Market Managed Multi-service Internet

M3I

European Fifth Framework Project IST-1999-11429

Deliverable 7.2
ISP Business Model Report
Prototype Descriptions

The M3I Consortium

Hewlett-Packard Ltd, Bristol UK (Coordinator)
Athens University of Economics and Business, GR
BT Research, Ipswich GB
Eidgenössische Technische Hochschule, Zürich CH
Forschungszentrum Telekommunication Wien, A
Technische Universität Darmstadt DE
Telenor, Oslo NO

For more information on this document or the M3I Project, please contact:

Hewlett-Packard Ltd,
European Projects Office,
Filton Road,
Stoke Gifford,
BRISTOL BS34 8QZ,
UK
Phone: (+44) 117-312-8631
Fax: (+44) 117-312-9285
E-mail: sandy_johnstone@hp.com

Document Control

Title: ISP Business Model Report: Prototype Descriptions

Type: Public Deliverable

Author: Jörn Altmann

E-mail: jorn_altmann@hpl.hp.com

Origin: HPLabs-Bristol

Doc ID: m3I:del07_2.doc

Amendment History

Version	Date	Author	Description/Comments
V 1.1	31-Dec-2000	HPLB	First published version
V 1.14	04-Feb-2002	HPLB	Formatting and language edit, public version

Contributors

Name	Company
Jörn Altmann, Huw Oliver, Hans Daanen	HPLB (User Direct; v17; 21-Dec-2000)
Panayotis Antoniadis, Ioanna Constantiou	AUEB
Martin Karsten	TUD (GSPx; v0.3; 21-Dec-2000)
Martin Karsten	TUD (GSPd; v0.3; 21-Dec-2000)
Bob Briscoe	BT (DPH/ECN; v0.2; 5-Dec-2000)
Burkhard Stiller, Placi Flury	ETHZ (CPS; v1; 30-Dec-2000)

TABLE OF CONTENTS

1 Introduction

This document contains a set of scenario descriptions, which illustrate possible future business models of ISPs. The analysis of current ISP business models and the impact of M3I technology on them has been made in Deliverable 7.1.

These scenarios will improve the understanding of fundamental factors, which will enable a positive evolution of currently successful business models, as well as the understanding of obstacles that may delay the development.

Our Telecom Company Partners within the M3I consortium are very interested in these scenarios because they meet the requirements that the underlying technology is simple and flexible. Flexibility is necessary to customise the presentation of the network to customers. The network presentation involves service, charging, and business interfaces.

British Telecom wishes to rigorously conform to standards for the first two of these interfaces, but wishes to have maximum flexibility with the business interface in order to synthesise innovative business models at the customer interfaces.

As an operator, Telenor is faced with the issue of how to efficiently implement mechanisms for segmentation of the market along quality/quantity dimensions in its communication service production platform. Even if development of the IntServ and DiffServ concepts have proceeded for several years, these and the products, in which they are implemented, have not been considered to be usable on a large-scale commercial basis until now. In the ongoing process within the company of how to arrive at sustainable business cases for network service differentiation, the additional models developed in the M3I-project, and presented in this document, will be used as inputs amongst others. A specific aspect of interest to Telenor is the question of how to arrive at the desired differentiation and still retain control of the operational complexity.

Each scenario provides a context in which special experiments concerning dynamic pricing can be executed. Each of those experiments examines certain objections to dynamic pricing. Some of the questions the experiments try to answer are: can stable prices be provided to end-users? What is the degree of stability in a market-managed network? What type of admission control system does a market-managed network need? What is the overall welfare improvement by using economic approaches? What are the requirements of the underlying network technology in order to run a market-managed network?

More specifically, five scenarios will be investigated within M3I, namely the **User Direct Scenario**, the **Dynamic Price Handler for Explicit Congestion Notification Scenario (DPH/ECN)**, the **Guaranteed Stream Provider for IntServ Scenario (GSPd)**, the **Guaranteed Stream Provider for Explicit Congestion Notification Scenario (GSPx)**, and the **Cumulus Pricing Scheme Scenario (CPS)**.

In the User Direct scenario and DPH/ECN scenario, end-users experience dynamic prices. End-users react to varying prices according to their utility of the service received. The difference between these two scenarios is the price reaction process and the frequency of price changes. In the User Direct scenario, the prices vary at a hourly, daily, or weekly rate (*user time scale*). End-users directly choose the type of network service from a selection of price/QoS levels combinations. In the DPH/ECN scenario, the prices might change every microsecond (*network time scale*) and an

agent (price reactor) reacts to price changes on behalf of an end-user according to a specified strategy.

In the Guaranteed Stream Provider scenarios (GSPd and GSPx), the market mechanisms are implemented in the production platform, hidden from users. The GSPx scenario provides the context to investigate that a production platform, which uses dynamic pricing for resource contention resolution, is favourable to more traditional approaches with regard to implementation complexity, operation, and maintenance. The GSPd scenario (Intserv) serves as reference to a pure Integrated Service approach.

The CPS scenario explicitly considers long time-scale pricing. CPS is a flat rate pricing scheme with a feedback mechanism to adapt flat rates to the actual usage over a longer time scale.

The responsibilities of each M3I partner (TU Darmstadt (TUD), British Telecom Research (BT), Telenor (TN), ETH Zürich (ETHZ), Hewlett Packard Labs Bristol (HPLB), Athens University of Economics and Business (AUEB)) for the scenarios are illustrated in the following Table 1-1. The responsibilities concern the design of the scenario, the implementation, the experiments, and the analysis of the experiment results. Certain fields under *Experiment* and *Analysis* of scenarios GSPx and GSPd do not show any partner being responsible. That means, the M3I consortium will not conduct this activity for these scenarios within the current project time-scale. However, these scenarios were designed and implemented to show the generality of the M3I technology developed.

Scenario			Design	Implementation	Experiments	Analysis
Name	**Technology**	**Owner**				
GSPx	ECN	TUD	TUD	TUD	---	---
GSPd	IntServ	TUD	TUD	TUD	TN	---
DPHa	ECN	BT	BT	BT/AUEB	BT	BT/HPLB
User Direct	DiffServ	HPLB	HPLB	HPLB	HPLB	HPLB
CPS	DiffServ	ETHZ	ETHZ	ETHZ	ETHZ	ETHZ

Table 1-1: Responsibilities for Scenarios

Each chapter of this document is an independent description of one scenario. In order to facilitate the identification of the different viewpoints and help the comparison between the scenarios, all chapters follow a similar structure. Each chapter is structured in sections about *overview and motivation,* the *design*, the *implementation*, the *experiment description*, and the *conclusion*. The references, abbreviation, and appendices are placed at the end of each chapter in order to keep the *independent* nature of this document and facilitate reading.

2 User Direct Scenario

This scenario describes a medium time scale control mechanism for Internet traffic. It is an economic mechanism that enables users to choose different price/QoS priority levels for network services at the user time scale. User time scale means that prices vary at a rate suitable for human beings to respond to those price changes. The changes might happen on an hourly, daily, or weekly basis. We believe that for this medium time scale, an economic control mechanism needs to be coupled with a technical rate control mechanism at a short time scale, in order to build a simple and cost-reducing system, which can provide quality of service to end-users. We describe the network hardware, the pricing and charging software of the architecture, and the experiments we intend to run on the network. In addition to this, we also discuss the advantages of the user time scale approach compared with a network time scale approach (prices change every microsecond) and a long-term time scale approach as used for network provisioning. We show that the approach proposed enables Internet service providers to react to network congestion sufficiently and to provide high flexibility in service selection to their customers.

2.1 Introduction

This work looks forward to a future Internet that is a multiple service network. The network will support different kinds of applications (i.e. email, real-time audio, and streaming video) and different needs of users. Such a network will be cost effective and will have the benefits of economies of scale.

We believe network service providers (also called connectivity service providers) will be forced to offer more flexibility to their customers in order to survive in the highly competitive Internet services market. The network service providers will have to provide highly customised services, which allow end-users to change QoS/price choices any time. Flat rated, tiered pricing plans will not be adequate in such an environment. Pricing plans of a finer granularity will be necessary.

Clearly, any successful solution for supporting multiple services cannot rely on technical solutions alone but also has to take into account the economic aspects. Different qualities of service must be priced differently. If not, people will always use the best one. Of course, user behaviour can be influenced by contracts and policing but this leads to complicated monitoring and policing systems. The economically efficient way to influence user behaviour is by giving them the correct financial incentives.

The challenge of designing a market-managed network with service differentiation is to provide an approach based both on pricing and on technical rate control that moves choice to the end customer and tackles the network problems of traffic management. In addition to this, the system has to be simple and has to generate predictable prices.

The next section describes the problem inherent with a market-managed network more fully. We discuss pricing principles in the Internet services market and give design motivations of our user time scale approach. Section 2.3 explains the design of our approach. The implementation choices and the hardware and software architecture is described in section 2.4. Finally, we close by describing the experiments we intend to perform.

2.2 Overview and Motivation

2.2.1 Usage-Based Pricing

In the future ISP market, usage-based pricing will be essential for a multi-services Internet. Usage-based pricing enables high customisation of network services, i.e. NSPs can provide their customers with the flexibility of switching between different qualities of network service. It is also a mean of allocating scarce resources among those end-users, who value the service the most.

As soon as end-users have to pay for their usage of the Internet, only information will be downloaded which is of more value to the user than the incurred costs (i.e. money spent on downloading or time spent waiting for the download to finish). Each download includes an evaluation of the utility of the information the end-user gets. The end-users' incentive is to maximise their utility when using the Internet. We believe over-utilisation of resources and, therefore, congestion will disappear if the prices are set correctly.

2.2.1.1 Flexibility in Service Selection

Assuming that there is sufficient competition and NSPs do not necessarily control or even know what content their users transmit, NSPs can only make profit by customising their network services. That means, they have to differentiate network services, add value to a basic network service, or offer their customers high flexibility in selecting qualities of network service. Customisation will help NSPs to differentiate themselves from their competitors.

By offering QoS selection on demand, end-users benefit from a broader selection of services and NSPs will get the surplus end-users are willing to pay in certain situations. Deploying this kind of customisation, pricing plans have to charge end-users based on the usage of the service. The service might differ from a basic service in a higher maximum transmission rate or a lower level of congestion.

2.2.1.2 Allocation of Scarce Network Resources

In addition to the need to highly customise network services, there is a fundamental need to adapt supply to demand for network services. With the unpredictability of needs of new Internet applications and the lack of historical traffic data, the adaptation is difficult.

Network capacity is increasing very rapidly (transmission capacity doubled each year during 1990 to 2000), but there is still the high probability that the network traffic will increase even faster in the short term. This traffic increase might be fuelled by running programs like Napster, Gnutella, or Freenet, that download bulk data without any direct interactions of users and in an "always-on" mode. Since the user is not charged for usage (when subscribed to a flat-rate service plan) and the download does not incur any time costs (since it does not require the user's presents), the user does not care about the amount of information being downloaded. The resulting increase in traffic and congestion on the network recently took many network planners by surprise.

In general, if the demand exceeds the supply, the network will become congested and the quality of service will go down for all users. If the demand is much lower than the supply, NSPs will have wasted investment. The ideal situation is the equilibrium of demand and supply but, because of the high variability in traffic volume, this is difficult to achieve by just applying technical solutions. It would add too much complexity to the network without giving any incentive for the end-user to use the system in an

economically efficient way. The network capacity should reflect a compromise between times of congestion and times of low network utilisation. However, in order to satisfy customers even during times of network congestion, the solution needs to take into account economic mechanisms too. These mechanisms will allocate resources to those end-users that value the service most. By distributing resources to users who value the service most, the overall welfare can be increased. In addition to this, it is important to give the NSPs the correct economic incentive to increase capacity in the long run when there is sufficient demand.

2.2.2 Design Motivation

The challenge in designing a market-managed network is to compose a system that allows service selection on demand and enables NSPs to deal with congestion. Within the next two subsections, considerations about frequency of price changes and predictability of prices are presented.

2.2.2.1 Service Control and Price Predicability

Important design aspects include service control and price predictability. The system must give the end-user control over choice of service and price according to her utility for those services. This is best done by giving the user direct access to a price / QoS selection that offers a service at a certain price for a certain duration. This also implies that the user's costs will be predictable. That these issues are important is shown in [1].

2.2.2.2 Frequency of Price Changes

One distinguishing feature of pricing alternatives is the frequency at which prices for services change. The alternatives can be classified according to the time scales at which they change. The classes we distinguish are *network time scale* (changing each microsecond), *user time scale* (changing hourly, daily, or weekly), and *long-term time scale* (only changing every month or year).

Methods that operate on network time scale change prices in real time as transient network overloads occur. While this allows congestion costs to be conveyed to the user at a fine grain and in real time, we see two drawbacks:

- The user cannot (and would not want to) respond sufficiently quickly to such rapid changes.
- It introduces greater unpredictability in the final charge.

Long-term time scale approaches try to predict the growth rate of the network traffic and find a compromise between times of congestion and times of under-utilisation. The basis of all these approaches are analyses of aggregated usage patterns. Based on these analyses, the network is upgraded to meet the predicted needs. Prices of network services only change very rarely. Although this approach helps to limit the cost of network over-provisioning, the disadvantage of this approach is that it does not consider the daily/hourly fluctuations of traffic. The difference between the traffic peaks during daytime and the lows at night is a factor of 20. In addition to this drawback, end-user's demand for future Internet services is very difficult to predict, resulting in even worse fluctuations if the demand is initially underestimated. This means, that there will probably be even more waste of capacity at some times, and network congestion during other times. Besides, the increase of bandwidth will always be a long-term process and not quick to correct.

The user time scale approach addresses these issues. User time scale means that prices for Internet services can change on an hourly, daily, or weekly basis, so that end-users can respond to those prices. Under this approach, end-users can express the value they place on the network service. Scarce resources are allocated to those users who value them most. For example, end-users could postpone emails until prices for network services go down (e.g. time of day pricing), or it could be by reducing the peak transmission rate or simply by reducing the amount of data sent. However, prices of network services are known to the end-user in advance and vary slow enough for them to keep track. From the NSP's perspective, the user time scale approach reveals more information about the usage pattern of their customers compared to the long-term time scale approach.

2.3 Design

2.3.1 Priority Levels

In our basic scheme the user is offered a list of priority levels at which he can send his traffic. Traffic sent at a higher level will be sent at a higher priority and at a higher price. The absolute quality of service of each priority level is not guaranteed. The quality of service depends on the current network state. The differences are relative and may change in real-time. Based on that, the user may choose to move up or down the levels accordingly. As lower levels become congested there will be more incentive to move up the levels. As congestion eases there will be lower incentives to pay the higher prices at the higher levels.

The end-user will have to have software available on his system that will allow him to control his system. Depending on the complexity of the pricing plan this can be a simple selector for the priority level or it may be a combination of traffic shapers and tariff aware applications.

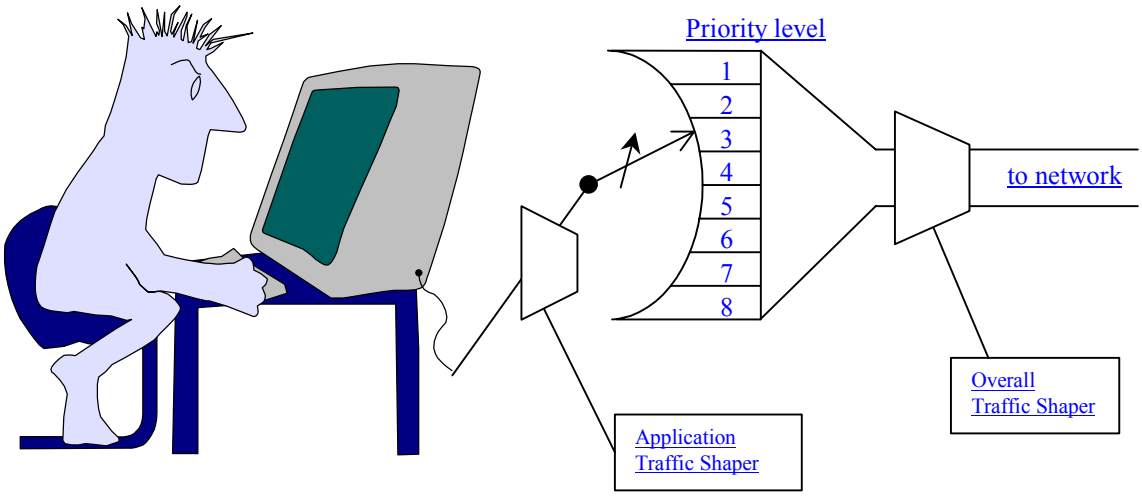

Figure 2-1: User Interaction with Different Priority Levels Offered

Depending on the pricing plan, the user may want to shape the traffic of the individual applications and the overall traffic he generates. If the individual applications are aware of the pricing plan they can adapt their own traffic, otherwise a per application traffic shaper needs to be available. The same goes for aggregate traffic. If the user needs to keep his overall traffic within certain profiles this can be done either by monitoring the

aggregate traffic and giving feedback to the individual applications or by an overall traffic shaper (see Figure 2-1).

2.3.2 Pricing

The basic pricing plan we propose is quite straightforward. Each priority level is priced at a different rate. The prices are strictly increasing with regard to the priority. If prices are set correctly (with regard to the price elasticity of end-users), the network load is different at each priority level. By changing the usage-based prices for the different DiffServ classes, the demand for each DiffServ class can be controlled. In order to get feedback on the demand, the usage of the different priority levels will be monitored and recorded. Usage will be metered, for example, as the number of bytes transmitted, the number of packets transmitted, or the amount of time connected to the network.

More complex pricing plans can be generated based on this basic pricing plan.

These pricing plans can be used in conjunction with split-edge pricing.

2.3.3 Receiver-Based Charging

A receiver-based charging scheme, like most others, requires some protocol to exchange the parameter settings for this charging scheme. Since the sender makes the type of service indication, so there needs to be some method of notification from the receiver to the sender if a different priority level (and cost) is preferred.

2.3.4 Multicast

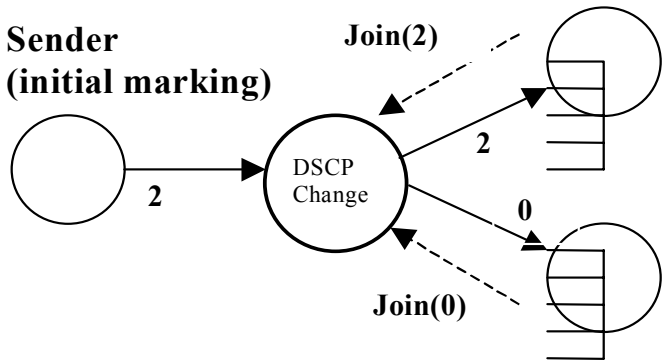

Figure 2-2: Example of a Marking Change by Multicast Routers

An increasingly large set of applications involves multicasting where a single process sends data to a large number of receivers simultaneously. In order to supply these multiple receivers with different quality of services, the QoS mechanism has to be adapted. That is an ongoing research area. For DiffServ, the proposals generally involve making a change to the TOS or DSCP code at an intermediate, multicast-enabled router (Figure 2-2). Coupled with a solution for receiver-based charging, the User Direct pricing scheme is compatible with these proposals. The receiver simply signals the required priority when he emits his *join* request.

2.4 Implementation

2.4.1 Implementation Choices

2.4.1.1 Mechanisms for Quality of Service

After describing why pricing on a user time scale and why service selection on demand can provide a business advantage, the technology has to be chosen which can deploy this kind of differentiation. One of the main requirements is that the technology has to be easy to understand and control for the end-user. In addition to this, if the technology is easily deployable in the current Internet, then the chance of adoption is much higher.

The alternatives come from QoS technologies such as IntServ or DiffServ. DiffServ provides a natural way of implementing our scheme. The NSP is given the opportunity to differentiate its services by implementing service classes with different quality parameters. NSPs set the quality of service that their customers will experience. From the user's perspective, each user can choose the priority based on their needs and money.

2.4.1.2 Traffic Shaping

In a future Internet, end-users have high capacity, always-on physical links to the Internet, capable of transmitting data at rates higher than 1.5 Mbit/sec. The problem of network congestion on the NSPs network occurs if end-users always sent their data at the maximum transmission rate (burstiness of traffic). A traffic shaper helps to avoid the problem, by altering the traffic profile with regard to the maximum transmission rate. However, since network responds times (i.e. the maximum transmission rate) is valuable to end-users, end-users should be given the choice of different transmission rates. The customer's decision will be based on the price for the maximum bandwidth and his needs.

The traffic shaper (or the IntServ router, which works as a traffic shaper) is located between the customer's computer and the DiffServ router. This raises the question whether the admission control system should reside on the NSP network or on the end-user's premises. In either case, the end-user has to have access to parameter setting of the traffic shaper, in order to specify the preferred transmission rate.

2.4.2 Topology

The description of the architecture design comprises the network architecture, and the pricing and charging software architecture.

2.4.2.1 Network Architecture

Our test network is composed of DiffServ routers, admission control hardware, and traffic generators (Figure 2-3). All components run on standard PCs that are connected via a 100Mbit/s Ethernet switch. The traffic generators represent a number of end-users/customers with a certain demand for network services. Thus, current traffic generators will need to be modified to model end users' price sensitivity.

Figure 2-3 also shows the hardware running the pricing and charging software as well as the metering and mediation software.

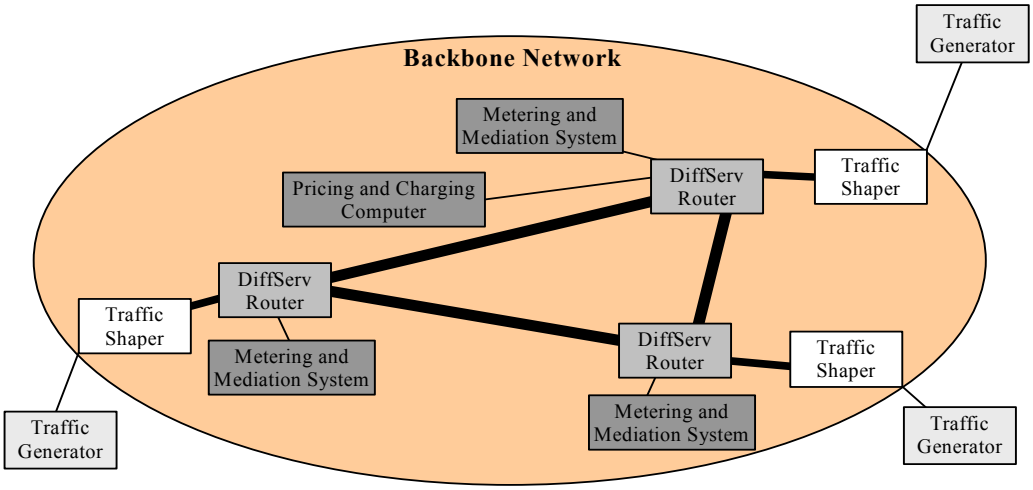

Figure 2-3: Example of the Test Network

2.4.2.2 Pricing and Charging Architecture

The components of the charging and pricing software can be grouped into two parts. One part runs on the end-user's computer and the other part runs on the NSP network. The software architecture is depicted in Figure 2-4 and Figure 2-5.

The software running on the NSP network comprises components for **Charging-and-Accounting, Price-Setting, Network-Mediation, Traffic-Shaper-Management**, and **User-Management**. The Charging-and-Accounting component is responsible for accounting of network services consumed by each user. The end-user usage data is rated with regard to choice of priority level, choice of maximum bandwidth, and choice of pricing plan. The Network-Mediation component, which aggregates usage data according to criteria describing the user-selected pricing plan, feeds into the Charging-and-Accounting component by sending periodically aggregated usage data. An example for such a mediation system is Hewlett-Packard's Smart Internet Usage (SIU). The Network-Mediation component's input comes from routers (e.g. Cisco's NetFlow) or network monitors. The Price-Setting component calculates and sets the prices for network services at the user time scale. Its input is the actual usage of individuals recorded by the Charging-and-Accounting component and the pricing policy specified by the ISP. The User-Management component manages the interaction with the end-

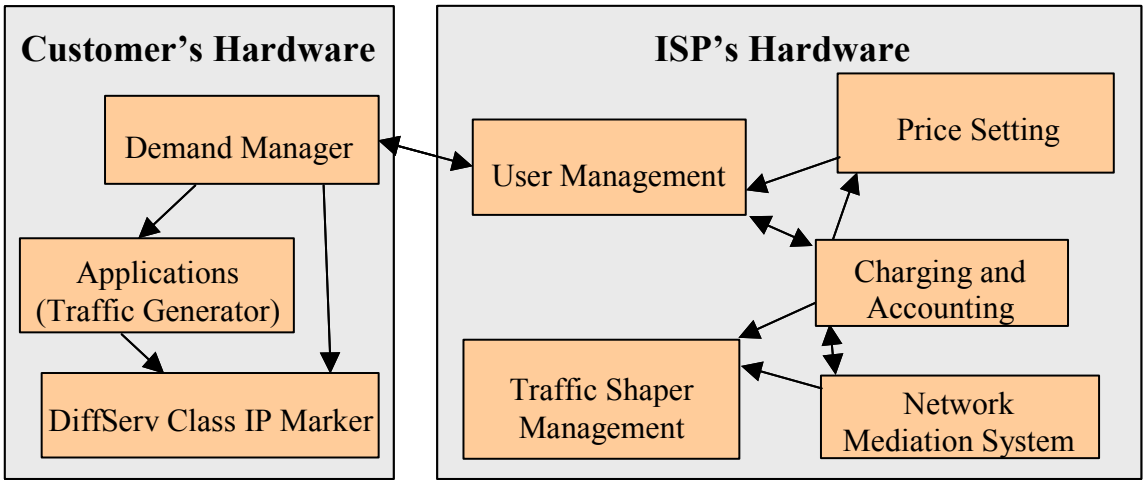

Figure 2-4: Software Archtecture

user. It offers services at the calculated prices to end-users. After a pricing plan (e.g. maximum transmission rate and DiffServ priority level) was accepted by the end-user, the information about the pricing plan is forwarded to the Charging-and-Accounting component, which needs the information to initialise the accounting and rating engine for this user accordingly. In addition to this, the User-Management component forwards updated charging information to end-users periodically. The Traffic-Shaper-Management component handles the interaction with the traffic shaper. It sets the maximum bandwidth capacity according to the end-user's selection. It received the settings from the Charging-and-Accounting component.

The software running on the end-user's computer presents the prices for the Internet services to the end-user. The end-user (represented by the **Demand-Manager** for the purpose of the experiments planned) makes choices of the preferred quality of service (i.e. priority level, and maximum bandwidth) based on his demand, which is determined by the kinds of applications he runs, the time of day, and his price elasticity. After a selection has been made, the acceptance of the offer is sent to the User Management component of the ISP. The Demand-Manager also sets the quality of service parameters for each application and forwards this information to the **Applications** (represented by the **Traffic Generator**) and to the **DiffServ Class IP Marker**, which marks the IP packets in case the applications can not do it.

2.4.2.3 Description in M3I architecture terminology

In order to facilitate the understanding of the functionality of the software components described, we show the relation between these software components and the components defined in the M3I architecture description (Deliverable D2) in the following paragraphs (refer to Figure 2-5).

The network provider specifies (1) its policy for pricing different network services using the **Enterprise-Policy-Agent** (**Ep**). This component places (2) those offers (i.e. pricing plans) in a **Public-Directory** (**O**), which can be accessed (5) by end-users.

End-users are represented by an **Enterprise-Policy-Agent** (**Ec**) component, which models (3) the end-users purchasing behaviour. The Enterprise-Policy-Agent component searches (5) for service offers in the Public Directories of network service providers. If the offer meets the requirements of the Enterprise Policy Agents then it accepts (7) the offer. The parameters of the pricing plan chosen are forwarded to the **Price-Reaction-Handler** (**Pc**). The Price Reaction Handler passes (9) the QoS parameter to the **Quality-of-Service-Manager** (**Qc**), which marks the IP packets according to the preferred priority level.

On the network service provider side, the **Charging-and-Accounting** component (**CAp**) gets initialised (10) with the prices and its scope (e.g. pricing plan and peak bandwidth) as defined in the offer acceptance as soon as an offer gets accepted from an end-user. The Charging and Accounting-System initialises (12)(16) the **Metering-Mediation-System** (**MMp**) and the **Quality-of-Service-Manager** (**Qp**). The Quality-of-Service-Manger gets (4) user's usage data from the Metering-Mediation-System. The measured usage data and the priority levels are reported to the Charging-and-Accounting component. After accounting and rating of the data, the Charging-and-Accounting component sends (15) the rated usage data of each end-user to the **Price-Handler** (**Pp**) component. In addition to this, the Charging-and-Accounting component sends (13) information about the currently accumulated charges to the end-user via the User-Management component. The Price-Setting component analyses the rated usage

data and calculates future prices based on the result of the analysis, which get posted on the Public Directory.

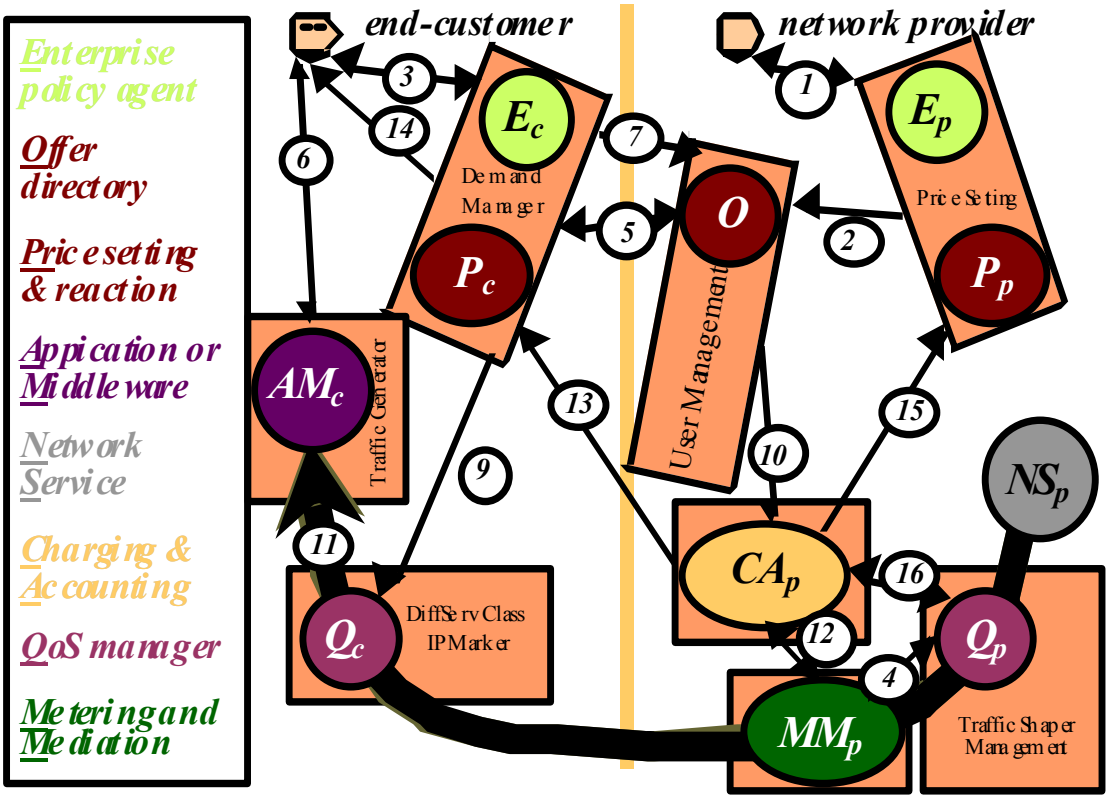

Figure 2-5: Software Architecture (Described with M3I Architecture Components)

2.5 Experiment Description

The prototype described will enable a wide spectrum of experiments. After some thought, we think it will be necessary to "play" with the system initially in order to gain an intuitive feel for its behaviour before we can test a set of hypotheses in a formal way.

There are a number of system parameters that we can vary in order to gain this understanding about the dynamics of the system:

- For the NSP:
  - Price ranges for the different priority levels
  - Scheduling algorithm of the DiffServ queues (WFQ, RED, priority, label coloring) that implement the priority levels.
  - Number of end-users on the different DiffServ classes
- For the end-user:
  - Willingness to pay
  - Type of traffic (e.g. UDP, TCP with certain characteristics)
  - Rate control (back-off algorithms)
  - Price/QoS selection strategies
- For the system:
  - Frequency of price changes

- Distribution of service requirements (e.g. file transfer vs. audio) across the user population

While varying the system parameters, the following questions will be kept in mind:

- Is such a system stable?

  It might happen that the load of the system starts to oscillate by end-users' responses to prices. For example, if the price for network services drops below a certain threshold all end-users might start using the network. Consequently, the load of the network increases and the prices have to be adjusted to reflect the load. But then, it might be that the prices are higher than the second user-defined threshold, which indicates to stop communication. Consequently, all end-users drop out of the network. The prices drop below the first threshold again and the entire process starts all over again.

  Although price variations are not as dynamic as an ECN based scheme, we want to investigate whether the User Direct scheme is near optimal as suggested by theoretical work done at Boston University [1].

- Is the overall network service economically efficient?

  The system should provide better service to those end-users, who value the network service more. More formal definitions of economic efficiency will be examined.

- Is the overall network service economically fair?

  The service allocation to end-users will be evaluated according to some formal definition of fairness. For example, a definition of fairness might include the criterion that no users get starved of services.

  The scheduling algorithms for the priority levels have a huge impact on the fairness of network services.

- What are the correct prices (at start of the system and during regular times)?

  The NSP will want to maximise his revenue and increase customer satisfaction. Prices might be set using the gathered information about current demand and customer usage profiles.

- Is the system incentive compatible?

  A system that is incentive compatible encourages users to be honest in expressing their preferences and willingness to pay.

In all of the experiments, traffic generators emulate the traffic of end-users and Demand-Managers emulate end-users' QoS/price choices. In particular, a traffic generator generates packet flows depending on a certain kind of application and the Demand-Manager's QoS choice (see Section 2.4.2.2).

2.6 Experiment Result

No results available yet.

2.7 Conclusion

We have presented a scenario for managing a network by market forces. The architecture combines network technology with economic theory, in order to maximise the overall benefit by allocating resources according to the end-users' demand. By setting prices according to current market situations on a user-time scale, the utilisation

of network resources can be controlled and gives users time to respond to price changes.

2.8 References

[1] J. Altmann and K. Chu, “A Proposal for a Flexible Service Plan that is Attractive to Users and Internet Service Providers", IEEE InfoCom2001, Conference on Computer Communications, Anchorage, Alaska, USA, April 2001.

[2] Ioannis.C. Paschalidis and J.N. Tsitsiklis, “Congestion-Dependent Pricing of Network Services,” IEEE/ACM Transactions on Networking, Vol. 8, No. 2, pages 171-184, August 2000.

2.9 Abbreviations

DiffServ	-	IETF Differentiated Services
DSCP	-	DiffServ Code Point
ECN	-	Explicit Congestion Notification
IntServ	-	IETF Integrated Services
ISP	-	Internet Service Provider
M3I	-	Market Managed Multi-service Internet
NSP	-	Network Service Provider
QoS	-	Quality of Service
SIU	-	Smart Internet Usage
ToS	-	Type of Service

3 Dynamic Price Handler for Explicit Congestion Notification Scenario (DPH/ECN)

The Dynamic Price Handler for Explicit Congestion Notification (DPH/ECN) scenario of the M3I project is fully defined in this section. The main intention is to ensure clarity on what the scenario entails for the benefit of the various parties in the M3I project working on it, whether for the purposes of implementation, architectural assessment, modelling and analysis or user-experiments.

3.1 Introduction

This section describes the Dynamic Price Handler for Explicit Congestion Notification (DPH/ECN) scenario developed in the M3I project. The scenario tests the concept of a dynamic price handler (DPH) reacting to priced Explicit Congestion Notification (ECN) marks. The partners planned to work on this scenario are specified in [2].

The scenario is already substantially described at high level in Sections 4.2 & 5.2 of Issue 1 of the M3I Requirements document [1]. Certain parts of this cited source are repeated here, but it is essential to refer to it directly for background and context. Where the two conflict, this description, as the later and more specific version, takes precedence. Every effort is made to identify all such cases with “Editor's comments”.

3.2 Overview and Motivation

3.2.1 Scenario Outline

The idea is to give hosts a price incentive to react to incipient congestion on the paths they are using through the Internet, while allowing them to pay to ignore a certain level of congestion if the value gained from so doing is greater than the charge levied. All that network providers have to do is to arrange for all routers to set the experimental congestion experienced (CE) bit [16] of the IP packet header with a probability related to the length of every queue[1] the packet traverses. The receiver's network provider then offers Internet network service at a charge calculated by placing an effectively fixed price on each such mark.

Unlike M3I's guaranteed stream provider scenarios, edge network customers are not insulated at all from a potentially variable quality or price. Instead, they run an agent called the dynamic price handler, which optimises their use of the available service within the constraints of a policy per task that they supply. Within these constraints the agent allows the quality to vary in order to avoid high cost periods, while at other times accepting a higher price in order to maintain a reasonable level of quality for the task in hand. The policy also allows the user to determine how unstable both quality and price may be for a particular task, which further constrains the agent's available strategies.

[1] In fact, the probability of setting the bit relates to the length of a virtual queue slightly smaller than the real queue to improve the early detection of congestion.

3.2.2 Scenario Purpose Motivation for Investigation

The DPH/ECN scenario is designed:

- to test the completeness of the M3I architecture [4];
- to test whether the concept of this scenario can be implemented;
- to test whether the implementations of M3I components intended for many scenarios are sufficiently generic, and otherwise to drive their re-design
- to be used as a demonstrator of M3I technology;
- to be used as a platform for possible future user experiments, although none is planned for this scenario within M3I so far;
- to be used to experiment with different algorithms within the price setting and reaction components, perhaps with a view to gathering performance data from the system to feed into simulations or models of much larger systems.

The essence of the scenario is a sending and a receiving application, each of which are operated by customers of different connectivity providers. Other customers' traffic is represented by a single flow across the path between these two customers, from a traffic generator. The receiver's provider charges a fixed price per ECN mark. Two sub-scenarios are considered, one where the receiver pays this charge, the other where it is settled by the sender.

3.3 Design

3.3.1 Scenario Topology

A minimum of six PCs is necessary for this scenario, in a layout shown in Figure 3-1[2]. The second flow is merely to supply congestion to cause the charge from the first flow to vary.

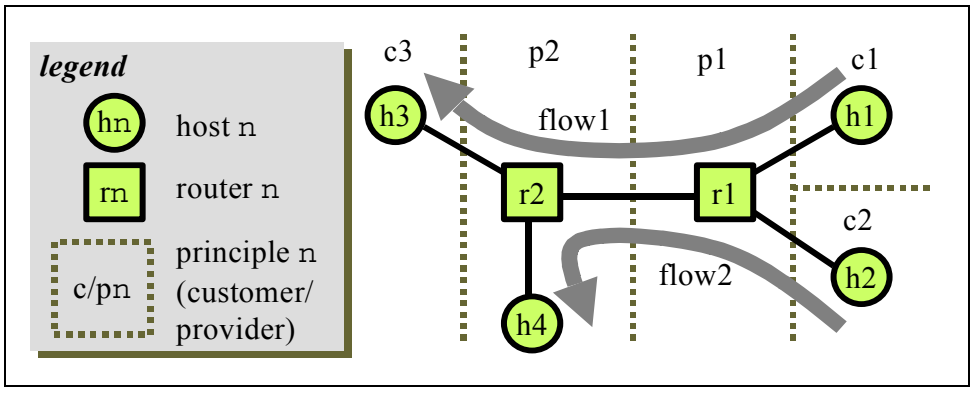

Figure 3-1: Scenario Topology

Notation that explicitly identifies a customer or a host as a sender or a receiver has been avoided, so that the notation can continue to be useful for duplex extensions to the scenario. However, for ease of understanding, the suffixes (snd) or (rcv) will be

[2] Note that a multi-provider scenario is specified, but the number of routers on the path across each provider is unrealistic. Adding a border router for each provider at their mutual boundary would have made the topology more realistic, but their presence or absence is irrelevant to the purpose of this scenario, which only requires that c1's (snd) and c3's (rcv) providers are different.

used after a customer or a host, to identify a sender or receiver for a particular data flow being discussed.

3.3.2 Trust Relationships

Here we specifiy the trust relationships that are assumed to hold in this scenario[3], i.e. the aspects highlighted as scenario specific in the M3I requirements document (end of section 4.2).

3.3.2.1 General Trust Relationships

Each instance of each type of stakeholder is termed a `principle', each of which are also identified in Figure 3-1. A trusted third party (TTP) whom everyone trusts to certify each principle's identity is assumed to exist outside this diagram. A TTP will not be implemented in this scenario, as nothing would be gained from such an exercise to further the research goals of M3I. Therefore identity certification will merely be assumed to have occurred.

The general trust scenario is that no-one trusts anyone else with the pragmatic exceptions listed in section 6.3.3 of [1] (draft version 7.1).

3.3.2.2 Trust Relationships Specific to DPH/ECN Scenario

With ECN, the sender, c1 (snd), ultimately controls the rate at which traffic is sent and the receiver, c3 (rcv), controls feedback of marking rate to the sender. Although the sender has ultimate control of the sending rate, the price reactor determines what the target rate should be and asks the sender to aim for it.

The sending rate chosen by c1 (snd) cannot be secret from c3 (rcv) who, as the receiver, directly experiences it. However, the marking rate is only known directly by c3 (rcv), while c1 (snd) relies on c3 (rcv) (or p2) to report it honestly (whether there is an incentive to lie depends on who is paying, but c3 (rcv) is always able to lie).

The price reactor runs under the buying policy of the payer. One assumes it is desirable not to reveal this policy to the non-payer where possible. It is certainly desirable not to reveal a buying policy to providers (although they can guess it by the traffic behaviour it produces). Thus, the price reactor should run at the payer's end whenever possible.

“Receiver pays” sub-scenario: the payer, c3 (rcv) sets the price-reactor's policy. The price reactor runs at the receiver, c3 (rcv), and every so often communicates the target rate to the sender, c1 (snd). The receiver, c3 (rcv), directly experiences the sender's rate, thus deterring the sender from not acting as instructed.

“Sender pays” sub-scenario: the payer, c1 (snd), sets the price reactor's policy. Ideally, for the security reason explained above, the price reactor runs at the sender, c1 (snd). It reacts to feedback of marks from the receiver (it also has to request price communications from p2 for the price per mark).

Clearing is achieved as follows in this scenario: p2 presents a signed[4] session characterisation for the session to its customer, c3 (rcv). c3 (rcv) in turn presents this to the clearinghouse, which also has access to p2's authenticated pricing. c3 (rcv) also

[3] Ignoring the trust status of c2 (snd), which is irrelevant to the scenario.

[4] [7] explains why c1 should be willing to trust the session characterisation that c3 gives it, whether or not p2 has signed it.

informs the clearinghouse of c1's (snd) signed intention to pay. The clearinghouse then settles c3's (rcv) debt with p2 directly. It also demands payment from c1 (snd) at its own prices. In this scenario, we will not implement settlement, but we will implement the messages necessary for it to be possible.

An aggregator is implemented at the receiver, which only sends marking rate feedback (session characterisations) to the price reactor at the same regularity that the price reactor wishes to re-calculate its target rate. This arrangement is just as efficient as placing the price reactor at the receiver to localise marking rate feedback, but it has the added benefit of secrecy of the buying policy.

3.3.3 Component Distribution

The locations of each M3I component are identified in Figure 3-2 and Figure 3-3 for the receiver pays and sender pays scenarios respectively. Most locations are obvious. The differences between the two diagrams are highlighted in red. The less obvious decisions are justified below.

3.3.3.1 Price Reactor

The price reactor component is closely tied to the application chosen for experiments with the scenario. In our case, this will be a real-time streaming application (see later). In both the RealSystem and Windows Media architectures, the rate control decisions are taken intermittently at the receiver, based on loss measurements at the receiver and on multiple available coding rates advertised to the receiver by the sender at session launch.

This is fine in the receiver pays sub-scenario, but if the sender pays, above we have recommended that the price reactor should ideally be at the sender.

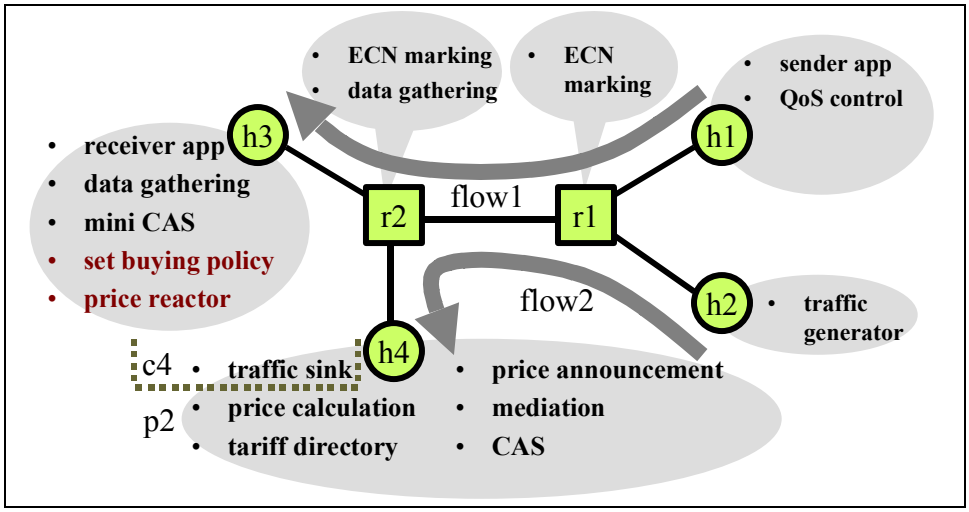

Figure 3-2: Component Distribution – "Receiver Pays" Sub-Scenario

3.3.3.2 QoS Controller

For a streaming application the equivalent of the QoS controller function is simply the sending application configured to stream at a certain rate. Therefore, this "component" is implicitly at the sender.

3.3.3.3 Host 4 (representing price setting and charging function provider systems)

The large collection of functions on this host does not imply these would normally all be instantiated on the same system. The scenario implementation should allow for all of them to be distributed (this is already fully taken into account in the CAS [19] & PM [12] designs).

The figures identify a further two security principles (c4 & p3) who will be considered to operate each function in this scenario:

- c4 (rcv): the traffic sink represents a function that would clearly normally reside on customer machines. This is simply located on h4 to save requiring another PC;
- p3: the clearinghouse operator will be considered to be a third party.

Although the CAS design allows for the charging function to be operated by a third party, this will not be investigated in this scenario, as the trust relationships are already complicated enough.

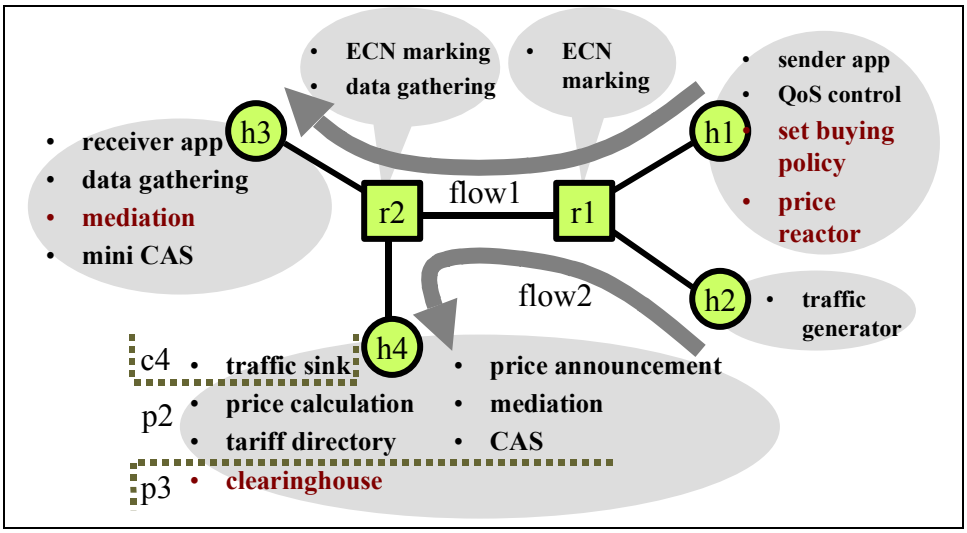

Figure 3-3: Component Distribution – "Sender Pays" Sub-Scenario

3.3.3.4 Data Gathering

Data gathering on h4 is for both accounting and price setting.

Data gathering on h3 is for the price reaction function.

3.3.3.5 Mini Charging and Accounting

In the "sender pays" sub-scenario, the receiver may need to produce charge advice if the issue of encrypted port numbers described earlier cannot be solved. A mini-charging and accounting system is placed on h3 only if this becomes necessary. Its design is discussed in the M3I architecture. This would initially be BT's QoteS, which would be upgraded with the internals of ETH's CAS as the project progresses. BT will maintain this mini-CAS if it proves necessary (unless ETH want to).

3.3.3.6 Mediation

Mediation on h4 is HP's SIU system.

In the “receiver pays” sub-scenario, the data gathering output can be local to h3. But in the “sender pays” scenario, it is aggregated on h3 and regularly reported to price reaction on h1. Thus a logical mediation function is required to do aggregation on h3. Clearly this will be a simple M3I implemented function, not requiring the full capabilities of an HP SIU-like system.

3.3.4 Component Behaviours

Most behaviours are obvious for this scenario, once the description of this scenario in the requirements document has been read. For instance, data gathering will only have to meter ECN marks, and the tariff (executed by the CAS) will only have to charge a fixed price for each mark.

The ECN marking algorithm will initially use the default (linear) RED algorithm in the ALTQ [8] distribution. Other algorithms and various configuration parameters may be tried by manual configuration.

Automatic alteration of queuing parameters (perhaps to exploit short term non-competitive situations) will not be required for this scenario.

The scenario is expected to work for sessions that span a change in price per mark, which is expected to be made to occur by, for example, a change in provider pricing policy.

The price calculation algorithm will initially just be implemented as a manually decided price per mark, until an initial algorithm is available from AUEB that takes account of discovered user utilities and of topology. It is likely this algorithm will be updated and improved over the life of the scenario.

The price reaction algorithm for this scenario is being developed by AUEB, with help from BT. BT have also been working on a price reaction algorithm for the later file transfer aspect of this scenario which has been delivered to AUEB for further development through simulation.

The frequency of check pointing, meter flushing etc. in the charging system will be set through the policy interface, rather than manually (although it may initially be the latter).

3.3.5 Interfaces

3.3.5.1 Service Interfaces

No changes from the description in the M3I Requirements document [1] and the Architecture [4].

3.3.5.2 Business/Policy Interfaces

For tariff communications interfaces, there is nothing unexpected - this scenario uses the interfaces of the Tariff class in the Price Mechanisms Design Pt I [12]. Only the push mode will be required.

For the provider policy interfaces for price setting, we assume those interfaces specified in [12] will be used.

Customer buying policy will be set through the user interfaces being developed at AUEB and BT. AUEB and BT intend to work together to each improve their approach

by learning from the other's. However, different user interfaces will be needed for attended and unattended use — AUEB's could be developed to be more suitable for continuous learning of user preference, while BT's is more suited to initial configuration, perhaps for subsequent unattended use. This is a subject for continued research in the M3I project. An extension to the session description protocol (SDP) [11] is being developed by BT (outside M3I, but the results should be available to M3I in time) to allow buying policies to be included in session descriptions.

QoS control policy will be applied using end-to-end protocols (see below) when agreeing what QoS to request between the ends, or the Microsoft generic QoS (GQoS) API [13] when actually requesting it. In the case of the ECN QoS controller, the protocol handler will not forward an RSVP request to the network, but handle it locally instead. This is similar behaviour to the protocol handler behind the Microsoft GQoS API, which appears like an intserv API to the application, but can send diffserv marked packets to the network if it receives a response from the network [3] telling it to use a certain code point instead of an RSVP request. In the case where the price reactor is at the receiver, this implies its output target rate is applied to the QoS controller at the sender over a remote interface. In the simpler alternative design described in the M3I price reaction design [6] (at the end of section 6 of issue 1.0), the suggested standards-based protocol for this remote interface was the session initiation protocol (SIP) [10]. For either RealSystem or Windows Media it will probably be necessary to use their proprietary protocols for this purpose, which are briefly described in [14], [15].

We will not set meter rules dynamically in this scenario, instead, using manual configuration.

Also, we will not automatically configure the mediation and charging systems.

3.3.5.3 Charging and Accounting Interfaces (Non-Policy)

These are broadly as described in the architecture, under session characterisation. NeTraMet will be used, and discussions are in progress on the exact nature of the interface to this meter.

3.4 Implementation

3.4.1 Platform Technology

Item	Platform
h1,h3	Win 2000
h2	Linux
h4	Win 2000 (for SIU)
r1,r2	FreeBSD

Table 3-1: Platform Technology

All links are 10/100BaseT.

3.4.2 Component Technology

Routers will be configured with active queue management using ECN.

The mediation system will be HP's SIU, with the adaptors and their configurations necessary for interoperation with the interfaces defined above.

Applications are defined in the experiment design below.

All other active components are to be produced within the M3I project, as described in the components sub-sections 3.3.3, 3.3.4 & 3.3.5.

3.5 Experiment Design

3.5.1 Applications

An on demand simplex video streaming application will be used in this scenario. It will consist of an on-demand video server, responding to real-time streaming protocol (RTSP) requests. Videos will each be available in multiple encodings. Buffering of the order of 5-10s will be allowed on the receiver.

From an application integration perspective, the main interfaces to the M3I architecture are those to QoS control. In turn, price reaction has to be configured in to the control loop of the QoS controller. Once this is achieved, the rest of the M3I architecture is much easier to slot into place, as we are free to write the price reactor ourselves.

We have evaluated both RealNetworks RealSystem and Microsoft Windows Media Technologies architectures for use as streaming applications in this scenario [14],[15]. The intention was to use applications that would be prevalent on people's desktops if it came to real customer trials. Although both rate control architectures were fairly similar, the plug-in architecture of RealSystem [17] was found to be more suited to adaptation for M3I's purposes. Its more flexible plug-in structure allows us to add new rate control algorithms more easily (if still rather tortuously).

We have modified RealPlayer to include an API for our own ECN measurements, and to include an M3I price reactor to instruct the sender on which rate to stream at.

3.5.1.1 Configuration and Launch

The launch steps given in the edge control use case in the M3I Architecture (sub-section 2.3.2 in issue 1.0) will be used for both sub-scenarios if possible. However, the edge-centric use case (sub-section 2.3.1) is likely to be necessary in the “sender pays” sub-scenario. This is to solve the issue of encrypted flow metering, where port numbers may not be available to the network providers in order to characterise and separate out the session to be settled by the sender.

The ideal, but extremely difficult aim in the M3I requirements, was to ensure that market control could be added to an application with no source code modifications required, so that, where a network offered M3I capabilities, applications could automatically take advantage of it. This is still our aim, but initially we will re-build each application structure ourselves, so that on launch we have previously arranged for it to work with M3I components.

Later, with this experience behind us, the approach we will probably take, is to modify session launching applications, in order to instantiate the necessary M3I components and linkages in the applications using these sessions. Session launching applications include browsers and the experimental session directory (sdr) [9] or its derivatives. Hopefully this will involve changing (or persuading the vendors to change) just a few key applications, without having to change every other application in the world.

We believe this approach will be fruitful, as more application writers realise that QoS can only be configured per session, not per application, as the results of Bouch [7] become better understood. This trend is already well developed, as witnessed by the converging research on policy controlled communications.

3.6 Experiment Results

No results available yet.

3.7 Conclusions

Not yet available.

3.8 References

[1] Ragnar Andreassen (ed) (Telenor R&D), "Requirements specifications; Reference model", M3I Eu VthFramework Project IST-1999-11429, Deliverable 1 (Jul 2000), <URL:http://www.m3i.org/>.

[2] Ragnar Andreassen & Bob Briscoe (Telenor R&D), “Scenarios responsibility matrix & space coverage,” M3I Eu VthFramework Project IST-1999-11429, Internal management presentation (Dec 2000), <URL:http://www.m3i.org/>.

[3] Yoram Bernet (MS), "Format of the RSVP DCLASS Object", Internet Draft, IETF work in progress (EXPIRED May 2000) (Oct 1999), <URL:http://www.ietf.org/internet-drafts/draft-ietf-issll-dclass-01.txt>.

[4] A. Bouch and M. Sasse and H.G. DeMeer (UC London), "Of packets and people: A user-centred approach to Quality of Service," IEEE IFIP Int'l W'kshp on QoS (IWQoS'00) (May 2000), <URL:http://www.cs.ucl.ac.uk/staff/A.Bouch/42-171796908.ps>.

[5] Bob Briscoe (BT) (ed), "Architecture", M3I Eu VthFramework Project IST-1999-11429, Deliverable 2 (Jul 2000), <URL:http://www.m3i.org/>.

[6] Bob Briscoe (BT) (ed), "Price Reaction Design", M3I Eu VthFramework Project IST-1999-11429, Deliverable 3 Pt II (Jul 2000), <URL:http://www.m3i.org/>.

[7] Bob Briscoe (BT), “Congestion Notification Incentives,” M3I Eu Vth Framework Project IST-1999-11429, Technical Report, Jan 2001, <URL:http://www.m3i.org/>.

[8] Kenjiro Cho, "A Framework for Alternate Queueing: Towards Traffic Management by PC-UNIX Based Routers," in Proc. USENIX 1998 Annual Technical Conference, New Orleans, LA, US (Jun 1998).

[9] Mark Handley (UC London), "On Scalable Internet Multimedia Conferencing Systems," UC London PhD thesis (14 Nov 1997), <URL:http://www.aciri.org/mjh/thesis.ps.gz>.

[10] M. Handley and H. Schulzrinne and E. Schooler and J. Rosenberg, "SIP: Session Initiation Protocol," IETF RFC 2481 (Mar 1999), <URL:rfc2543.txt>.

[11] Mark Handley and Van Jacobsen, "SDP: Session Description Protocol," IETF RFC 2327 (Mar 1998), <URL:rfc2327.txt>.

[12] Martin Karsten (TUD) (ed), "M3I Pricing Mechanisms (PM); Design" M3I Eu Vth Framework Project IST-1999-11429, Deliverable 3, Jun 2000, <URL:http://www.m3i.org/>.

[13] Microsoft, "Windows 2000 Features in Support of Differentiated Services," Microsoft white paper (Jul 2000),

<URL:http://www.microsoft.com/windows2000/library/howitworks/communications/trafficmgmt/diffserv.asp>.

[14] Peterjan van Njeuwenhuizen, "Real System Rate Control", BT Technical Report, (17 Oct 2000).

[15] Peterjan van Nieuwenhuizen, "Windows Media Technologies Rate Control", BT Technical Report, (17 Oct 2000).

[16] K. K. Ramakrishnan (AT&T Labs Research) and Sally Floyd (LBNL), "A Proposal to add Explicit Congestion Notification (ECN) to IP", IETF RFC 2481 (Jan 1999), <URL:rfc2481.txt>.

[17] RealNetworks <URL:http://www.realnetworks.com/devzone/>.

[18] David Songhurst, "Utility, willingness to pay and rate control," M3I Eu VthFramework Project IST-1999-11429, Internal Technical Report (25 May 2000), <URL:http://www.m3i.org/>.

[19] Burkhard Stiller (ETH) (ed), "Charging and Accounting System (CAS) Design," M3I Eu Vth Framework Project IST-1999-11429, Deliverable 4, Jun 2000, <URL:http://www.m3i.org/>.

3.9 Abbreviations

API	-	Application programming interface
AUEB	-	Athens University of Economics and Business
BT	-	British Telecommunications plc
CE	-	Congestion Experienced (part of the ECN protocol)
DiffServ	-	IETF differentiated services
DPH	-	Dynamic Price Handler
ECN	-	Explicit congestion notification
Echo	-	Echo of congestion experienced (part of the ECN protocol)
GSP	-	Guaranteed Service Provider
IETF	-	Internet Engineering Task Force
GUI	-	Graphical user interface
IntServ	-	IETF integrated services
IP	-	Internet Protocol
ISP	-	Internet Service Provider
M3I	-	Market Managed Multi-service Internet
QoS	-	Quality of Service
QoteS	-	QoS around the edge
RSVP	-	Resource Reservation Protocol
TCP	-	Transmission Control Protocol
VoD	-	Video on Demand

4 Guaranteed Stream Provider for IntServ Scenario (GSPd)

This section describes a particular incarnation of the M3I system, termed Guaranteed Stream Provider over IntServ scenario, or GSP/IntServ for short. This scenario serves as a reference to evaluate accomplishments and trade-offs of other M3I scenarios and additionally, as a further test for M3I's flexibility. By assuming a fixed-priced, guaranteed service network provider, the GSP functionality is essentially empty. The network technology consists of RSVP/IntServ-capable routers, which communicate with the market management system by means of COPS. Because service invocations are explicitly signalled, the effort for metering and the load imposed on the market management system are assumed to be rather limited.

4.1 Introduction

This section describes the *Guaranteed Stream Provider over IntServ (GSP/IntServ)* scenario developed in the M3I project. This scenario represents the "classical" Integrated Services approach [BCS94] to provide QoS for individual application flows within a packet-switched network. The purpose of this scenario is to serve as a reference for comparative performance tests with the more innovative approaches pursued in M3I, specifically GSP/ ECN (see [And00] for details).

Besides the pricing and charging system, the technical and implementation focus in M3I is on the edge device between a network provider and its customer(s). Design of this scenario is straightforward, as indicated in later sections of this document. The network technology part is mainly realised. A first version of this scenario's implementation is planned to be available at the end of April 2001.

The partners involved in this scenario are TUD for network technology and pricing system, ETH for the CAS, Telenor for performance experiment design, BT for end-systems technology and HP for system integration.

4.2 Overview and Motivation

The major goals of the M3I project are twofold. First, we want to develop a pricing and charging system, which flexibly and efficiently supports a large variety of service offerings. Second, we plan to investigate the potential of providing stable resource allocation over a highly dynamic network system, which manages congestion by transmitting congestion indications (ECN marks) to edge respectively end systems. The GSP/IntServ scenario supports both goals as follows. By offering a fixed-price, admission-controlled service, the M3I pricing and charging system is assessed to support such services. Since IntServ is already known to provide stable and reliable QoS to individual application flows, it is further used as a reference system to analyse the achievements and trade-offs that can be accomplished by GSP/ECN.

4.3 Design

The following sections describe the overall design of this scenario. First, an overview is given in terms of the M3I architecture [Bri00]. Then, a use case is presented to further illustrate the scenario.

4.3.1 Overview

The scenario is illustrated in the notion of the M3I architecture in Figure 4-1. Generic components and interfaces are as described in [Bri00]. A use case referring to the numbers in this figure is presented in the next section.

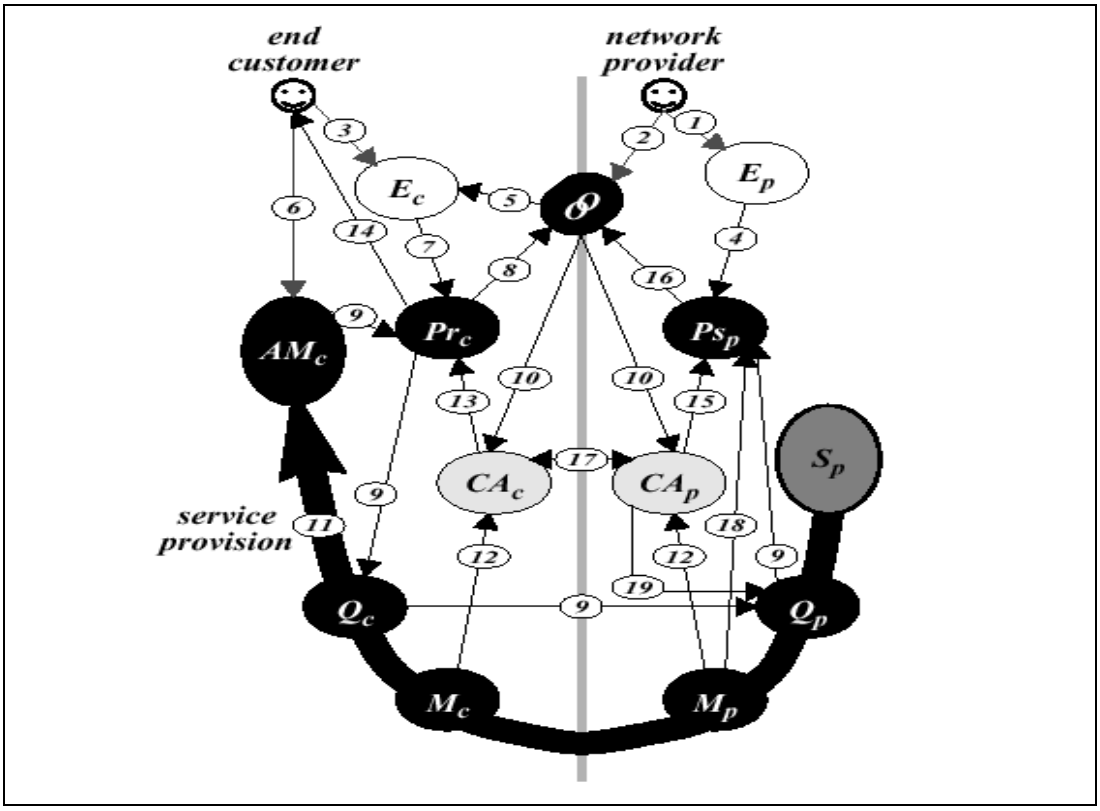

Figure 4-1: Overview of the GSP/IntServ Scenario

4.3.2 Use Case

We present a detailed use case, consisting of a provider and a customer, for this scenario. Individual interactions of this use case are shown in Figure 4-1 by numbered arrows. The provider sets up its enterprise policy control (1) and offer directory (2), and the customer configures its enterprise policy control, respectively (3). The provider's enterprise policy control configures the price setting module (4) and an offer is received from the customer (5), which launches an application (6). The customer's enterprise policy control configures the price reaction module of the customer, which chooses an appropriate tariff offer (8) and appropriately configures the QoS manager to request the corresponding service (9). The tariff offer is reported to both charging and accounting systems (10). Then, the actual service is delivered (11). During service invocation, current usage is metered and reported to the charging and accounting systems (12), which report to price reaction (13) and eventually the user (14) on the customer's side, as well as to the price setting module on the provider's side (15). Price setting in turn might update the currently offered tariffs (16), however, in this scenario it is expected to

happen only on a rather long time-scale. If both provider and customer run independent charging systems, the calculated charges are reconciled periodically (17). Metering of usage is reported to the price setting module (18) and finally, the charging and accounting system needs a feedback channel (19) to the provider's QoS manager in order to potentially prohibit service delivery for certain customers.

4.4 Implementation

An overview about the implementation of the GSP/IntServ scenario is given in Figure 4-2. Only the provider's system is shown here. Implementation details of the components and interfaces are discussed in this section.

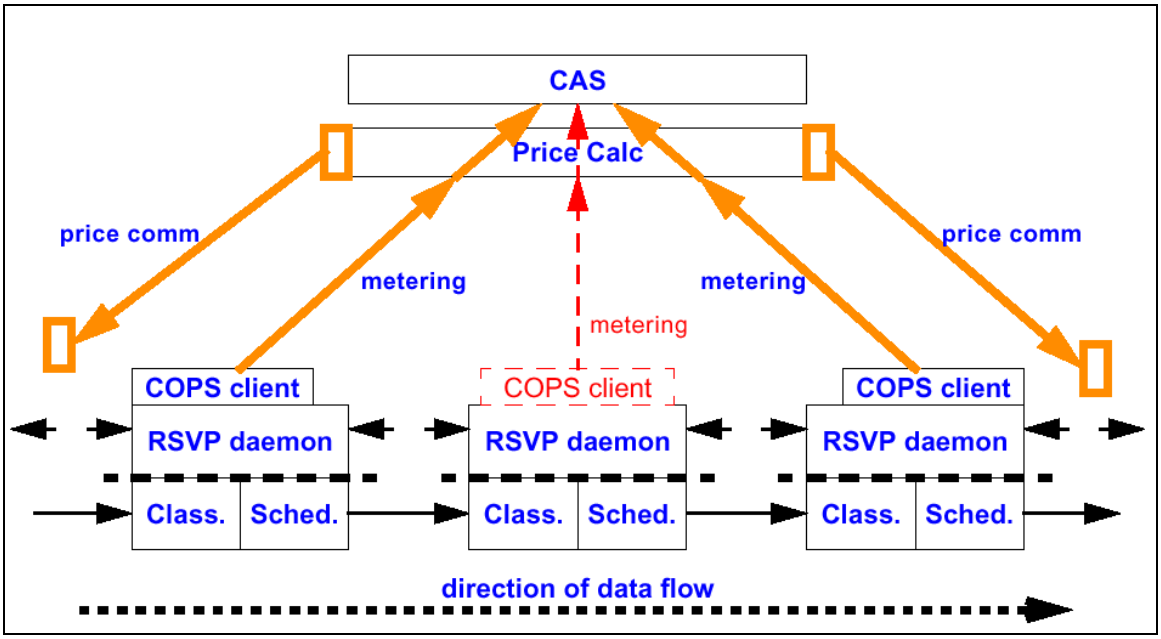

Figure 4-2: Implementation Overview of the GSP/IntServ Scenario

4.4.1 Components

In this section, the implementation details of system components as outlined in [Kar00b] are described.

4.4.1.1 IP Router

FreeBSD-based router including ALTQ [Cho98], employing HFSC scheduling [SZN97]. Resources are allocated to IntServ services classes according to [SPG97] respectively [KSWS99].

4.4.1.2 Resource Broker

KOM RSVP daemon [Kar00a], co-located with the router

4.4.1.3 Gateway = Guaranteed Stream Provider

No functionality.

4.4.1.4 End-System

There are two types of end systems required for this scenario:

- application systems to run sensible application services:
  - FreeBSD/Linux end system, running KOM RSVP and simple audio/video streaming
  - Windows end system, requesting resources through RSVP
- traffic generators that emit background traffic to load the network:
  - FreeBSD/Linux end systems, using netperf or other tools, or
  - Telenor's traffic generator

4.4.1.5 Data Gathering

KOM RSVP daemon on router, extended to export RSVP state information via COPS.

4.4.1.6 Mediation

Translation (and storage) of COPS events into NME, support of push model for Price Calculation.

4.4.1.7 Price Calculation

Java framework as described in [Kar00b], configured by hand (user interface or hard-coded), potentially co-located with Charging and Accounting System.

4.4.1.8 Price Communication

Java-based middleware as described in [Kar00b]. Tariff Directory, for pull-mode access from clients. C++ client for FreeBSD/Linux end system alternative.

4.4.1.9 Charging and Accounting System

According to M3I Deliverable 4 [Sti00].

4.4.2 Interface Definition

4.4.2.1 M3I-3 (Mediation - CAS)

Unidirectional COPS: PEP→PDP according to RFCs 2748/2749 [DBC+00,HBC+00], using only the Resource-Allocation context.

4.4.2.2 M3I-5 (Price Communication - CAS)

The main interface to obtain tariff information is defined by class TariffReader and appropriate subclasses of class Tariff. See the documentation of the price mechanisms package for details.

4.4.2.3 M3I-10 (CAS - QoS Component)

(later): back channel from COPS: PDP→PEP similar to RFCs 2748/2749 [DBC+00, HBC+00], but in an optimistic mode (only rejections are transmitted), using only the Resource-Allocation context.

4.4.2.4 M3I-12 (Data Gathering -> Mediation)

Unidirectional COPS: PEP→PDP according to RFCs 2748/2749 [DBC+00,HBC+00], using only the Resource-Allocation context.

4.5 Experiment Description

The main purpose of experimenting is to test basic functional correctness of all components. Furthermore, it is interesting to find out, whether and to what extend application flows receive their QoS objective under load conditions. As shown in the topological scenario in Figure 4-3, at least 7 PCs are needed to carry out basic experiments. It is very likely that more are required to generate reasonable data.

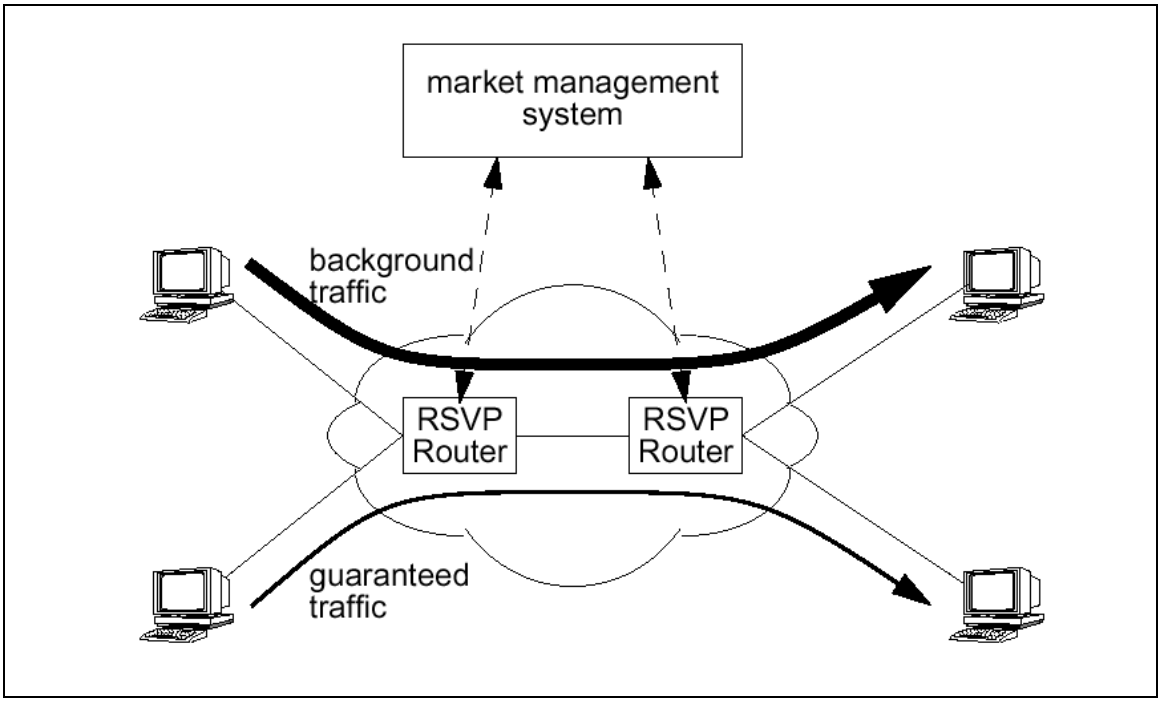

Figure 4-3: Experiment Set-up for GSP/IntServ

4.5.1 Functional Experiments

The following steps are required to test the basic functionality of this scenario. All programs mentioned here are part of the RSVP distribution package. Both the guaranteed traffic as well as the background traffic are essentially constant-bit-rate transmissions, so they are mainly useful for functional experiments:

1) run “RSVPD” on router machines;
2) run “sendVideo” and “receiveVideo” on reserved path end systems;
3) run “sender” and “receiver” on unreserved path end systems;
4) verify that video transmission receives bandwidth, despite cross traffic;
5) verify that sensible prices are produced and transmitted;
6) verify that market management system (essentially CAS) stores correct data.

4.5.2 Performance Experiments

Given our current knowledge, it is clear that one bottleneck of this scenario will be the classification and scheduling modules on the routers, namely ALTQ. A large number of traffic generators have to generate reservation requests and appropriate data. Receivers must record the level of QoS achievement despite cross traffic. The open question is how many reserved flows of certain sizes can be efficiently supported in this scenario. We have to keep in mind that the ALTQ implementation (certain scheduling disciplines) is known to be not optimal in terms of implementation efficiency. Another open question is the appropriate performance of the market-management system.

4.6 Experiment Results

No results available yet.

4.7 Conclusions

Not yet available.

4.8 References

[And00] Ragnar Andreassen, editor. *M3I Deliverable 1 - Requirements Specification*. June 2000.

[BB00] Anders P. Bergsten and Niklas Borg. Implementation and Evaluation of the Common Open Policy Service (COPS) Protocol and its use for Policy Provisioning, 2000. http://extwww.lulea.trab.se/i-lab/cops/index.htm.

[BCS94] Robert Braden, David Clark, and Scott Shenker. RFC 1633 - Integrated Services in the Internet Architecture: an Overview. Informational RFC, June 1994.

[Bri00] Bob Briscoe, editor. *M3I Deliverable 2 - System Architecture / Overview (Initial)*. June 2000.

[Cho98] Kenjiro Cho. A Framework for Alternate Queueing: Towards Traffic Management by PC-UNIX Based Routers. In *Proceedings of USENIX 1998 Annual Technical Conference, New Orleans, LA, USA*, June 1998.

[DBC+00] David Durham, Jim Boyle, Ron Cohen, Shai Herzog, Raju Rajan, and Arun Sastry. RFC 2748 - The COPS (Common Open Policy Service) Protocol. Standards Track RFC, January 2000.

[HBC+00] Shai Herzog, Jim Boyle, Ron Cohen, David Durham, Raju Rajan, and Arun Sastry. RFC 2749 - COPS usage for RSVP. Standards Track RFC, January 2000.

[Int00] Intel. Free COPS implementation, 2000.

[Kar00a] Martin Karsten. KOM RSVP Protocol Engine, 2000. http://www.kom.e-technik.tu-darmstadt.de/rsvp/.

[Kar00b] Martin Karsten, editor. *M3I Deliverable 3, Part I - Pricing Tools Design*. June 2000.

[KSWS99] Martin Karsten, Jens Schmitt, Lars Wolf, and Ralf Steinmetz. Provider-Oriented Linear Price Calculation for Integrated Services. In *Proceedings of the Seventh IEEE/IFIP International Workshop on Quality of Service (IWQoS'99), London, UK*, pages 174–183. IEEE/IFIP, June 1999. ISBN 0-7803-5671-3.

[SPG97] Scott Shenker, Craig Partridge, and Roch Guerin. RFC 2212 - Specification of Guaranteed Service. Standards Track RFC, September 1997.

[Sti00] Burkhard Stiller, editor. *M3I Deliverable 4 - CAS Design*. June 2000.

[SZN97] Ion Stoica, Hui Zhang, and T. S. Eugene Ng. Hierarchical Fair Service Curve Algorithm for Link-Sharing, Real-Time and Priority Service. *ACM Computer Communication Review*, 27(4), October 1997. Proceedings of SIGCOMM'97 Conference.

[Vov00] Vovida. Free COPS implementation, 2000.

4.9 Abbreviations

ALTQ	-	Alternate Queueing
CAS	-	Charging and Accounting System
COPS	-	Common Open Policy Service
ECN	-	Explicit Congestion Notification
HFSC	-	Hierarchical Fair Service Curve
GSP	-	Guaranteed Stream Provider
IntServ	-	Integrated Services
M3I	-	Market Managed Multi-service Internet
NME	-	Normalised Meter Event
QoS	-	Quality of Service
PDP	-	Policy Decision Point
PEP	-	Policy Enforcement Point
RSVP	-	Resource ReSerVation Protocol
SIU	-	Smart Internet Usage

4.10 Appendix - Current Implementation Status

4.10.1 Available Modules

- plain KOM RSVP implementation
- various COPS implementations [BB00,Int00,Vov00]
- ALTQ implementation providing HFSC scheduling
- RSVP-capable video-streaming application
- traffic generators: netperf and periodic CBR traffic generator

4.10.2 To Do

- choose and extend COPS implementation to carry RSVP requests
- extend RSVP daemon to emit COPS requests
- COPS mediation into NMEs for CAS and Price Calculation
- integrate Price Communication and CAS
- develop Tariff Directory
- integrate Price Communication with end system application
- *later*: extend RSVP daemon to handle COPS rejections

4.10.3 Available System

Basic network QoS system available. No M3I modules yet.

5 Dynamic Priced, Guaranteed Stream Provider for Explicit Congestion Notification Scenario (GSPx)

This chapter describes a particular incarnation of the M3I system, termed *Dynamically Priced Guaranteed Stream Provider over ECN* scenario, or *DP-GSP/ECN* for short. This scenario serves as an intermediate development step towards the pure GSP/ECN scenario and additionally, as a further test for M3I's flexibility. As opposed to the GSP/ECN scenario, the DP-GSP does not attempt to stabilise prices, but only to offer a signalled and admission-controlled network service. Thereby, the problem domains of providing stable QoS and providing stable prices are decoupled. The latter is off-loaded to the customer of the GSP. The network technology is given by ECN-capable core routers, which are surrounded by RSVP-capable GSP-systems. Service requests between end-customers and the GSP are transmitted using RSVP and tunnelled through the core network. The interaction between network technology and market management system is expected to be high within the core network, while the load imposed by metering of signalled service requests is presumably limited.

5.1 Introduction

This section describes the *Dynamically Priced Guaranteed Stream Provider over ECN (DP-GSP/ECN)* scenario developed in the M3I project. In this scenario, a GSP offers a guaranteed service with dynamic prices to customers. The main purpose of this scenario is to serve as an intermediate development step towards GSP/ECN.

Besides the pricing and charging system, the technical and implementation focus in M3I is on the edge device between a network provider and its customer(s). Design of this scenario is straightforward, as indicated in later sections of this document. The network technology part is partially realised. The "ECN-side" of this scenario is intended to be identical to scenario GSP/ECN. The "GS-side" is intended to be similar to GSP/IntServ. The GSP box itself is supposed to share the technical basis with GSP/ECN. A first version of this scenario's implementation is planned to be available at the end of April 2001.

The main partners involved in this scenario are TUD for network technology and pricing system, ETH for the CAS, BT for end-systems technology and HP for system integration.

5.2 Overview and Motivation

The major goals of the M3I project are twofold. First, we want to develop a pricing and charging system, which flexibly and efficiently supports a large variety of service offerings. Second, we plan to investigate the potential of providing stable resource allocation over a highly dynamic network system, which manages congestions by transmitting congestion indications (ECN marks) to edge respectively end systems. The DP-GSP/ECN scenario supports both goals as follows. By offering a dynamic-price, admission-controlled service, the M3I pricing and charging system is assessed to support such services. Furthermore, this scenario represents an approach to provide predictable QoS and dynamic prices for individual application flows over an ECN-priced network. Thereby, it serves as an intermediate development step towards the more innovative approaches pursued in M3I, specifically GSP/ECN (see [And00] for details). In this scenario, the focus is on the GSP between a ECN-priced network provider and

customers who request a predictable performance of network transmission, but are willing to handle dynamic prices.

Research results suggest we may achieve a stable rate allocation within a best-effort network with FIFO queues by transmitting shadow prices to the end systems and relying on economic rationale at those end systems. The statistical mechanism to transmit shadow prices is supposed to be ECN [RF99]. In theory, in has been shown that such a market system provides stable, optimal and proportional fair resource allocation [Kel00]. A further scenario is given by using edge devices at network borders to hide the dynamics of this process to end systems and essentially to transform the market-managed resource allocation into a signalling-based QoS assignment (GSP/ECN) [GK99]. Besides the general open issues regarding the time scales of operation and resulting concerns about actual stability, two additional aspects can be identified when applying this concept to reality. These two aspects are termed *multicast problem* and *commitment problem* in this document.

5.2.1 Multicast Problem

The multicast problem applies to rate allocation through transmission of statistical shadow price information in general. If packets are replicated within core nodes, different receiver (edge-)systems might experiences different levels of congestion along the respective path. The amount of resulting feedback leads to a multicast implosion problem and furthermore, the highest congested path essentially determines the rate allocation for all receivers. Consequently, no heterogeneity can be supported. This issue is depicted in Figure 5-1, which shows two receivers, A and B, both experiencing different levels of congestion.

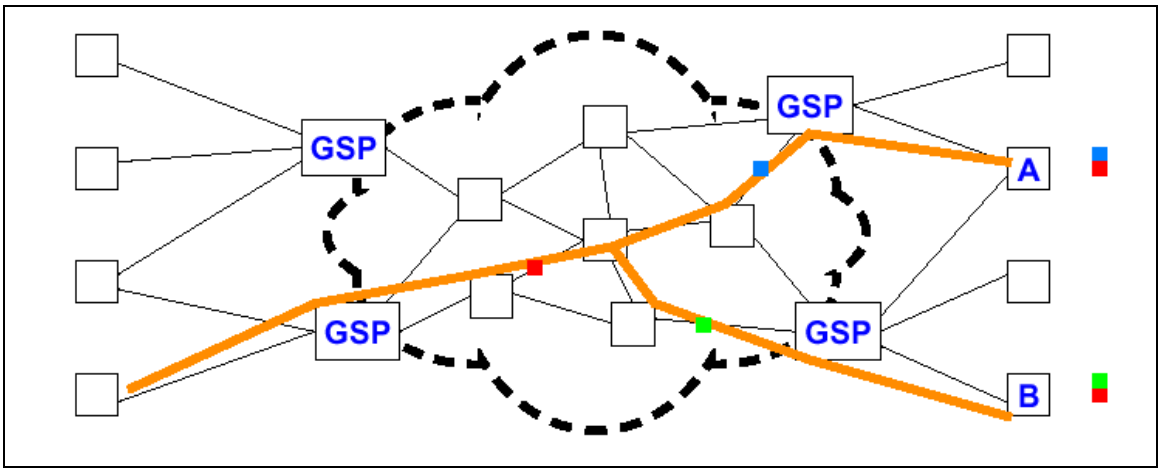

Figure 5-1: Multicast Problem

It is debatable whether this problem exists in the first place. If interior nodes are multicast-capable, they have to keep state information per multicast group. In this case, it can be argued that additional management state to address multicast heterogeneity does not increase the overall level of complexity. However, in the rest of this section, we assume the relevance of this problem.

5.2.2 Commitment Problem

The commitment problem specifically applies to the GSP scenario. If several GSPs have committed to transmit traffic at certain rates, but end systems have not fully exploited their shares, the economic and technical risk of these commitments is based on incorrect information. When end systems begin to transmit at the allocated rate, the

resulting congestion level in the network is much higher than the experienced level at the time of commitment and thus, it is not guaranteed that all commitments can be satisfied. This potential problem is illustrated in Figure 5-2.

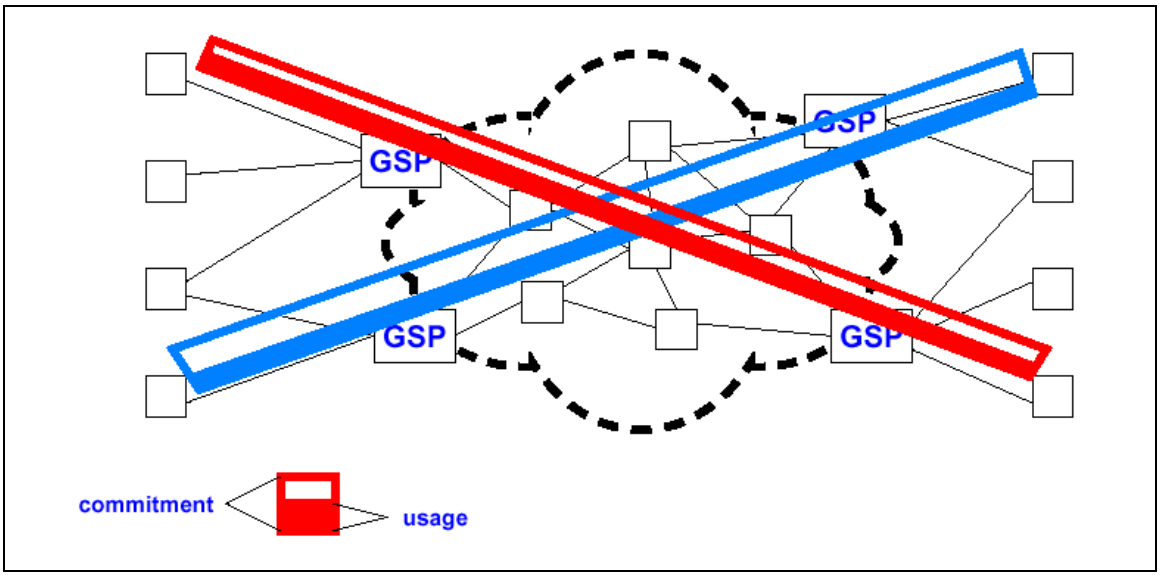

Figure 5-2: Commitment Problem

A straw-man solution to this problem is for the GSP to insert artificial traffic whenever an end system does not use its rate allocation. For several reasons, this solution does not seem favourable. Firstly, it leads to inefficient resource usage, because the artificial traffic competes with otherwise legitimate network usage. Additionally, in the case of uncoordinated GSPs, such a requirement can hardly be enforced.

5.2.3 Advantages of DP-GSP/ECN

The DP-GSP/ECN provides stable resource allocation to its customers, albeit coupled with highly dynamic prices. It addresses both problems mentioned above. Additionally, it serves as an intermediate development step, synchronisation point and fallback solution for the M3I project.

Multicast Problem

The DP-GSP/ECN addresses the multicast problem by enabling additional solutions. Transmission to a heterogeneous multicast group can be mapped onto several homogeneous local groups. As an extreme, each egress node is served through unicast transmission. By employing the DP-GSP/ECN, mapping strategies, such as described in the context of IntServ of ATM [SWKS99], can be applied. Based on the end systems' price thresholds, an egress node can select between a discrete set of transmission groups to choose the optimal resource allocation.

Commitment Problem

The commitment problem is leveraged to the end systems by not guaranteeing stable prices. End systems request services up a price threshold and their service is pre-empted if the actual price exceed this threshold. The main service of the GSP is to alleviate end systems from the details of shadow price transmission and its actual implementation. Furthermore, the GSP acts as a protection gateway for the interior network by ensuring that overall demand does not exceed the capacity. Such protection is an essential requirement for the system's stability [Kel00].

5.3 Design

The following sections describe the overall design of this scenario. First, an overview is given in terms of the M3I architecture document [Bri00]. Then, a use case is presented to further illustrate the scenario.

5.3.1 Overview

The scenario is illustrated in the notion of the M3I architecture in Figure 5-3. Generic components and interfaces are as described in [Bri00]. A use case referring to the numbers in this figure is presented in the next section.

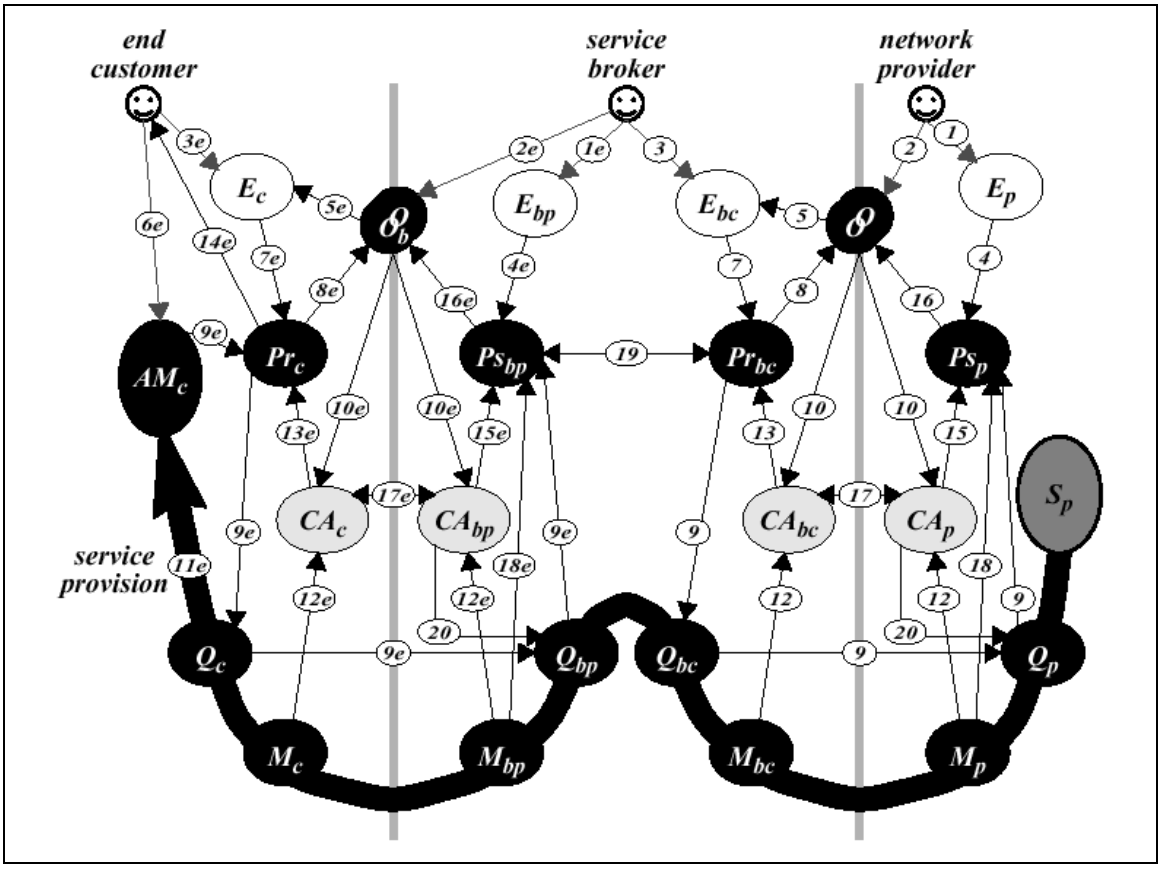

Figure 5-3: Overview of the DP-GSP/ECN Scenario

5.3.2 Use Case

We present a detailed use case for this scenario. Individual interactions of this use case are shown in Figure 5-3 by numbered arrows. This use case consists of three entities, which are given by a network provider, a service broker (the GSP) and a customer. All steps related to the end-customer a suffixed with an “e”. Both network provider and service broker set up their enterprise policy control (1) and offer directory (2), and the service broker and customer configure their enterprise policy control, respectively (3). Each enterprise policy control configures the respective price setting module (4) and an offer is received from the service broker respectively the customer (5), which launches an application (6e). Each enterprise policy control configures the respective price reaction module, which in turn chooses an appropriate tariff offer (8) and appropriately configures the QoS manager to request the corresponding service (9). Both QoS managers of the service broker have to cooperate in order to deliver the guaranteed service to the customer. The tariff offer is reported to all instances of the charging and

accounting system (10). Then, the actual service is delivered (11). During service invocation, current usage is metered and reported to the charging and accounting systems (12), which report to price reaction (13) and eventually the user (14) on the customer's side, as well as to both price setting modules (15). The service provider's price setting is expected to regularly update the currently offered tariffs (16e) whereas the network provider might only occasionally update its prices (16). If both provider and customer run independent charging systems, the calculated charges are reconciled periodically (17). Metering of usage is reported to the price setting module (18) and the service broker uses the prices experienced from the ECN-provided to update its own price setting module (19). Essentially, the service broker's price setting module uses a simple algorithm to calculate prices for session-oriented service invocations from ECN-based prices and the currently experienced marking. It makes no attempt to stabilise prices. Finally, the charging and accounting system needs a feedback channel (20) to the provider's QoS manager in order to potentially prohibit service delivery for certain customers.

5.4 Implementation

An overview about the implementation of the DP-GSP/ECN scenario is given in Figure 5-4. Only the provider's system is shown here. Implementation details of the components and interfaces are discussed in this section.

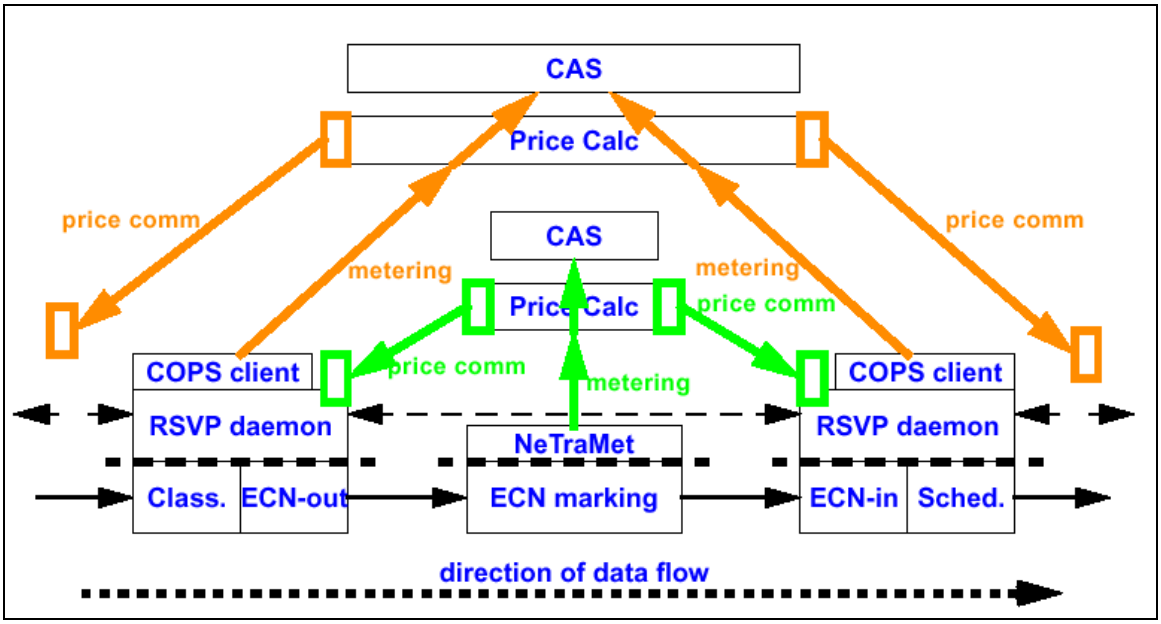

Figure 5-4: Implementation Overview of the DP-GSP/ECN Scenario

5.4.1 Components

In this section, the implementation details of system components as outlined in [Kar00] are described.

5.4.1.1 IP Router

FreeBSD-based router including ALTQ [Cho98], employing RED [FJ93].

5.4.1.2 Resource Broker

No functionality.

5.4.1.3 Gateway = Guaranteed Stream Provider

FreeBSD-based router including ALTQ and KOM RSVP

ECN(CE)-analyser and -classifier, implemented as ALTQ kernel module (NB: algorithmic details are to be defined later.)

ECN(ECT)-classifier and -marker, implemented as ALTQ kernel module

5.4.1.4 End-System

There are two types of end systems required for this scenario.

- application systems to run sensible application services:
  - FreeBSD/Linux end system, running KOM RSVP and simple audio/video streaming
  - Windows end system, requesting resources through RSVP
- traffic generators that emit background traffic to load the network:
  - FreeBSD/Linux end systems, using netperf or other tools, or
  - Telenor's traffic generator

5.4.1.5 Data Gathering

ECN-priced network

ECN-capable NeTraMet meter

Guaranteed Stream Provider

KOM RSVP daemon on router, extended to export RSVP state information

ECN-capable NeTraMet meter and/or ECN(CE)-analyser

5.4.1.6 Mediation

ECN-priced network

SIU for aggregation of load information from different NeTraMet readers, support push model for Price Calculation. Algorithmic details are to be defined later

Guaranteed Stream Provider

Translate (and store) COPS events into NME, support push model for Price Calculation. The question whether to use HP's SIU or own code, or even to leave empty is open at the time of writing.

5.4.1.7 Price Calculation

ECN-priced network

Java Framework as described in [Kar00], internal algorithms to be defined later

Guaranteed Stream Provider

Java Framework as described in [Kar00], internal algorithms to be defined later

5.4.1.8 Price Communication

ECN-priced network

Java-based middleware as described in [Kar00]. Push mode distribution of current price information to the GSP.

Guaranteed Stream Provider

Java-based middleware, as described in [Kar00]. Tariff Directory, for pull-mode access from clients. C++ client for FreeBSD/Linux end system alternative.

5.4.1.9 Charging and Accounting System

According to M3I Deliverable 4 [Sti00].

5.4.2 Interface Definition

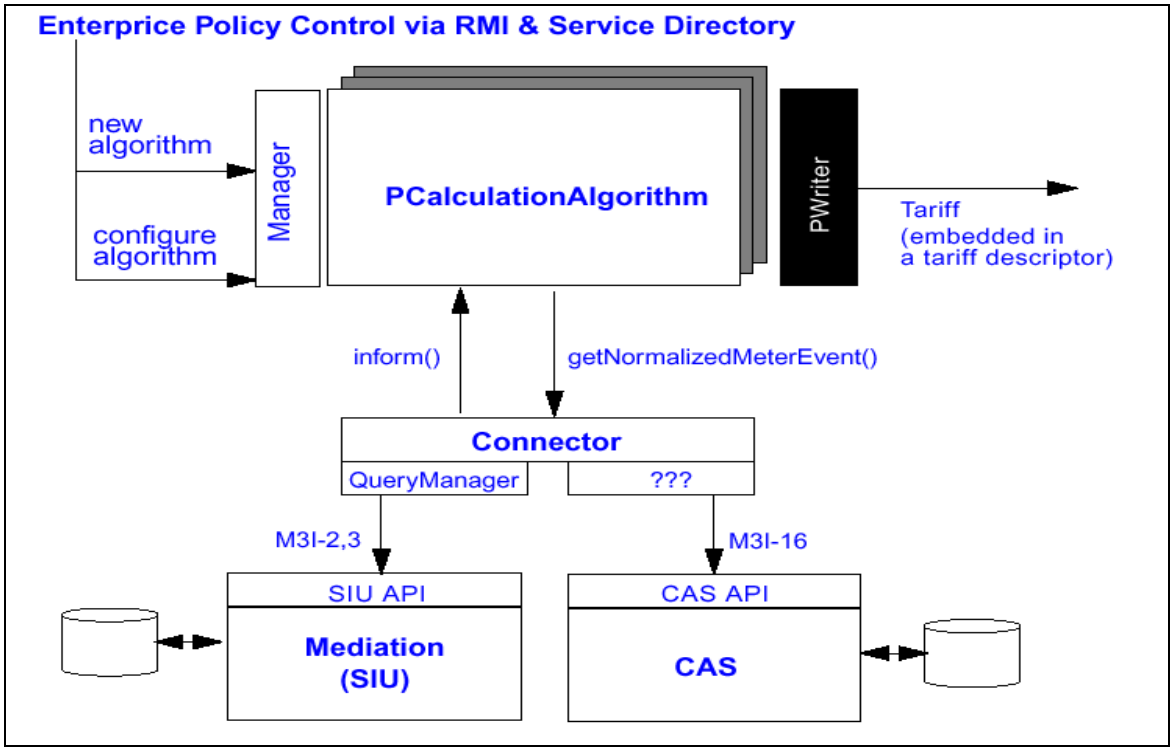

Figure 5-5: Interface of Price Calculation Module

5.4.2.1 M3I-2 (Mediation - Price Calculation)

ECN-priced network

The Price Calculation Module is connected to the Mediation and to the CAS via a connector class. For communication with SIU as mediation module, the connector uses SIU's Query Manager that will retrieve the necessary information from SIU using it's built-in RMI methods. The information is stored in an instance of SIU's NormalizedMeterEvent (NME) class. The process is triggered by a price calculation algorithm requesting new data about current network usage using a local method call to getNormalizedMeterEvent() of class Connector. The NME is pulled out of SIU and passed on to the price calculation algorithm. The content of the NME is specified during the configuration of SIU (M3I-14) and must fulfil the requirements of the price calculation algorithm. This is shown in Figure 5-5. One simple example for the content of such an NME is the average queue length (measured over a configured interval). Detailed specification of the exchanged information requires knowledge about particular price calculation algorithms.

Guaranteed Stream Provider

Unidirectional COPS: PEP→PDP according to RFCs 2748/2749 [DBC+00,HBC+00], using only the Resource-Allocation context.

5.4.2.2 M3I-3 (Mediation - CAS)

ECN-priced network

See Section 5.4.2.1.

Guaranteed Stream Provider

Unidirectional COPS: PEP→PDP according to RFCs 2748/2749 [DBC+00,HBC+00], using only the Resource-Allocation context.

5.4.2.3 M3I-5 (Price Communication - CAS)

The main interface to obtain tariff information is defined by class TariffReader and appropriate subclasses of class Tariff. See documentation of price mechanisms package.

5.4.2.4 M3I-7 (Price Communication - Price Reaction)

The main interface to obtain tariff information is defined by class TariffReader and appropriate subclasses of class Tariff. See documentation of price mechanisms package.

5.4.2.5 M3I-10 (CAS - QoS Component)

ECN-priced network

Not used.

Guaranteed Stream Provider

(later): back channel from COPS: PDP→PEP similar to RFCs 2748/2749 [DBC+00, HBC+00], but in an optimistic mode (only rejections are transmitted), using only the COPS Resource-Allocation context.

5.4.2.6 M3I-12 (Data Gathering -> Mediation)

ECN-priced network

SIU SNMP collector

Exchange of managed objects via SNMP according to RFC 2720, extended to allow counting of ECN marks, for details see Metering Tools/mibs/rtfm-mib-BT.txt in BT's QoteSV distribution.

Load measurement (for price calculation) → obtain/set RED configuration in IP router?

Guaranteed Stream Provider

Unidirectional COPS: PEP→PDP according to RFCs 2748/2749 [DBC+00,HBC+00], using only the Resource-Allocation context.

5.5 Experiment Description

The main purpose of experimenting is to test whether and to what extend application flows receive their QoS objective under load conditions. As illustrated in the topological scenario in Figure 5-6, at least 9 PCs are needed to carry out such experiments. It is very likely that more are required to generate reasonable data.

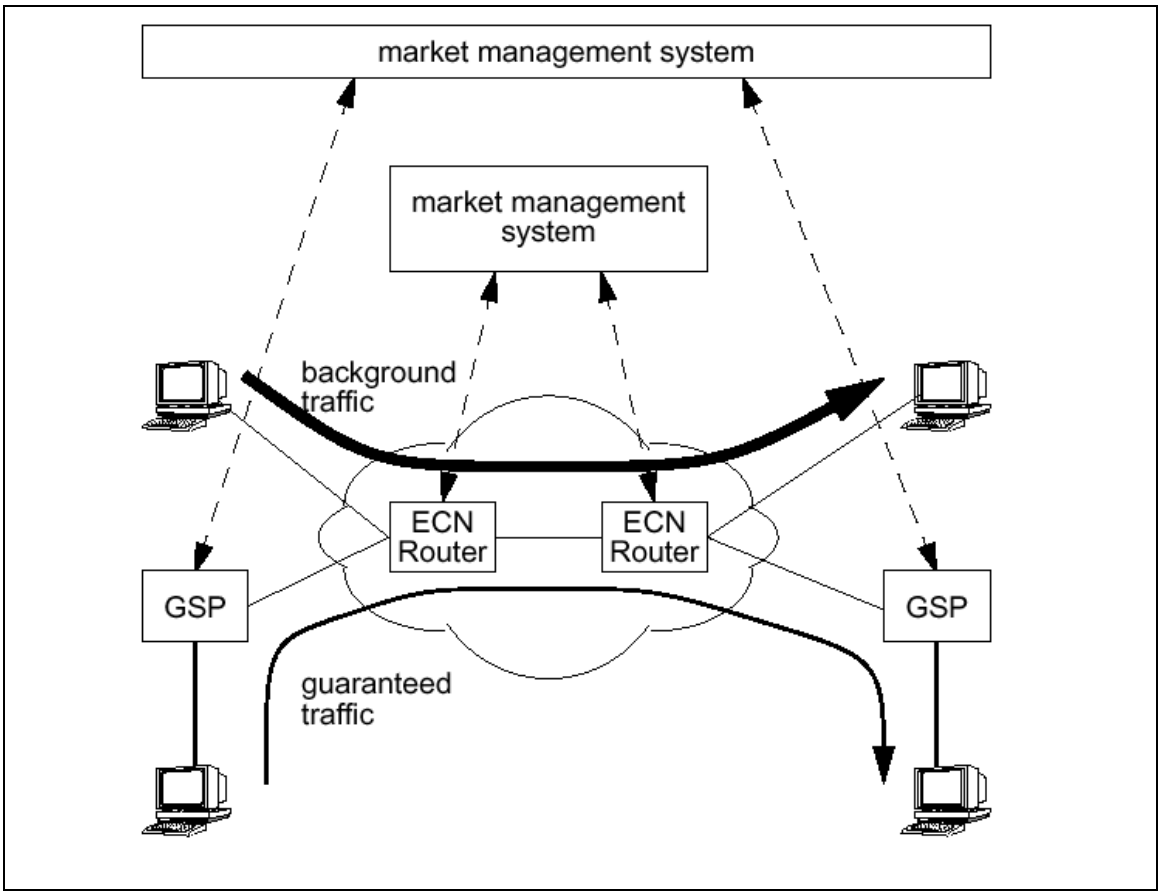

Figure 5-6: Experiment Set-up for DP-GSP/ECN

5.5.1 Functional Experiments

The following steps are required to test the basic functionality of this scenario. All programs mentioned here are part of the RSVP distribution package. Both the guaranteed traffic as well as the background traffic are essentially constant-bit-rate transmissions, so they are mainly useful for functional experiments:

1) run “RSVPD” on router machines;
2) run “sendVideo” and “receiveVideo” on reserved path end systems;
3) run “sender” and “receiver” on unreserved path end systems;
4) verify that video transmission receives bandwidth, despite cross traffic;
5) verify that sensible prices are produced and transmitted;
6) verify that both market management systems (core net and GSP) store correct data.

5.5.2 Performance Experiments

Given our current knowledge, the major open question is given by the trade-off in terms of resource utilisation that has to be made in the core network in order to achieve desirable QoS objectives. A large number of traffic generators have to generate reservation requests and appropriate data. Receivers must record the level of QoS achievement despite cross traffic. The open question is how many reserved flows of certain sizes can be efficiently supported in this scenario. Another open question is the appropriate performance of both instances of the market-management system.

5.6 Experiment Results

No results available yet.

5.7 Conclusions

Not yet available.

5.8 References

[And00] Ragnar Andreassen, editor. *M3I Deliverable 1 - Requirements Specification*. June 2000.

[BB00] Anders P. Bergsten and Niklas Borg. Implementation and Evaluation of the Common Open Policy Service (COPS) Protocol and its use for Policy Provisioning, 2000. http://extwww.lulea.trab.se/i-lab/cops/index.htm.

[Bri00] Bob Briscoe, editor. *M3I Deliverable 2 - System Architecture / Overview (Initial)*. June 2000.

[Cho98] Kenjiro Cho. A Framework for Alternate Queueing: Towards Traffic Management by PC-UNIX Based Routers. In *Proceedings of USENIX 1998 Annual Technical Conference, New Orleans, LA, USA*, June 1998.

[DBC+00] David Durham, Jim Boyle, Ron Cohen, Shai Herzog, Raju Rajan, and Arun Sastry. RFC 2748 - The COPS (Common Open Policy Service) Protocol. Standards Track RFC, January 2000.

[FJ93] Sally Floyd and Van Jacobson. Random Early Detection gateways for Congestion Avoidance. *IEEE/ACM Transactions on Networking*, 1(4):397–413, August 1993.

[GK99] Richard J. Gibbens and Frank P. Kelly. Distributed Connection Acceptance Control for a Connectionless Network. In *Proceedings of 16th International Teletraffic Congress*, June 1999.

[HBC+00] Shai Herzog, Jim Boyle, Ron Cohen, David Durham, Raju Rajan, and Arun Sastry. RFC 2749 - COPS usage for RSVP. Standards Track RFC, January 2000.

[Int00] Intel. Free COPS implementation, 2000.

[Kar00] Martin Karsten, editor. *M3I Deliverable 3, Part I - Pricing Tools Design*. June 2000.

[Kel00] Frank P. Kelly. Models for a self-managed Internet, 2000. submitted to Royal Society.

[RF99] Kadangode K. Ramakrishnan and Sally Floyd. RFC 2481 - A Proposal to add Explicit Congestion Notification (ECN) to IP. Experimental RFC, January 1999.

[Sti00] Burkhard Stiller, editor. *M3I Deliverable 4 - CAS Design*. June 2000.

[SWKS99] Jens Schmitt, Lars Wolf, Martin Karsten, and Ralf Steinmetz. VC Management for Heterogeneous QoS Multicast Transmissions. In *Proceedings of the 7th International Conference on Telecommunications Systems, Analysis and Modelling, Nashville, Tennessee*, pages 105–125, March 1999.

[Vov00] Vovida. Free COPS implementation, 2000.

5.9 Abbreviations

ALTQ	-	Alternate Queueing
ATM	-	Asynchronous Transfer Mode
CAS	-	Charging and Accounting System
CE	-	Congestion Experienced
COPS	-	Common Open Policy Service
DP	-	Dynamically Priced
ECN	-	Explicit Congestion Notification
ECT	-	ECN-Capable Transport
HFSC	-	Hierarchical Fair Service Curve
GSP	-	Guaranteed Stream Provider
IntServ	-	Integrated Services
M3I	-	Market Managed Multi-service Internet
MIB	-	Managed Information Base
NME	-	Normalised Meter Event
QoS	-	Quality of Service
PDP	-	Policy Decision Point
PEP	-	Policy Enforcement Point
RSVP	-	Resource ReSerVation Protocol
SNMP	-	Simple Network Management Protocol
SIU	-	Smart Internet Usage

5.10 Appendix - Current Implementation Status

5.10.1 Available Modules

- plain KOM RSVP implementation
- various COPS implementations [BB00,Int00,Vov00]
- ALTQ implementation providing HFSC scheduling
- RSVP-capable video-streaming application
- traffic generators: netperf and periodic CBR traffic generator
- plain NeTraMet for UNIX, new release supposed to appear soon
- ECN-capable NeTraMet for Windows (binary only)

5.10.2 To Do

- choose and extend COPS implementation to carry RSVP requests
- extend RSVP daemon to emit COPS requests
- COPS mediation into NMEs for CAS and Price Calculation
- integrate Price Communication and CAS
- develop Tariff Directory
- integrate Price Communication with end system application
- develop ECN(CE)-analyser and -classifier
- develop ECN(ECT)-classifier and -marker
- integrate RSVP with ECN(CE)-analyser and ECN(ECT)-marker
- design & implementation of price calculation algorithm(s) for ECN-priced subnet
- design & implementation of price calculation algorithm(s) for RSVP-based subnet
- SIU NeTraMet encapsulator
- *later*: extend RSVP daemon to handle COPS rejections

5.10.3 Available System

No full system yet

6 Cumulus Pricing Scheme Scenario (CPS)

The Cumulus Pricing Scheme (CPS) scenario is the scenario in the M3I project considering explicitly long time-scale pricing. In a sense, CPS can be stated as a dynamic flat rate pricing scheme with an appropriate feedback mechanism. Indeed the scope of CPS claims, since it defines a new approach, investigation on contracting by Service Level Agreements (SLA) and as a consequence investigation on traffic heuristics for correct estimation of customer requirements.

In the M3I project, CPS is applied onto a Differentiated Services (DiffServ) environment. The idea is to merge the two systems and to profit from synergies in the areas of contract negotiation and of contract supervision. Currently the necessary experimental platform is set up, heuristics are in the process of collection, and an evaluation is intended.

6.1 Introduction

Pricing schemes form the essential part of a business model for Internet Service Providers (ISP). A pricing scheme applied to the transport of data in an IP network needs to cope with a number of issues of the IP technology utilised. Therefore, the scheme designed at this stage was termed Cumulus Pricing Scheme (CPS) and has been explicitly developed for the Differentiated Services Internet Architecture (DiffServ).

CPS proposes a paradigm shift and argues that the problem of Internet pricing is not a matter of complexity, but instead a problem of mapping multiple and multi-dimensional time-scales. The developed scheme shows a simple, transparent, market-managed, and feasible Internet pricing scheme[5].

CPS is a flat rate scheme founding on SLA contracts between customers and ISP, whereby the customer may be an ISP. It provides individual and dynamic adaptation of flat rates on long-time scales due to SLA contract ruptures and/or renegotiations. The compliance of the contract is motivated and supported by a feedback mechanism, the Cumulus Points (CP), and the liberality for deviations on short-time scales, due to statistical metering and average CP accumulation mechanisms [7], [8].

The scenario is under the responsibility of ETHZ. Since CPS is a rather new idea, conceptual and theoretical topics are under close investigation. This concerns mainly the process of gathering knowledge and experiences in contract metrics, i.e. contract terminology and contract negotiation. Furthermore, heuristics are collected with the intention to define appropriate stimuli and parameters for a simulation initialisation and a clear scenario definition. The mapping and implementation of CPS will start right after that.

6.2 Overview and Motivation

The assignment of CPs works as follows: first of all, customer and ISP are supposed to agree on a contract specifying the expected customer requirements in terms of service usage as well as on a flat rate to be paid for them. Following this agreement, the

[5] In M3I terminology [6], the developed scheme is determined by all features of a tariff scheme. However, for comparisons with “traditional” Internet “pricing work”, the older and less precise term has been utilised. Pricing Mechanisms, as described in [3], are applicable.

factual usage may not match the prediction given by the user (for whatever reason, be it, e.g., an incorrect statement, changing habits, or new applications). As soon as these discrepancies exceed some threshold, the user receives feedback in terms of the mentioned CPs. They exist as red and green flags: a red CP indicates that the user has been overusing her capacities, a green one indicates the opposite, i.e. that the user might have been allowed to use more resources than she actually did. The larger the discrepancy between contract and reality, the more CPs may be assigned.

CPs remains valid for a dedicated number of consecutive billing periods, and it is their accumulation that finally triggers certain consequences. Hence, receiving CPs requires no immediate reaction. However, their successive accumulation over consecutive billing periods eventually may exceed a CP threshold and have consequences for the user, depending on ISP policies.

Figure 6-1 describes a typical example of how CPs are used. Customer C has stated her expected bandwidth requirements to be x MB/s, but the actual bandwidth consumption exceeds the agreed upon one slightly in January and heavily in February. Accordingly the consumer receives one red CP at the end of January and two additional red CPs at the end of February. Afterwards, her consumption falls below the expected value (one green CP in March), before it behaves exactly according to the contract in April (which is apparently the ideal situation). Later on, in May and June this value is exceeded again. The accumulation of the CPs as of end of June sums up to five red CPs and eventually requires a renegotiation of the original contract.

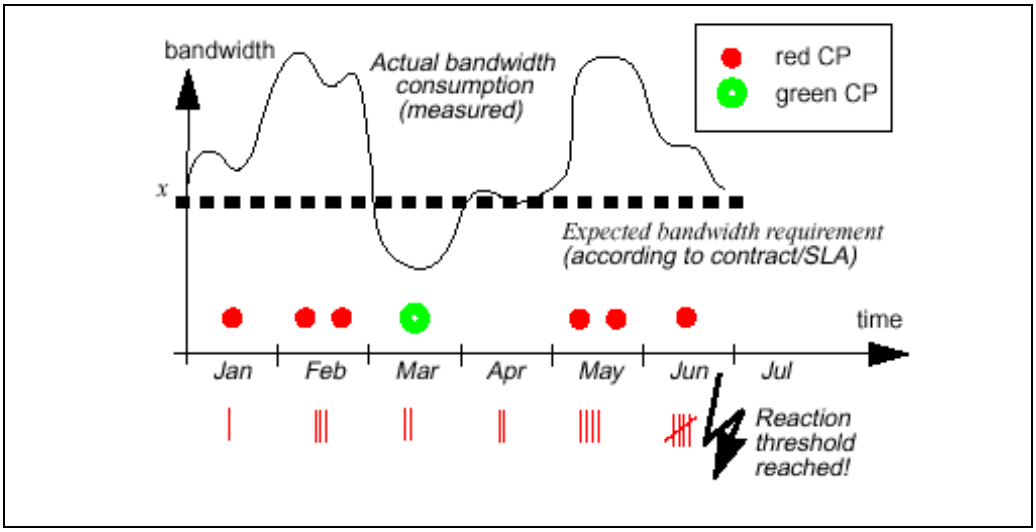

Figure 6-1: Red and Green Cumulus Points and their Accumulation over Time

The motivation and advantages for using CPS as a pricing scheme bases on its long time-scale behaviour, its transparency for customers, i.e. simplicity to understand, on the predictability of prices due to the negotiated flat rate and on appropriate feedback mechanisms, which help to complain with the contract.

6.3 Design

The discussion of the CPS scenario is based on full M3I requirements and scenarios, which are part of [1], and follows the M3I Architecture as described in [2]. In the CPS scenario only one single stakeholder is required, the ISP as indicated in Figure 6-2. It provides communication as well as contract negotiation and management mechanisms.

Furthermore, this stakeholder maintains an offer repository, where customers see all services the stakeholder is able to provide.

Since CPS rather is not service-oriented, but is based on accumulated and statistical usage of services over a longer term billing period, contract negotiation, contract enforcement, and contract management become very important.

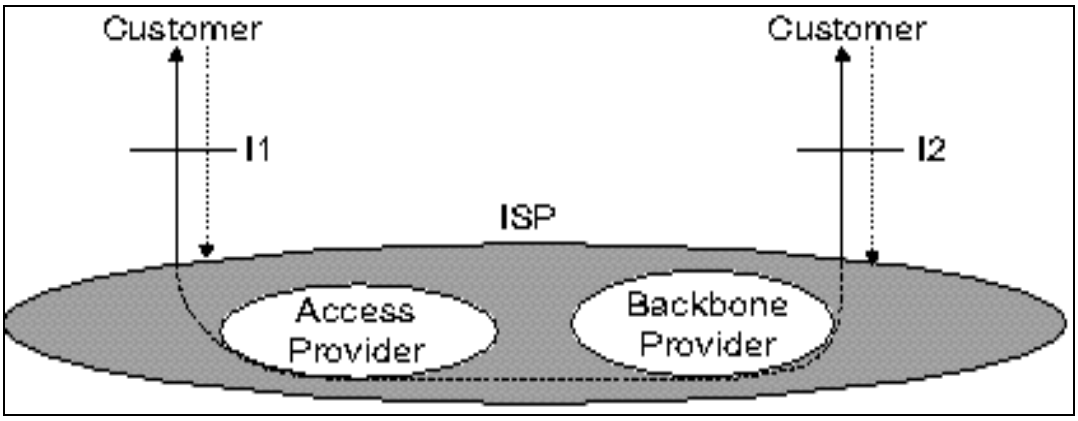

Figure 6-2: CPS Scenario

6.3.1 Requirements Interface I1/I2

There will be no charging interactions between the customer and the ISP. Business interactions initiated by a negotiation phase aim to establish an exhaustive Service Level Agreement. M3I will focus on the financial part of the SLA, i.e. setting a fee (in advance) according to estimated customer service consumption, the agreement upon billing periods (the validity period of the contract) as well as on feedback and auditing mechanisms. All legal aspects of the SLA will be neglected for the time being.

The communication technology provided by the stakeholder is based on DiffServ, i.e. Assured Forwarding (AF), Expedited Forwarding (EF) as well as best effort per hop behaviours. Technical aspects of the SLA, like signalling and admission control for DiffServ, are out of scope of this work.

Generally the interfaces I1 and I2 are asymmetric, i.e. on the customer offering the service, a different charging policy can be applied. As a consequence for the scenario, only interface I1 remains to be investigated. Further on, contracts and charging issues between Access and Backbone Provider are neglected. After these refinements, all charging interactions are reduced to two phases, both applied on interface I1. These encompass:

1) Phase1: define and set up the individual contract according to the customer needs. The scenario only defines the flat-rate per volume and DiffServ forwarding class, the applied feedback mechanism (i.e. what is the equivalence of a CP) and the thresholds, which provoke a contract rupture or re-negotiation. Requirements of Phase 1 are the investigation of heuristics for the initial traffic estimation, i.e. to find out what are the requirements of the customer.
2) Phase2: survey of contract compliance and feedback provisioning. The goal is to apply the same heuristics, i.e. the resulting functions and parameters, as for Phase1 for survey, so that the metering effort can be reduced to a reasonable minimum. A result of the survey is the feedback in form of CPs and if necessary an “ultima ratio”, the immediate contract violation.

6.3.2 CPS-Relevant SLA Components

Based in these major assumptions and interface restrictions, the following components of an SLA are relevant with respect to CPS:

- *Fee (flat-rate price)*: defines the monetary amount the customer has to pay, regardless of service usage;
- *Billing period*: determines the long time-scale in which the fee is valid and which a feedback is given to the customer for the service utilisation;
- *Validity period*: specifies the overall contract duration, which includes normally several billing periods and shows the longest time-scale of interest for CPS;
- *Re-negotiation conditions*: these conditions determine the critical amount of cumuli, which can enforce a contract re-negotiation;
- *Feedback*: this includes mechanisms for customer to receive information on the actual cumulus status;
- *Optional cumulus price equivalent*: this equivalent defines a setting of a possible price per cumulus point to avoid a contract re-negotiation.

These components form the basic information and according mechanisms required to establish an operational CPS approach.

6.3.3 Mapping of CPS to CAS

Figure 6-3 depicts the mapping of the CPS scenario to the CAS (Charging and Accounting System). Several components have been left empty compared to the full CAS design since they have no function in CPS.

An additional internal interface between the *Charging* component and the *Agent Interface* has been introduced, since its the important path to provide a feedback to the customer on the actual cumulus set. The feedback mechanism is activated, whenever new charging records have been processed.

An essential question to be solved tackles the issue, where the CPS-SLA will be managed. The major characteristics of this CPS-SLA implies:

1) the SLA is the contract between customer and network provider. Most of the agreement will be human readable and used for legal and financial issues but nevertheless it necessarily contains commitments on service provisioning, i.e. technical aspects.;
2) from the SLA a Service Level Specification (SLS) is deduced. The SLS is a mapping of technical commitments to technical parameters and/or functions. The SLS makes it possible for network providers to fulfil the contract in a transparent way, i.e. invisible for the customer;
3) furthermore, the SLS consists of logical separable parts. In the scenario we distinguish a part concerned with service provisioning i.e. with DiffServ configuration, signalling, admission control, traffic conditioning, furthermore, a part concerned with charging. The M3I architecture of the charging system, permits and suggests due to the distribution and separation of duties, that the SLS part concerned with charging will be split up in several areas of interest.

The momentary state proposes that the charging issues of the SLS are kept in the Customer/ User Support entity of the CAS. The SLA instead will be archived in a external repository under responsibility of the billing centre and the network provider.

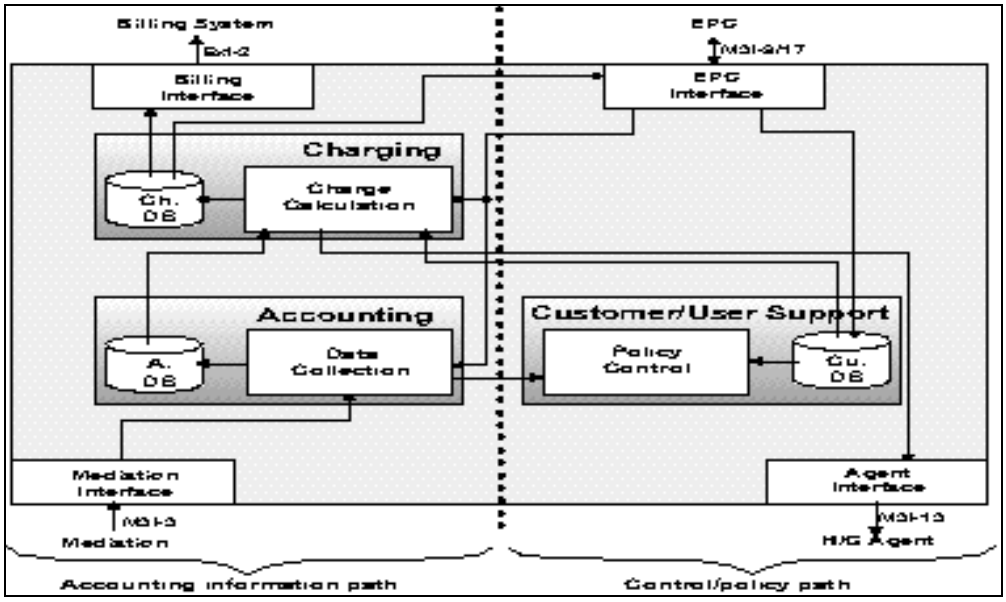

Figure 6-3: Mapping the CPS Scenario to the CAS

6.3.3.1 Interface Message Specification

The following parameters required for CPS are used in several messages. These ones include:

```
<contract_id>    ::= <number>
<customer_id>    ::= <name@address>
<name@address>   ::=  to be refined
<user_id>        ::= <name@address>, [ <customer_id>]
<source>         ::= <IP_address>
<destination>    ::= <IP_address>
<IP_address>     ::= <IP_of_host> | <IP_of_network>
<date>           ::= <date_rough>, [<date_precise>]
<date_rough>     ::= <year> , <month>, [<day>]
<date_precise>   ::= <day>, <hour>, <min>, <sec>, [<msec>]
```

Even though many details of these parameters are straightforward, the *contract_id* is used to identify the tariff to be applied on the accounting record. In addition, the different time-scales are reflected by the “rough” and “precise” date parameters.

The following subsections investigate in a fine-granular manner all interfaces, which are important for the mapping of the CPS approach onto the CAS. For an easy-to-read description an Extended Backus-Naur form has been applied, which includes for explanatory reasons a set of comments, indicated by the "#".

6.3.3.2 Interface M3I-3 (Mediation-Accounting)

The I-MA (Interface Mediation-Accounting) [7] follows the “push” interaction paradigm, where all data are taken from the metering component, mediated, and pushed into the accounting component. A single record will be part of a message, which includes all relevant data.

```
   Message ::= <user_id>, <source>, <date>, # general parameters
               <contract_id>, #contract parameter
               <usage>, <service>
   <usage> ::= "to be refined"
 <service> ::= <PHB>, <DSCP> # Service definition in DiffServ
```

```
    < PHB> ::= <best-effort> | <AF> | <EF> # AF:assured
                                                forwarding
                                          #EF:expedited
                                                forwarding
    <DSCP> ::= to be refined              # DiffServ Codepoint
```

Remark: the *contract_id* as well as *user_id* are needed for a detailed billing, especially if a customer represents several users. The set of DiffServ Codepoints (*DSCP*) is automatically defined by the DiffServ technology in use.

6.3.3.3 Interface M3I-9 (EPC-Customer Support)

The I-ECS (Interface Enterprise Policy Control (EPC)-Customer Support) follows a push interaction paradigm as well. The push is initiated from the EPC to inform the customer support on contractual information discussed above.

```
         Message :: = <customer_id>, <contract_id>,
                      # customer may have several contracts
                      <user_list> <threshold_para>
<threshold_para> ::= "to be refined"
     <user_list> ::=  <user_id>+
                      # list of users belonging to customer
```

The user list need to be refined to be able to associate the user, the customer, and the contract with minimal overhead. However, redundancy is required to provide failure safety.

6.3.3.4 Interface M3I-13 (Charging-Host/Gateway Agent)

The I-CHGA (Interface Charging-Host/Gateway AGent) follows two interaction modes: push and pull. On one hand, this is required to inform the customer via the host/gateway agent on the current setting of the cumulus points received so far. On the other hand, this interface informs the CAS on the customer's reaction on these cumulus point setting, such as an abort of the contract (based on contract restrictions), an establishment of a new contract with new estimates, or an intermediate payment for over-utilisations.

```
       Message   ::= <user_id>, <contract_id>,
                     <destination>, <actual_charge>
<actual_charge>  ::= <cumuli_amount>
<cumuli_amount>  ::= <number>   # a positive number equals red
                                                          CPs
                           # a negative number equals green
                                                          CPs
```

6.3.3.5 Ext-2 (Billing Interface)

Finally, the I-BI (Interface Billing) defines the details to be presented to the customer after an end of a billing period has been reached. The interaction mode has been defined on a pull-basis, to allow a billing system to collect billing-relevant information at any time.

```
             Message ::= <customer_id>, <contract_id>,
                         <cumuli_amount>, <measurement_period>
<measurement_period> ::= <start_date>, <end_date>
        <start_date> ::= <date>
          <end_date> ::= <date>
```

Note, that the details of the billing system are out of the scope of M3I, only the interface and relevant information for a billing system are maintained.

6.3.4 CPS Close-Up

The considerable pool of related work on pricing models has been investigated for the new pricing scheme to comprise most of the advantages while avoiding most of their problems. The new Cumulus Pricing Scheme (CPS) has been developed with respect to three main requirements:

- transparency and predictability for customers;
- economic efficiency (by introducing market mechanisms) for the ISPs;
- technical feasibility of the accounting.

6.3.4.1 General Idea of Cumulus Pricing

The fundamental decision between static and dynamic schemes touches immediately customer's desires concerning price stability, e.g., highly fluctuating auctions, whereas orienting a pricing scheme strictly according to the forces of the market readily induces technical infeasibility. In this situation, CPS is an approach to reconcile all three requirements. An example has been outlined above.

CPS is basically a flat rate scheme (but rates may vary over long time-scales), it provides a feedback mechanism to bring market forces into play (where this feedback is not an immediate one, but requires the accumulation of a sufficient number of discrete "flags" indicating user behaviour), and it allows a huge flexibility in terms of the technical prerequisites, especially concerning the measuring and accounting mechanisms of the required data records.

The key to the new solution proposed lies in building the contract between customer and ISP upon suitable information about the expected usage pattern of the service plus influencing the actual customer behaviour by a new type of feedback mechanism that is specific in terms of its relation to different time scales. *Measurements* take place *over a short time scale* and allow evidence about *user behaviour on a medium time scale*. This evidence is expressed in terms of discrete flags (the so-called "Cumulus Points"), yet not triggering some sort of *reaction* by themselves, but only as a result of their accumulation over a *long time scale*.

Hence, if the user has been detected by the ISP to strongly overuse capacities or to misbehave in some sense, she will usually receive some sort of warning that in case of unchanged behaviour the contract may sooner or later may be finished by the ISP. On the other hand, if the user is under-utilising her capacities, this behaviour may be rewarded by some sort of bonus system.

This feedback system strongly depends on measurement activities that are deliberately left open to the ISP. Hence, one could think of (1) an ISP monitoring each packet or each connection in the one extreme, (2) ISPs undertaking systematic monitoring, (3) ISPs measuring every now and then (maybe in some sort of statistical framework), or (4) at the other extreme ISPs not measuring at all.

6.3.4.2 Mathematical Description

Suppose that ISP *I* offers only one service, and initially customer *C* has stated her expected bandwidth requirements according to a contract (the SLA) to be *x* MB/s, whereupon ISP *I* has offered a flat rate tariff of *a* $/month for this service which

customer *C* has accepted. In reality, the volume consumed by *C* is described by a function *V(t)* of time, which naturally may differ arbitrarily from the stated expected requirement *x*.

Let $\Delta_i = \Delta(t_i)$ describe the monthly over- or under-utilisation, respectively, of the customer with respect to her statement *x*, i.e.

$$\Delta_i = \int_{t_{i-1}}^{t_i} (V(t) - x)dt = \int_{t_{i-1}}^{t_i} V(t)dt - x(t_i - t_{i-1})$$

where t_i describes the end of measurement period *i*, e.g., the end of month, *i* = 0,1,2... (note that t_0 describes the start of the contract between ISP and customer).

Cumulus Points are assigned by the ISP *I* according to a rule (the so-called "CP Rule") whose content is up to the ISP, but typically might look like the following:

CP Rule: *define* θ_n, *n = -N,…, -1, 0, 1, 2,…, N, to be the CP thresholds,* $\theta_0 = 0$ *and* $\theta_{\pm(N+1)} = \pm\infty$, *where N describes the maximal number of CPs that could possibly be assigned for one measurement period. Then for measurement period i, the customer is assigned cumulus points iff*

(1) $0 \le \theta_{c_i} \le \Delta_i \le \theta_{c_i+1}$ or

(2) $\theta_{c_i-1} < \Delta_i \le \theta_{c_i} \le 0$

the choice between (1) and (2) depending on $\operatorname{sgn}\Delta_i$.

Hence, if Δ_i is positive (i.e. overuse in period *i*) and lies between thresholds θ_c and θ_{c+1}, then *c* cumulus points are assigned. If Δ_i is negative and between thresholds θ_{c-1} and θ_c, then *c* cumulus points are assigned, where *c* now is a negative number, hence the cumulus points are referred to as "green" ones, whereas for positive c the cumulus points are "red". Now the cumulus points c_i are accumulated over time according to

$$\Gamma_n = \sum_{i=1}^{n} c_i ,$$

hence, Γ_n describes the total sum of cumulus points assigned since the start of the contract.

The reaction to CP accumulation is again basically up to the ISP and is the content of a second rule, the so-called "Reaction Rule", typically looking like this:

Reaction Rule: define Θ to be the reaction threshold. Then the contract between customer and ISP is in the state of imbalance and needs to be renegotiated after period *n* if

$$|\Gamma_n| \ge \Theta$$

Depending on sgn Γ_n, there may as well be two different thresholds Θ^+ and Θ^- for red and green CPs, respectively.

6.3.4.3 ISP Policies

Dealing with these Cumulus Points, different ISP policies include especially the following ones:

- **Measurements: i**t has been argued that it is almost impossible to find a standard way of network monitoring and accounting that is compulsory for all ISPs. Hence, in the approach proposed it is up to the ISP, on which data measurements the distribution of CP is based;
- **CP assignment:** being assigned one or more CPs depends on violating thresholds in terms of utilisation or bandwidth. Fixing these thresholds is up to the ISP. Note that setting prevents from smaller oscillations in x and result in a superfluous assignment of CPs;
- **Accumulation: u**sually, CPs are supposed to be accumulated over subsequent billing periods. However, it may be allowed for CPs to expire, or for red CPs to be charged up against green ones;
- **Contract renegotiation:** another threshold to be set freely by the ISP concerns the point at which the contract with the customer is supposed to be renegotiated. The way of renegotiating is also open. Either the customer delivers a new statement about expected QoS requirements, and the provider offers a new charge, or the old contract remains valid, and in one way or the other the delta requirements plus the old red CPs are dealt with by a separate new contract, e.g., by an extra payment.

6.3.5 Architectural Embedding of CPS

The CPS approach follows a well-defined behaviour as expressed above and in [7], [8]. To ensure that the details and mechanisms available in M3I, mainly driven by the architecture in place [2], are applicable to CPS, the architectural embedding has been performed based on the notation of [2]. The resulting diagram is presented in Figure 6-4 and explains the important interactions as well as parameters exchanged.

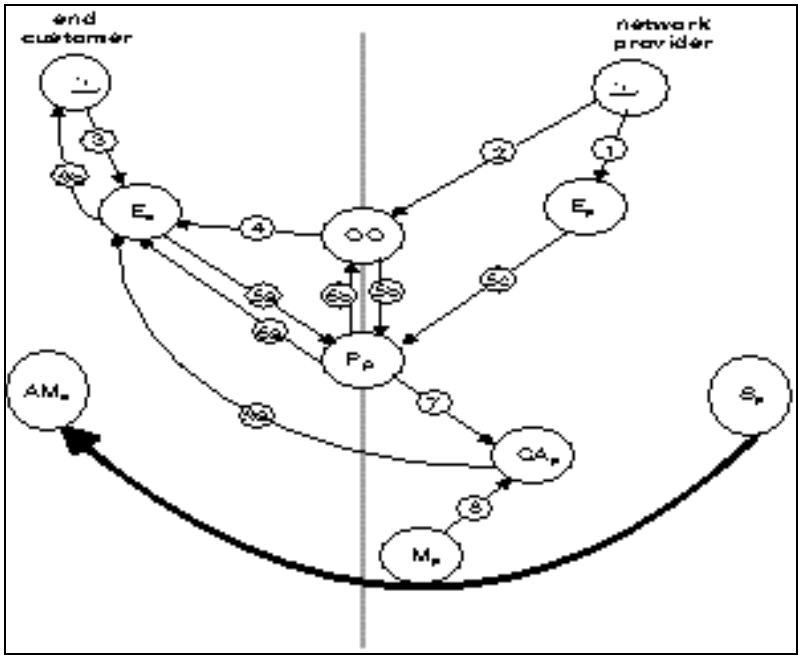

Figure 6-4: CPS Scenario Architecture Diagram

According to a step-wise description of these interactions, the arcs are labelled:

1. configuration of enterprise policy control;
2. announcement of available services, not yet creation of them, since they may be configured according to customer preferences;
3. configuration of enterprise policy (customer);
4. announcement of available services;
5. SLA-negotiation:
   - 5.a. presumption on resource consumption i.e. estimation of customer what amount of services she will use during a billing period;
   - 5.b. what services are still available;
   - 5.c. setting of: fee, threshold for cumulus generation, threshold for contract break and possible fixation of an equivalent price of a cumulus point;
6. SLA-negotiation feedback mechanism:
   - 6.a. announcements of fee and thresholds. The last is important for feedback and auditing;
   - 6.b. updating of newly available services (different to other scenarios);
7. configuration of CAS;
8. metering data;
9. (9a/b) feedback of actual cumulus status, i.e. of the additional cumuli assigned during actual billing period. Keeping track on the elder cumulus status is a matter of the billing centre.

Note that the P_{rc} and Q_c smiles are not listed, since CPS makes usage of long time-scales. Dynamic adoption of QoS on behalf of a price reaction policy (better cumulus reaction) needs not to be available on short time scales, especially since the generation of cumuli is absolutely independent on the actual market price of the service. Instead it is up to the customer to either estimate her behaviour correctly, or to change the behaviour according to the estimation or to change initiate a contract renegotiation. In any case a right estimation is still the cheapest and most favourable solution for both customer and network provider.

6.4 Implementation

The implementation of CPS will allow for two distinct ways. The first one includes the mathematical evaluation of the scheme. This covers the implementation of a simulation model for studying the relevant effects, e.g., with respect to thresholds, as described below in Section 6.5. This implementation includes the real collection of existing traffic data to ensure that the model input parameters are as realistic as possible. With these two sets of data (the collected ones and generated ones) the CPS is implemented by its mathematical functions.

In a second step a real network technology, the DiffServ platform as the underlying networking technology choice, is set up and the integration of CPS into the M3I CAS will follow as described. Within this combined implementation of CPS in the M3I CAS and a suitable networking technology selection, presumably three traffic types will be supported, Assured Forwarding, Expedited Forwarding, and Best-effort. Each of them

will be under a different CPS regime. An appropriate CPS mapping scheme for thresholds and the CPs is to be developed accordingly.

6.5 Experiment Description

CPS acts on long time-scales, therefore, experiments for the CPS scenario require to run long enough as well. It is important to note that the focus is not set purely on the traffic generated by applications, but on the customer behaviour, i.e. the stochastic nature of the traffic. The goal of these experiments is to verify and tune those functions, which are developed by mathematical approaches for CPS and are formed by simulations results, to deduce a set of fully applicable and mandatory parameters for those SLA defined for DiffServ and the CPS scheme.

The current experiment proposal includes a set of applications generating AF, EF, and Best-effort traffic. An SLA-related estimation of customer requirements and behaviour will happen. In addition, the SLA is set by observing the traffic during several short-time periods. Finally, the definition of CPS parameters will be performed according to a SLA and an experiment starts with stochastically running applications over a long-term period.

The field of application of CPS is not defined by metering and charging of characteristic network traffic caused by representative applications. CPS is not a charging approach to apply on a specific application or family of applications with similar characteristics. Thus the intention is not to charge for each single application traffic, but for an approximation of the total traffic amount generated by all applications. This approximation implies a shift from “measure everything” to statistical and stochastic measuring, e.g., this could be the estimation of the mean value and the variance in the most primitive case. The consequence for those applications of the CPS scenario is that not single applications are relevant, but the stochastic appliance of applications. Those characteristics of applications are not so relevant, since the distribution of CPs is coupled to the initial traffic estimation of the contract.

A set of important questions is to be answered by simulation:

1) are the CP thresholds calculated reasonable? Especially, are they consistent to traffic that does not follow a normal distribution?
2) if so, how are they to be determined? Is a linear version or a non-linear version to be preferred?
3) using thresholds derived in question (1) above, what is the confidence level that their distances are large enough to yield CP assignment being widely independent of the measurement process? Is this confidence level realistic?
4) how big is the influence of the number of measurements per customer? Is there a bound on this number due to the asymptotic nature of the Student-t distribution, where it does no longer make sense to increase the number of measurements beyond?

6.6 Experiment Results

No results available yet.

6.7 Conclusions

The CPS scenario deals with important design issues for Internet pricing schemes (basically an Internet tariff on a long time-scale). It proposes an approach that is able to explain why a long-term pricing proposal looks like as it does. Moreover, it is intended to experiment how this scheme allows for the design of a tariff that eventually even solves the so-called “feasibility problem”, i.e. the trade-off between technical, economic, and user-based requirements [4].

The Cumulus Pricing Scheme CPS is presented as leading example of a new tariff structure and the experiments will show its feasibility.

6.8 References

[1] R. Andreassen (Edt.): *Requirements Specifications, Part I Reference Model;* M3I Deliverable 1; Version 7, July 6, 2000.

[2] B. Briscoe (Edt.): *Architecture, Part I Primitives & Compositions;* M3I Deliverable 2; Version 1, July 7, 2000

[3] M. Karsten (Edt.): *Pricing Mechanisms Design (PM);* M3I Deliverable 3, Version 1.0, June 30, 2000.

[4] P. Reichl, B. Stiller: *Notes on Cumulus Pricing and Time-scale Aspects of Internet Tariff Design;* Computer Engineering and Networks Laboratory, ETH Zürich, Switzerland, TIK Report No. 97, November 2000.

[5] P. Reichl, B. Stiller (Edt.): *ISP Cost Model (ICOMO) Design;* M3I Deliverable 8; Version 2.0, December 18, 2000.

[6] B. Stiller, J. Gerke, P. Flury: *Charging and Accounting System Design (CAS);* M3I Deliverable 4, Version 1.01, July 7, 2000.

[7] B. Stiller, J. Gerke, P. Reichl, P. Flury: *The Cumulus Pricing Scheme and its Integration into a Generic and Modular Internet Charging System for Differentiated Services*; Computer Engineering and Networks Laboratory, ETH Zürich, Switzerland, TIK Report No. 96, September 2000.

[8] B. Stiller, J. Gerke, P. Reichl, P. Flury: *Management of Differentiated Services Usage by the Cumulus Pricing Scheme and a Generic Internet Charging System;* To appear: IEEE/IFIP Symposium on Integrated Network Management (IM'2001), Seattle, Washington, U.S.A., May 14-17, 2001.

6.9 Abbreviations

AF	-	Assured Forwarding
CAS	-	Charging and Accounting System
CPS	-	Cumulus Pricing Scheme
EF	-	Expedited Forwarding
DiffServ	-	Differentiated Services Internet Architecture
DSCP	-	DiffServ Codepoint
SLA	-	Service Level Agreement
SLS	-	Service Level Specification
ISP	-	Internet Service Provider
M3I	-	Market Managed Multi-service Internet
QoS	-	Quality-of-Service
TUD	-	Technische Hochschule Darmstadt, Germany