

Policy-based Congestion Management for an SMS Gateway

Roberto Cosenza
Infoflex
Stockholm, Sweden
roberto.cosenza@infoflex.se

Alberto Gonzalez Prieto, Rolf Stadler
KTH, Royal Institute of Technology
Stockholm, Sweden
gonzalez, stadler@imit.kth.se

December 19th, 2003

Abstract

We present a policy-based approach to managing congestions in Short Message Service (SMS) systems. Congestion situations typically occur on SMS Gateways (SMSGs), which route SMS messages between different networks and domains. In our architecture, an SMS operator can dynamically define the maximum acceptable loss of messages of a non-guaranteed SMS service class, thereby controlling the trade-off between minimal message loss and maximum throughput in an SMS system. We present the functional architecture of a manageable SMSG and discuss the realization of the Policy Decision Point (PDP), which applies the congestion policy on the SMSG. An implementation of our architecture on a commercial SMSG, the EMG, is underway.

1. Introduction

The Short Message Service (SMS) is an out-of-band message delivery system that permits users to send and receive text messages to/from their mobile phones.

SMS was introduced in 1992 and, since then, it has experienced a remarkable success: by the end of 2002, 30 billion messages were exchanged monthly. This figure was increasing at a rate of 0.5 billions per month. This makes SMS to represent about 10% of the revenue of mobile operators.

We consider two differentiated services. The first is the *guaranteed service*. It guarantees delivery, offering zero losses. The second is the *non-guaranteed service*. The *non-guaranteed service* is of interest to users that present elasticity to prices. For instance, consider a newspaper sending its highlights to its readers via SMS. This involves bulk transfers of messages. In such a case, the price of the message plays an important role.

Figure 1 presents the network architecture for SMS deployment¹. The key element is the SMSC, which acts

as a store-and-forward system for short messages. Upon receiving an SMS, the SMSC queries the HLR database for routing information for the addressee of the message. With this information, the SMSC determines the servicing MSC for the addressee. Finally, the appropriate MSC delivers the message to the terminal of the receiver. This includes finding the appropriate base station to reach her.

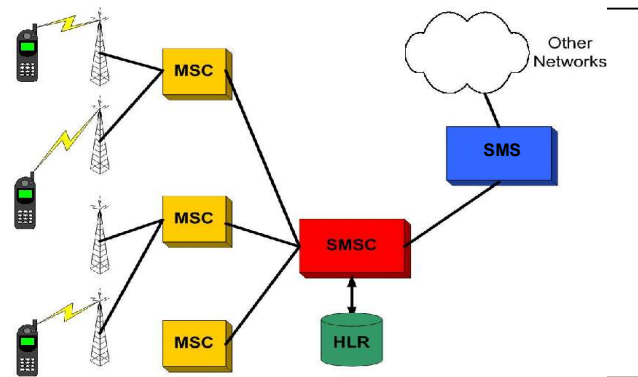


Fig 1: Architecture for SMS Deployment

The SMSC receives messages from two different parties. First, mobile terminals. This is the case when a user sends a message from his mobile terminal. The second source of messages is SMS gateways. SMSGs interconnect the wireless network to others, like other mobile operator's network or TCP/IP networks. When the interconnected networks do not speak the same protocols, the gateway must perform protocol translations.

In this work, we will focus on the Enterprise Messaging Gateway (EMG), a commercial UNIX-based SMSG. We use the model of the EMG shown in figure 2. On the left, we have the incoming ports, which receive the messages the EMG has to deliver. On reception, the message is routed to the appropriate

¹ In this work, we consider SMS in the GSM context

outgoing port. After that, the message may need to be converted to a protocol understood by the receiving network. This conversion phase is irrelevant for this work.

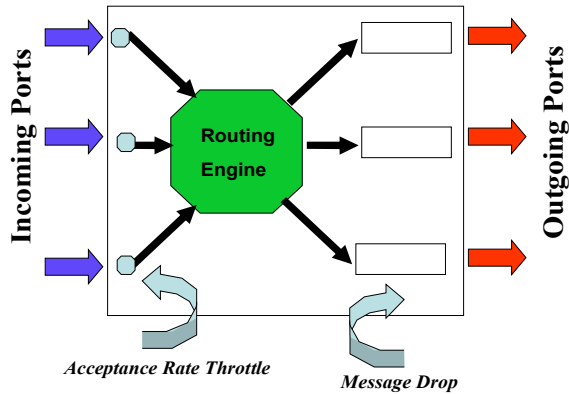


Fig 2: EMG Model

Each outgoing port of the EMG has an associated queue. This permits the EMG to cope with brief situations of congestion. However, long periods of congestion require the introduction of additional techniques.

Assuming the EMG is well-provisioned, long situations of congestion can be caused by a persistent performance degradation of an outgoing port. In the rest of the paper, we will refer to such a port as "congested port"

If congestion is not addressed, the queue associated to the congested port will, eventually, get full. At that point, the EMG is designed to completely stop accepting incoming messages. Otherwise, all accepted messages routed through the congested port would be dropped. Such a situation is not acceptable.

The focus of this work is to provide the EMG with congestion management capabilities that permits us to address long situations of congestion.

In this paper, we propose a policy-based approach to congestion management for the EMG. We have chosen this approach for three main reasons. The first is that the use of policies permits us to raise the level of abstraction of the interaction with the EMG. This permits the operator to skip learning the details of the EMG implementation, focusing on the management goals. This raise of abstraction is especially relevant for commercial SMSGs, like the EMG, due to the lack of specialists in such a specialized software.

The second reason is that a policy-based framework permits to modify the behavior of the system without having to re-implement it. In our case, we can dynamically adapt the behavior of the queuing system without having to recode it. This feature gives more flexibility to the EMG operator.

The third reason is that task automation can be specified—in a relatively intuitive way—using event-condition-action policies. In this work, we aim to automate the reaction to congestion.

2. Congestion Management in the EMG

In this section, we present the two strategies we can use to address congestion in the EMG.

The first strategy consists in reducing the load on the congested outgoing port by reducing the *acceptance rate* at the incoming ports (see figure 2). The acceptance rate at an incoming port controls the admission of messages from that port into the EMG. It indicates the number of messages per second an incoming port accepts. In the EMG, we can set the acceptance rate of incoming ports individually, with a granularity of 1 message per second (m/s).

By reducing the acceptance rate, we avoid the queue associated with the congested outgoing port from growing. However, this reduction affects the loads on all outgoing ports. Therefore the overall throughput of the EMG is compromised.

The second strategy consists in reducing the load on the congested port by dropping some of the non-guaranteed messages that are routed to its associated queue.

Both strategies—reducing the acceptance rate and dropping non-guaranteed messages—present a trade-off. If we want to maximize the overall throughput, we must avoid reducing the acceptance rate. But then, we are forced to drop messages to avoid the outgoing queue from overflow.

In contrast, if we want to minimize losses, we must avoid dropping messages. Then, we need to reduce the acceptance rate.

Therefore the EMG manager has to choose between giving priority to (i) having low losses or (ii) having high throughput.

3. The Congestion Control Policy and its Realization

In this section, we describe our approach to congestion management in the EMG. Figure 3 shows the functional architecture of our approach.

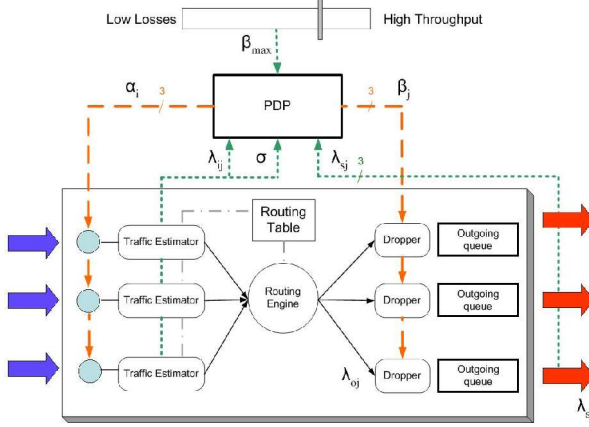


Fig 3: Functional Architecture for Realizing Congestion Control Policies on an EMG

3.1. Policy Specification

First of all, the operator specifies how he wants the system to behave during congestion. He chooses between giving priority to high throughput or to low losses.

As we have presented before, these two strategies present a trade-off: we achieve one at the cost of the other. Considering this, we propose to use a graphical interface where giving priority to one implies possibly penalizing the other: a sliding bar with one of the ends labeled "high throughput", and the other "low losses" (see figure 3).

The performance metric to favor is indicated by the position of the slider. The closer the slider is to the "high throughput" end, the more priority is given to throughput—at the cost of losses—and vice versa.

This qualitative priority is mapped to a quantitative parameter: the *maximum allowable dropping percentage* (β_{max}). It indicates the maximum percentage of non-guaranteed messages that can be dropped under congestion. β_{max} takes values from 0% (low losses) to 100% (high throughput), varying linearly with the position of the slider.

The *maximum allowable dropping policy* provides the maximum overall throughput under the constraint that, at most, $\beta\%$ of non-guaranteed messages is dropped.

3.2. Policy Refinement

The *maximum allowable dropping policy* is refined into two *second-level* policies: a policy for dropping messages and a policy for reducing the acceptance rate of incoming ports. These policies have the following forms:

TYPE 1.- ON congestion DO drop β % of non-guaranteed messages routed to the congested port

TYPE 2.- ON congestion DO reduce acceptance rate to α messages per second.

We have one policy of type 2 for each incoming port and one policy of type 1 for each outgoing port. That is, we have as many α_i as incoming ports and as many β_j as outgoing ports.

The appropriate values for β_j and α_i are calculated by the PDP. They depend on the throughput performance of the congested port (λ_{sj}), the EMG traffic matrix (λ_{ij}), the percentage of messages using the guaranteed service (σ), and β_{max} .

The traffic matrix and σ are estimated by the blocks labeled *traffic estimator*. They do so considering the incoming messages and the routing table of the EMG

The PDP calculates $(\alpha_1, \alpha_2, \dots, \alpha_i, \dots, \alpha_n)$ in two steps. The first consists in calculating the maximum traffic that can be routed to the congested port (λ_{oj_max}). It is calculated—for the steady state—with the following formula :

$$\lambda_{oj_max} = \frac{\lambda_{sj}}{\sigma + (1 - \beta_{max})(1 - \sigma)}$$

In the second step, the PDP calculates the set $(\alpha_1, \alpha_2, \dots, \alpha_i, \dots, \alpha_n)$ that—keeping λ_{oj} smaller or equal to λ_{oj_max} —provides the maximum overall throughput for the EMG. This set depends on λ_{oj_max} and λ_{ij} . This calculation consists in the optimization of a linear system of equations.

β_j is set to β_{max} for the dropper associated to the congested port. For the rest of droppers, it is set to 0.

We have discussed the case of a single congested port. The case of multiple congested ports is addressed analogously.

3.3. Policy Evaluation

In our context, congestion arises when $\lambda_{oj} > \lambda_{sj}$. During congestion, the PDP calculates the appropriate $(\alpha_1, \alpha_2, \dots, \alpha_i, \dots, \alpha_n)$ and $(\beta_1, \beta_2, \dots, \beta_i, \dots, \beta_m)$, and configures the incoming ports and the droppers (figure 3) accordingly.

This calculation is performed whenever any of the incoming parameters of the PDP (λ_{sj} , λ_{ij} , σ or β_{max}) changes significantly. If this re-calculation results in new values for α_i and β_j , the incoming ports and the droppers are re-configured.

3.4. Study Case

In this section, we illustrate the behavior of an EMG under congestion policies in a specific scenario. It consists of an EMG configured with three incoming

ports (ip1, ip2 and ip3) and three outgoing ports (op1, op2, op3), like in figure 3. The port that will suffer the performance degradation is op3.

Each incoming port is configured to accept 6 m/s, and that is the load it is offered. The percentage of messages using the *non-guaranteed* service is 90%.

The combined traffic of the three incoming ports (18 m/s) is evenly distributed among the outgoing ports: the load of each outgoing port is 6 m/s. Nevertheless, each incoming port contributes to the load of op3 differently. 70% of the traffic received in ip1 is routed through op3. For ip2 and ip3, the percentages are 20% and 10%, respectively.

Figure 4, shows the overall throughput of the EMG as a function of β for different performances of the congested port (from 1 m/s to 5 m/s). The values in the figure correspond to the steady state after the EMG has adapted to the new congested situation.

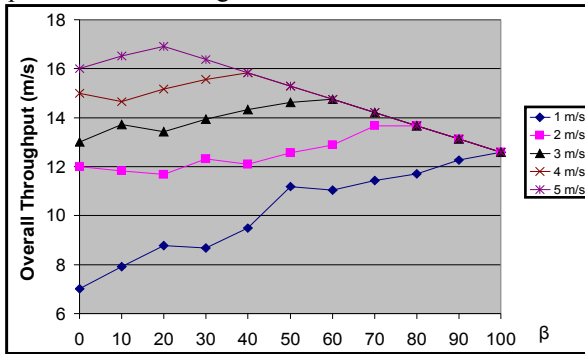


Fig 4: Overall Throughput of an EMG in function of β , the drop rate of messages of the non-guaranteed service, for various rates of a congested output port

We want to remark that the throughput does not always increase with β . There are two situations when a smaller β can provide higher throughput. The first situation is characterized by local minimums. For instance, when the performance of the congested port is 2 m/s, the overall throughput is lower for $\beta = 20\%$ than for $\beta = 0\%$. However, if we keep increasing β (to 30%), the throughput increases as well.

The reason is a constraint of the EMG: all α_i must be integers. This is imposed by the internals of the EMG. Due to this, the λ_{oj} produced by the calculated $(\alpha_1, \alpha_2, \dots, \alpha_i, \dots, \alpha_n)$ might be smaller than λ_{oj_max} . In such cases, the PDP calculates the smallest β ($\beta < \beta_{max}$) that keeps the load on the congested port smaller or equal to λ_{oj} for the calculated $(\alpha_1, \alpha_2, \dots, \alpha_i, \dots, \alpha_n)$. This permits non-guaranteed messages to utilize a capacity that would be unused otherwise.

The second situation is characterized by the monotonous decrease of the throughput. One example can be seen in figure 4, for a congested port

performance of 5 m/s. In this case, from $\beta = 20\%$ on, the overall throughput decreases monotonously.

The reason is that for $\beta = 20\%$, the percentage of dropped messages is high enough to permit setting the acceptance rate of all incoming ports at its initial value (6 m/s). Therefore, since we can not accept more messages, dropping a higher percentage only decreases the overall throughput.

The existence of these two situations implies that different *maximum allowable dropping* policies can result in the same configuration parameters $(\alpha_1, \alpha_2, \dots, \alpha_i, \dots, \alpha_n)$, $(\beta_1, \beta_2, \dots, \beta_i, \dots, \beta_m)$. For instance, for a congested port performance of 1 m/s, the configuration for both policies (i) $\beta_{max} = 50\%$, and (ii) $\beta_{max} = 60\%$ is $\alpha_i = (0, 6, 6)$, $\beta_j = (0, 0, 50)$.

4. Current state and Future Work

In this paper, we presented our approach to congestion management in the EMG, a UNIX-based SMS gateway. Our architecture permits the EMG manager to choose the performance metric to prioritize during congestion: throughput or loss. In situations of congestion, the system auto-configures, following the manager's selected policy.

We are currently implementing our architecture on the EMG platform, version 2.4. The prototype is programmed in C.

One of the issues we want to evaluate on the prototype is the behavior of different algorithms for the estimation of the parameters that influence the PDP.

Currently, messages receive different treatments, based on the service they use: guaranteed or non-guaranteed. We plan to study congestion policies based on characteristics, such as the sender of the message, the message contents or the source network.

References

- [1] GSM Association, www.gsmworld.com
- [2] Nordic Messaging, www.nordicmessaging.se
- [3] M. J. Masullo, S. B. Calo, "Policy management: an architecture and approach". Proc. of IEEE Workshop on Sys. Management, UCLA, Cal., April 1993
- [4] M. Sloman, "Policy Driven Management for Distributed System", Journal of Networks and Systems Management, Vol.2, No.4, 1994