NTNU

Norwegian University of Science and Technology

Performance study of the 3LIHON output scheduling part

Gaia Leli

Master of Telematics - Communication Networks and Networked Services (2 year)

Submission date: February 2012

Supervisor: Norvald Stol, ITEM

Co-supervisor: Michele Savi, ITEM

Carla Raffaelli, DEIS - Università di Bologna

Norwegian University of Science and Technology
Department of Telematics

Problem Description

A new switch architecture concept for the future optical core network has been proposed called the "3-Level Integrated Hybrid Optical Network" (3LIHON) [3]. Within this architecture concept some actual realization alternatives are possible. Challenges remain with regard to evaluation of both the general concept and the actual realizations with respect to performance, dependability, cost and energy consumption/environmental impact. Comparisons should be done, both between alternative realizations of the 3LIHON concept, and with other suggested architectures for the future optical network.

In this thesis the focus will be on examining the wavelength utilization and performance for statistically multiplexed traffic (with a given scheduling algorithm) towards the wavelengths on the output side of a 3LIHON switching node. Different input traffic distributions and load distributions across the three different transport classes of 3LIHON will be examined. More specifically we want to observe loss for packets belonging to the Statistically Multiplexed Real Time (SM/RT) transport class, loss and delay for packets belonging to the Statistically Multiplexed Best Effort (SM/BE) transport class, and the utilization of each wavelength.

Initially we will assume:

- no FDL delay available in the Optical Packet Switch (OPS) part of the architecture, i.e. for SM/RT packets;
- a single buffer with FIFO priority order and no (link level) retransmissions from the Electronic Packet Switch (EPS) part of the architecture, i.e. for SM/BE packets;

If time allows the study may be extended to include some of the following:

- use N shared FDL delay line buffers available in the OPS (i.e. for SM/RT packets) for a given output fiber (with N small);
- allow link level retransmission of interrupted packets from the EPS (i.e. examine the trade-off between reducing loss but increasing delay for the SM/BE transport class);

The Master thesis is formally started September 1st, 2011 at NTNU and should be finished by March 7th, 2012 (6 months master thesis period for exchange students plus one week for the Christmas's holiday).

Supervisor (NTNU):	Associate professor Norvald Stol
Supervisor (UniBo):	Associate professor Carla Raffaelli
Co-supervisor (NTNU):	Post doctor Michele Savi
Assignment given:	September 1st, 2011

Acknowledgments

It is a great pleasure to thank my supervisor, Carla Raffaelli, for having offered me the opportunity of coming to NTNU and working on an exciting research project that after nine enjoyable months finally culminated in this thesis.

I am heartily thankful to my co-supervisor, Norvald Stol, whose encouragement, guidance, support and feedbacks from the initial to the final level enabled me to develop an understanding of the subject.

I am especially grateful to Michele Savi for his support, encouragement and the many conversations.

I would like to extend my gratitude to all the numerous people who contribute in making the NTNU such a lively and friendly place. I am particularly indebted to Mona Nordaune and Randi Schrøder Flønes for their administrative assistance and for having been so kind and helpful since my first day.

It would not have been possible to write this thesis without the help and support of the kind people around me. It is impossible to give a particular mention here so... I would like to thank my friends from Italy, Korea, Lithuania, France, Norway, Germany, Spain, Catalunya, Bulgaria and Belgium for their support and encouragement throughout (any many unforgettable nights!).

The last thank you goes to my parents, for allowing me to go and make my own path.

Abstract

In the last years hybrid optical networking is a topic of increasing interest for graceful migration to future high capacity integrated service networks. A new hybrid network architecture is proposed to harmonize different transport technologies and to support a suitable set of services: "3-Level Integrated Hybrid Optical Network" (3LIHON) [3].

The aim of this thesis is to study the performance of 3LIHON focusing on examining the Quality of Service (QoS) in the output part of the node. In particular we study the performance for Statistically Multiplexed (SM) traffic.

In Chapter 1 we present the motivation of our study and the current work. We give the problem definition and define the goal of the thesis.

Chapter 2 shows concepts and architecture of 3LIHON. Firstly we introduce the reference classes used and the Quality of Service (QoS) requirement. Furthermore we give a complete description of 3LIHON architecture[3] describing transport services, architecture in detail, input and output part of the node. Finally we describe the advantages of 3LIHON network.

To simulate the 3LIHON architecture we use a programming language called Simula and a context class for discrete event simulation called DEMOS.

In Chapter 3 firstly we describe the simulation model implemented, moreover we give a code description. We show the sources characterization and the packets characterization for all type of traffic that 3LIHON is able to handle: Guaranteed Service Transport (GST) traffic, Statistically Multiplexed (SM) Real Time (RT) traffic and Statistically Multiplexed (SM) Best Effort (BE) traffic. The code used in this work is available in Appendix C.

In Chapter 4 firstly we present the simulation scenario and then the obtained results. To evaluate the accuracy's level of our results we use a 95% confidence interval and more theoretical details about that are given in Appendix A. We consider three study cases and for each of them we analyze in details the Packet Loss Probability (PLP) of Statistically Multiplexed Real Time (SM/RT) packets, the Packet Loss Probability (PLP) of Statistically Multiplexed Best Effort (SM/BE) packets and the delay of Statistically Multiplexed Best Effort (SM/BE) packets in the Best Effort queue. Some additional results used to obtain the study case called Series Two in Chapter 4.4 are shown in Appendix B.

In Chapter 5 are presented some conclusions of this work and in Chapter 6 we show some hints that can be the sparkle for furher works.

List of Abbreviations

3LIHON	3-Level Integrated Hybrid Optical Network
OpMiGua	Optical Migration Capable Nerwork with service Guarantees
GST	Guaranteed Service Transport
SM/RT	Statistically Multiplexed Real Time
SM/BE	Statistically Multiplexed Best Effort
FDL	Fiber Delay Line
OXC	Optical Cross Connection
OPS	Optical Packet Switch
EPS	Electronic Packet Switch
FIFO	First In First Out
WRON	Wavelength Routed Optical Network
QoS	Quality of Service
HQ	High Quality
LQ	Low Quality
FEC	Forward Error Connection
DPT	Detect Packet Type
OC	Optical Code
E/D	Encoder/Decoder
SOA	Semiconductor Optical Amplifiers
LiNbO3	Lithium Niobate
ITU-T	International Telecommunication Union Telecommunication standard sector
HQ	High Quality
LQ	Low Quality
CA	Collision Avoidance
CR	Contention Resolution

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation and Current Work

Hybrid optical networking is a topic of increasing interest for graceful migration to future high capacity integrated service networks.
Future networks require high flexibility and re-configuration capability to handle the wide variety of services that they will bring into the scene. The aim is to guarantee high speed interconnection between users and to support a wide range of services and applications.

One of the earliest hybrid network architecture proposed for this purpose is called "Optical Migration Capable Nerwork with service Guarantees" (OpMiGua) [1]. This is a model of Optical Packet Switched (OPS) hybrid network that supports high throughput efficiency and Guaranteed Service Transport (GST) with no packet loss and constant delay.
OpMiGua divides the traffic into two service classes and uses the capacity of the same wavelength in a Wavelength Routed Optical Network (WRON).

The traffic is divided into:

- Guaranteed Service Transport (GST) : service class for the circuit-switched traffic;
- Statistically Mutiplexed Best Effort (SM/BE) : service class for

the Best Effort packed switched traffic;

Beginning from the OpMiGua architecture and from its extensions [2] it is proposed a new hybrid optical network called ”3-Level Integrated Hybrid Optical Network” (3LIHON) [1] that it is studied to harmonize different transport technologies and to handle future networks.

The new architecture introduces a third service level:

- Statistically Mutiplexed Real Time (SM/RT);

with the purpose to support strict real time services.

The advantages that 3LIHON is able to introduce are mainly three:

- increase the Quality of Service (QoS) for short packets with high realtime demands;
- better bandwidth utilization;
- cheaper switch architecture;

Previously results from 3LIHON architecture are presented in [1]. They consider a mix of GST and SM/RT traffic and leave for further works the chance to implement the complete set of service classes.

The aim of this work is to implement a simulation model of full 3LIHON architecture and to study the performance for statistically multiplexed traffic towards the wavelengths on the output side of a 3LIHON switching node.

1.2 Problem Definition and Goal

1.2.1 Problem Definition

We study the behavior of the 3LIHON architecture using different input traffic distributions and different load distributions per each type of transport class.

The focus is on the loss for packets belonging to the Statistically Multiplexed (SM) transport classes.
More specifically we want to observe loss for packets belonging to the Statistically Multiplexed Real Time (SM/RT) transport class and loss and delay for packets belonging to the Statistically Multiplexed Best Effort (SM/BE).

In the previous work, presented in [1], to evaluate the loss for the SM/RT traffic the attention is focused on a single output link with M wavelengths. In this work we evaluate the loss for the SM/RT and for the SM/BE traffic focused again on a single output link with M wavelengths.

The model studied in the previous work [1] is discrete-time with multi-slot packet generation but in this thesis we use an asynchronous and un-slotted model.

1.2.2 Goal

The goal of this work is to study the performance of a complete 3LIHON architecture, composed by GST, SM/RT and SM/BE services, with regard to contention at the output side of the node.
More specifically we want to observe loss for packets belonging to the Statistically Multiplexed Real Time (SM/RT) transport class, loss and delay for packets belonging to the Statistically Multiplexed Best Effort (SM/BE) transport class and the utilization of each wavelength.

1.3 Outline

The outline of this work is as follows: Chapter 2 presents 3LIHON architecture and its characteristics. Chapter 3.1 introduces the simulation model adopted. Chapter 3.2 describe in detail the simulation model implemented. Chapter 4.1 introduces the simulation scenario. Chapter 4.2 presents results for different series study case. Chapter 5 gives some conclusions of this work. Finally, Chapter 6 presents further works.

Chapter 2

3LIHON: Concepts and Architecture

The "3-Level Integrated Hybrid Optical Network" (3LIHON) is a new hybrid network architecture developed as an extension of OpMiGua [1].
It is introduced to handle a future network and to support a wide range of services that require higher level of Quality of Service (QoS).

2.1 Reference classes and QoS requirements

Classify future and still unknown services is impossible but we can introduce seven general classes and hope that they will be able to match with the future services.

A generic but at the same time exhaustive classification is the follow one:

1. **Video Streaming.** Examples of this type of services are broadcasting television and digital television signals between studios. They require high bandwidth demands, especially when they are High Quality (HQ). Video Streaming services treat one-way transport of vision and sound for semi-professional and professional use. They are characterize by low jitters to avoid large

buffers at receivers. Could be delay sensitive and have very high loss sensitivity depending on use. In the high demand cases it is maybe necessary to introduce Forward Error Correction (FEC).

2. **Video Conversational.** They are two-way transport of vision and sound to handle the conversation between humans. They require high bandwidth demands, like the video streaming class, and high real time demands. They are less loss sensitive then the previous type. When the HQ video is required for professional use, like medicine use, it is better to use parallel video streaming.
3. **Music Streaming.** The demand of bandwidth in this case is limited. It is a one-way transport that require low jitters to do not have large buffers at receivers; in this case they can be mobile. The delay sensitive is not so strict but in any case change radio station can not take too long. In the HQ music a little noise is acceptable and then it is possible to have high loss sensitivity.
4. **Voice Conversational.** Voice conversational services consist in two-way transport of voice to facilitate the conversation between two humans. The main characterize of this type of service is the high real time demand. The bandwidth demand and the loss sensibility are both low.
5. **Interactive Messaging.** This service involves humans, sensors and/or "machine" and an example is transport of information from a critical supervisory system. The demand of bandwidth and the loss sensitivity are low. The real time demand is in general high but for many uses of this service it can be relaxed.
6. **Control Traffic.** Control traffic examples are messages to handle routing/segnaling information in a network or to exchange status/failure. The bandwidth request is low but usually it has both high loss sensitivity and high real time demands. This behavior is important when the system is for example carrying a signaling of fault in a network.
7. **General Data Transfer.** Examples of this typer of services are program updates, database use and back-up, Low Quality (LQ)

video streaming or LQ music streaming for not professional use. The bandwidth demands is variable in this case. We can say that in general it has very high loss sensibility but with low real time demands which allow retransmissions.

It is possible to introduce new services just combining the basic classes presented above.
For example the "on-line gaming" can be easy added like combination of General Data Transfer and Interactive Messaging. The first has high bandwidth but low real time demand required to establish a gaming environment. The second one provides low bandwidth and high real time demand that are suitable with the necessity to keep the gaming environment synchronized during the current gameplay.

Every basic class has a different QoS requirement and it is indicated in the following table, according with the ITU-T Recommendation Y.1541 [5] :

Table 2.1: QoS targets for reference services classes [5]

Service Class	Y.1541 QoS Class	Upper Bound Packet Loss Rate	Upper Bound Delay	Upper Bound Jitters	Bandwidth need
1.Video Streaming	6 or 7	10^{-5}	100 ms or 400 ms	50 ms	High
2. Video Conversational	0 or 1	10^{-3}	100 ms or 400 ms	50 ms	High
3.Music Streaming	6 or 7	10^{-5}	100 ms or 400 ms	50 ms	Low to Medium
4.Voice Conversational	0 or 1	10^{-3}	100 ms or 400 ms	50 ms	Low
5.Interactive Messaging	3 or 2	10^{-3}	100 ms or 400 ms	undef.	Low
6.Control Traffic	2	10^{-3}	100 ms	undef.	Low
7.General Data Transfer	4 or 5	10^{-3}	1 s or undef.	undef.	Low to High

2.2 3LIHON architecture

2.2.1 Transport Services Description

3LIHON handles three different transport services:

- Guaranteed Service Type (GST): class of transport resembling an optical circuit switched service without information loss into the network;

- Statistically Multiplexed Real Time (SM/RT): class of transport like a packet switched service with possibility of loss inside the node and contest for bandwidth;

- Statistically Multiplexed Best Effort (SM/BE): class of transport like a packet switched service with small overall packet loss but no guaranteed delay inside the nodes;

All this transport services are carried by the same set of shared wavelengths.

The first one is the same studied in OpMiGua architecture [1] and the second service is split in two.
We introduce a new class service to support the transport of real time services. This kind of services indeed are defined by short packets with low bandwidth demand but high real time request. This is unsuitable with both the class services presented in OpMiGua.
The GST transport service introduces an undesired delay in the input nodes and the SM/BE transport service may bring delay and potentially loss.

2.2.2 Architecture Description

Figure 2.1 represents 3LIHON architecture with N input fibers and N output fibers carrying M wavelengths each:

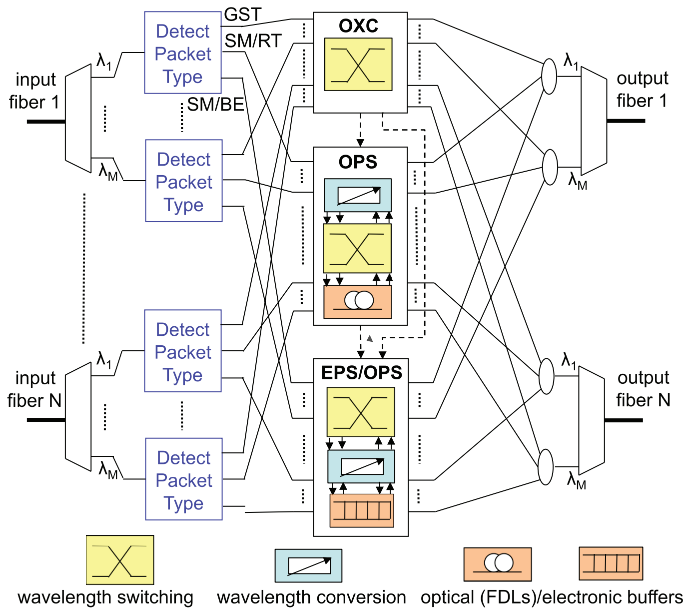

Figure 2.1: General 3LIHON scheme architecture with N input/output fibers carrying M wavelength [3]

Per each input wavelength is given a block able to identify which kind of traffic is coming, called Packet Detected Types (PDT). It uses optical coding techniques (proposed in [4]) to distinguish the different services.

After that the packet goes into the switching stage.

2.2.3 Input to Node Description

In the 3LIHON architecture the first input block is the Detect Packet Type (DPT).
A possible implementation of DPT using Optical Code (OC) is represented in Figure 2.2:

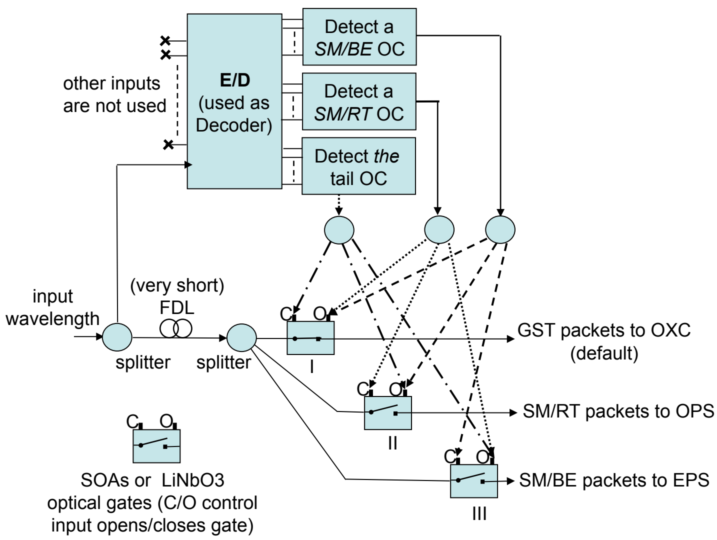

Figure 2.2: Detect Packet Type (DPT) subsystem implemented using OC detection [3]

These encoder/decoder are passive devices and for this reason they have the advantage to do not increase the power consumption of the architecture.
The default operation of this device is to send an unmarked traffic flow to the OXC.

In this way it is not necessary to mark the GST packets with OC. It is necessary for the SM/RT and SM/BE, in order to allow the DPT to send them in the right switching subsystem.

It is possible to split the OCs available in two group: one dedicated to the SM/RT and the other one to the SM/BE packets. Doing in this way it is possible to distinguish the different nature of the SM traffic directly into the optical domain and the priority of the coming traffic is defined as well.

The DPT receive packets with variable length and it is able to recognize that the received packet is finished because of the tail-OC, introduced ad hoc.

When an optical packet arrives at the input wavelength, an OC Decoder detects the OC corresponding to either a SM/RT or a SM/BE packet and consequently opens (O) and closes (C) the gates at the input to forward the packet to its correct switching subsystem.
The packet is then forwarded into the correct switching subsystem. When the DPT detects the tail OC, the system comes back to the default configuration. That means all unmarked packets are sent to OXC.

2.2.4 Core Node Description

Per each service presented the 3LIHON architecture introduces a different switching subsystem:

- Optical Cross Connection (OXC);
- Optical Packet Switching (OPS);
- Electronic Packet Switching/ Optical Packet Switching (EPS/OPS);

The GST packets are switched by using Optical Cross Connection (OXC). They traverse the OXC through optical circuits already established end-to-end through the core network, thus without any loss of packet. In this way the GST packet travels into the network following fixed path from the input node to the output node through pre-established wavelengths.

The OPS handles the SM/RT packets that have no buffering, except a limited Fiber Delay (FD). In this way the real time service is guaranteed.

The SM/BE packets are switched by Electronic Packet Switching (EPS).
We expect that the largest part of the traffic is GST packets or SM/BE packets. The first treats large volume traffic, like video services, the second one is able to handle packets smaller than the GST, like services characterize by limited real time demands.

In both cases we can use cheap switches already commercially available based on existing technology. On the other hand, to realize the switching of SM/RT traffic we have to introduce a new and more expensive system: Optical Packet Switching (OPS). But the volume of the SM/RT traffic is very limited if compared with the other two classes. For this reason also the OPSs are less requiring in terms of expensive optical hardware and it is possible to introduce concentrators, as proposed in [8]. Even if 3LIHON needs further switching subsystem, the overall cost of the architecture should be lower than OpMiGua.

2.2.5 Output from Node Description

In the output from node we have three different types of traffic services that compete to use the same wavelengths on switch output links.

To handle this behavior we introduce those additional services:

- Collision Avoidance (CA): to handle the possible collision among different traffic types;
- Contention Resolution (CR): to treat the contention among packets of the same class;

We consider a generic wavelength j on a generic fiber k that we indicate like (j, k) and introduce a Detect signal, like showed in Figure 2.3:

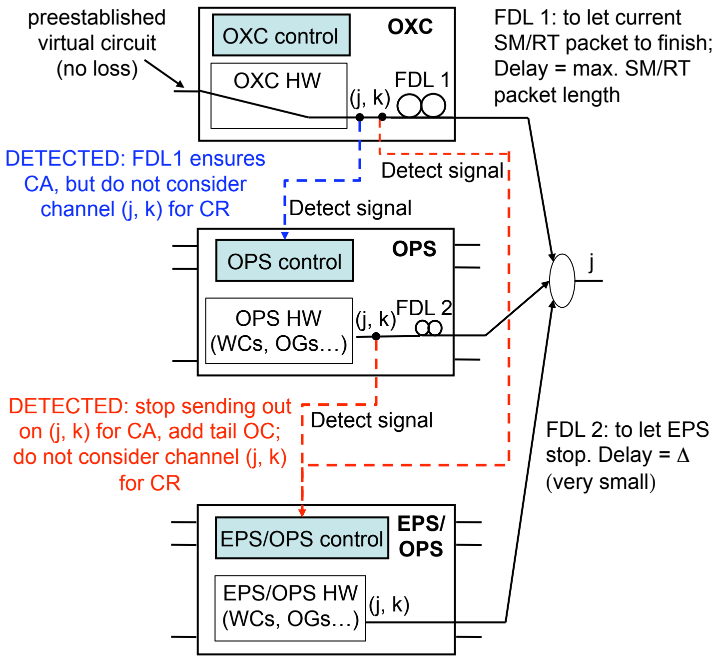

Figure 2.3: Collision Avoidance mechanism [3]

The GST packets are characterize by maximum priority and that means that they are forwarded from input to output without any loss in fixed and pre-established circuits.
Every time that a GST packet is detected, the output of the OXC sends a Detect signals to the OPS control and another Detected signal to the EPS control with the purpose to update their status.
The GST packet continues its path through a fixed Fiber Delay Line, indicated in the figure like FDL1.
The length of this delay has to be equal to the maximum SM/RT packet length to allow the potential SM/RT packet incoming to be delivered out on channel (j, k). The GST packets have higher priority level but they are not allow to interrupt any SM/RT packets that are already crossing the system. The result is there is no collision between GST and SM/RT packets. When a GST packet is coming the OPS considers the channel (j, k) not available to handle incoming packets. To solve the contention problems the OPS has to test the other wave-

lengths looking for one of them that are not carrying another GST packet.

To implement the Collision Avoidance (CA) on channel (j, k), when the OPS detects the GST is over, it does not send immediately new SM/RT packets. It has to wait for a time equal to the delay introduced by FDL1, to let the GST packet finish its transmission.
The maximum length of a SM/RT packet should be much shorter than a GST packet to do not increase much the length of FDL1. In this way the GST packets will have short delays in the OXCs.

The EPS/OPS must immediately stop to send out packets for CA reasons and it must not consider channel (j, k) for contention resolution for incoming packets.
Only a short FDL2 (delay Δ in Figure 2.3) is needed for the SM/RT packets to allow the EPS to stop its transmission.

Contention Resolution (CR) is simple to handle in our system since we receive just one packet at each time instant. Our simulation model is asynchronous, like illustrated in Chapter 3.1.1.

2.2.6 3LIHON Advantages

Comparing the 3LIHON with the OpMiGua architecture we can notice the following extra advantages:

- Better utilization of the bandwidth. The relative length of GST packet compared to the length of the SM/RT packet is larger than in the previous architecture.
- Benefit with GST packet delay. The GST packets have shorter constant delays after the switch.
- More flexibility to handle the GST packets. Since short real time demand packets are not carried by GST services the types of packets that are left will either fill up GST packets quickly (e.g. video with high real-time demands) or they are not real-time and can also accept a longer timeout value if short/few

packets are arriving to be merged into a GST packet. The result is that we will have less short GST packets into the network.

Chapter 3

Simulation Modeling and Code Description

3.1 Simulation Modeling

The focus of this work is on implementing a simulation model of full 3LIHON architecture and to study the performance for Statistically Multiplexed (SM) traffic towards the wavelengths on the output side of a 3LIHON switching node.

3.1.1 Model Description

The 3LIHON architecture showed in Figure 2.1 presents N input/output carrying $M = 32$ wavelengths.
In this work the Packet Loss Probability (PLP) is evaluated in a single output fiber.

A mix of GST, SM/RT and SM/BE traffic flow is considered. GST traffic travels through the network without loss, while the SM traffic is sent in gaps between GST packets. A given scheduling algorithm is proposed to handle the traffic in the network.

Negative exponential distributions are used to describe the arrival process for GST traffic and for Statistical Multiplexed (SM) traffic.

GST packets are collected in burst and then forwarded in the network by OXC.

The packet length distribution used to describe the length of the GST bursts is constant distribution. We use Poisson distributions to model the length of the Statistical Multiplexed (SM) traffic.

The simulation model that we present in this work is an asynchronous and un-slotted model.

We assume no FDL available in the Optical Packet Switching (OPS) for the SM/RT and one buffer with FIFO priority order and no retransmission for SM/BE.

3.1.2 Quality of Service

The purpose of this work is to study the Quality of Service (QoS) in the output nodes.
The focus of our work is on the follow parameters:

- Packet Loss Probability (PLP) for Statistically Multiplexed Real Time (SM/RT) service;
- Packet Loss Probability (PLP) for Statistically Multiplexed Best Effort (SM/BE) service;
- Delay of SM/BE packets in the BE queue;

3.2 Code Description

In this work we simulate the 3LIHON architecture using the programming language Simula and a context class for discrete event simulation called DEMOS.

Simula is a pure object-oriented programming language developed in Norway in the 60's for simulation purpose, like its name implies. It

represents the framework for many of the features of object-oriented languages.
Discrete Event Modelling On Simula (DEMOS) is a context class written by Graham Birtwistle that has many building blocks for discrete event simulators.

Our purpose is to implement the 3LIHON architecture with 32 output wavelengths ($M = 32$) and with a bit channel rate equal to 1 *G bit per sec*.

3.3 Sources Characterization

To handle the different types of services that 3LIHON is able to transmit we use different kind of sources.

Since the GST bursts travel into the network following pre-established path, we consider 32 GST sources: one per each output wavelength. In this way we realize a fixed link and assure the delivery of each GST burst.
To define the rate of these sources we use a negative exponential distributions, with mean value that is function of the length of GST and bit channel rate, to achieve a wanted load from GST traffic on each wavelength.

To generate the SM/RT traffic we consider a single source generates packets according with a negative exponential distribution. The mean value is function of relative percentage of SM/RT packet that we are considering, the bit channel rate, the length of the SM/RT packets and the number of output wavelengths considered.

To handle the SM/BE packets we use a similar system: we consider a single SM/BE source with negative exponential distribution. Now this value is function of relative percentage of SM/BE packet, bit channel rate, length of the SM/BE packets and number of output wavelengths considered.

3.4 Packets Characterization

We define a different length per each service to treat the collection of packets that the network is able to transmit.

The GST bursts are characterized by constant distribution with fixed mean value length.
In order to study the performances of the 3LIHON architecture we considered three values for the mean value of GST bursts:

- GST burst length 4000 *bytes*;
- GST burst length 40000 *bytes*;
- GST burst length 400000 *bytes*;

The SM/RT packets are characterized by a negative exponential distribution, with mean value equal to 40 *bytes*. This is a reasonable choice because compatible with the SM/RT QoS requirements illustrated in the Chapter 2.1.

Furthermore we use a negative exponential distribution to describe the SM/BE length, with mean value function of the SM/RT's packets length.

So far we consider a value equal to 40 times the SM/RT's packets length, that means 1600 *bytes*.
Then we test the simulator using two new values: 70 times (2800 *bytes*) and 100 times (4000 *bytes*) the length of SM/RT packets.
Nowadays we have not SM/BE traffic with this size but this is a possible require for the future networks.

For this study we estimate the FDL 5 times the SM/RT's packets length. This also means that SM/RT packets are truncated to this maximum length when they are generated.

3.4.1 Guaranteed Service Transport Traffic Description

We implement the GST traffic by Demos like illustrated in Figure 3.1:

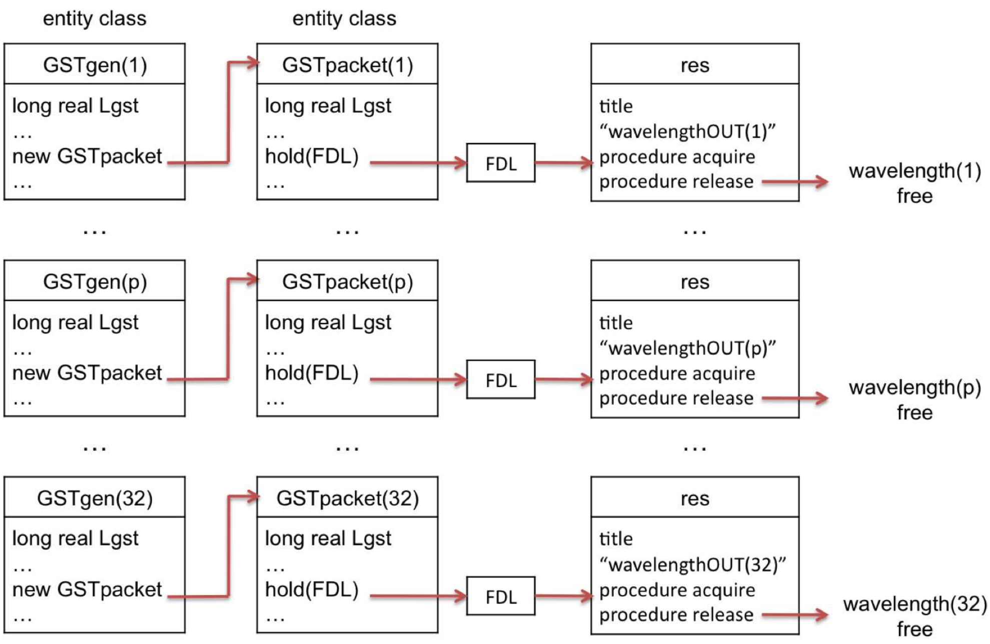

Figure 3.1: Demos implementation for GST traffic

To describe the behavior of the GST traffic we introduce two entity classes:

- *GST generator*

  We have 32 generators and every time that one of the them generates a packet the correspondent wavelength is busy until the end of the transmission.
  Packets generate from this type of sources can not be interrupted and this guarantees the higher priority level to GST traffic.

- *GST burst*

  There is a direct link between the GST source and the corresponding output wavelength. Every time that a GST burst is been generated we mark the correspondent FDL with flag equal to 1, to remember that it is used to carry a packet with higher priority level and that it can not be used.

  We introduce a boolean variable that we set true to remind that GST burst is crossing the FDL. Hence we keep track of the correspondent wavelength that is still busy and we guarantee that generated packet arrives at destination.

  When a GST burst is crossing the network the corresponding resource is not available.

  There is a free wavelength for each GST burst generated. It uses only one specific wavelength, which is always available (after exiting from the FDL). A GST burst uses the resource for all the time needed to send the burst and it releases the wavelength at the end of the transmission.

  To indicate that now it is free and available per each type of services we set the flag to 0 and send this information to the "BEserver" class (details in Chapter 3.4.3).

3.4.2 Statistically Multiplexed Real Time Traffic Description

To implemented the SM/RT traffic by Demos we used the schema illustrated in Figure 3.2:

We describe the behavior of SM/RT traffic by two entity classes:

- *RT generator*

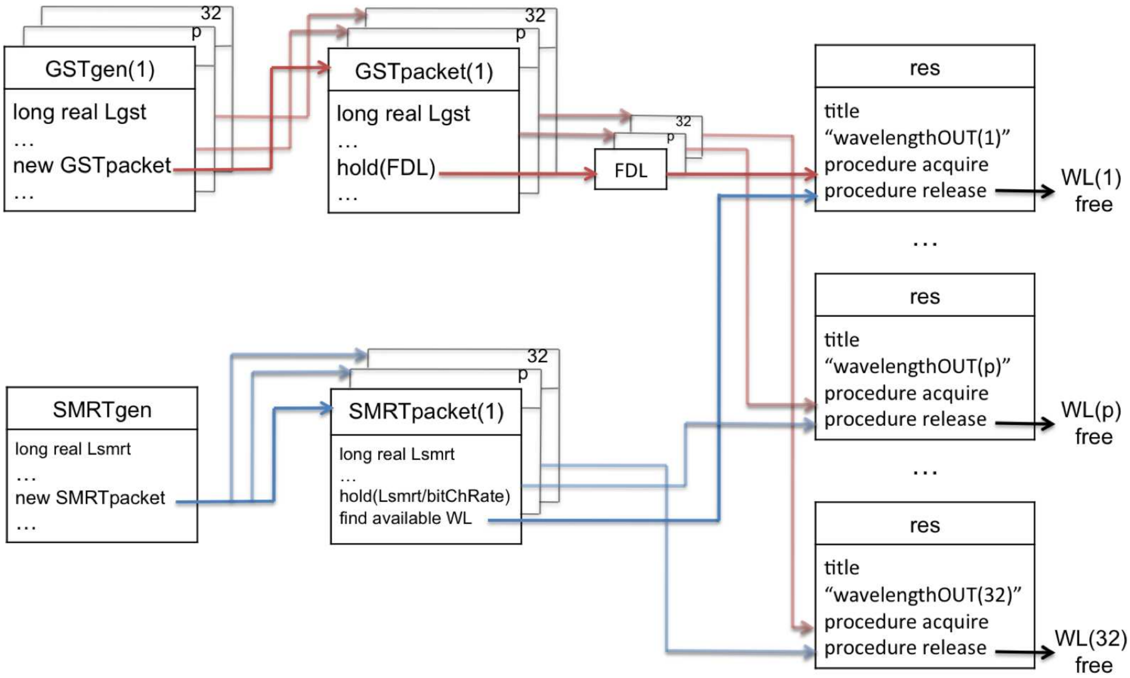

Figure 3.2: Demos implementation for SM/RT traffic

In this case we have a single SM/RT generator that assigns the length value to the generated packet according with the distributions illustrated in Chapter 3.3.
After that we take care of the Collision Avoidance: if the packet is longer than the max mean length SM/RT's packet than we fix equal to max mean length SM/RT's packet.

- *RT packet*

  We give the correspondent length to the SM/RT packet, like showed in Chapter 3.4.

  In order to send a SM/RT packet through the system we need to find a free wavelength or to find a resource that is carrying

a SM/BE packet. In this case we are allow to interrupt the incoming packet and to send the SM/RT.
We check all the wavelengths beginning from the first one and define a local variable. The purpose is to keep track of which is the last output wavelength interrupted: in this way we do not interrupt always the same resource.
If we find an output wavelength with flag equal to 0 that means that it is free and then we use to transmit the SM/RT packet. If we do not, according with the priority traffic law in 3LIHON architecture, we start to search a resource that is carrying a SM/BE packet to interrupt and we use to send a SM/RT packet. If we find a wavelength marked with flag value 3 we are allow to interrupt the transmission because the SM/RT services have high priority level than the SM/BE.

When a SM/BE packet is interrupted we must register the loss. Hence we send an interrupt with power value equal to 2 to the SM/BE packet and memorize which wavelength is used to send the SM/RT packet.
Finally we use the wavelength to transmit the SM/RT packet: we acquire its and set the flag to the correspondent value, that is 2.
The SM/RT packet uses the resource for all time that needs and no one can interrupt.
When the transmission of the SM/RT packet is over the wavelength is released and the flag is reset to 0, free resource.
To inform that the resource is released and available we send this information to "BEserver" class like illustrated in in Chapter 3.4.3.

3.4.3 Statistically Multiplexed Best Effort Traffic Description

We describe the SM/BE traffic by Demos using the schema illustrated in Figure 3.3:

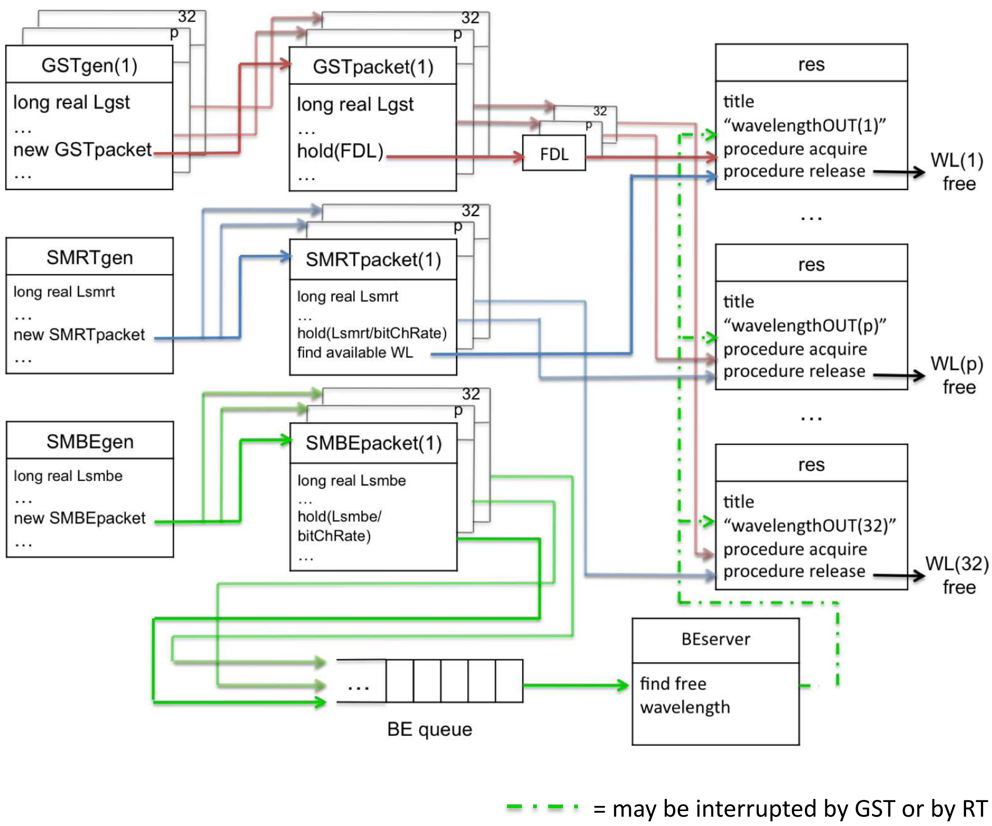

Figure 3.3: Demos implementation for SM/BE traffic

To implement the behavior of the SM/BE traffic is necessary to introduce three different classes:

- *BE generator*

  A single source generates a new SM/BE packet and it assigns them a matching length.
  The generator is busy until the arrive of the next packet and the distribution that governs this behavior is presented in Chapter 3.3.

- *BE packet*

The entity class "SMBEpacket" works with the entity class "BE-server" to handle the transmission of the SM/BE packets in the system. GST bursts and RT packets are both allow to interrupt the transmission of SM/BE traffic so we have to consider and implement that.

All packets generated by the previous class go into a queue, called "BE_Q".
The role of the entity "BEpacket" is to mark the packet with flag's value 3, remembering that it has less priority level.

When there is no competition for the resource, the packet acquires and uses a wavelength until the end of the information unit carried. In the end it changes the flag's value from 3 to 0 to indicate that the resource is now free to carry all types of packets.
We introduce a specific object to synchronization purpose, called "serverBEwaiting_wl". It informs all system's classes that the resource is busy or that the "BEqueue" is empty.
This is a specific type bin object implemented in Demos and used to realize cooperation between multiple entities.
The Demo's description of "serverBEwaiting_wl" is the follow one:

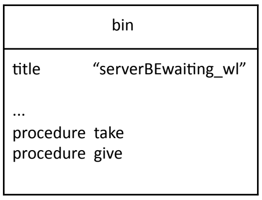

Figure 3.4: Bin object ”serverBEwaiting_wl”

It is possible that more packets require the same wavelength and in this case we handle the resource competition like illustrated follow.

The SM/BE packets can be interrupted:

- by GST burst: a SM/BE packet is crossing the wavelength but receives an interrupt with power value 1 because ”serverBEwaiting_wl” is taken in the class ”GST packet” from a packet with high priority level;
- by SM/RT packet: a SM/BE packet is crossing the wavelength but receives an interrupt with power value 2 because ”serverBEwaiting_wl” is taken in the class ”BE packet” from a packet with high priority level;

We memorize the number of output wavelength that carries the interrupted SM/BE packet. Hence we do not interrupt always the same resource in case of loss.

We register every SM/BE packet loss and the time that each SM/BE packet spend into the queue for statistical purpose.

- *BE server*

This class is in charge to take the packets from the BE queue and send in the output wavelengths.

The server checks all the output wavelengths beginning from the first one and looking for one of them available.
If it find a free resource then checks if there is a SM/BE packet into the BE queue. If both the conditions are true the server takes the first packet that is waiting in queue (FIFO policy), memorizes the output wavelength that is going to use and then starts to send the packet in output resource.
After that the server is free and ready to handle a new SM/BE packet.

Chapter 4

Simulation Scenario & Results

4.1 Simulation Scenario

The focus of this work is on performance for statistically multiplexed traffic in 3LIHON architecture.
We study few series of simulation to analyze the behavior of our network with different input traffic distribution and different load distribution.
The plan of the simulation is illustrated in Figure 4.1:

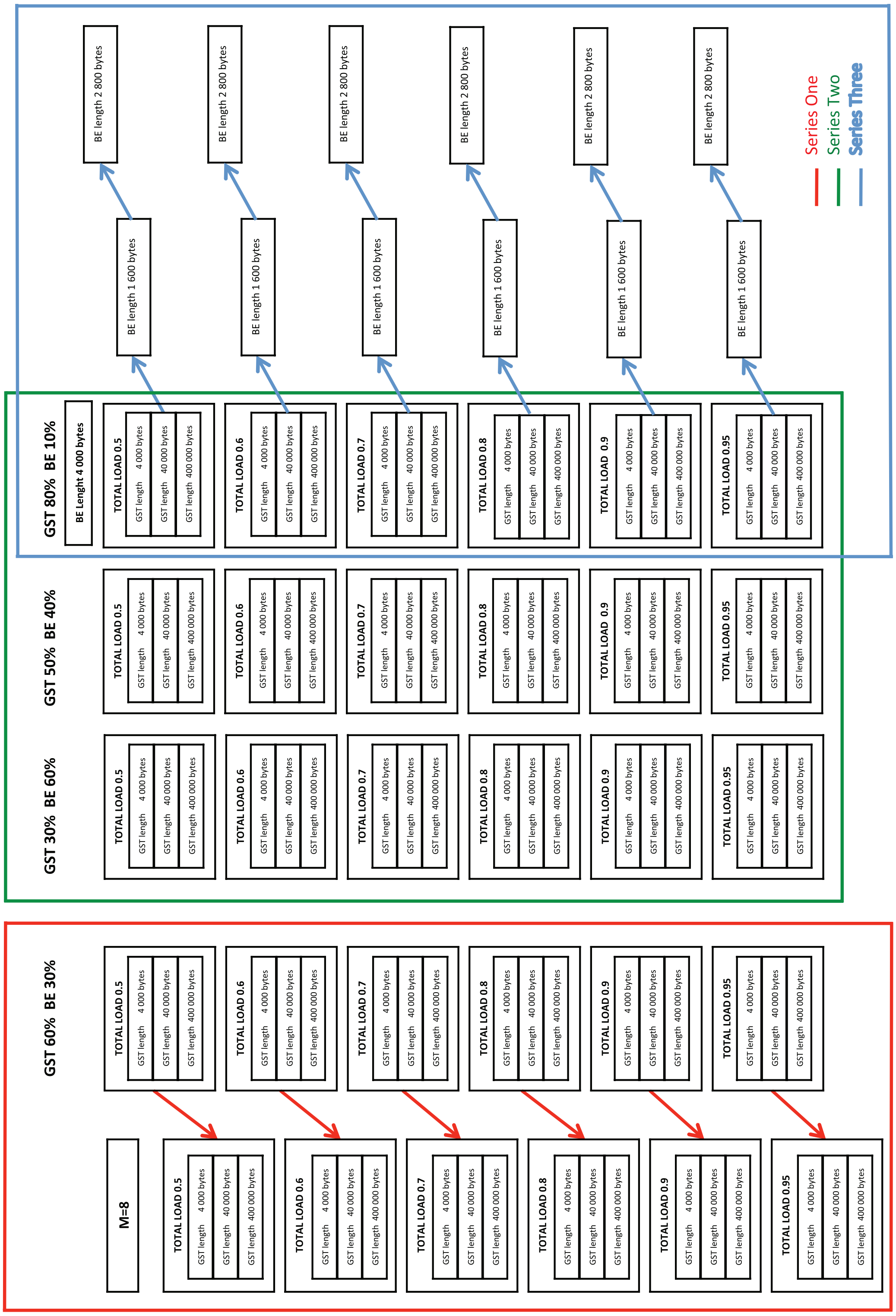

Figure 4.1: Simulation Scenario

4.2 Results

In this section we present the results of 3LIHON simulation model in the following cases:

- **Series One**
  We consider a mix of relative traffic flow:
  GST 60%, SM/RT 10% and SM/BE 30%.
  We decide to analyze the behavior varying the mean value of GST bursts length:
  4000 *bytes*, 40000 *bytes* and 400000 *bytes*.
  We fix the number of output wavelength $M = 32$.
  Then we study the Packet Loss Probability (PLP) for Statistically Multiplexed Real Time (SM/RT) traffic, the Packet Loss Probability (PLP) for Statistically Multiplexed Best Effort (SM/BE) traffic and the delay of the SM/BE packets in the BE queue.
  In order to consider the model's behavior varying the number of output wavelength change we decrease M to 8 and study the Packet Loss Probability (PLP) for Statistically Multiplexed Real Time (SM/RT) traffic.

- **Series Two**
  We study the system with four different mix of relative traffic:

  - GST 60% and SM/BE 30%;
  - GST 50% and SM/BE 40%;
  - GST 30% and SM/BE 60%;
  - GST 80% and SM/BE 10%;

  and with a percentage of SM/RT equal to 10%.
  We decide to analyze what happen when the mean value of GST bursts length is:
  4000 *bytes*, 40000 *bytes* and 400000 *bytes*.
  We study the Packet Loss Probability (PLP) for Statistically Multiplexed Real Time (SM/RT) traffic, the Packet Loss Probability (PLP) for Statistically Multiplexed Best Effort (SM/BE) traffic and the delay of the SM/BE packets in the BE queue.

- **Series Three**
  We consider a mix of relative traffic flow:
  GST 80%, SM/RT 10% and SM/BE 10%.
  We analyze the behavior varying the mean value of SM/BE packets length:
  1600 *bytes*, 2800 *bytes* and 4000 *bytes*.
  We fix the mean value of GST bursts length to 40000 *bytes*.
  Then we study the Packet Loss Probability (PLP) for Statistically Multiplexed Real Time (SM/RT) traffic, the Packet Loss Probability (PLP) for Statistically Multiplexed Best Effort (SM/BE) traffic and the delay of the SM/BE packets in the BE queue.

The simulations are obtained using a student's T distribution and running $n = 10$ independent replications to establish 95% confidence intervals. We represent the confidence intervals in our results.

More details about confidence interval, student's T distribution and parameters chosen are shown in Appendix A.

4.3 Series One Results: GST 60% SM/BE 10% SM/RT 30%

In this section we study 3LIHON architecture with the follow relative load partition:

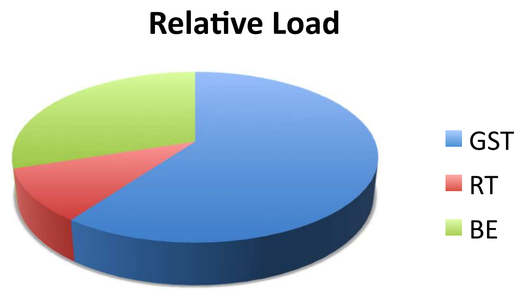

Figure 4.2: Relative Load Series One GST = 0.6 SM/RT = 0.1 SM/BE = 0.3

4.3.1 Packet Loss Probability of the SM/RT

Firstly we focus on the Packet Loss Probability (PLP) of the Statistically Multiplexed Real Time (SM/RT) traffic.

This value is the ratio between the SM/RT packets lost and the SM/RT packets that require a free wavelength.
Like showed in Chapter 3 the SM/RT traffic is allow to interrupt SM/BE packets but not GST. If all wavelengths are busy to transmit GST bursts or other SM/RT packets then the SM/RT packet is lost.

We simulate 5 seconds with a transient time of 0.25 seconds.
Each simulation counts $n = 10$ sub-simulation, per 10 different seeds. In this way we can describe the parameters with confidence interval of 95%, like illustrated in Appendix A.

4.3.1.1 PLP of SM/RT with $M = 32$

We consider a 3LIHON model with $M = 32$ wavelength and we study the behavior for three different length's value of GST bursts.
We obtain the following results:

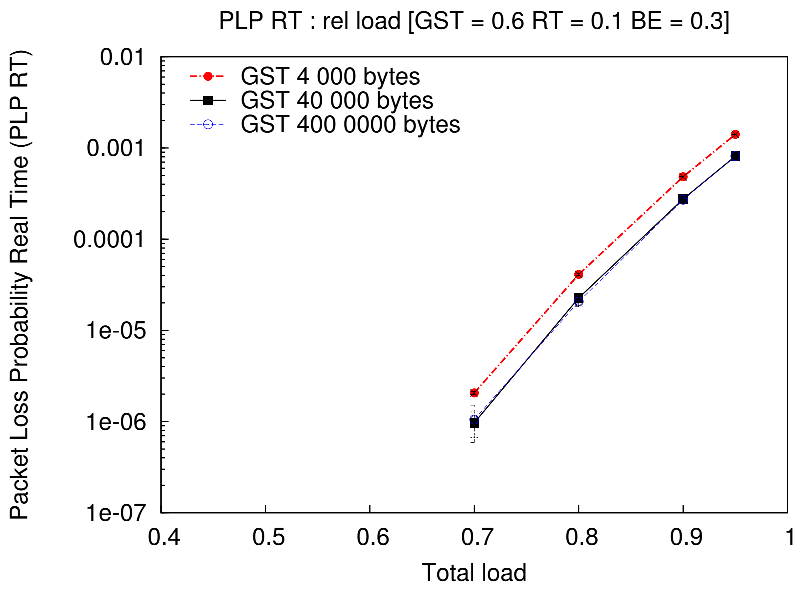

Figure 4.3: PLP SM/RT Series One

We studied the system for total load values low, e.g. 0.5 and 0.6, but the simulation time that we implement is not enough to get statistically significant values.

In Figure 4.3 we have a complete study of the PLP SM/RT packets in the relative load conditions illustrated before.
We vary the mean value of length of GST bursts. For smaller values of the GST length the loss probability of the SM/RT packet is bigger. For longer values of GST length is lower.
In the second case the GST bursts are less but longer and the free space between the end of one GST burst and the beginning of the next one is longer. The SM/RT packets use this free space to acquire

a resource and then the chances to find one of them free are higher. Hence the PLP of the SM/RT packet is smaller when the distribution that describe the length of GST burst has a bigger mean value.

The Figure 4.3 shows that the trend of the curves for mean value GST length equal to 40000 and 400000 *bytes* is almost the same.
The reason is because in any case SM/RT packets are allow to interrupt SM/BE transmission so the GST length does not influence so much SM/RT traffic. The relative load of the SM/RT packets is in general not too high and in all our studies is 10%.

4.3.1.2 PLP of SM/RT with $M = 8$

In order to consider the model's behavior when the the number of output wavelength change we make a complete set of simulations considering a smaller number of output wavelength, $M = 8$.

We study the PLP of SM/RT and the results are shown in Figure 4.4 :

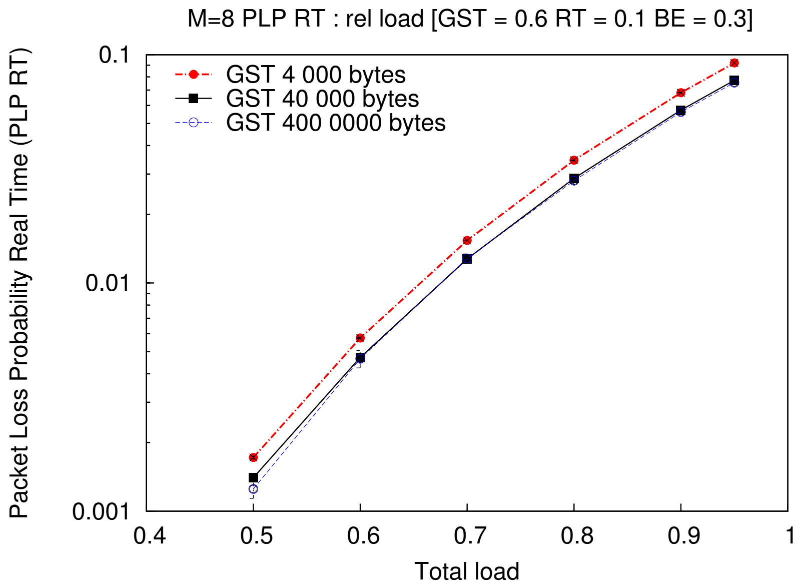

Figure 4.4: PLP SM/RT Series One with $M = 8$

We analyze this study case decreasing the number of the output wavelengths from 32 to 8 because we want to observe how the PLP SM/RT changes with less wavelengths available.
The conclusion is that GST length does not influence so deeply the PLP SM/RT.
Comparing the behavior of the system with $M = 32$ and $M = 8$ is easy to conclude that in the second case the loss is greater.
We have less available output resources to use and the probability that all of them are carrying GST bursts or previous SM/RT packets is higher.
In the next study cases we keep the number of output wavelength $M = 32$.

4.3.1.3 PLP of SM/RT varying GST bursts length

In this section we consider the PLP SM/RT in function of GST length values for different total load values.

The results are shown in Figure 4.5:

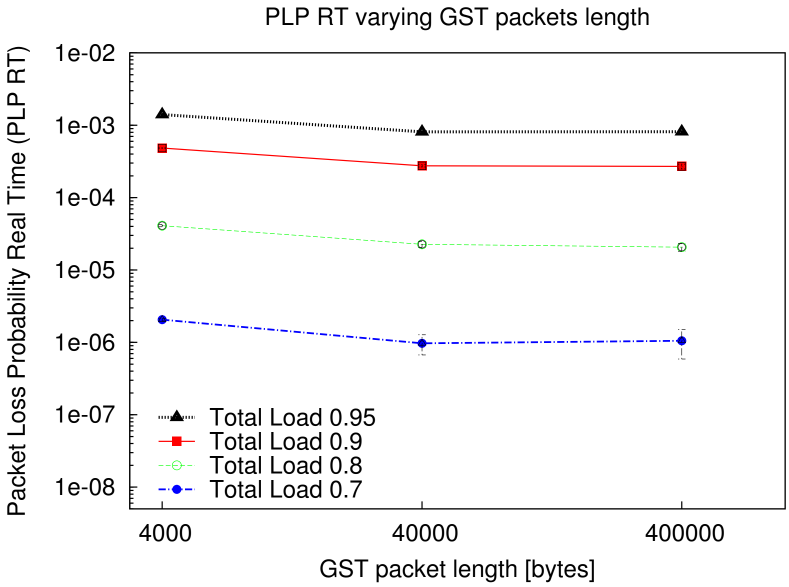

Figure 4.5: PLP SM/RT Series One varying GST length

The value of the PLP SM/RT is almost the same for all total load values when the mean length of GST burst is equal to 40000 or 400000 .
Longer is the GST burst and lower is the probability to lose a SM/RT packet.

4.3.2 Packet Loss Probability of the SM/BE

Secondly we analyze the Packet Loss Probability (PLP) of the Statistically Multiplexed Best Effort (SM/BE) traffic.

We define this parameter like ratio between the number of SM/BE packets lost and the number of the SM/BE packets taken out of the BE queue.
Each SM/BE packet generated goes into a queue, waits until one of

the output wavelength is free and finally the transmission starts.
If a GST or SM/RT packet needs to use a resource and no one is free then the SM/BE is interrupted and we register a packet loss. Each single packet that goes into the queue is a packet that is allowed to require an output wavelength, if it arrives or not to its destination depends of the other two classes.

We simulate 5 seconds with transient time 0.25 seconds and study the 3LIHON system with $M = 32$ output wavelengths.
Each simulation counts $n = 10$ sub-simulations, with 10 different seeds.

In the Figure 4.6 we present the PLP for SM/BE using a logarithmic scale:

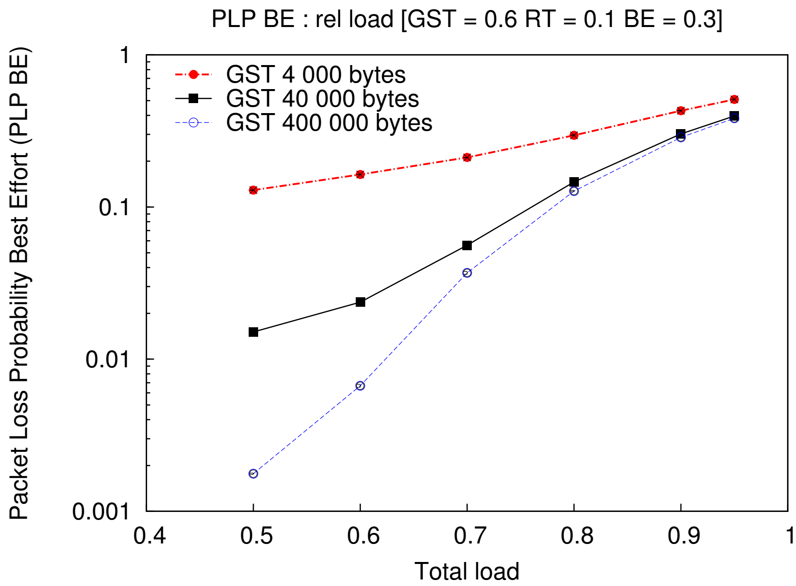

Figure 4.6: PLP SM/BE Series One using logarithm scale

The probability to lose a packet in the SM/BE case is higher than in SM/RT case and that is logical because of how we implemented the traffic priority.

We observe that if longer is the size of the GST bursts than lower is the probability that a SM/BE packet is interrupted. If we have long GST bursts with a fixed relative load the GST generator produces less packets than in the case with GST shorter bursts. There are less GST bursts that can interrupt SM/BE packets. That explain the results plot in Figure 4.6.

Figure 4.7 shows PLP of SM/BE using a linear scale to focus on high total load values.

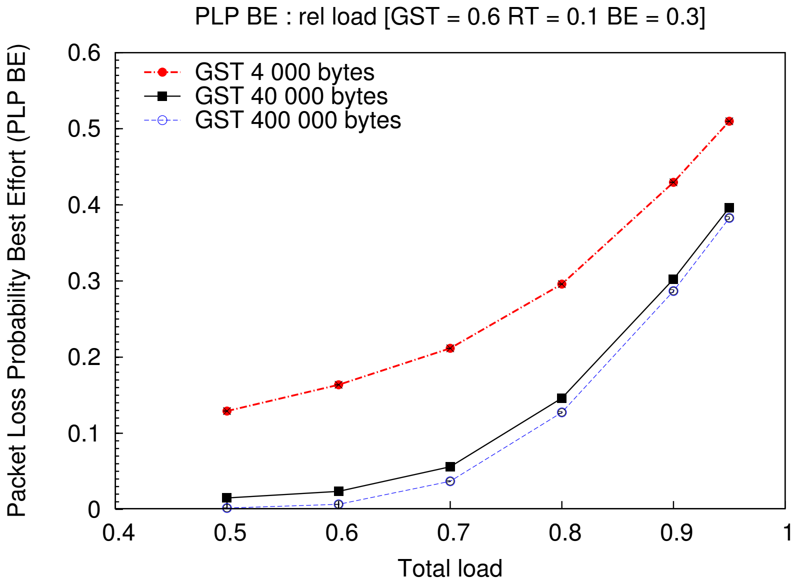

Figure 4.7: PLP SM/BE Series One using linear scale

The PLP of SM/BE is lower for GST bursts length equal to 400000 *bytes* than for 40000 *bytes*.

SM/BE traffic is characterized by the lowest priority level then we think that is interesting to see which is the influence of the other two services class to interrupt the SM/BE packets.

We study the relative percentage of SM/BE packets interrupted by GST bursts (Figure 4.8)

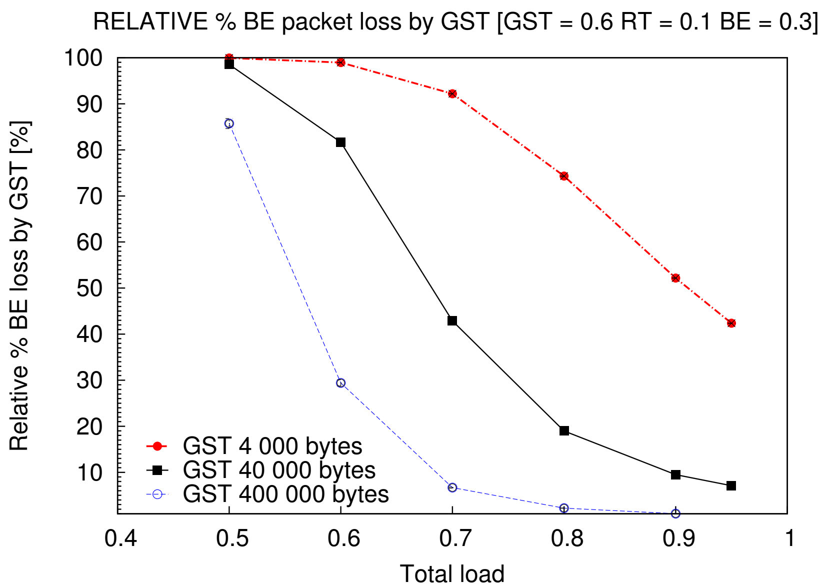

Figure 4.8: Relative percent SM/BE loss by GST Series One

and the relative percentage of SM/BE packets interrupted by SM/RT packets (Figure 4.9).

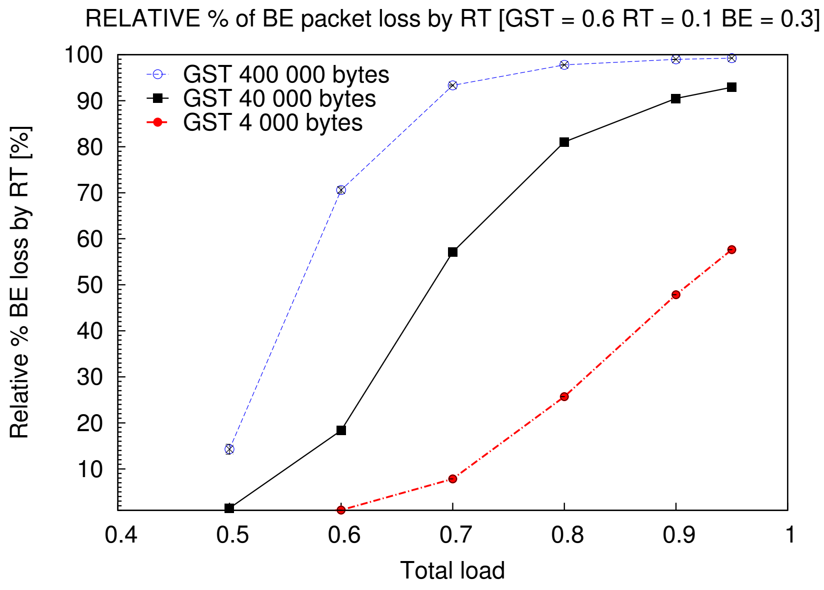

Figure 4.9: Relative percent SM/BE loss by SM/RT Series One

In this case the statistical parameter that we study with 95% of confident interval is the ratio between the number of BE packets interrupted by GST and the total number of SM/BE packets interrupted. We called this relative percentage of SM/BE packets interrupted by GST services.
The trend of this curves shows that when the total load is low the SM/BE packets are interrupted mostly by GST bursts. The relative percentage of SM/BE packet interrupted in this case is very high then it depends completely to the behavior of the GST bursts.
In the other hand when the total load is high the behavior of the GST bursts does not influence so much the relative percent of SM/BE packet lost.

We analyze the ratio between the number of SM/BE packets interrupted by SM/RT and the total number of SM/BE packets interrupted calling relative percentage of SM/BE packets interrupted by SM/RT. The trend of this curves shows that for high total load the SM/BE packets are interrupted mostly by SM/RT packets but for low total load they do not influence so much the percentage of interrupted packets.

4.3.3 Delay of the SM/BE packet

Finally we study the delay of the Statistically Multiplexed Best Effort (SM/BE) packets.
All SM/BE packets that want to use a wavelength have to wait in a queue called "BE_Q" (Chapter 3.4.3). We think that is interesting to study the mean value of delay of SM/BE packets in this queue.

Per each SM/BE packet that is generated we calculate the difference between the time that it arrives in queue and that it is taken out from the queue by the BE server. This value is the delay of a single BE packet. We sum all this partial delays and divide for the number of the BE packets that are taken out of the BE queue obtaining the mean value of BE packet delay.
This is the parameter that we study with 95% of confident interval.

The results presented in Figure 4.10 show mean delay for SM/BE packets as a function of total load for three different mean GST burst length values:

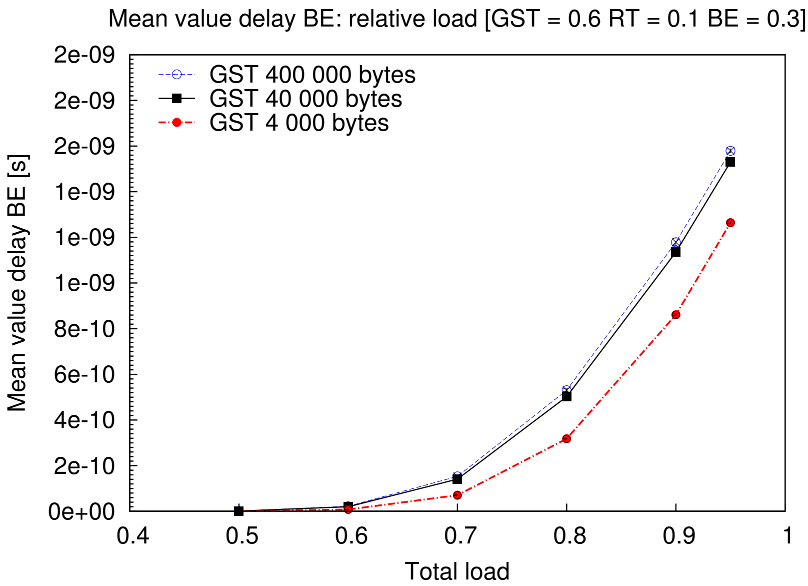

Figure 4.10: Mean value of SM/BE packets delay Series One

The y-axis represents the mean delay for SM/BE packet in second.

We observe that higher is the total load then higher is also the mean value delay observed for SM/BE packet. If the total amount of the packets increases also the probability that BE services are interrupted is higher.

We can observe different behaviors varying GST length. When the length is short then the delay is small. When GST packets are long then they take the output wavelengths longer and the waiting time of the SM/BE packet in the BE queue is longer.

Because of the nature of this statistic parameter we decided to study also the moment of the second order of the SM/BE packet delay: the variance of the mean value of the SM/BE packet delay.

The results are in Figure 4.11:

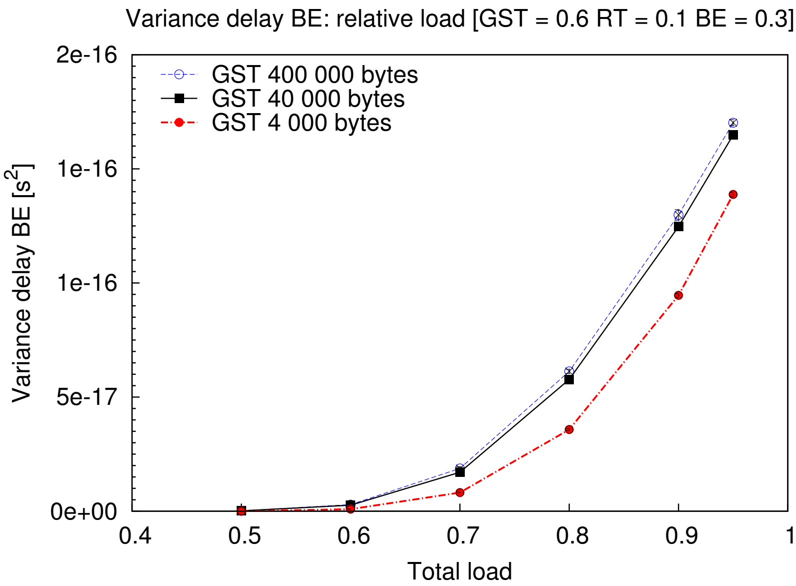

Figure 4.11: Variance of SM/BE packets delay Series One

The values of the delays in this second case are much smaller. Focusing on relative load value 0.8 and on the GST mean value of 4000 *bytes* we can notice that there is a difference of more than ten orders of magnitude and the moment of the second order is smaller than the moment of the first. We have a delay of 2.6 10^-6 for the mean value and the corresponding one for the variance is 4.7 10^-17.

To higher total load correspond longer waiting time into the BE queue and if the GST length is shorter then the delay is smaller.

4.4 Series Two Results

4.4.1 GST burst length 40 000 bytes

In this section we show the results of 3LIHON simulation model varying the mix of the relative percentage of services like illustrated in Chapter 4.2 and keeping the mean value of GST bursts length equal to 40000 *bytes*.

4.4.1.1 Packet Loss Probability of the SM/RT

Firstly we study the trend of the Packet Loss Probability (PLP) of Statistically Multiplexed Real Time (SM/RT).

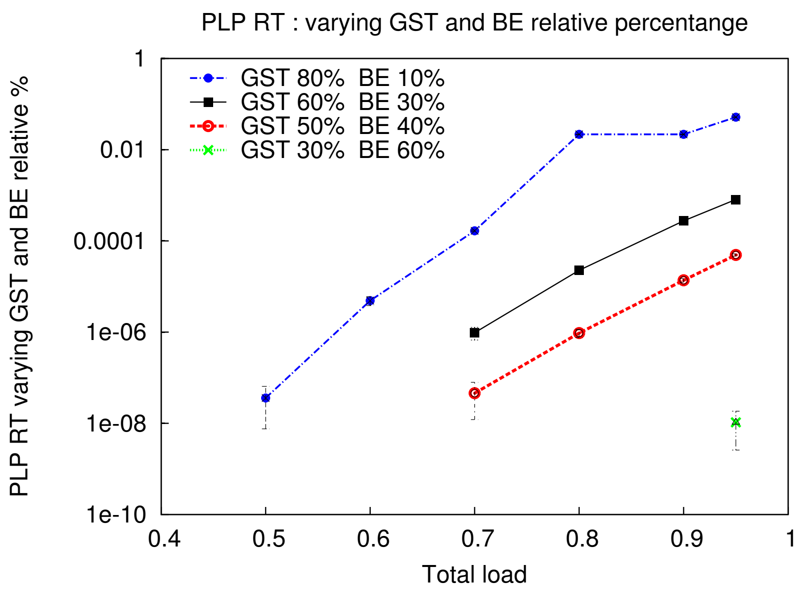

Figure 4.12: PLP SM/RT varying relative percentage GST & BE

Figure 4.12 shows that the PLP of SM/RT for total load equal to 0.8 is higher than the value obtained for total load 0.9 when the relative percentage of guaranteed traffic is high. This behavior looks unusual if compared with the study cases so far, however it is consistent with the OpMiGua's results showed in [11] and [12].

[12] demonstrates that the relative share and length of GST packets influence the loss mechanism related to the Statistically Multiplexed (SM) classes and then the Packet Loss Probability(PLP). Varying GST share we have significant improvement in the PLP.
[11] presents an analysis of packet loss in OpMiGua node with given reservation technique. For shorter GST packets and GST share below 85% the mean difference between simulation and analysis is approximately 10%.
The results in Figure 4.12 show that when the relative percentage of guaranteed traffic is high the PLP of SM/RT for total load equal to 0.8 is higher than for total load 0.9. Hence the results obtained for 3LIHON model are consistent with the previous OpMiGua studies.

Furthermore Figure 4.12 shows what happen when the simulation time chosen is not enough to have appropriate level of accuracy. This is the share GST 30% percentage case.

4.4.1.2 Packet Loss Probability of the SM/BE

Secondly we study the Packet Loss Probability (PLP) of Statistically Multiplexed Best Effort (SM/BE) packets, showed in Figure 4.13:

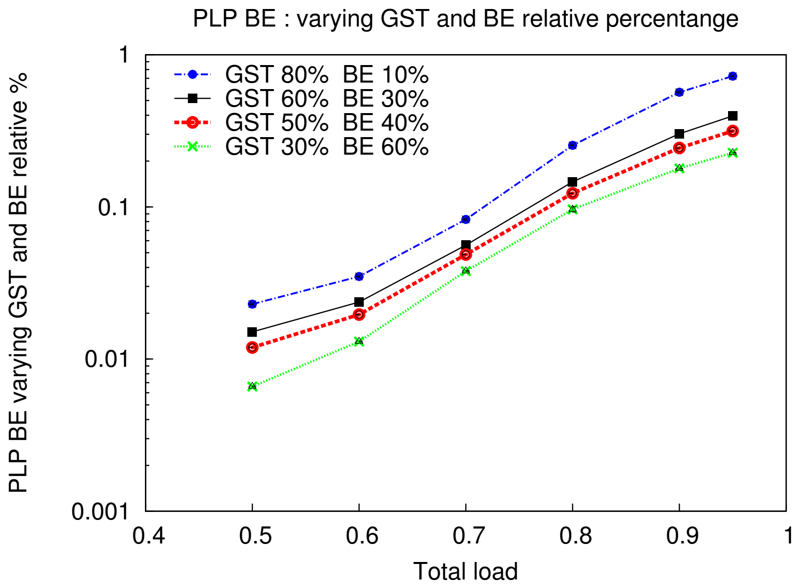

Figure 4.13: PLP SM/BE varying relative percentage GST & BE

We observe that the trend is increasing with the total load value. More bursts and packets are crossing the system and higher is the probability that a SM/BE packet is interrupted because has lower priority level.
When the percentage of GST bursts is higher then the PLP of SM/BE is higher too.

4.4.1.3 Delay of SM/BE packet

In this section we consider the mean value of the delay of SM/BE in the BE queue.

The results are in Figure 4.14:

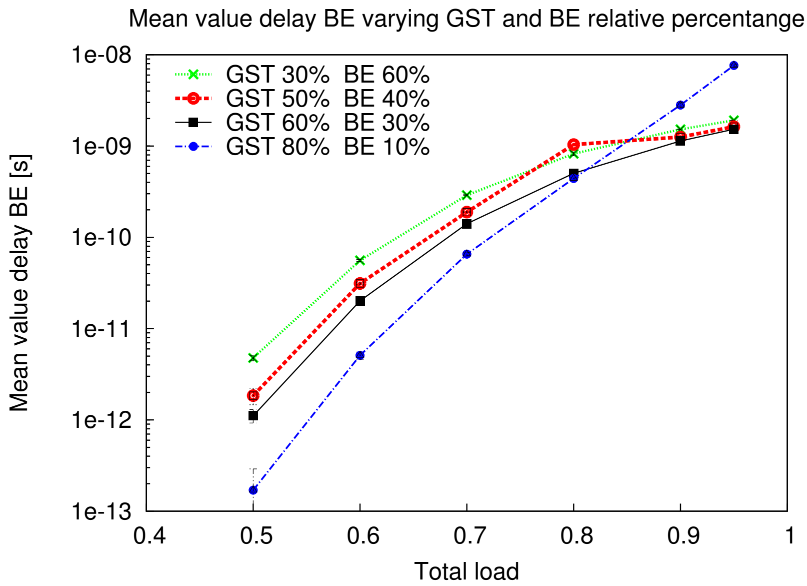

Figure 4.14: Mean Value Delay SM/BE packets varying relative percentage GST & BE

When the total load is below 80% the delay of SM/BE packets in the BE queue increases with the relative percentage of SM/BE packets. For total load 80% the value for share GST percentage 60% and 80% is quite similar and the higher value is registered for share GST packets 50%.

This behavior is different from the previous study cases and to understand we focus on the blue case (80% GST and 10% BE) at loads 0.90 and 0.95. The few SM/BE packets in queue have low probability of finding a free wavelength in-between GST bursts and SM/RT packets since the total load on the wavelengths is high. Hence they are stuck in the queue longer. At lower loads this effect is not the dominating one since then the SM/BE packets are more likely to find free wavelengths and stays shorter in queue. In lower loads case the relative amount of SM/BE packets seems to dominate, giving the other color graphs higher delay values.

In this context is interesting to study the delay of SM/BE traffic varying the relative percentage of SM/BE packets.

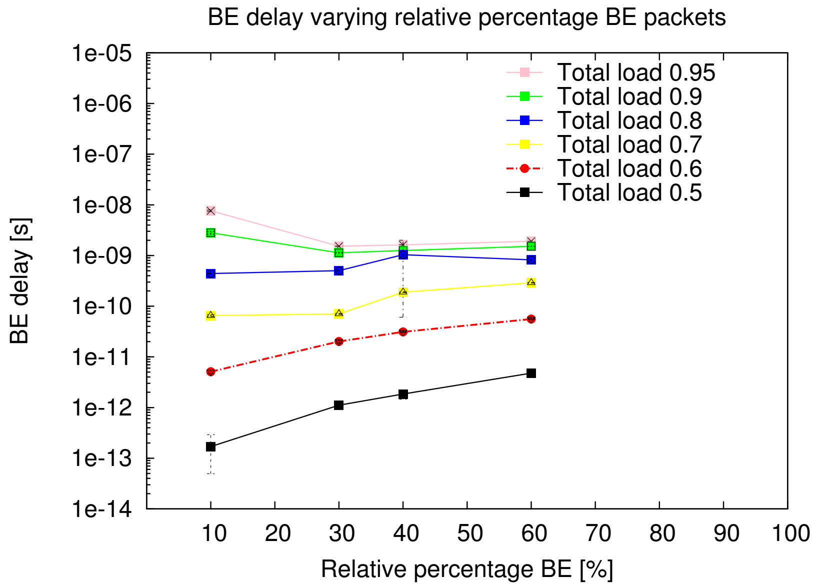

Figure 4.15: SM/BE packets delay varying the relative percentage of SM/BE traffic

Figure 4.15 shows that when the total load is low the SM/BE service is not congested. The delay of SM/BE packets in BE queue increases with the relative percentage of SM/BE traffic.

On the other hand when the total load is high the SM/BE service is congested. The BE queue is not able to handle the SM/BE packets incoming and GST traffic improves its performance.

4.4.2 GST burst length 4 000 bytes

In this section we study the simulation model varying the mix of the relative percentage of services like illustrated in Chapter 4.2 and keeping the mean value of GST bursts length equal to 4000 *bytes*.

4.4.2.1 Packet Loss Probability of the SM/RT

We consider the Packet Loss Probability (PLP) of Statistically Multiplexed Real Time (SM/RT) packets.

Results are shown in Figure 4.16:

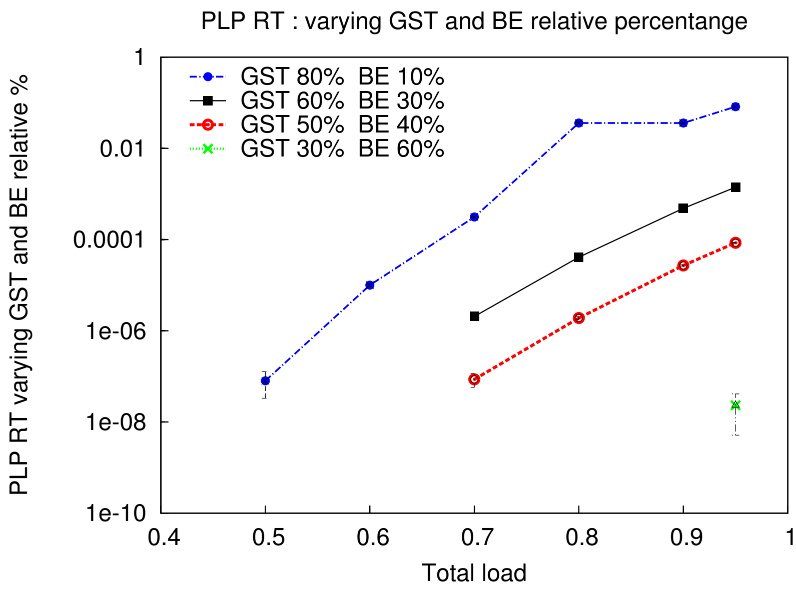

Figure 4.16: PLP SM/RT varying relative percentage GST & BE

Consideration about this results are similar to the previous study case Series One.
When the relative percentage of GST burst is high the probability to lose SM/RT packet is high too.
When the share GST burst percentage and the relative load are 80% we observe an unexpected peak. The result is consistent with previous OpMiGua results showed in [11] and [12].

4.4.2.2 Packet Loss Probability of the SM/BE

In this section we study the Packet Loss Probability (PLP) of Statistically Multiplexed Best Effort (SM/BE) packets.

The results are shown in Figure 4.17:

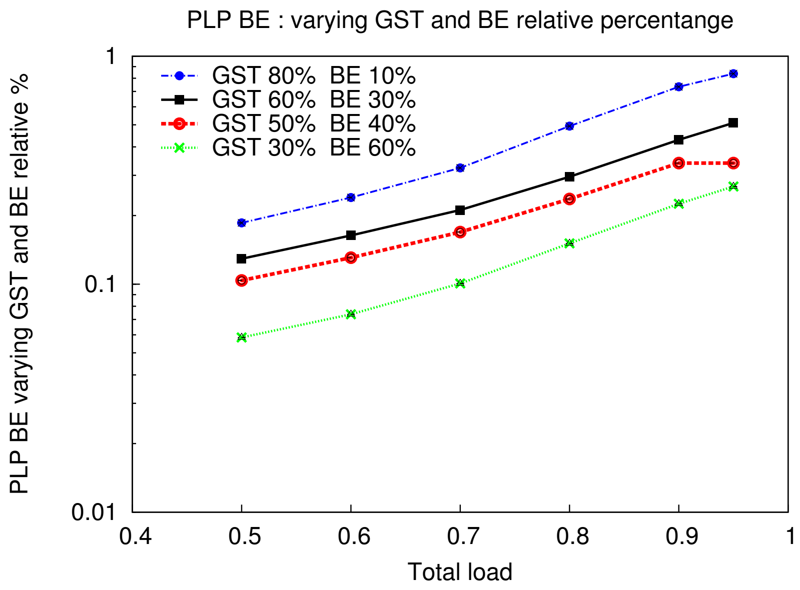

Figure 4.17: PLP SM/BE varying relative percentage GST & BE

Higher is the relative percentage of GST traffic and higher the probability to lose SM/BE packets too.
When the total load is low SM/BE packets are interrupted mostly by GST burst, in the other hand when the total load is high they are interrupted by SM/RT. This result was found out in OpMiGua model [1] and is still consistent in 3LIHON like showed in Figure 4.17. Comparing this results with Figure 4.12 we observe that shorter is the length of GST bursts and lower the PLP of SM/BE packets.

The PLP of SM/BE for the share GST percentage equal to 50% is quite the same for total load 0.9 and 0.95.
Further works can be focus on that perhaps focusing on the study case with relative percentage of GST burst 50% (Appendix B.1) and varying some others background parameters.

4.4.2.3 Delay of SM/BE packet

We study the mean value of the delay of the SM/BE in the BE queue:

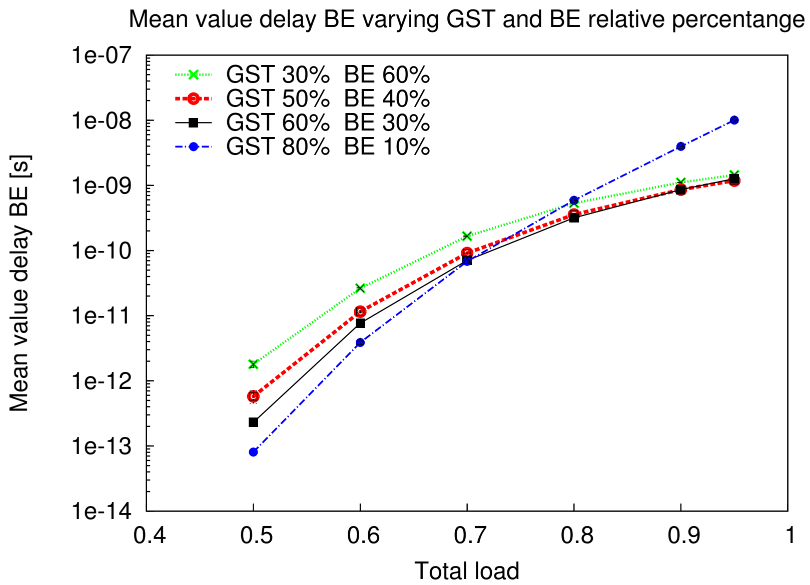

Figure 4.18: Mean Value Delay SM/BE packets varying relative percentage GST & BE

The Figure 4.18 shows that below 0.7 the delay increases for each percentage of GST packets relative load. When the total load is 0.7 the delay is almost the same for relative percentage of guaranteed traffic equal to 50 %, 60% and 80% but not 30%. Above total load 0.7 the results for relative percentage of GST bursts 30%, 50% and 60% start to converge on the same value and it happens for total load 0.95. However the curve of the GST 80% increases above total load 0.7.

When the relative percentage of GST traffic is high our simulation results show a particular behavior, like in OpMiGua model ([11] and [12]).

A possible explanation is showed above (cf. Chapter 4.4.1.3). When the total load is high there are few SM/BE packets in queue.

They have low probability to find a free wavelength, thus they are stuck in the queue longer. In lower loads case the relative amount of SM/BE packets seems to dominate giving the other color graphs higher delay values.

4.4.3 GST burst length 400 000 bytes

4.4.3.1 Packet Loss Probability of the SM/RT

In this section we study the trend of the Packet Loss Probability (PLP) of Statistically Multiplexed Real Time (SM/RT) when the GST burst length is 40000 *bytes*.

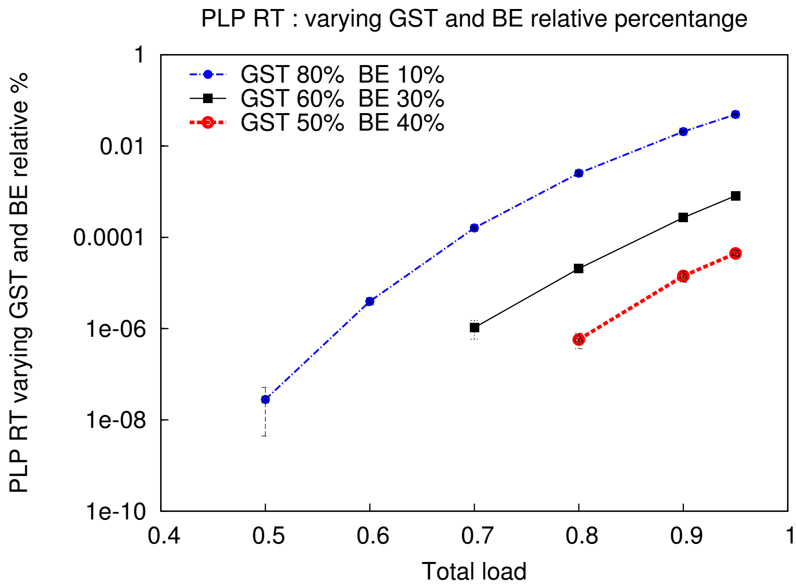

Figure 4.19: PLP SM/RT varying relative percentage GST & BE

Figure 4.19 shows three different mix of relative percentage of GST and SM/BE traffic because the obtained results for GST 30% and SM/BE 60% are not consistent. The time simulation used is not enough to give result with a good level of accuracy. Then we decide to do not include this result.

The curve with relative percentage of GST bursts higher does not show a peak for total load equal 0.8. Hence is possible that choosing a relative percentage of guaranteed traffic high enough we can influence the PLP of SM/RT.

This is an interesting point to focus on in further works.

4.4.3.2 Packet Loss Probability of the SM/BE

In this section we consider the Packet Loss Probability (PLP) of SM/BE packets.

The results are in Figure 4.20:

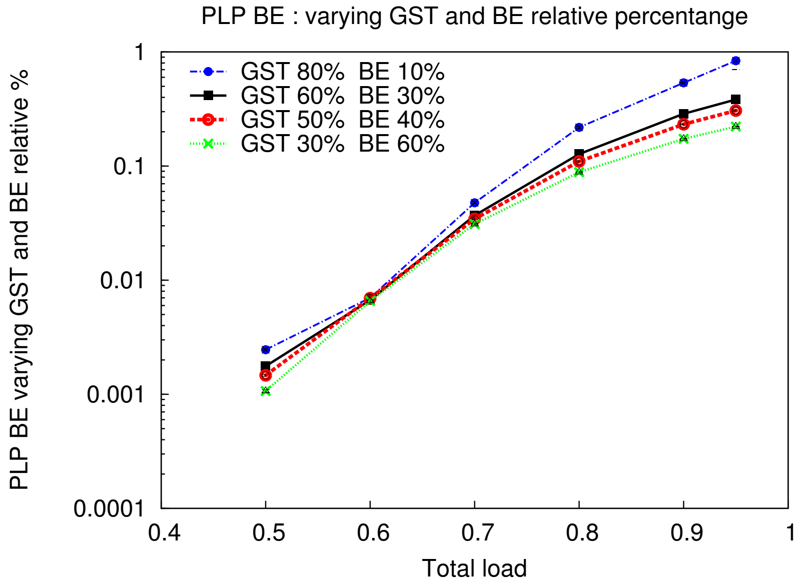

Figure 4.20: PLP SM/BE varying relative percentage GST & BE

Our simulation results show that the trend of the curves is the same of the previously cases: higher is the relative percentage of GST burst and higher is the probability to lose SM/BE packets.
However Figure 4.20 shows that when the total load value is 60% the PLP of SM/BE packets is the same.

This behavior is interesting and it is left for further works.

4.4.3.3 Delay of SM/BE packet

This section studies the delay of the SM/BE packets in the BE queue.

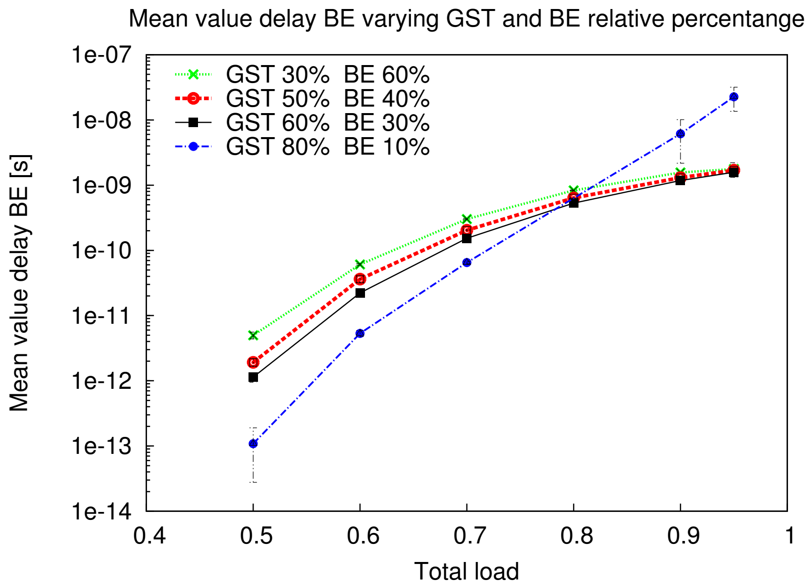

Figure 4.21: Mean Value Delay SM/BE packets varying relative percentage GST & RT

The Figure 4.21 shows results consistent with the two previous studies (Chapter 4.4.1.3 and Chapter 4.4.2.3).
Firstly the curves increase for all mix of relative percentage of GST & SM/BE. When the total load is 0.8 the curves for GST 30%, 50% and 60% start to converge. Moreover they converge for total load 0.95. However the curve that represent the higher percentage of GST burst keeps an increasing trend also above the total load value 0.8. The explanation for this behavior is given in 4.4.1.3.

4.5 Series Three Results: GST 80% BE 10% RT 10% varying BE length

The last study is focus on the behavior of the system when the size of the SM/BE packets changes.

The relative load partition used in this case is illustrated in Figure 4.22:

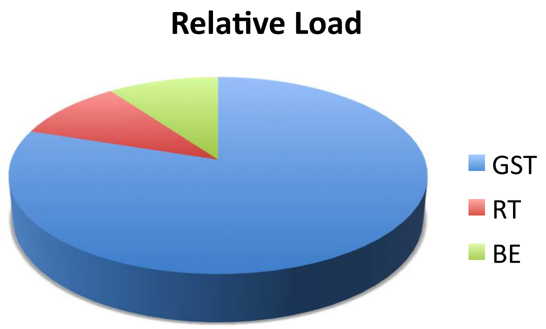

Figure 4.22: Relative Load Series Three GST = 0.8 RT = 0.1 BE = 0.1

It is necessary to define a value for the mean value of distribution that provides the length of the guaranteed services. We can not decide this value without consider the relation with the mean value of the SM/BE packets length. To manage the Collision Avoidance (CA) is necessary to respect what illustrated in Chapter 2 and then the GST bursts have to be always longer that SM/BE packets.

We set the mean value of the GST bursts to 40000 *bytes* and for the mean value of SM/BE packets we consider the follow cases:

- SM/BE packet length 4000 *bytes*;
- SM/BE packet length 2800 *bytes*;

- SM/BE packet length 1600 *bytes*;

4.5.1 Packet Loss Probability of the SM/RT

The study of the PLP RT varying the length of the SM/BE packets shows something that we know since the description of the 3LIHON architecture: the SM/BE traffic is not allow to influence SM/RT service performance because it has lower priority level.

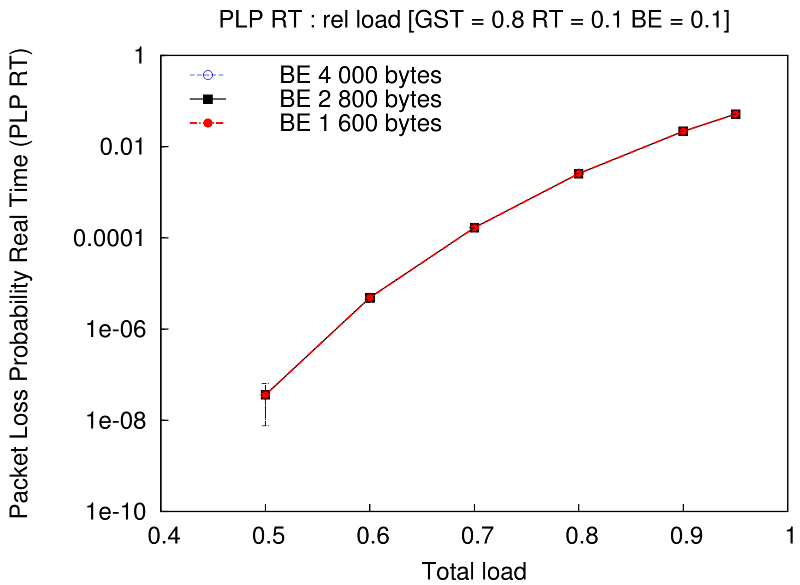

Figure 4.23: PLP of SM/RT Series Three

The SM/RT packets can always interrupt SM/BE packets then the PLP SM/RT does not change varying the length of the SM/BE packets.

Figure 4.23 shows this behavior, as expected.

4.5.2 Packet Loss Probability of the SM/BE

In this section we study the Packet Loss Probability (PLP) of the SM/BE for different values of the SM/BE packets length.

The results are shown in Figure 4.24:

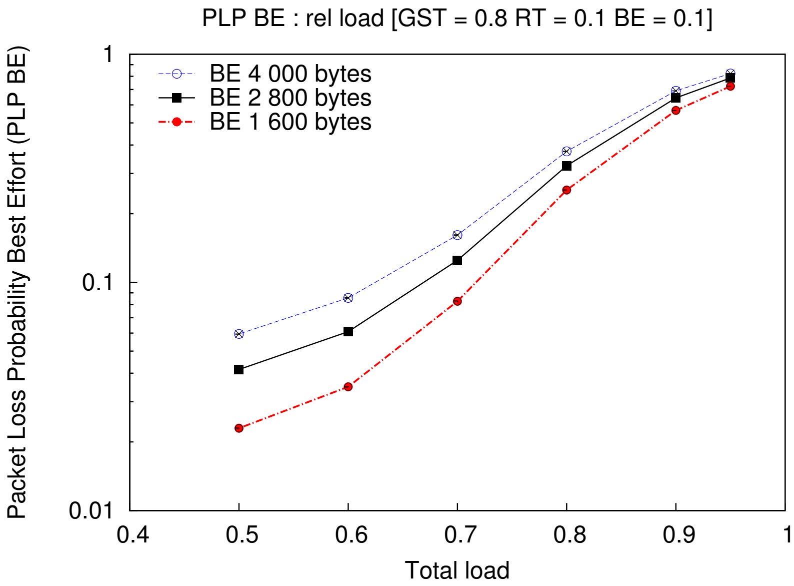

Figure 4.24: PLP of SM/BE Series Three using logarithmic scale

The probability to interrupt a SM/BE packet increases with the length of the packet. If the SM/BE packet is shorter then the time to transmit through an output wavelength is shorter. If the time needed to forward the information is smaller then the probability that a GST burst or a RT packet needs to use the resource is smaller.

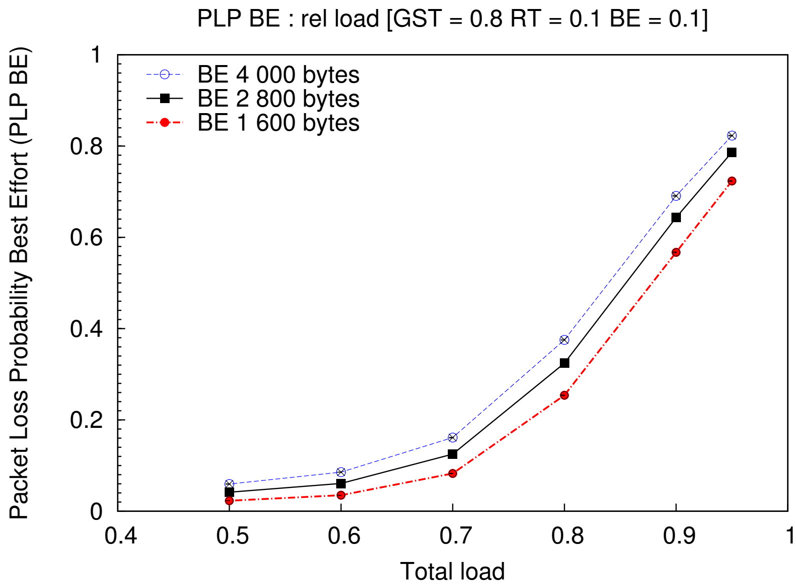

Figure 4.25: PLP SM/BE Series Three using linear scale

Figure 4.25 shows the same result but with linear scale to observe in detail the behavior with high total load.

4.5.3 Delay of the SM/BE packet

In this section we study the delay of the SM/BE packets in the BE queue.
In Figure 4.26 we show the results for the mean value of the delay:

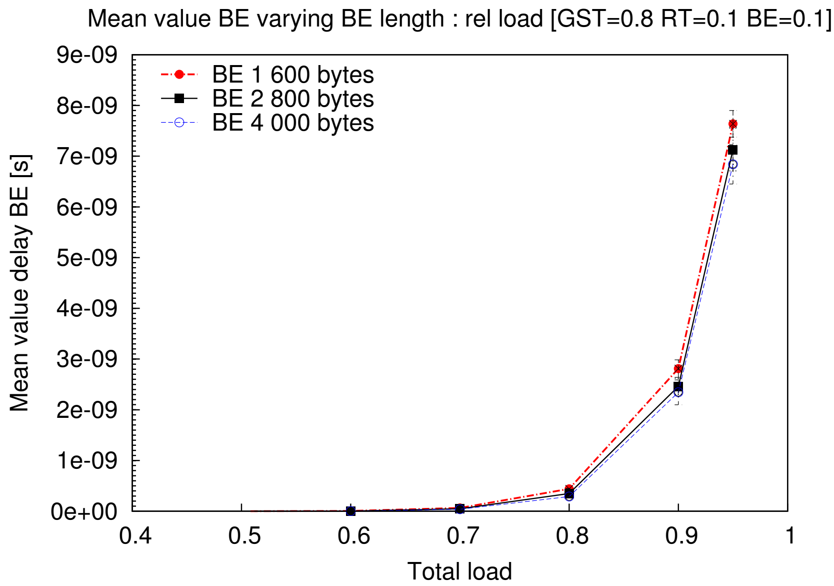

Figure 4.26: Mean Value of SM/BE packets delay Series Three

The probability that a SM/BE packet is stuck into the queue is greater when the size of the packets is smaller because the SM/BE generator produces more packets. All the generated packets have to wait in the queue before to be forwarded in the matched output wavelength like illustrated in Chapter 3.4.3. The delay increases if the percentage of GST burst increases too.

In the Figure 4.27 we show the result for the variance of the delay of SM/BE packets:

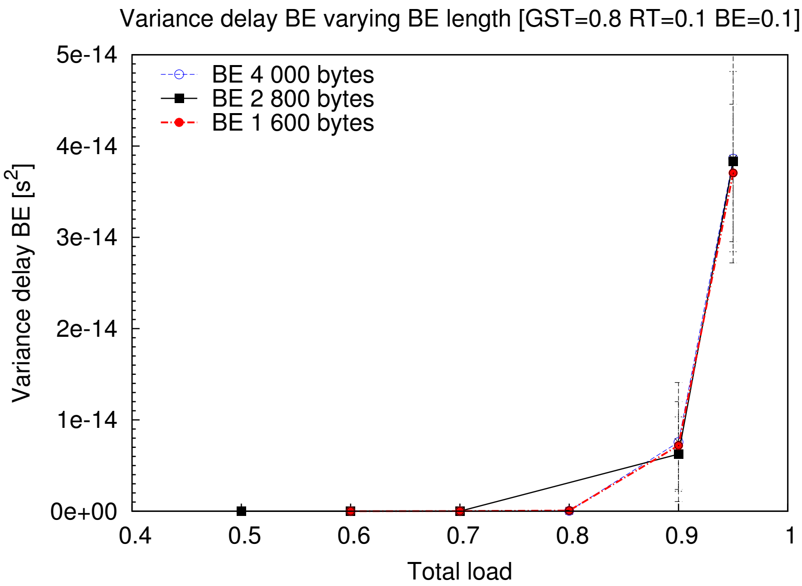

Figure 4.27: Variance of SM/BE packets delay Series Three

Considerations for this result are similar to Series One case (Chapter 4.3.3). The values of the variance of delay are smaller than mean value one. The time that the packet SM/BE spends in the queue is longer when the total load is higher.

Chapter 5

Conclusions

This thesis focus on Quality of Service (QoS) aspects in the "3-Level Integrated Hybrid Optical Network "(3LIHON) architecture.

A simulation model is developed using a programming language called Simula and a context class for discrete event simulation called DEMOS. The result is an asynchronous and un-slotted model able to handle a variety of simulation's scenarios.

We consider the performance on the output side of a 3LIHON switching node with a given scheduling algorithm towards wavelength and our attention is on Statistically Multiplexed traffic.

Three study cases are considered to evaluate the network's performance.

The Series One is a complete study on the output side of a 3LIHON node varying the length of GST bursts and with fixed relative percentage of GST bursts, SM/RT and SM/BE packets. Results show that the PLP for SM/RT packets is quite good when the total load of the simulator is below 80%, and that when reducing the number of output wavelengths the performance becomes worse. The PLP of SM/BE shows again good results for total load around 80% however the behavior is quite bad for higher values. We demonstrate that when the total load is low SM/BE packets are mostly interrupted by GST bursts but when it is high by SM/RT packets. Observing the delay of SM/BE packets in the BE queue we show that the waiting time increases with the total load and when the GST bursts length increase so do the time in the queue.

The Series Two is a study varying the mix of relative percentage of services with fixed length value of GST bursts. The consideration about the PLP of SM/RT are similar to the first study case but we can observe a more specific behavior: when the relative percentage of GST bursts is high we have an improvement in PLP as the load increase over 80%. Observing the PLP of SM/BE results we can say that it increases when the percentage of GST bursts increases too. About the delay of SM/BE packets in the BE queue we observe a special behavior when the share of GST burst is high and it is the same for each GST burst length.

The Series Three is a complete study on the output side of 3LIHON node varying the length of SM/BE packets. Firstly we provide a result demonstration that the SM/BE traffic does not influence the SM/RT and GST traffics. Then we observe that the probability to interrupt a SM/BE packet increases with the length of the SM/BE and that the delay of SM/BE packets depends on the length of SM/BE packets. Finally we can say that the performance of the simulated model are quite good below 80% total load and sometimes are good also for higher values. In any case it is not recommendable to work above 90% total load because the risk to loose SM/RT packets is too high.

To sum up the implemented algorithm gives good performance on the output side of a 3LIHON switching node, for reasonable load values and studied traffic mixes.

Chapter 6

Further works

The proposed thesis opens many possible scenarios for further works.

The presented simulation model has not Fiber Delay Line (FDL) delay in the Optical Packet Switching (OPS) part of the architecture and has a single buffer with First In First Out (FIFO) policy. Moreover it does not implement a retransmission mechanism for the SM/BE packets that are interrupted.
It might be interesting to compare the currently results with N shared FDL buffers available in the OPS for a given output fiber and/or to consider the performance allowing link level retransmission of interrupted packets from the EPS.

Furthermore it would be interesting to study the network's behavior assigning priority levels to SM/BE packets. In this case we should consider more than one BE queue and implement different retransmission mechanism.

In the proposed work we consider fixed distributions to implement the packets generators, hence it would be interesting to vary those distributions and study the consequences on the network's performance.

Finally we should consider different relative percentage of share traffic and varying length and relative percentage of both the Statistically Multiplexed (SM) traffic types.

Appendix A

Confident Inteval

We have to evaluate the accuracy's level of our result after each simulation.

All the estimated parameters are stochastic variables characterized by mean value and by variance. If we repeat n times the same simulation we are going to obtain n different and independent observations. For this reason we must define a confident interval to evaluate the accuracy's level of result.

To study the performance of 3LIHON we need to introduce two estimators:

- Mean Value Estimator;
- Variance Estimator;

A.0.4 Mean Value Estimator

The parameter that we want evaluate is called x and its mean value is μ.
The independent observations that we obtain are $X_1, X_2, ..., X_n$.

An estimator of the mean value is the sample average defined like:

$$\overline{X}(n) = \frac{1}{n}\sum_{i=1}^{n} X_i$$

The obtained estimator is an stochastic variable so if we repeat few times the simulation we are going to find different $X(n)$ values. In general we can say that $X(n) \neq \mu$ and then we need to evaluate the confidence interval that is an interval around $X(n)$. In this way we are confident ("confident interval") that μ is in that range.

This obtained estimator of μ is not polarized:

$$E\{X(n)\} = \mu$$

The definition of the confident interval for the considered estimator is:

$$P\{|\overline{X}(n) - \mu| < \delta\} = 1 - \alpha$$

where δ is the half amplitude of the confident interval:

$$[\overline{X} - \delta; \overline{X} + \delta]$$

If $1 - \alpha = 0,95$ then we evaluate the accuracy of all simulations with a confidence interval of 95%.

It has gaussian distribution with mean value equal to 0 and variance equal to 1.

A.0.5 Variance Estimator

We can calculate an estimator also for the second moment.
The variance of of$X(n)$ is:

$$Var\{X(n)\} = \frac{\sigma^2}{n}$$

When the number of samples increases then the estimation of the parameter is better.
To estimate this parameter we can use the sample variance:

$$S^2(n) = \frac{1}{n-1}\sum_{i=1}^{n}[X_i - \overline{X}(n)]^2$$

Again we have a non polarize estimator.

$$Var[\overline{X}(n)] = \frac{S^2(n)}{n} = \frac{1}{n(n-1)}\sum_{i=1}^{n}[X_i - \overline{X}(n)]^2$$

A.0.6 Student's T Disrtibution

If the number of observations is high, e.g. $n > 30$, we can represented $X(n)$ by a gaussian distribution and introduce the stochastic variable Z_n:

$$Z_n = \frac{[\overline{X}(n) - \mu]}{\sqrt{\frac{\sigma^2}{n}}}$$

Z_n is a gaussian distribution with mean value equal to zero and variance equal to 1.

The Z_n distribution in represented in Figure A.1:

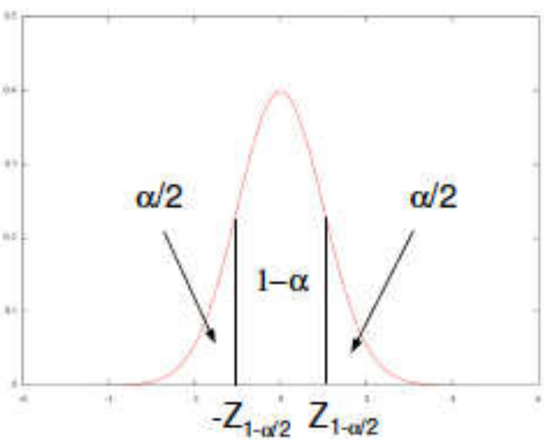

Figure A.1: Gaussian Distribution of Z_n

The value $z_{1-\alpha/2}$ is such that the integral under the curve between $-z_{1-\alpha/2}$ and $z_{1-\alpha/2}$ is equal to $1 - \alpha$:

$$P\{-z_{1-\alpha/2} \leq z \leq z_{1-\alpha/2}\} = 1 - \alpha$$

We can use $S^2(n)$ instead that σ^2 in the Z_n expression because we assumed that the number of observations n is big.
We obtain the follow expression:

$$P\{-z_{1-\alpha/2} \leq \frac{[\overline{X}(n) - \mu]}{\sqrt{\frac{\sigma^2}{n}}} \leq z_{1-\alpha/2}\}\} = P\{\overline{X}(n) - z_{1-\alpha/2}\sqrt{\frac{S^2(n)}{n}} \leq \mu$$

$$\leq \overline{X}(n) + z_{1-\alpha/2}\sqrt{\frac{S^2(n)}{n}}\} \cong 1 - \alpha$$

Then the half amplitude of the confident interval is:

$$\delta = z_{1-\alpha/2}\sqrt{\frac{S^2(n)}{n}}$$

If all the samples X_i have a normal distribution we can define the variable t_n like:

$$t_n = \frac{[\overline{X}(n) - \mu]}{\sqrt{\frac{\sigma^2}{n}}}$$

This distribution is called student's T with $n - 1$ degrees of freedom and the confident interval is exactly:

$$\delta = t_{n-1,1-\alpha/2}\sqrt{\frac{S^2(n)}{n}}$$

We called it confident interval t.
The values of the T-distribution for different values of n are represented in specific tables. When we use the confident interval we are making an approximation because is rare that the samples X_i have a normal distribution.

In our study we use a student's T distribution with confidence interval of 95%, then with $1-\alpha = 0.95$, and with $n = 10$ observations. The matching table is represented in Table A.1:

Table A.1: Student's T table

$n-1$	$t_{n-1,1-\alpha/2}$
1	12.706
2	4.303
3	3.182
4	2.776
5	2.571
6	2.447
7	2.365
8	2.306
9	**2.262**
10	2.228
11	2.201
12	2.179
13	2.160
14	2.145
15	2.131
16	2.120
17	2.110
18	2.101
19	2.093
20	2.086
21	2.080
22	2.074
23	2.069
24	2.064
25	2.060
26	2.056
27	2.052
28	2.048
29	2.045
30	2.042
40	2.021
50	2.009
75	1.992
100	1.984
∞	1.960

Appendix B

Extra Results

The purpose of this Appendix is to show same extra results that we do not include in the main work.

The Series Two presented in Chapter 4.4 is obtained by mixing four different relative traffic percentage. We obtain separately results for each of this study cases and then we mix into the Series Two.

The detailed results obtained per each of this cases are presented in Chapter 4.3 and in the following section.

B.1 Extra Series: GST 50% & RT 10% & BE 40%

In this section we show the results obtained for the follow mix of relative traffic flow:

- GST 50%
- SM/RT 10%
- SM/BE 40%

and for three different GST burst length:

- 40000 *bytes*

- 4000 *bytes*
- 400000 *bytes*

B.1.1 PLP of SM/RT

Firstly we show the Packet Loss Probability (PLP) of Statistically Multiplexed Real Time (SM/RT) traffic, showed in Figure B.1:

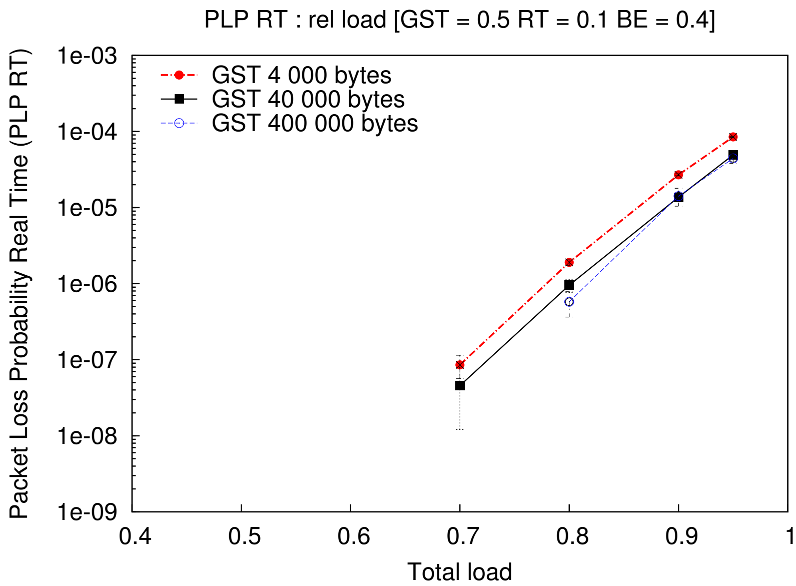

Figure B.1: PLP of SM/RT Extra Series GST = 0.5 SM/RT = 0.1 SM/BE = 0.3

B.1.2 PLP of SM/BE

Secondly we present the Packet Loss Probability (PLP) of Statistically Multiplexed Best Effort (SM/BE) traffic.

Figure B.2 shows the results for logarithmic scale:

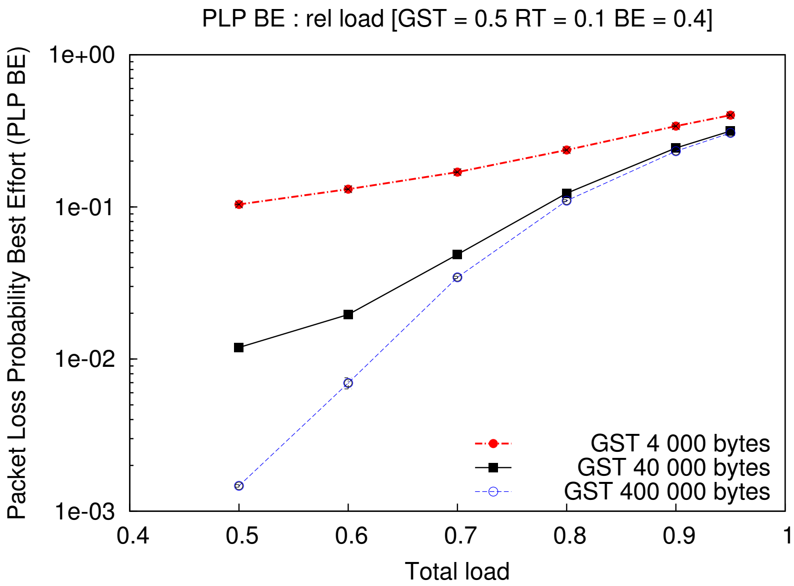

Figure B.2: PLP of SM/BE Extra Series GST = 0.5 SM/RT = 0.1 SM/BE = 0.3 using logarithmic scale

Figure B.3 shows the result for linear scale:

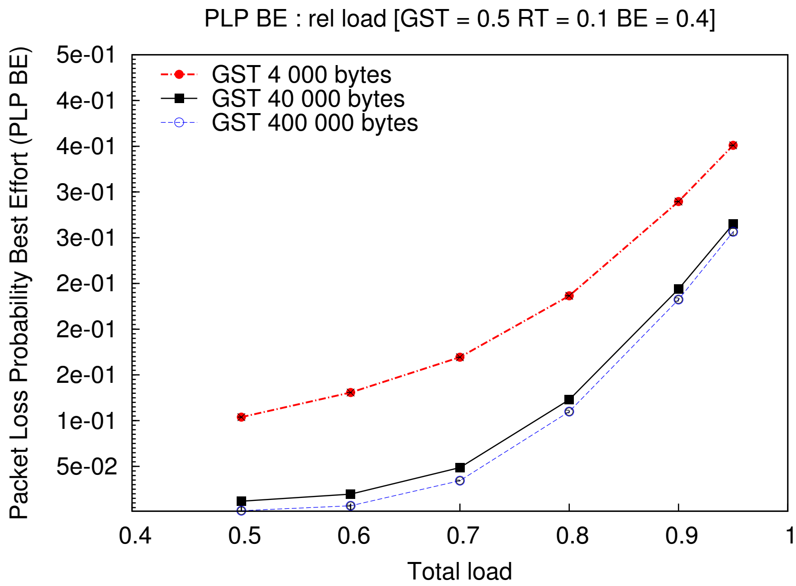

Figure B.3: PLP of SM/BE Extra Series GST = 0.5 SM/RT = 0.1 SM/BE = 0.3 using linear scale

B.1.3 Delay of SM/BE packets in BE queue

In the end we study the delay of the Statistically Multiplexed Best Effort (SM/BE) packets in the Best Effort (BE) queue.

The results for the mean value are shown in Figure B.4:

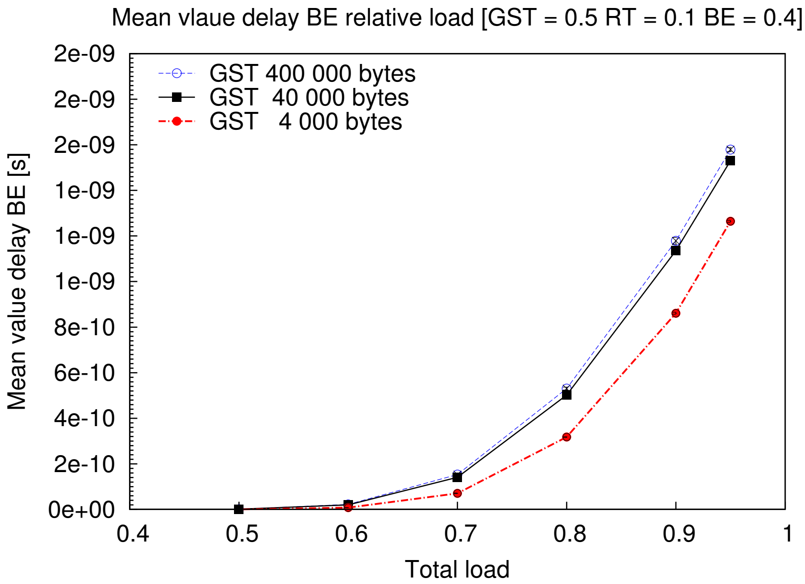

Figure B.4: Mean value delay of SM/BE in BE queue Extra Series GST = 0.5 SM/RT = 0.1 SM/BE = 0.3

The results for the mean variance are shown in Figure B.5:

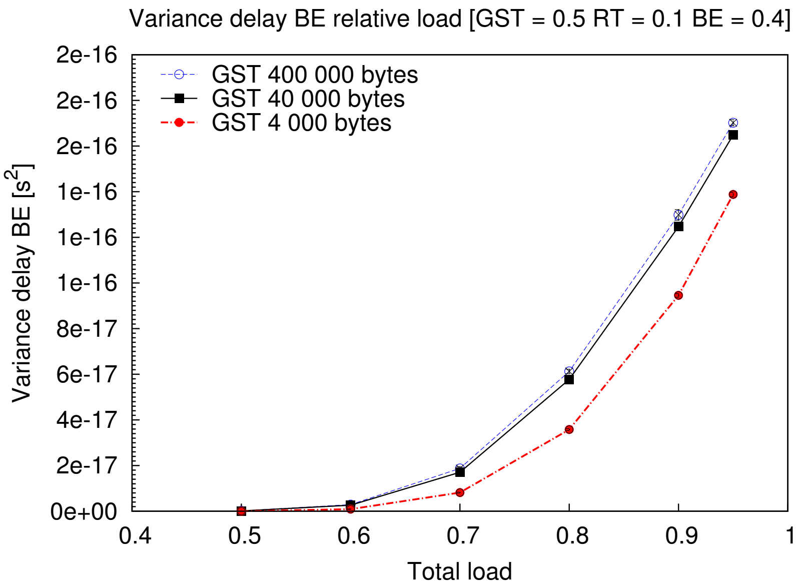

Figure B.5: Variance delay of SM/BE in BE queue Extra Series GST = 0.5 SM/RT = 0.1 SM/BE = 0.3

B.2 Extra Series: GST 30% & RT 10% & BE 60%

In this section we show the results obtained for the follow mix of relative traffic flow:

- GST 30%
- SM/RT 10%
- SM/BE 60%

and for three different GST burst length:

- 40000 *bytes*
- 4000 *bytes*
- 400000 *bytes*

B.2.1 PLP of SM/BE

We present the Packet Loss Probability (PLP) of Statistically Multiplexed Best Effort (SM/BE) traffic.
Figure B.6 shows the results for logarithmic scale:

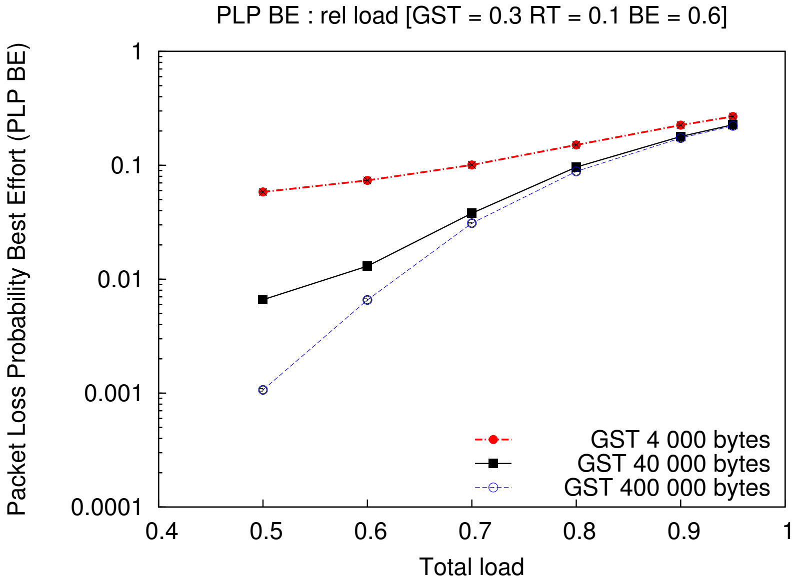

Figure B.6: PLP of SM/BE Extra Series GST = 0.3 SM/RT = 0.1 SM/BE = 0.6 using logarithmic scale

Figure B.7 shows the result for linear scale:

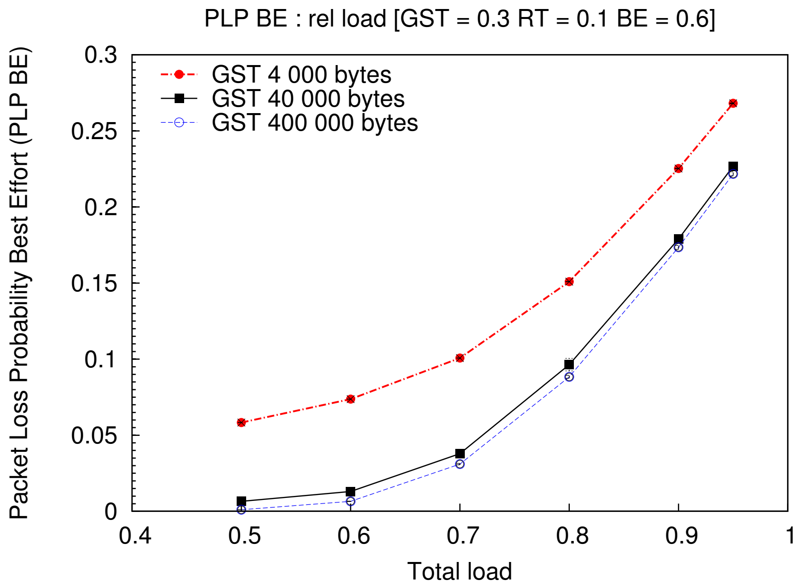

Figure B.7: PLP of SM/BE Extra Series GST = 0.3 SM/RT = 0.1 SM/BE = 0.6 using linear scale

B.2.2 Delay of SM/BE packets in BE queue

In the end we study the delay of the Statistically Multiplexed Best Effort (SM/BE) packets in the Best Effort (BE) queue.

The results for the mean value are shown in Figure B.8:

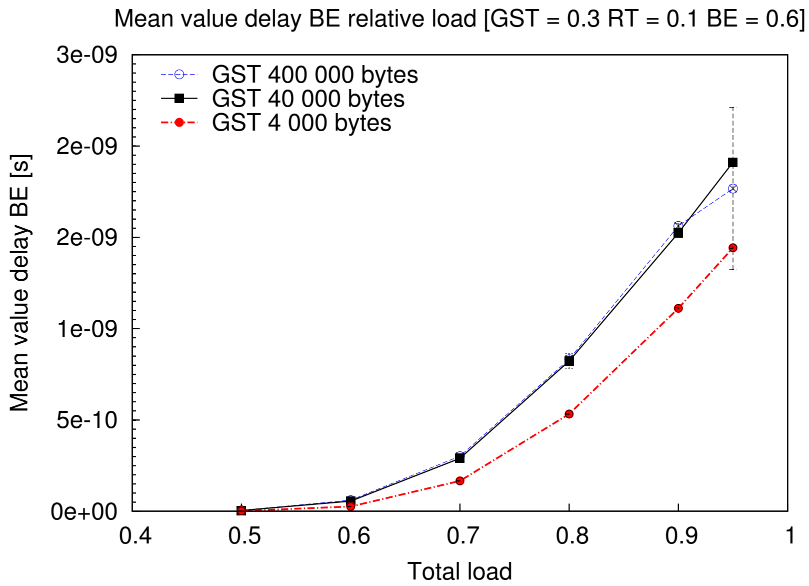

Figure B.8: Mean value delay of SM/BE in BE queue Extra Series GST = 0.3 SM/RT = 0.1 SM/BE = 0.6

The results for the mean variance are shown in Figure B.9:

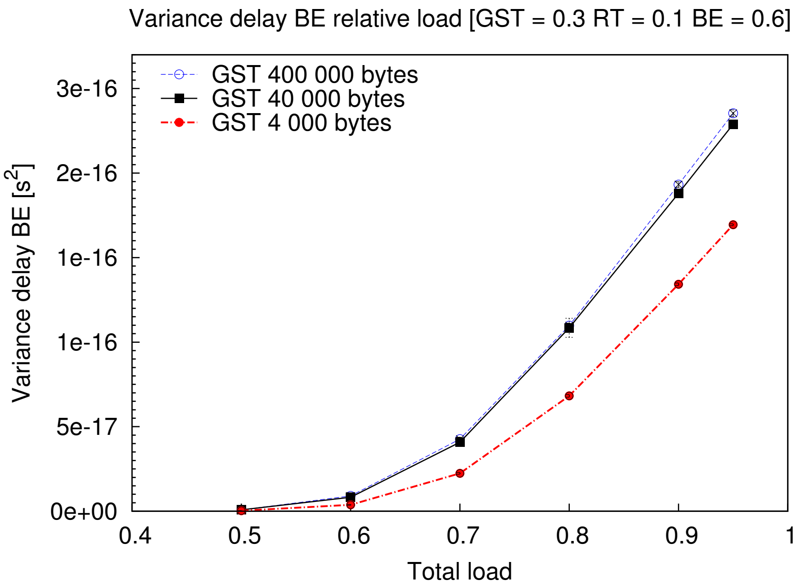

Figure B.9: Variance delay of SM/BE in BE queue Extra Series GST = 0.3 SM/RT = 0.1 SM/BE = 0.6

B.3 Extra Series: GST 80% & RT 10% & BE 10%

In this section we show the results obtained for the follow mix of relative traffic flow:

- GST 80%
- SM/RT 10%
- SM/BE 10%

and for three different GST burst length:

- 40000 *bytes*
- 4000 *bytes*
- 400000 *bytes*

B.3.1 PLP of SM/RT

We show the Packet Loss Probability (PLP) of Statistically Multiplexed Real Time (SM/RT) traffic, showed in Figure B.10:

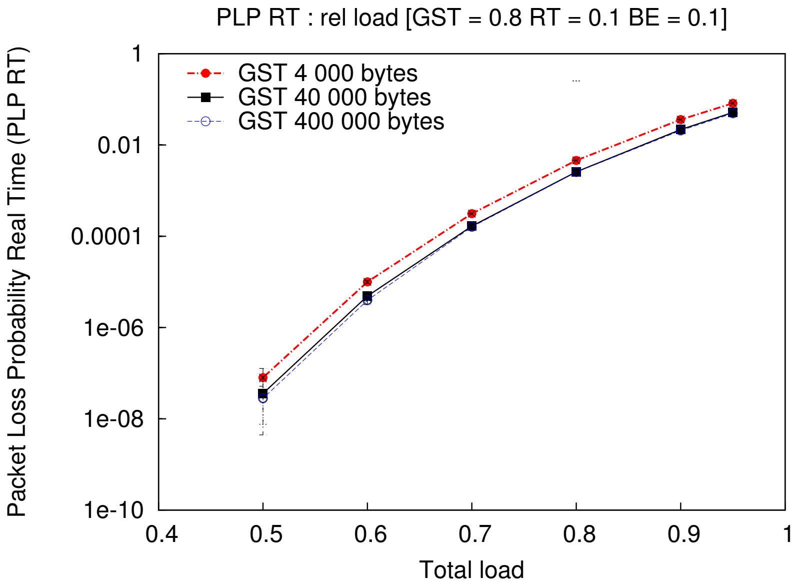

Figure B.10: PLP of SM/RT Extra Series GST = 0.8 SM/RT = 0.1 SM/BE = 0.1

B.3.2 PLP of SM/BE

We present the Packet Loss Probability (PLP) of Statistically Multiplexed Best Effort (SM/BE) traffic.

Figure B.11 shows the results for logarithmic scale:

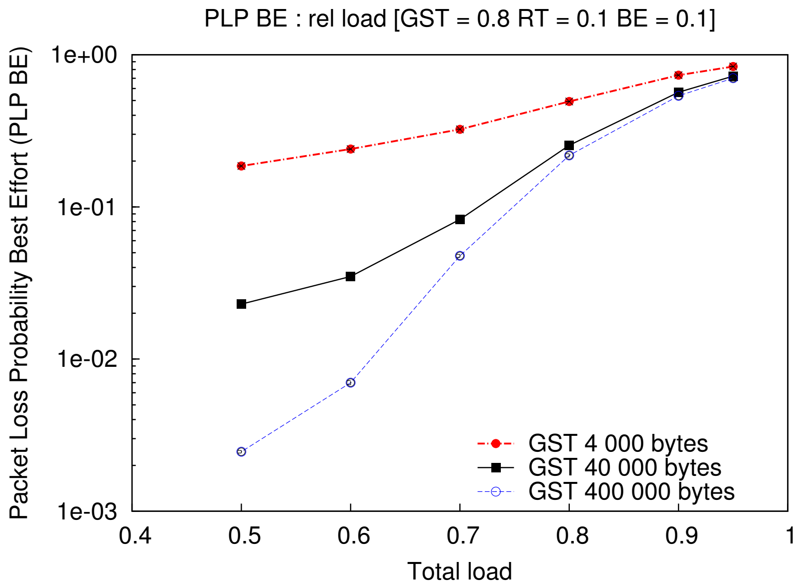

Figure B.11: PLP of SM/BE Extra Series GST = 0.8 SM/RT = 0.1 SM/BE = 0.1 using logarithmic scale

Figure B.12 shows the result for linear scale:

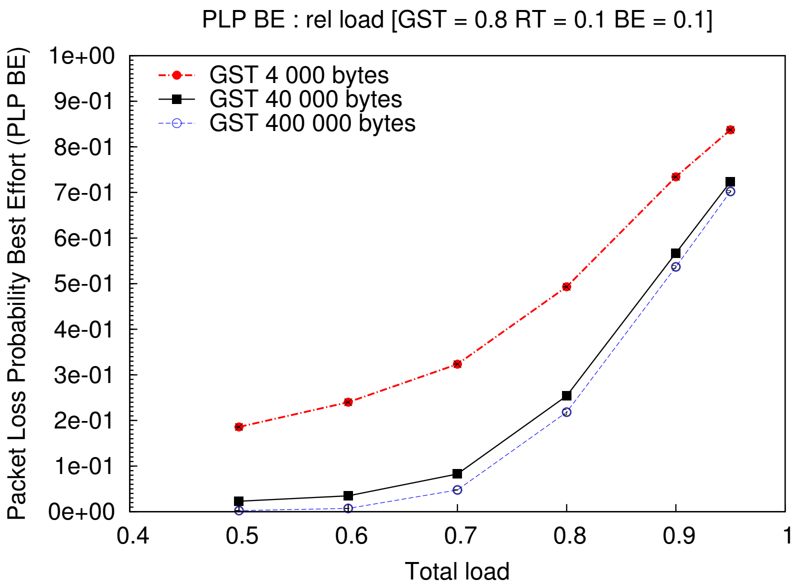

Figure B.12: PLP of SM/BE Extra Series GST = 0.8 SM/RT = 0.1 SM/BE = 0.1 using linear scale

Appendix C

Demos Code

```
BEGIN

integer conta_osservazioni;
long real array PLPrt(0:9);
long real array PLPrt2(0:9);
long real array PLPbe(0:9);
long real array PLPbe2(0:9);
long real PLP_RT, PLP_RT2;
long real PLP_BE, PLP_BE2;
long real msum, m2sum;
long real array msum_vettore(0:9);
long real array m2sum_vettore(0:9);
long real vsum, v2sum;
long real array vsum_vettore(0:9);
long real array v2sum_vettore(0:9);
long real percABS, perc2ABS;
long real array percABS_vettore(0:9), perc2ABS_vettore
    (0:9);
long real percREL1, perc2REL1;
long real array percREL1_vettore(0:9), perc2REL1_vettore
    (0:9);
long real percREL2, perc2REL2;
long real array percREL2_vettore(0:9), perc2REL2_vettore
    (0:9);

long real SUM_PLPrt, sum_meanPLPrt;
long real SUM_PLPrt2, sum_meanPLPrt2;
long real SUM_PLPbe, sum_meanPLPbe;
long real SUM_PLPbe2, sum_meanPLPbe2;
```

```
long real sum_msum_vettore, sum_m2sum_vettore;
long real sum_vsum_vettore, sum_v2sum_vettore;
long real sum_percABS_vettore, sum_perc2ABS_vettore;
long real sum_percREL1_vettore, sum_perc2REL1_vettore;
long real sum_percREL2_vettore, sum_perc2REL2_vettore;

long real deviazione_std_rt;
long real deviazione_std_be;
long real deviazione_std_delay1;
long real deviazione_std_delay;
long real deviazione_std_percABS,
    deviazione_std_percREL1, deviazione_std_percREL2;
long real ConfInter95rt;
long real ConfInter95be;
long real ConfInter95_delay1;
long real ConfInter95_delay;
long real ConfInter95_percABS, ConfInter95_percREL1,
    ConfInter95_percREL2;
long real lower_rt;
long real lower_be;
long real lower_delay1;
long real lower_delay;
long real lower_percABS, lower_percREL1, lower_percREL2;
long real upper_rt;
long real upper_be;
long real upper_delay1;
long real upper_delay;
long real upper_percABS, upper_percREL1, upper_percREL2;
long real adjustDELTA_rt;
long real adjustDELTA_be;
long real adjustDELTA_delay1;
long real adjustDELTA_delay;
long real adjustDELTA_percABS, adjustDELTA_percREL1,
    adjustDELTA_percREL2;

for conta_osservazioni := 0 step 1 until 9 do

BEGIN
   !****************************************;
   !************INPUT DATA*****************;
   !****************************************;

         long real sim_time;
         long real transient_time;
         integer OUTPUT_WL;
```

```
        long real FDL, FDLinBit;
        long real bitChannelRate;
        long real length_GST, mean_length_RT,
           mean_length_BE;
        long real max_mean_length_RT,
           max_mean_length_BE;
        long real TOTAL_LOAD;
        long real GSTx100_TOTAL, RTx100_TOTAL,
           BEx100_TOTAL;
        long real RELATIVExcentGST, RELATIVExcentBE,
           RELATIVExcentRT;
        long real A, B;
        long real num_pRT_sec, num_pBE_sec;

        integer array my_seeds(0:9);

        long real PLP_tot;

        transient_time:= 0.25;
        ![s];

        sim_time := 5.0;
        ![s];

        OUTPUT_WL := 32;
        !number of fiber;

        mean_length_RT := 40*8;
        ![bits];

        bitChannelRate := 1*10**9;
        ![bit/s];

        FDL := 5*mean_length_RT/bitChannelRate;
        ![sec];

        FDLinBIT := 5*mean_length_RT;
        ![bits];

        length_GST := 1000*mean_length_RT;
        ![bits];

        max_mean_length_RT:= 5*mean_length_RT;
        ![bits];

```

```
        mean_length_BE:= 40*mean_length_RT;
        ![bits];

        TOTAL_LOAD:= 0.5;

        RELATIVExcentGST := 0.6;
        RELATIVExcentRT:= 0.1;
        RELATIVExcentBE:= 0.3;

        A:= (RELATIVExcentGST*TOTAL_LOAD*bitChannelRate
           )/length_GST;
        B:=(1/A-length_GST/bitChannelRate);

        num_pRT_sec:= (RELATIVExcentRT*TOTAL_LOAD*
           bitChannelRate)/mean_length_RT;  ![Rate];
        num_pBE_sec:= (RELATIVExcentBE*TOTAL_LOAD*
           bitChannelRate)/mean_length_BE;  ![Rate];

        GSTx100_TOTAL:= RELATIVExcentGST*TOTAL_LOAD;
        RTx100_TOTAL:= RELATIVExcentRT*TOTAL_LOAD;
        BEx100_TOTAL:= RELATIVExcentBE*TOTAL_LOAD;

        my_seeds(0):= 907;
        my_seeds(1):= 234;
        my_seeds(2):= 326;
        my_seeds(3):= 104;
        my_seeds(4):= 711;
        my_seeds(5):= 523;
        my_seeds(6):= 883;
        my_seeds(7):= 113;
        my_seeds(8):= 417;
        my_seeds(9):= 656;

        outtext("*******INPUT DATA*******"); outimage;
        outtext("SIMULATION TIME [s]");
        outfix(sim_time,2,20);
        outimage;
        outtext("TRANSIENT TIME [s]");
        outfix(transient_time,2,20);
        outimage;
        outtext("NUMBER OF OUTPUT WAVELENGTH");
        outfix(OUTPUT_WL,0,20);
        outimage;
        outtext("BIT CHANNEL RATE [bit/s]");
        outfix(bitChannelRate,0,40);
```

```
          outimage;

          outtext("GST [bytes]");
          outfix(length_GST/8,0,20);
          outimage;
          outtext("RT [bytes]");
          outfix(mean_length_RT/8,0,20);
          outimage;
          outtext("BE [bytes]");
          outfix(mean_length_BE/8,0,20);
          outimage;

          outtext("TOTAL LOAD");
          outfix(TOTAL_LOAD,2,20);
          outimage;
          outtext("RELATIVE GST LOAD");
          outfix(RELATIVExcentGST,2,20);
          outimage;
          outtext("RELATIVE RT LOAD");
          outfix(RELATIVExcentRT,2,20);
          outimage;
          outtext("RELATIVE BE LOAD");
          outfix(RELATIVExcentBE,2,20);
          outimage;

!*****************************************************;

BEGIN

EXTERNAL CLASS demos="/Users/gaialeli/cim-3.37/demos.atr
    ";

DEMOS

 BEGIN
 integer k,p,i,u;
 integer cont_res1, cont_res2;
 integer finish, BE_delay;
 integer numberBE, totalBE;

 integer array counter_GSTpacket_generate(1:OUTPUT_WL);
 integer array counter_GSTpacket_successful(1:OUTPUT_WL)
    ;
 integer GSTp_intoWL,  GSTp_outtoWL;
 integer GST_tot, GST_successful;
```

```
long real gst_insec, gst_insec_tot;

integer counter_RTpacket_generate,
   counter_RTpacket_successful;
integer RT_wl_directly_free, RT_wl_busy;
integer counter_RTpacket_lost;
integer RTp_intoWL,  RTp_outtoWL;
long real rt_insec, rt_insec_tot;

integer counter_BEpacket_generate,
   counter_BEpacket_successful;
integer counter_BEpacket_interrupted,
   counter_BEpacket_interrupted_by_GSTp;
integer counter_BEpacket_interrupted_by_RTp;
integer BEp_inQ;
integer BEp_intoWL,  BEp_outtoWL;
long real be_insec, be_insec_tot;
integer counter_BEpacket;

long real PLP_GST;
integer TOTAL_P_GENERATED, TOTAL_P_intoWL;
long real somma, somma2, delayBE;

!***GSTtraffic: DISTRIBUTION;
ref(rdist) array         nextGST(1:OUTPUT_WL);
ref(rdist) array         lengthGST(1:OUTPUT_WL);

!***SMRTtraffic: DISTRIBUTION;
ref(rdist)               nextSMRT;
ref(rdist)               lengthSMRT;

!***SMBEtraffic: DISTRIBUTION;
ref(rdist)               nextSMBE;
ref(rdist)               lengthSMBE;

ref(res) array           wavelengthOUT(1:OUTPUT_WL);
ref(waitQ)               BE_Q;
ref(bin)                 serverBEwaiting_wl;

integer array            flag(1:OUTPUT_WL);
ref(BEserver)            BE_S;
ref(SMBEpacket) array    SMBEpacket_pointer(1:OUTPUT_WL
   );

boolean array FDL_in_use(1:OUTPUT_WL);
```

```
!******************************************************;
!               GST PACKETS GENERATOR                  ;
!******************************************************;

entity class GSTgen(p); integer p;

BEGIN

 long real Lgst;
 ref(GSTpacket) p_GSTp;

 CICLO:
  counter_GSTpacket_generate(p) :=
     counter_GSTpacket_generate(p) + 1;
  Lgst:= lengthGST(p).sample;
  p_GSTp:- new GSTpacket("GSTpacket   ", p);
  p_GSTp.Lgst:=Lgst;
  p_GSTp.schedule(0.0);
  hold(Lgst/bitChannelRate + nextGST(p).sample);
 goto CICLO;

END***GSTgen(p)***;

!******************************************************;
!                     GST PACKET                       ;
!******************************************************;

entity class GSTpacket(p); integer p;

BEGIN

 long real Lgst;
 ref(SMBEpacket) BE_p;

  flag(p):= 1;
  FDL_in_use(p) := true;

  if Lgst/bitChannelRate >= FDL then
  BEGIN
   hold(FDL);
   if SMBEpacket_pointer(p)=/=NONE then
    BEGIN
     BE_P:-SMBEpacket_pointer(p);
     BE_P.interrupt(1);
```

```
    END;
   wavelengthOUT(p).acquire(1);
   GSTp_intoWL := GSTp_intoWL + 1;
   hold(Lgst/bitChannelRate-FDL);

   FDL_in_use(p) := false;
   hold(FDL);
  END
  else
  BEGIN
   hold(FDL-Lgst/bitChannelRate);
   FDL_in_use(p) := false;
   hold(Lgst/bitChannelRate);
   if SMBEpacket_pointer(p)=/=NONE then
    BEGIN
     BE_P:-SMBEpacket_pointer(p);
     BE_P.interrupt(1);
    END;
   wavelengthOUT(p).acquire(1);
   GSTp_intoWL := GSTp_intoWL + 1;
   hold(Lgst/bitChannelRate);
  END;
  wavelengthOUT(p).release(1);
  gst_insec:= Lgst/bitChannelRate;
  gst_insec_tot:= gst_insec_tot+gst_insec;
  GSTp_outtoWL := GSTp_outtoWL  + 1;
  if not FDL_in_use(p) then flag(p):=0;
  serverBEwaiting_wl.give(1);
  counter_GSTpacket_successful(p):=
     counter_GSTpacket_successful(p)+1;

END***GSTpacket***;

!*******************************************************;
!                 SMRT PACKETS GENERATOR               ;
!*******************************************************;
entity class SMRTgen;

BEGIN

long real Lsmrt;
ref(SMRTpacket) p_SMRTp;

CICLO:
```

```
 counter_RTpacket_generate := counter_RTpacket_generate
    + 1;
 Lsmrt:= lengthSMRT.sample;
 p_SMRTp :- new SMRTpacket("SMRTpacket   ");
 if Lsmrt > max_mean_length_RT then Lsmrt :=
    max_mean_length_RT;

 p_SMRTp.Lsmrt := Lsmrt;
 p_SMRTp.schedule(0.0);
 hold(nextSMRT.sample);
goto CICLO;

END***SMRTgen***;


!*****************************************************;
!                    SMRT PACKET                      ;
!*****************************************************;

entity class SMRTpacket;

BEGIN
 long real Lsmrt;
 integer interrupt_wl, i;
 ref(SMBEpacket) BE_p;
 counter_BEpacket:= counter_BEpacket+1;

 for i:= 1 step 1 until OUTPUT_WL do
 BEGIN
  if flag(i) = 0  then BEGIN
                      RT_wl_directly_free :=
                         RT_wl_directly_free + 1;
                      goto USE_FREE_WL;
                      END;

  if flag(i) = 3  AND interrupt_wl = 0 then interrupt_wl
      := i;

 END;

 if interrupt_wl > 0 then
  BEGIN
   BE_P :- SMBEpacket_pointer(interrupt_wl);
   BE_P.interrupt(2);
   i := interrupt_wl;
```

```
    RT_wl_busy := RT_wl_busy + 1;
    goto USE_FREE_WL;
   END
   else
   BEGIN
   counter_RTpacket_lost:= counter_RTpacket_lost +1 ;
   END;
  goto FINE;

  USE_FREE_WL:
   wavelengthOUT(i).acquire(1);
   RTp_intoWL := RTp_intoWL + 1;
   flag(i) := 2;
   hold(Lsmrt/bitChannelRate);
   wavelengthOUT(i).release(1);
   rt_insec:= Lsmrt/bitChannelRate;
   rt_insec_tot := rt_insec_tot+rt_insec;
   RTp_outtoWL := RTp_outtoWL + 1;
   if flag(i) = 2 then flag(i) := 0;
   serverBEwaiting_wl.give(1);
   counter_RTpacket_successful:=
      counter_RTpacket_successful+1;
  FINE:

END***SMRTpacket***;

!*****************************************************;
!                 SMBE PACKETS GENERATOR             ;
!*****************************************************;

entity class SMBEgen;

BEGIN

 ref(SMBEpacket) p_SMBEp;
 long real Lsmbe;

   CICLO:
    counter_BEpacket_generate :=
       counter_BEpacket_generate + 1;
    Lsmbe:= lengthSMBE.sample;
    p_SMBEp :- new SMBEpacket("SMBEpacket   ");
    p_SMBEp.Lsmbe := Lsmbe;
    p_SMBEp.schedule(0.0);
    hold(nextSMBE.sample);
```

```
  goto CICLO;

END***SMBEgen***;

!***************************************************;
!                    SMBE PACKET                   ;
!***************************************************;

entity class SMBEpacket;

BEGIN

 integer actual_wl;
 long real Lsmbe;
 long real start_BEp;

 serverBEwaiting_wl.give(1);
 start_BEp := time;
 BE_Q.wait;
 delayBE := time - start_BEp;
 somma := somma + delayBE;
 somma2:=somma2+delayBE**2;
 flag(actual_wl) := 3;
 SMBEpacket_pointer(actual_wl) :- this SMBEpacket;
 wavelengthOUT(actual_wl).acquire(1);
 BEp_intoWL := BEp_intoWL + 1;
 hold(Lsmbe/bitChannelRate);
 wavelengthOUT(actual_wl).release(1);
 be_insec := Lsmbe/bitChannelRate;
 be_insec_tot := be_insec_tot+be_insec;
 BEp_outtoWL := BEp_outtoWL + 1;
 if interrupted = 0
 then
  BEGIN
   serverBEwaiting_wl.give(1);
   counter_BEpacket_successful:=
      counter_BEpacket_successful+1;
  END;
  if flag(actual_wl) = 3 then
  flag(actual_wl):= 0;

 SMBEpacket_pointer(actual_wl) :- NONE;
 if interrupted > 0 then counter_BEpacket_interrupted :=
     counter_BEpacket_interrupted + 1;
```

```
 if interrupted = 1 then
    counter_BEpacket_interrupted_by_GSTp :=
    counter_BEpacket_interrupted_by_GSTp + 1;
 if interrupted = 2 then
    counter_BEpacket_interrupted_by_RTp  :=
    counter_BEpacket_interrupted_by_RTp + 1;

 END;

!*******************************************************;
!                   SMBE SERVER                         ;
!*******************************************************;

entity class BEserver;

 BEGIN
 integer i;
 ref(SMBEpacket) p_SMBEpacket;

 INIZIO:
  for i:= 1 step 1 until OUTPUT_WL do

     if flag(i) = 0 then if BE_Q.length > 0 then

                         BEGIN
                          p_SMBEpacket :- BE_Q.coopt;
                          p_SMBEpacket.actual_wl := i;
                          p_SMBEpacket.schedule(0.0);
                          BEp_inQ:= BEp_inQ + 1;
                        END;

     serverBEwaiting_wl.take(1);

 goto INIZIO;

 END;

!*******************************************************;
!*******************MAIN*******************************;
!*******************************************************;
outtext("*********************************"); outimage;
outtext("                                 "); outimage;
outtext("                                 "); outimage;

setseed(my_seeds(conta_osservazioni));
```

```
outtext("OBSERVATION NUMBER");
outfix(conta_osservazioni , 0, 5);
outimage;
outtext("SEED:");
outfix(my_seeds(conta_osservazioni), 0, 20);
outimage;

outtext("                                  "); outimage;
outtext("                                  "); outimage;
outtext("**********************************"); outimage;

!***GST: PACKET'S LENGTH;
for k := 1 step 1 until OUTPUT_WL do
lengthGST(k)         :- new Constant("L GSTp",
   length_GST);

for k := 1 step 1 until OUTPUT_WL do
!***GST: NEXT PACKET;
nextGST(k)           :- new NegExp("NextGSTp", 1/B);

!***SMRT: PACKET'S LENGTH;
lengthSMRT           :- new NegExp("L SMRTp", 1/
   mean_length_RT);

!***SMRT: NEXT PACKET;
nextSMRT            :- new NegExp("NextSMRTp",
   num_pRT_sec*OUTPUT_WL);

!***SMBE: PACKET'S LENGTH;
lengthSMBE          :- new NegExp("L SMBEp", 1/
   mean_length_BE);

!***SMBE: NEXT PACKET;
nextSMBE            :- new NegExp("NextSMBEp",
   num_pBE_sec*OUTPUT_WL);

!WAVELENGTH OUT;
for k := 1 step 1 until OUTPUT_WL do
wavelengthOUT(k) :- new res(edit("wlOUT  ", k), 1);

!FLAG;
for k := 1 step 1 until OUTPUT_WL do flag(k) := 0;

BE_Q :- new waitQ("BEtQueue");
serverBEwaiting_wl :- new bin("ServBE wait", 0);
```

```

for k := 1 step 1 until OUTPUT_WL do
new GSTgen("GSTgen", k).schedule(nextGST(k).sample);

new SMRTgen("RTgen").schedule(nextSMRT.sample);

new SMBEgen("BEgen").schedule(nextSMBE.sample);

new BEserver("BEserver").schedule(0.0);

hold(transient_time);
reset;

!***************reset****************;

BEGIN
for cont_res1:= 1 step 1 until OUTPUT_WL do
 counter_GSTpacket_generate(cont_res1):=0;
END;

BEGIN
for cont_res2:= 1 step 1 until OUTPUT_WL do
 counter_GSTpacket_successful(cont_res2):=0;
END;

GSTp_intoWL:=0;
GSTp_outtoWL:=0;
GST_tot:=0;
GST_successful:=0;
gst_insec:=0.0;
gst_insec_tot:=0.0;

counter_RTpacket_generate:=0;
counter_RTpacket_successful:=0;
RT_wl_directly_free:=0;
RT_wl_busy:=0;
counter_RTpacket_lost:=0;
RTp_intoWL:=0;
RTp_outtoWL:=0;
rt_insec:=0.0;
rt_insec_tot:=0.0;
PLP_RT:=0.0;
PLP_RT2:=0.0;

finish:=0.0;
```

```
BE_delay:=0.0;
numberBE:=0;
totalBE:=0;
counter_BEpacket_generate:=0;
counter_BEpacket_successful:=0;
counter_BEpacket_interrupted:=0;
counter_BEpacket_interrupted_by_GSTp:=0;
counter_BEpacket_interrupted_by_RTp:=0;
BEp_inQ:=0;
BEp_intoWL:=0;
BEp_outtoWL:=0;
be_insec:=0.0;
be_insec_tot:=0.0;
counter_BEpacket:=0;
somma:=0.0;
somma2:=0.0;
delayBE:=0.0;
PLP_BE:=0.0;
PLP_BE2:=0.0;
msum:=0.0;
m2sum:=0.0;
vsum:=0.0;
v2sum:=0.0;
percABS:=0.0;
percREL1:=0.0;
percREL2:=0.0;
perc2ABS:=0.0;
perc2REL1:=0.0;
perc2REL2:=0.0;

TOTAL_P_GENERATED:=0;
TOTAL_P_intoWL:=0;

!************end reset****************;

for u:=1 step 1 until 100 do
begin
    hold(sim_time/100.0);
end;

for i:= 1 step 1 until OUTPUT_WL do
BEGIN
  GST_tot:= GST_tot + counter_GSTpacket_generate(i);
END;
```

```
for i:= 1 step 1 until OUTPUT_WL do
BEGIN
  GST_successful:= GST_successful +
     counter_GSTpacket_successful(i);
END;

PLP_RT:= counter_RTpacket_lost/counter_BEpacket;
PLP_RT2:=PLP_RT**2;

PLP_BE:= counter_BEpacket_interrupted/BEp_inQ;
PLP_BE2:=PLP_BE**2;

TOTAL_P_GENERATED:= GST_TOT + counter_BEpacket_generate
   + counter_RTpacket_generate;
TOTAL_P_intoWL:= GSTp_intoWL + counter_BEpacket +
   BEp_inQ;

!********************PRINT*************************;
!                   GST traffic                    ;
!**************************************************;
outtext("                                   ");
   outimage;
outtext("**************GST*****************");
   outimage;

outtext("GSTpacket Generate per each GSTsource: ");
outimage;
for i:= 1 step 1 until OUTPUT_WL do
BEGIN
  outint(i,2); outint(counter_GSTpacket_generate(i), 16)
     ;
  outimage;
END;
outimage;

outtext("GSTpacket successful per each GSTsource: ");
outimage;
for i:= 1 step 1 until OUTPUT_WL do
BEGIN
  outint(i,2); outint(counter_GSTpacket_successful(i),
     16);
  outimage;
END;
outimage;
```

```
outtext("GST tot generate : ");
outfix(GST_tot, 0, 20);
outimage;

outtext("GST Successful: ");
outfix(GST_successful, 0, 20);
outimage;

outtext("Generate GST packets that are not succesful: ")
   ;
outfix(GST_tot-GST_successful, 0, 20);
outimage;

outtext("GST packets that are in the FDL: ");
outfix(GST_tot-GSTp_intoWL, 0, 20);
outimage;

outtext("GSPps that are actively using wavelengths when
   the simulation stops  : ");
outfix(GSTp_intoWL-GSTp_outtoWL, 0, 20);
outimage;

outtext("GSTp that acquire WL : ");
outfix(GSTp_intoWL, 0, 20);
outimage;

outtext("GSTp that release WL = GST Successful :");
outfix(GSTp_outtoWL, 0, 20);
outimage;

outtext("sec to transmit all GSTp :");
outfix(gst_insec_tot,16,20);
outimage;
outtext("sec to transmit all GSTp ratio number of
   OUTPUT_WL :");
outfix((gst_insec_tot/OUTPUT_WL)/sim_time, 16, 20);
outimage;


!**********************PRINT*************************;
!                     RT traffic                      ;
!***************************************************;
outtext("                                    ");
   outimage;
```

```
outtext("***************RT*****************");
   outimage;

outtext("RTpacket generate : ");
outfix(counter_RTpacket_generate, 0, 20);
outimage;

outtext("RTpacket Successful : ");
outfix(counter_RTpacket_successful, 0, 20);
outimage;

outtext("Generate RT packets that are not successful: ")
   ;
outfix(counter_RTpacket_generate-
   counter_RTpacket_successful, 0, 20);
outimage;

outtext("RT packets that are crossing the system : ");
outfix(counter_RTpacket_generate-RTp_intoWL, 0, 20);
outimage;

outtext("RTps that are actively using wavelengths when
   the simulation stops  : ");
outfix(RTp_intoWL-RTp_outtoWL, 0, 20);
outimage;

outtext("RTp that acquire WL: ");
outfix(RTp_intoWL, 0, 20);
outimage;

outtext("RTp that release WL = RT Successful: ");
outfix(RTp_outtoWL, 0, 20);
outimage;

outtext("RT lost packets: ");
outfix(counter_RTpacket_lost, 0, 20);
outimage;

outtext("RTp that find a free WL: ");
outfix(RT_wl_directly_free, 0, 20);
outimage;

outtext("RTp that find a busy WL in the beginning = BE
   interrupted by RT: ");
outfix(RT_wl_busy, 0, 20);
```

```
outimage;

outtext("PLP RT : ");
outfix(PLP_RT, 16, 20);
outimage;

outtext("sec to transmit all RTp: ");
outfix(rt_insec_tot,16,20);
outimage;
outtext("sec to transmit all RTp ratio number of
   OUTPUT_WL :");
outfix((rt_insec_tot/OUTPUT_WL)/sim_time,16,20);
outimage;


!**************************************************;
!**********************PRINT***********************;
!                       BE traffic                ;
!**************************************************;
outtext("                                      ");
   outimage;
outtext("*****************BE***************");
   outimage;

outtext("BEpacket generate : ");
outfix(counter_BEpacket_generate, 0, 20);
outimage;

outtext("BEpacket Successful : ");
outfix(counter_BEpacket_successful, 0, 20);
outimage;

outtext("Generate BE packets that are not successful: ")
   ;
outfix(counter_BEpacket_generate-
   counter_BEpacket_successful, 0, 20);
outimage;

outtext("BEp_inQ = BEp into the queue = BEintoWL :");
outfix(BEp_inQ, 0,20);
outimage;

outtext("BEp that acquire WL : ");
outfix(BEp_intoWL, 0, 20);
outimage;
```

```

outtext("BEp that release WL = BE Successful:");
outfix(BEp_outtoWL , 0, 20);
outimage;

outtext("BEps that are actively using wavelengths when
   the simulation stops  : ");
outfix(BEp_intoWL-BEp_outtoWL , 0, 20);
outimage;

outtext("BEpacket Interrupt = BEpacket Interrupt by GST
   + BEpacket Interrupt byRT : ");
outfix(counter_BEpacket_interrupted, 0, 20); outimage;

outtext("BEpacket Interrupt by GST : ");
outfix(counter_BEpacket_interrupted_by_GSTp, 0, 20);
outimage;

outtext("BEpacket Interrupt by RT : ");
outfix(counter_BEpacket_interrupted_by_RTp, 0, 20);
outimage;

outtext("BEpacket Interrupt by GST + BEpacket Interrupt
   byBE : ");
outfix(counter_BEpacket_interrupted_by_RTp+
   counter_BEpacket_interrupted_by_GSTp, 0, 20);
outimage;

outtext("PLP BE : ");
outfix(PLP_BE, 16, 20);
outimage;

outtext("BEinterrupted ratio BEinQueue:");
outfix(counter_BEpacket_interrupted/BEp_inQ, 16, 20);
outimage;

outtext("BEinterrupted by GST ratio BEinQueue:");
outfix(counter_BEpacket_interrupted_by_GSTp/BEp_inQ, 16,
    20);
outimage;

outtext("BEinterrupted by RT ratio BEinQueue:");
outfix(counter_BEpacket_interrupted_by_RTp/BEp_inQ, 16,
   20);
outimage;
```

```
outtext("sec to transmit all BEp :");
outfix(be_insec_tot ,16 ,20);
outimage;
outtext("sec to transmit all BEp ratio number of
   OUTPUT_WL :");
outfix((be_insec_tot/OUTPUT_WL)/sim_time,16,20);
outimage;

outtext("Sum all delay BE packets : ");
outfix(somma, 16, 20);
outimage;

outtext("Sum**2 all delay BE packets : ");
outfix(somma2, 16, 20);
outimage;

outtext("Mean value delay BE packet : ");
outfix(somma/BEp_inQ, 16, 20);
outimage;

msum:=msum+somma/BEp_inQ;
vsum:=vsum+abs(BEp_inQ*somma2-somma**2.0)/(BEp_inQ*(
   BEp_inQ-1.0));
percABS := percABS + (counter_BEpacket_interrupted/
   BEp_intoWL)*100.0;
percREL1 := percREL1 + (
   counter_BEpacket_interrupted_by_GSTp/
   counter_BEpacket_interrupted)*100.0;
percREL2 := percREL2 + (
   counter_BEpacket_interrupted_by_RTp/
   counter_BEpacket_interrupted)*100.0;

m2sum:=m2sum+(somma/BEp_inQ)**2;
v2sum:= v2sum+(abs(BEp_inQ*somma2-somma**2.0)/(BEp_inq*(
   BEp_inQ-1.0)))**2.0;

perc2ABS := perc2ABS + ((counter_BEpacket_interrupted/
   BEp_intoWL)*100.0)**2.0;
perc2REL1 := perc2REL1 + ((
   counter_BEpacket_interrupted_by_GSTp/
   counter_BEpacket_interrupted)*100.0)**2.0;
perc2REL2 := perc2REL2 + ((
   counter_BEpacket_interrupted_by_RTp/
   counter_BEpacket_interrupted)*100.0)**2.0;
```

```
outtext("percABS");
outfix(percABS, 20, 45);
outimage;

outtext("percREL1");
outfix(percREL1, 20, 45);
outimage;

outtext("percREL2");
outfix(percREL2, 20, 45);
outimage;

outtext("                                     ");
   outimage;
outtext("*************************************");
   outimage;

outtext("TOTAL PACKETS GENERATED : ");
outfix(TOTAL_P_GENERATED, 0, 20);
outimage;

outtext("TOTAL PACKETS into the WL : ");
outfix(TOTAL_P_intoWL, 0, 20);
outimage;

outtext("GST% : ");
outfix(GSTx100_TOTAL, 5, 20);
outimage;
outtext("GST% simulation result : ");
outfix((gst_insec_tot/OUTPUT_WL)/sim_time, 16, 20);
outimage;

outtext("BE% : ");
outfix(BEx100_TOTAL, 5, 20);
outimage;
outtext("BE% simulation result : ");
outfix((be_insec_tot/OUTPUT_WL)/sim_time, 16, 20);
outimage;

outtext("RT% : ");
outfix(RTx100_TOTAL, 5, 20);
outimage;
outtext("RT% simulation result: ");
outfix((rt_insec_tot/OUTPUT_WL)/sim_time, 16, 20);
```

```
outimage;


!****************************************;

END***DEMOS***;

END;

PLPrt(conta_osservazioni):= PLP_RT;
PLPrt2(conta_osservazioni):= PLP_RT2;
sum_PLPrt := sum_PLPrt+PLPrt(conta_osservazioni);
sum_PLPrt2 := sum_PLPrt2+PLPrt2(conta_osservazioni);
PLPbe(conta_osservazioni):= PLP_BE;
PLPbe2(conta_osservazioni):= PLP_BE2;
sum_PLPbe := sum_PLPbe+PLPbe(conta_osservazioni);
sum_PLPbe2 := sum_PLPbe2+PLPbe2(conta_osservazioni);
msum_vettore(conta_osservazioni):=msum;
m2sum_vettore(conta_osservazioni):=m2sum;
vsum_vettore(conta_osservazioni):=vsum;
v2sum_vettore(conta_osservazioni):=v2sum;
percABS_vettore(conta_osservazioni):= percABS;
perc2ABS_vettore(conta_osservazioni):= perc2ABS;
percREL1_vettore(conta_osservazioni):= percREL1;
perc2REL1_vettore(conta_osservazioni):= perc2REL1;
percREL2_vettore(conta_osservazioni):= percREL2;
perc2REL2_vettore(conta_osservazioni):= perc2REL2;

sum_msum_vettore:= sum_msum_vettore+msum_vettore(
   conta_osservazioni);
sum_m2sum_vettore:= sum_m2sum_vettore+m2sum_vettore(
   conta_osservazioni);

sum_vsum_vettore:= sum_vsum_vettore+vsum_vettore(
   conta_osservazioni);
sum_v2sum_vettore:= sum_v2sum_vettore+v2sum_vettore(
   conta_osservazioni);

sum_percABS_vettore:= sum_percABS_vettore+
   percABS_vettore(conta_osservazioni);
sum_perc2ABS_vettore:= sum_perc2ABS_vettore+
   perc2ABS_vettore(conta_osservazioni);

sum_percREL1_vettore:= sum_percREL1_vettore+
   percREL1_vettore(conta_osservazioni);
```

```
sum_perc2REL1_vettore:= sum_perc2REL1_vettore+
    perc2REL1_vettore(conta_osservazioni);

sum_percREL2_vettore:= sum_percREL2_vettore+
    percREL2_vettore(conta_osservazioni);
sum_perc2REL2_vettore:= sum_perc2REL2_vettore+
    perc2REL2_vettore(conta_osservazioni);

outtext("OBSERVED VALUE = PLP of RT : ");
outint(conta_osservazioni,10); outfix(PLPrt(
    conta_osservazioni), 40, 45);
outimage;

outtext("SQUARED OBSERVED VALUE: PLP**2 of RT ");
outint(conta_osservazioni,10); outfix(PLPrt2(
    conta_osservazioni), 40, 45);
outimage;

outtext("OBSERVED VALUE = PLP of BE : ");
outint(conta_osservazioni,10); outfix(PLPbe(
    conta_osservazioni), 40, 45);
outimage;

outtext("SQUARED OBSERVED VALUE: PLP**2 of BE ");
outint(conta_osservazioni,10); outfix(PLPbe2(
    conta_osservazioni), 40, 45);
outimage;

outtext("OBSERVED VALUE=msum : ");
outint(conta_osservazioni,10 ); outfix(msum_vettore(
    conta_osservazioni), 40, 45);
outimage;

outtext("SQUARED OBSERVED VALUE=m2sum  ");
outint(conta_osservazioni,10); outfix(m2sum_vettore(
    conta_osservazioni), 40, 45);
outimage;

outtext("OBSERVED VALUE=vsum : ");
outint(conta_osservazioni,10 ); outfix(vsum_vettore(
    conta_osservazioni), 40, 45);
outimage;

outtext("SQUARED OBSERVED VALUE=v2sum  ");
```

```
outint(conta_osservazioni,10); outfix(v2sum_vettore(
    conta_osservazioni), 40, 45);
outimage;

outtext("OBSERVED VALUE= percABS : ");
outint(conta_osservazioni,10 ); outfix(percABS_vettore(
    conta_osservazioni), 40, 45);
outimage;

outtext("SQUARED OBSERVED VALUE= perc2ABS  ");
outint(conta_osservazioni,10); outfix(perc2ABS_vettore(
    conta_osservazioni), 40, 45);
outimage;

outtext("OBSERVED VALUE = percREL1 : ");
outtext("% relative of BE interrupted by GST");
outint(conta_osservazioni,10 ); outfix(percREL1_vettore(
    conta_osservazioni), 40, 45);
outimage;

outtext("SQUARED OBSERVED VALUE= perc2REL1  ");
outint(conta_osservazioni,10); outfix(perc2REL1_vettore(
    conta_osservazioni), 40, 45);
outimage;

outtext("OBSERVED VALUE= percREL2 : ");
outtext("% relative of BE interrupted by RT");
outint(conta_osservazioni,10 ); outfix(percREL2_vettore(
    conta_osservazioni), 40, 45);
outimage;

outtext("SQUARED OBSERVED VALUE= perc2REL2  ");
outint(conta_osservazioni,10); outfix(perc2REL2_vettore(
    conta_osservazioni), 40, 45);
outimage;

END;

sum_meanPLPrt:=sum_PLPrt/conta_osservazioni;
sum_meanPLPrt2:=sum_PLPrt2/conta_osservazioni;
deviazione_std_rt:= sqrt(abs(10.0*sum_PLPrt2-sum_PLPrt*
    sum_PLPrt)/90.0);
ConfInter95rt:= (deviazione_std_rt*2.262)/sqrt(10.0);
lower_rt:= sum_meanPLPrt-ConfInter95rt;
upper_rt:= sum_meanPLPrt+ConfInter95rt;
```

```
if lower_rt>0 then adjustDELTA_rt:=ConfInter95rt
   else adjustDELTA_rt:=ConfInter95rt+lower_rt;

outtext("Mean value of sum of PLP RT : ");
outfix(sum_meanPLPrt, 40, 45);
outimage;

outtext("Sum of PLP**2 of: ");
outfix(sum_PLPrt2, 40, 45);
outimage;

outtext("Mean value of sum of PLP**2 of RT : ");
outfix(sum_meanPLPrt2, 40, 45);
outimage;

outtext("Standard deviation RT: ");
outfix(deviazione_std_rt, 40, 45);
outimage;

outtext("95% Confidential Intervall : ");
outfix(ConfInter95rt, 40, 45);
outimage;

outtext("LOWER of RT : ");
outfix(lower_rt, 40, 45);
outimage;

outtext("UPPER of RT : ");
outfix(upper_rt, 40, 45);
outimage;

outtext("Adjusted lower 95% Confidence interval of RT : ");
outfix(adjustDELTA_rt, 40, 45);
outimage;

sum_meanPLPbe:=sum_PLPbe/conta_osservazioni;
sum_meanPLPbe2:=sum_PLPbe2/conta_osservazioni;
deviazione_std_be:= sqrt(abs(10.0*sum_PLPbe2-sum_PLPbe*sum_PLPbe)/90.0);
ConfInter95be:= (deviazione_std_be*2.262)/sqrt(10.0);
lower_be:= sum_meanPLPbe-ConfInter95be;
upper_be:= sum_meanPLPbe+ConfInter95be;
if lower_be>0 then adjustDELTA_be:=ConfInter95be
   else adjustDELTA_be:=ConfInter95be+lower_be;
```

```
outtext("Mean value of sum of PLP BE : ");
outfix(sum_meanPLPbe, 40, 45);
outimage;

outtext("Sum of PLP**2 of BE: ");
outfix(sum_PLPbe2, 40, 45);
outimage;

outtext("Mean value of sum of PLP**2 of BE : ");
outfix(sum_meanPLPbe2, 40, 45);
outimage;

outtext("Standard deviation BE: ");
outfix(deviazione_std_be, 40, 45);
outimage;

outtext("95% Confidential Intervall of BE: ");
outfix(ConfInter95be, 40, 45);
outimage;

outtext("LOWER of BE : ");
outfix(lower_be, 40, 45);
outimage;

outtext("UPPER of BE : ");
outfix(upper_be, 40, 45);
outimage;

outtext("Adjusted lower 95% Confidence interval of BE :
    ");
outfix(adjustDELTA_be, 40, 45);
outimage;

deviazione_std_delay1:=sqrt(abs(10.0*sum_m2sum_vettore-
    sum_msum_vettore**2)/90.0);
ConfInter95_delay1:= (deviazione_std_delay1*2.262)/sqrt
    (10.0);
lower_delay1:= sum_msum_vettore/10.0-ConfInter95_delay1;
upper_delay1:= sum_msum_vettore/10.0+ConfInter95_delay1;
if lower_delay1>0 then adjustDELTA_delay1:=
    ConfInter95_delay1
    else adjustDELTA_delay1:=ConfInter95_delay1+
        lower_delay1;
```

```
outtext("Mean value of msum: ");
outfix(sum_msum_vettore/10.0, 40, 45);
outimage;

outtext("Standard deviation mean value delay: ");
outfix(deviazione_std_delay1, 40, 45);
outimage;

outtext("95% Confidential Intervall mean value delay: ")
    ;
outfix(ConfInter95_delay1, 40, 45);
outimage;

outtext("LOWER of mean value delay : ");
outfix(lower_delay1, 40, 45);
outimage;

outtext("UPPER of mean value delay: ");
outfix(upper_delay1, 40, 45);
outimage;

outtext("Adjusted lower 95% Confidence interval mean
    value delay : ");
outfix(adjustDELTA_delay1, 40, 45);
outimage;

deviazione_std_delay:=sqrt(abs(10.0*sum_v2sum_vettore-
    sum_vsum_vettore**2)/90.0);
ConfInter95_delay:= (deviazione_std_delay*2.262)/sqrt
    (10.0);
lower_delay:= sum_vsum_vettore/10.0-ConfInter95_delay;
upper_delay:= sum_vsum_vettore/10.0+ConfInter95_delay;
if lower_delay>0 then adjustDELTA_be:=ConfInter95_delay
   else adjustDELTA_delay:=ConfInter95_delay+lower_delay
       ;

outtext("Mean value of vsum: ");
outfix(sum_vsum_vettore/10.0, 40, 45);
outimage;

outtext("Standard deviation variance delay: ");
outfix(deviazione_std_delay, 40, 45);
outimage;

outtext("95% Confidential Intervall variance delay: ");
```

```
outfix(ConfInter95_delay, 40, 45);
outimage;

outtext("LOWER variance delay : ");
outfix(lower_delay, 40, 45);
outimage;

outtext("UPPER variance delay: ");
outfix(upper_delay, 40, 45);
outimage;

outtext("Adjusted lower 95% Confidence interval variance
     delay : ");
outfix(adjustDELTA_delay, 40, 45);
outimage;

deviazione_std_percABS:=sqrt(abs(10.0*
   sum_perc2ABS_vettore-sum_percABS_vettore**2)/90.0);

ConfInter95_percABS:= (deviazione_std_percABS*2.262)/
   sqrt(10.0);
lower_percABS:= sum_percABS_vettore/10.0-
   ConfInter95_percABS;
upper_percABS:= sum_percABS_vettore/10.0+
   ConfInter95_percABS;
if lower_percABS>0 then adjustDELTA_percABS:=
   ConfInter95_percABS
   else adjustDELTA_percABS:=ConfInter95_percABS+
      lower_percABS;

outtext("Mean value % absolute of BE interrupted: ");
outfix(sum_percABS_vettore/10.0, 40, 45);
outimage;

outtext("Standard deviation % absolute of BE interrupted
   : ");
outfix(deviazione_std_percABS, 40, 45);
outimage;

outtext("95% Confidential Intervall % absolute of BE
   interrupted: ");
outfix(ConfInter95_percABS, 40, 45);
outimage;

outtext("LOWER % absolute of BE interrupted: ");
```

```
outfix(lower_percABS, 40, 45);
outimage;

outtext("UPPER % absolute of BE interrupted: ");
outfix(upper_percABS, 40, 45);
outimage;

outtext("Adjusted lower 95% Confidence interval %
   absolute of BE interrupted : ");
outfix(adjustDELTA_percABS, 40, 45);
outimage;

deviazione_std_percREL1:=sqrt(abs(10.0*
   sum_perc2REL1_vettore-sum_percREL1_vettore**2)/90.0);

ConfInter95_percREL1:= (deviazione_std_percREL1*2.262)/
   sqrt(10.0);
lower_percREL1:= sum_percREL1_vettore/10.0-
   ConfInter95_percREL1;
upper_percREL1:= sum_percREL1_vettore/10.0+
   ConfInter95_percREL1;
if lower_percREL1>0 then adjustDELTA_percREL1:=
   ConfInter95_percREL1
   else adjustDELTA_percREL1:=ConfInter95_percREL1+
      lower_percREL1;

outtext("Mean value % relative of BE interrupted by GST
   : ");
outfix(sum_percREL1_vettore/10.0, 40, 45);
outimage;

outtext("Standard deviation % relative of BE interrupted
    by GST: ");
outfix(deviazione_std_percREL1, 40, 45);
outimage;

outtext("95% Confidential Intervall % relative of BE
   interrupted by GST: ");
outfix(ConfInter95_percREL1, 40, 45);
outimage;

outtext("LOWER % relative of BE interrupted by GST: ");
outfix(lower_percREL1, 40, 45);
outimage;

```

```
outtext("UPPER % relative of BE interrupted by GST: ");
outfix(upper_percREL1, 40, 45);
outimage;

outtext("Adjusted lower 95% Confidence interval %
   relative of BE interrupted by GST : ");
outfix(adjustDELTA_percREL1, 40, 45);
outimage;

deviazione_std_percREL2:=sqrt(abs(10.0*
   sum_perc2REL2_vettore-sum_percREL2_vettore**2)/90.0);

ConfInter95_percREL2:= (deviazione_std_percREL2*2.262)/
   sqrt(10.0);
lower_percREL2:= sum_percREL2_vettore/10.0-
   ConfInter95_percREL2;
upper_percREL2:= sum_percREL2_vettore/10.0+
   ConfInter95_percREL2;
if lower_percREL2>0 then adjustDELTA_percREL2:=
   ConfInter95_percREL2
   else adjustDELTA_percREL2:=ConfInter95_percREL2+
      lower_percREL2;

outtext("Mean value % relative of BE interrupted by RT :
    ");
outfix(sum_percREL2_vettore/10.0, 40, 45);
outimage;

outtext("Standard deviation variance % relative of BE
   interrupted by RT: ");
outfix(deviazione_std_percREL2, 40, 45);
outimage;

outtext("95% Confidential Intervall variance % relative
   of BE interrupted by RT: ");
outfix(ConfInter95_percREL2, 40, 45);
outimage;

outtext("LOWER variance % relative of BE interrupted by
   RT: ");
outfix(lower_percREL2, 40, 45);
outimage;

outtext("UPPER variance % relative of BE interrupted by
   RT: ");
```

```
outfix(upper_percREL2, 40, 45);
outimage;

outtext("Adjusted lower 95% Confidence interval variance
    % relative of BE interrupted by RT : ");
outfix(adjustDELTA_percREL2, 40, 45);
outimage;

END;
```

Bibliography

[1] Steinar Bjornstad, Dag Roar Hjelme and Norvald Stol: "A packet switched hybrid optical network with service guarantees", *IEEE Journal on Selected Areas in Communications*, Volume: 24, Issue: 8, Part: Supplement, pp. 97-107, 2006.

[2] Norvald Stol, Carla Raffaelli, Michele Savi and Gabriella Cincotti, "Optical Codes for Packet Detection in the OpMiGua Switch Architecture", *Photonics in Switching (PS) 2010*, Monterey, California, 25.-29. July, 2010.

[3] N. Stol, M. Savi, C. Raffaelli, "3-Level Integrated Hybrid Optical Network (3LIHON) to Meet Future QoS Requirements", IEEE Globecom 2011, 5-8 Dic., Houston (TX), 2011.

[4] C. Raffaelli, M. Savi, A. Stavdas, "Sharing Wavelength Converters in Multistage Optical Packet Switches", *in proceedings of Workshop on High Performance Switching and Routing (HPSR) 2006, Poznan, Poland, 7-9 June 2006.*

[5] ITU-T Recommendation Y.1541 (02/2006) "Internet protocol aspects Quality of service and network performance", including Y.1541 Amendment 3 (05/2008).

[6] Graham Birtwistle, "DEMOS - A system for Discrete Event Modelling on Simula", 2003.

[7] Bjorn Kirkerud, "OBJECT-ORIENTED PROGRAMMING WITH SIMULA", International Computer Science Series, 1989.

[8] K. Vlachos, E. Van Breusegem, K. Christodoulopoulos, D. Colle, K. Ramantas, P. Demeester: "Constrained and Unconstrained overspill routing in optical networks: a detailed performance evaluation", *Springer Photonic Netw, Commun.*, 13 (3) pp 227-240, 2007.

[9] A. Kimsas, S. Bjornstad, N. Stol, D. R. Hjelme, "AWGshare: A highly scalable and strictly non-blocking optical switch" *Optical Network Design and Modeling (ONDM), 2011 15th International Conference on Optical Networking Design and Modeling*

[10] Michael S. Borella, Jason P. Jue, Dhritiman Barnerjee, Byrav Ramamurthy and Biseanath Mukherjee, "Optical Components for WDM Lightware Networks" *Proceedings of the IEEE, VOL. 85, NO. 8, August 1997*

[11] Andreas Kimsas, Harlad Overby and Norvald Stol, "Analysis of a Bufferless OpMiGua Node" *Proceeding of Fifth Advanced International Conference on Telecommunications, AICT'09, pp. 381-388, 2009*

[12] Andreas Kimsas, Steinar Bjornstad, Harlad Overby and Norvald Stol, "Reservation Techniques in an OpMiGua Node" *in LNCS Optical Network Design and Modeling, Springer, pp 146-155, May 2007*

[13] Andreas Kimsas "Network layer Studies of the Hybrid Optical Migration Capable Network with Service Guarantees" *Thesis for the degree of Doctor Philosophiae* Oslo, 25.10.2011

[14] C. Raffaelli, S. Aleksic, F. Callegati, W. Cerroni, G. Maier, A. Pattavina, M. Savi, "Optical Packet Switching", *in Enabling Optical Internet with Advanced Network Technologies, J. Aracil, F. Callegati editors, Springer, UK, 2009, pp. 31-85, ISBN: 978-1-84882-277-1.*

[15] C. Raffaelli, M. Savi, A. Stavdas, 'Multistage Shared-Per-Wavelength Optical Packet Switch: Heuristic Scheduling Algorithm and Performance' *Journal of LightWave Technology, VOL. 27, NO 5, MARCH 1, 2009*