirements of the MESCAL components. In addition, the GAL drivers should be enriched to support additional router capabilities (e.g. BGP).

5.5.2.8 TEQUILA Traffic Generation and Emulation Tools

5.5.2.8.1 Overview

Traffic generation and emulation tools have been developed by TEQUILA to enable controlled and extensive testing of the developed traffic engineering and service admission control prototypes. These test tools are:

- The Network Topology Generator tool takes as input the number of network edge and core routers and generates fully meshed physical topologies, which can automatically be installed in the TEQUILA system.
- The Bulk Service Request Generator generates service subscription requests. It is configured with the number of SLSs per service type and service type specific parameters, e.g. throughput, QoS class, for which subscription requests are to be generated. Given a number of customers already registered in the TEQUILA system, the Bulk Service Request Generator assigns generated subscriptions to customers using a uniform distribution. An alternative version of the Bulk Service Request Generator tool can directly generate the traffic demand matrix for ingress, egress QoS class tuples.
- The Traffic Emulation tool generates service invocation requests for explicitly invoked established subscriptions and subsequently offered load events for all potentially active traffic sources, either explicitly or implicitly activated. Invocation requests are generated based on pre-configured stochastic models regarding inter-arrival and holding times per invocation, per user and per service type. Offered load events are generated for the duration of an invocation following the ON-OFF traffic source model using a pre-configured distribution, average ON and OFF times, per service type. The peak throughput transmitted during the ON periods is the contractual peak rate per established subscription.
- The Network Emulation tool emulates network performance given the load offered to the network and a set of paths for distributing QoS traffic across the network. The tool operates at discrete time-units. At each time-unit, offered load events per active invocation are matched to the traffic conditioning and routing settings and the admitted load (to actually enter the network) is calculated, according to the logic of the employed service admission control algorithm. Based on PHB configuration settings at the core network, the edge-to-edge performance per OA is assessed

at each time-unit based on load-related performance settings per PHB. Appropriate events reflecting network state for QoS delivery per OA are generated and fed to the service admission control instances.

5.5.2.8.2 Applicability to Mescal

The above tools were proven very helpful for testing the TEQUILA system in terms of scalability, stability and cost-benefit. With the appropriate enhancements the tools, in particular the network emulation tool, can assist in the testing and validation of the MESCAL system, especially of its intra-domain aspects.

5.5.2.9 Policy Management Protocol Implementations

5.5.2.9.1 PONDER

Ponder is a language for specifying management and security policies for distributed systems.

Separating policies from the managers that interpret them allows the behaviour and strategy of the management system to be changed without re-coding the managers. The management system can then adapt to changing requirements by disabling policies or replacing old policies with new ones without shutting down the system. The Ponder Language Policy Specification provides administrators with the necessary support for specifying and compiling Ponder policies.

As part of the Ponder development effort, a complete toolkit has been developed to support the users of this language. Available components include:

- **Ponder Compiler (v0.2.1)** The Ponder Compiler consists of a Syntax Analyser, a two-pass Semantic Analyser and a default Java Code Generator for Policies. The Compiler Framework is generic enough to allow the addition of more Code Generators dynamically, without modifying the existing Compiler code. The Compiler takes a Ponder language specification and generates Intermediate Code, which contains the various Policy objects of the specification according to the interfaces described by the Ponder API. The Ponder Compiler is implemented in Java using the SableCC compiler tool.
- **Ponder Policy Editor (v1.0.1)** A customisable text editor for the Ponder language written in Java. It has all the basic features of a text editor and additionally includes features that facilitate the editing of Ponder Policies. These additional features include syntax highlighting, policy templates, library events and library constraint templates. The aforementioned Ponder Compiler is integrated with the Ponder Policy Editor and can be invoked and customised from within the Editor.
- **Ponder Management Toolkit (v1.0)** A Management Toolkit Framework designed to facilitate the management of all Ponder tools from a single management console. The current release includes the main Console of the toolkit, the Policy Editor (with the Ponder Compiler) and a Configuration Management tool that allows the specification of the various parameters, which can then be shared by all the tools in the system. The current implementation of the domain storage uses an LDAP server. The framework allows easy addition of new tools (a tool can be added by implementing a specific interface).

5.5.2.9.2 LDAP

LDAP is a standard, extensible directory access protocol that enables directory clients and servers to interact using a common language. LDAP, as the name suggests, is a lightweight implementation of the X.500 Directory Access Protocol (DAP), first published in 1990. The X.500 protocol grew out of a need for a directory model that bridged applications and operating systems. However, it proved cumbersome, partly because it runs over the OSI networking stack. LDAP by contrast runs directly over TCP/IP, which is popular, fast, simple and relatively inexpensive to implement.

Oracle Internet Directory is Oracle's directory service compliant with LDAP version 3. It runs as an application on the Oracle9i database, which may or may not reside on the same operating system. To communicate with the database, Oracle Internet Directory uses Oracle Net Services, remote data-access software that enables client-to-server and server-to-server communication across any network. The information flow is presented at (Figure 12). Oracle Internet Directory's scalability, high availability and security features make it the directory of choice for enterprise applications.

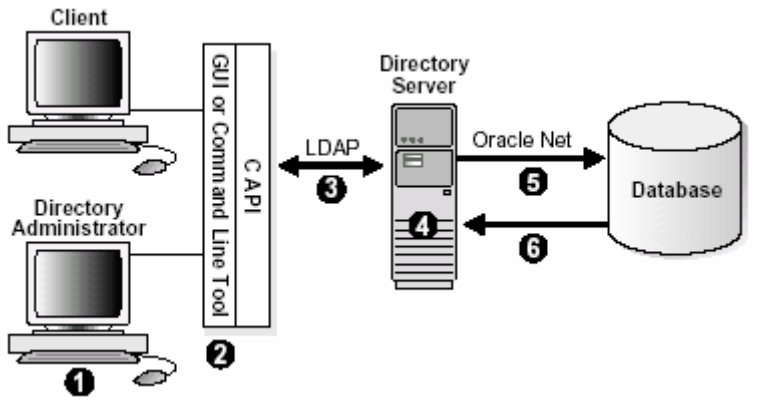

Figure 12 Information flow for Oracle Internet Directory

LDAP might be used in MESCAL as a protocol for accessing the policy repository. The Oracle Internet Directory provides an implementation of LDAP v3 plus the Oracle DB that can be used as a policy repository (for storing the policies).

5.5.2.9.3 SNMP

The Simple Network Management Protocol (SNMP) is the most well known protocol for Internet network management services.

One of the most widely spread implementations of SNMP was written at the University of California at Davis. It includes various tools relating to the SNMP: an extensible agent, an SNMP Library, tools to request or set information from SNMP agents, tools to generate and handle SNMP traps, a version of the Unix netstat command using SNMP and a graphical Perl/Tk/SNMP based MIB browser. The package is freely available and supports all versions of SNMP (SNMPv1, SNMPv2, SNMPv3) and it can run on most Unix environments, *BSD, Linux, IRIX, MS-Windows [SNMPa].

Another commercially available SNMP software is the one from AdventNet providing Web- and Java-based SNMP protocol stack and network management tools [SNMPb].

A list of SNMP related software is available in [SNMPc].

5.5.2.9.4 RADIUS

The RADIUS (Remote Authentication Dial-In User Service) is a protocol for carrying authentication, authorisation and configuration information between a Network Access Server, which desires to authenticate its links, and a shared Authentication Server.

RADIUS has been deployed since 1994 and there are many available implementations. One freely available implementation is by Cistron, which has become widely used in the free software community -written by Miquel van Smoorenburg (miquels@cistron.nl) from the original Livingston source [RADIUS].

5.5.2.9.5 DIAMETER

The DIAMETER protocol is a proposed protocol which can be used for policy, AAA (Authentication, Authorisation and Accounting) and resource control. The protocol has been set up to coexist with RADIUS. DIAMETER is currently under development. There is a homepage, which contains the

latest information on DIAMETER at [DIAMETER]. The Open Diameter project in sourceforge provides an initial version of DIAMETER implementation code [DIAMETER2].

5.5.2.9.6 Applicability to MESCAL

Following the current state-of-art, principles of the policy framework may be used in the architecture and design of the MESCAL system; mainly for increasing the flexibility of the design. Policy based interactions may be considered not only at the level of customer to provider or provider to provider interactions but also at the level of network functions. As such, the protocol implementations described above are candidates for being used for realising policy-based interactions between components of the MESCAL system and/or storing policy information. The protocol to be used in MESCAL can only be decided after the specifications of the functional architecture, which will pose requirements on the features that need to be supported by the protocol.

5.5.3 Summary of Applicability to MESCAL

The following table summarises the applicability of the components presented to MESCAL.

Objects	Functional Suitability (1-3)*	Availability (1-3)*	Migration			
			Documentation (1-3)*	Platform (1-3)*	Technology (1-3)*	Modifications /Support (1-3)*
TEQUILA SrNP	2	3	2	3	3	3
TEQUILA SSM	2	3	2	3	3	3
TEQUILA ND	3	3	2	3	3	3
TEQUILA RSIM	2	3	2	3	3	2
TEQUILA DRtM	2	3	2	3	3	2
TEQUILA DRsM	2	3	2	3	3	2
TEQUILA Monitoring	2	3	2	3	3	3
TEQUILA GAL	1	2	2	3	3	1
TEQUILA Traffic Generation and Emulation tools	2	3	2	3	3	2
Policy Management Protocol Implementations	2	3	2	2	2	0

(*) 0: low, 1: average, 2: good, 3: excellent

Table 10 Applicability of background objects with MESCAL

As already outlined, the purpose of surveying various background components was to check upon tools that may facilitate the design and implementation of the MESCAL system. There is no scope in selecting one of more of the presented background objects, as the project implementation plans do not depend on them.

Based on well-defined selection criteria, the applicability of the presented background components was rationalised. As it can be seen from Table 10, the considered background objects present viable options, to a lesser or greater degree, to facilitate project design and implementation activities. In particular,

- The TEQUILA traffic engineering (ND, DRtM, DRsM) and GAL components may be used, with appropriate enhancements, to cover the intra-domain traffic engineering, while the underlying concepts and notions and the gained know-how could be utilised in the design of the MESCAL inter-domain traffic engineering solutions

- The concepts, notions of and experience gained from the TEQUILA service subscription and invocation management components may be utilised in building the MESCAL inter-domain service architecture; the use of the corresponding TEQUILA components may require significant adaptations.

Finally, it should be noted that the project will continuously review existing components that may be used to facilitate system design and realisation, according to relevant state-of-art developments and the on-going specifications of the MESCAL functional model.

6 REFERENCES

[Barabasi] Albert-Laszlo Barabasi and Reka Albert, “Emergence of Scaling in Random Networks”, Science pp 509-512, October 1999

[BRITE] http://www.cs.bu.edu/BRITE/

[Calvert] Calvert, K., Doar, M., and Zegura, E. "Modelling Internet topology". IEEE Communications Magazine (June 1997).

[D0.2] MESCAL Deliverable D0.2, "Dissemination and use plan"

[D1.1] MESCAL Deliverable D1.1, "Specification of business models and a functional architecture for inter-domain QoS delivery"

[D1.2] MESCAL Deliverable D1.2, "Initial specification of protocols and algorithms for inter-domain SLS management and traffic engineering for QoS-based IP service delivery and their test requirements"

[D1.3] MESCAL Deliverable D1.3, "Final specification of protocols and algorithms for inter-domain SLS management and traffic engineering for QoS-based IP service delivery"

[D1.4] MESCAL Deliverable D1.4, “Issues in MESCAL Inter-Domain QoS Delivery: Technologies, Bi-directionality, Inter-operability, and Financial Settlements”

[DIAMETER] http://www.diameter.org

[DIAMETER2] http://sourceforge.net/projects/diameter/

[Ellen] Ellen W. Zegura, Kenneth L. Calvert and Michael J. Donahoo. "A Quantitative Comparison of Graph-Based Models for Inter Topology". IEEE/ACM Transaction on Networking Vol 5. No 6. December 1997.

[Ellen2] Ellen W. Zegura, Kenneth L. Calvert and Samrat Bhattacharjee. "How to Model an Internetwork", Proceedings of IEEE INFOCOM, April 1996.

[Enns-Netconf 03] R. Enns, Editor “NETCONF Configuration Protocol”, Internet-Draft <draft-ietf-netconf-prot-01>, Expires April 2004.

[Enns-xml 03] R. Enns, “XMLCONF Configuration Protocol” Internet draft <draft-enns-xmlconf-spec-00.txt>, Feb. 2003.

[ERLANG] http://www.erlang-software.com/

[Eval02] Evaluation of J-Sim, June 2002. Available at: www.j-sim.org/comparison.html

[EX-COMM] Internet Draft “Controlling the redistribution of BGP routes”, draft-bonaventure-bgp-redistribution-02.txt

[Faloutsos] M. Faloutsos, P. Faloutsos, C. Faloutsos, "On Power-Law Relationships of the Internet Topology", In proceedings of SIGCOMM: IEEE Conference on Communication, 1999

[GateD] http://www.gated.org

[gtitm] http://www.cc.gatech.edu/projects/gtitm/

[Hazewinkel 03] H. Hazewinkel ,D. Partain, “The Differentiated Services Configuration MIB”, Internet draft <draft-ietf-snmpconf-diffpolicy-08.txt>, Expires April 2004.

[inet] http://topology.eecs.umich.edu/inet/

[J-SIM] http://www.j-sim.org

[MAISIE] http://may.cs.ucla.edu/projects/maisie/

[MATH] http://www.mathworks.com/

[Medina] Alberto Medina, Ibrahim Matta and John Byers. "On the Origins of Power Laws in Internet Topologies". ACM SIGCOMM Computer Communications Review, Vol. 30, No. 2, April 2000

[Minoli] "ISP Business Relationships", Chapter 3 of Daniel Minoli, Andrew Schmidt, "Internet Architectures", Wiley Press 1999, ISBN 0-471-19081-0

[Mitzen] Michael Mitzenmacher, "A Brief History of Generative Models for Power Law and Lognormal Distributions" 39th Annual Allerton Conference on Communication, Control, and Computing, 2001

[MRT] www.merit.edu/~mrt

[Nicol02] D. M. Nicol, Simulation performance analysis, April 2002. Available at: www.ssfnet.org/Exchange/gallery/dumbbell/index.html

[NLANR] http://moat.nlanr.net/AS/Data/

[NS] www.isi.edu/nsnam/ns

[OMNET] http://whale.hit.bme.hu/omnetpp/

[OPNET] http://www.opnet.com/

[Palmer] Christopher R. Palmer and J. Gregory Steffan, "Generating Network Topologies That Obey Power Laws". Proceedings of the Global Internet Symposium, Globecom 2000, November 2000, San Francisco.

[pch] http://www.pch.net/documents/data/routing-tables/

[QRS] www.tct.hut.fi/tutkimus/ironet/QRS/index.html

[QUALNET] www.qualnet.com

[Quotin02] B. Quotin, S. Uhlig and O. Bonaventure, "Using redistribution communities for Interdomain Traffic Engineering," QoFIS'02, Zurich, Switzerland, October 2002.

[RADIUS] http://www.cistron.nl/~miquels/radius/

[REAL] http://minnie.tuhs.org/REAL/

[Rekhter98] Y. Rekhter and T. Li, "A border gateway protocol 4," Internet-Draft (RFC1771), February 1998

[RFC3512] M. MacFaden, D. Partain, J. Saperia, W. Tackabury, "Configuring Networks and Devices with Simple Network Management Protocol (SNMP)" Informational RFC 3512, April 2003.

[routeviews] http://archive.routeviews.org/

[SNMPa] http://net-snmp.sourceforge.net/

[SNMPb] http://www.adventnet.com

[SNMPc] http://wwwsnmp.cs.utwente.nl/software/

[SSFNET] www.ssfnet.org

[TA] MESCAL Technical Annex

[Tang] "Network Topology Generators: Degree-Based vs Structural" Hongsuda Tangmunarunkit et al. SIGCOMM: IEEE Conference on Communication, 2001

[Waldbusser 03] Steve Waldbusser, Jon Saperia, Thippanna Hongal, "Policy Based Management MIB", Internet draft <draft-ietf-snmpconf-pm-14.txt>, September 2003.

[Waxman] B.M.Waxman. "Routing of Mulitpoint Connections". IEEE Journal on Selected Areas in communications, 6(9):1617-1622, 1988

[Zebra] www.zebra.org

[Zhang03] Raymond Zhang, JP Vasseur, “MPLS Inter-AS Traffic Engineering requirements”, draft-zhang-mpls-interas-te-req-01.txt, Expires: July, 2003