January 2006

Admission control approaches in the IMS presence service

Muhammad T. Alam
Bond University, Muhammad_Alam@bond.edu.au

Zheng da Wu
Bond University, Zheng_Da_Wu@bond.edu.au

Admission Control Approaches in the IMS Presence Service

Muhammad T. Alam, Graduate Student Member *IEEE*, Zheng Da Wu, Member *IEEE*

***Abstract*—** In this research, we propose a weighted class based queuing (WCBQ) mechanism to provide class differentiation and to reduce the load for the IMS (IP Multimedia Subsystem) presence server (PS). The tasks of admission controller for the PS are demonstrated. Analysis and simulation models are developed to quantify the performance of WCBQ scheme. An optimized dropping time frame has been developed based on which some of the pre-existing messages are dropped from the PS-buffer. Cost functions are developed and simulation comparison has been performed with FCFS (First Come First Served) scheme. The results show that the PS benefits significantly from the proposed queuing and dropping algorithm (WCBQ) during heavy traffic.

***Keywords*—**Admission control, presence, queuing.

I. Introduction

IP Multimedia Subsystem (IMS) is a new framework, basically specified for mobile networks, for providing Internet Protocol (IP) telecommunication services [1]. It is the technology that will merge the Internet (packet switching) with the cellular world (circuit switching). It will make Internet technologies, such as the web, email, instant messaging, presence, and videoconferencing available nearly everywhere. Furthermore, the aim of IMS is not only to provide new services but also to provide all the services, current and future, that the Internet provides. Presence is one of the basic services that is likely to become omnipresent in IMS. It is the service that allows a user to be informed about the reachability, availability, and willingness of communication of another user. The presence service is able to indicate whether other users are online or not and if they are online, whether they are idle or busy. Additionally the presence service allows users to give details of their communication means and capabilities.

The presence framework defines various roles as shown in Fig. (1). The person who is providing presence information to the presence service is called a presence entity, or for short a presentity. In the figure, Alice plays the role of a presentity. The presentity is supplying presence information such as status, capabilities, communication address etc. A given presentity has several devices known as Presence User Agents (PUA) which provide information about her presence. All PUAs send their pieces of information to a presence agent (PA). A presence Agent can be an integral part of a Presence Server (PS). A PS is a functional entity that acts as either a PA or as a proxy server for SUBSCRIBE requests. Fig. (1) also shows two watchers: Bob and Cynthia. A watcher is an entity that requests (from the PA) presence information about a presentity. A subscribed watcher asks to be notified about future changes in the presentity's presence information, so that the subscribed watcher has an updated view of the presentity's presence information.

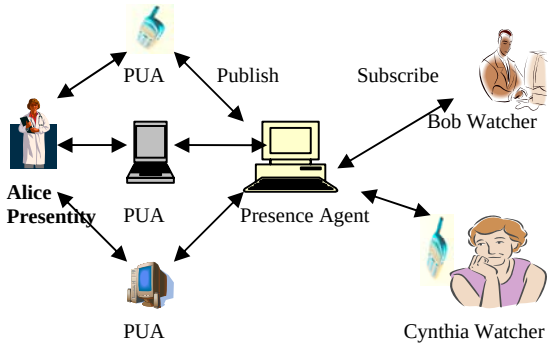

Fig. 1 SIP presence overview

3GPP TS 23.141 [2] provides the architecture to support the presence service in the IMS. The used interfaces for this service are *Pen, Pw, Pi, Px Ut* and the protocols used are SIP (Session Initiation Protocol), Diameter and XCAP (XML Configuration Access Protocol) as described by the IMS technical specification. The presentity publishing flow is illustrated in the Fig. (2). The PUBLISH requests containing an Event header set to presence ought to be forwarded to the PS where the presentity's presence information is stored. So, the S-CSCF (serving-call/session control function) forwards the PUBLISH request (3) to that Presence Server. The PS authorizes the publication and sends a 200 (OK) response (4). After the publishing message has been received, the PS uses the NotifyPresUp message to notify the corresponding watchers, the presentity's presence information via the respective S-CSCF and P-CSCF (Proxy-CSCF) of the watchers [2].

A presence application in an IMS mobile terminal device may contain a list of up to 100 presentities. A watcher receives NotifyPresUp message every time any of its

M. T. Alam is a doctoral candidate at Bond University, Gold Coast, QLD 4229, Australia (e-mail: malam@bond.edu.au, phone: 61-7-55953395, fax: 61-7-55953320)

Z. D. Wu is an associate professor at Bond University, Gold Coast, QLD 4229, Australia (phone: 61-7-55953311, fax: 61-7-55953320)

presentity changes state. Although the SIP (Session Initiation Protocol) event notification framework (RFC 3265, [10]) offers powerful tool, in some situations the amount of information that the Presence Server has to process might be large. When presence information reaches a small device that has constraints in memory, processing capabilities, battery lifetime and available bandwidth, the device may be overwhelmed by the large amount of information and might not be able to acquire or process in real time. So, there has to be tradeoffs between the amounts of information sent, the frequency of the notifications, and the bandwidth usage to send that information. The flow of messages will be massive for large amount of publishers and watchers joining an IMS system. Every time, a presentity changes state, all its watchers will have to be notified. In this research, we propose a weighted class based queuing (WCBQ) mechanism to drop the low priority pre-existing messages from the PS based on the optimal sojourn time. We also discuss some of the ongoing presence optimization techniques that are proposed by the IETF engineers and compare them with WCBQ in terms of admission control for a PS. The simulation result suggests that the PS can benefit vastly from our queuing and dropping technique during heavy traffic. It saves a great deal of message generation for the PS, thus reducing load in busy time.

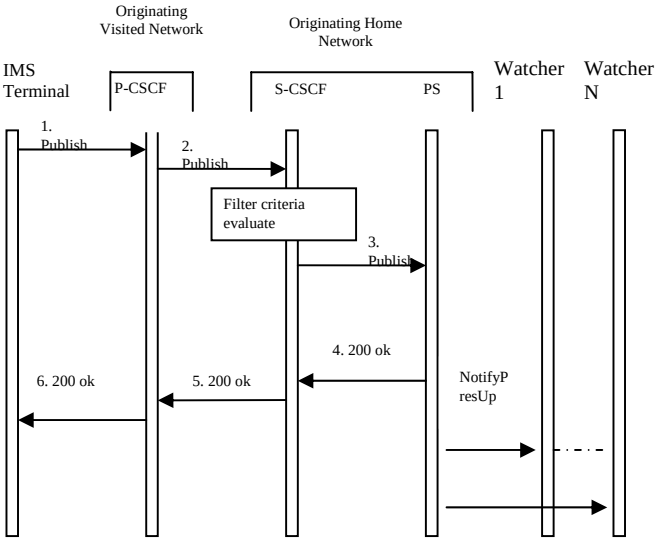

Fig. 2 the IMS terminal publishing presence information

The paper is structured as follows. Section II shows the related work and section III proposes the WCBQ for the PS. The server analysis and admission control mechanisms are described in Section IV. Section V shows the performance results for the proposed scheduler. Section VI describes the effective bandwidth criteria for the discussed admission control mechanisms. The transition probabilities are computed in section VII whereas the cost consumption of a presence system is presented in section VIII based on the steady state probabilities of section VII. The simulation for cost consumption is provided in section IX and finally, section X concludes the paper.

II. RELATED WORK

It is indeed important that the user gets accurate and rich presence information while the bandwidth usage is reduced. The Presence Information Data Format (PIDF) is a protocol-agnostic document that is designed to carry the semantics of presence information across two presence entities. The PIDF (RFC 3863) is specified in the Internet-Draft "Presence Information Data Format (PIDF)" [3]. The PIDF encodes the presence information in an XML (Extensible Mark-up Language) document that can be transported, like any other MIME (Multipurpose Internet Mail Extension) document, in presence publication (PUBLISH transaction) and presence subscription/notification (SUBSCRIBE/NOTIFY transaction) operations. The Rich Presence Information Data Format (RPID) is an extension to the PIDF that allows a presentity to express detailed and rich presence information to his/her watchers. Like the PIDF, RPID is encoded in XML. The RPID extension is specified in RFC 4480 [4]. A watcher receives NotifyPresUp messages from the PS based on this RPID, every time a presentity of its list changes state. Obviously, this mechanism does not scale well, particularly in wireless environment since the heavy transmission rate can easily overload an IMS network with message flows. In order to solve this problem, the IETF has created a number of concepts as described below.

1. Event filtering (RFC 4660, [21]) is one mechanism on which IETF engineers are working to reduce the amount of presence information transmitted to watchers. A weight or preference is indicated through a SUBSCRIBE request. The mechanism defines a new XML body that is able to transport partial or full state. Thus, the document size is reduced at the cost of information transmitted. Also, the implementers need to be aware of the computational burden on the PS.

2. Event-throttling mechanism [22] allows a subscriber to an event package to indicate the minimum period of time between two consecutive notifications. So, if the state changes rapidly, the notifier holds those notifications until the throttling timer has expired. Usually, the PS will buffer notifications that do not comply with the throttle interval, and batch all of the buffered state changes together in a single notification when allowed by the throttle. The throttle applies to the overall resource list (RFC 4662, [5]), which means that there is a hard cap imposed by the throttle to the amount of traffic the presence application can expect to receive. With partial-state notifications, the notifier will always need to keep both a copy of the current full state of the resource F, as well as the last successfully communicated full state view F' of the resource in a specific subscription. The construction of a partial notification then involves creating a difference of the two states, and generating a notification that contains that difference. When a throttle is applied to the subscription, it is important that F' is replaced with F only when the throttle is

reset. Additionally, the notifier implementation checks to see that the size of an accumulated partial state notification is smaller than the full state, and if not, the notifier sends the full state notification instead. The disadvantage is that batching and matching will introduce additional processing delay in the PS. Currently, a subscription refresh is needed in order to update the throttle interval. However, this is highly inefficient, since each refresh automatically generates a (full-state) notification carrying the latest resource state. Also, with this mechanism the watcher does not have a real-time view of the subscription state information. Moreover, holding the information will require additional buffer space. Nonetheless, this policy may be helpful for IMS terminals with low processing power capabilities, limited battery life or low bandwidth accesses. We will discuss the tradeoffs of such service later in this paper.

3. Compression of SIP messages is another technique to minimize the amount of data sent on low-bandwidth access. RFC 3486 [11], RFC 3320 [13], RFC 3321 [12] defines signaling compression mechanisms. Usually these algorithms substitute words with letters. The compressor builds a dictionary that maps the long expressions to short pointers and sends this dictionary to the de-compressor. However, the frequency of data transmission is not reduced in such techniques.

Clearly each of the abovementioned works has limitations and tradeoffs. The admission control technique with effective bandwidths allocations is a mature topic today [6-8, 19, 23]. Janevski and Spasenovski [9] proposed a wireless class based flexible fair queuing algorithm that supports QoS demands of different traffic classes both at error free and error states for wireless IP networks. Marbach analyzed the optimized pricing scheme with packet loss in a game theoretic priority servicing framework in [16]. Moorman and Lockwood developed and analyzed a wireless scheduling algorithm in [17] to provide QoS bounds to the ATM traffic classes of [18]. However, an efficient queuing of presentities publishing information to the PS is yet to be defined in the IMS environment. In this paper, we are proposing a weighted class based queuing mechanism to reduce the load of the IMS PS during heavy traffic. The literature review on class-based queuing is hand-full [24-25]. Our WCBQ distinguishes classes according to message arrival rates and weighs flows inside them according to the number of watchers who are watching a presentity. An optimal sojourn time for the heavily weighted messages has been proposed. Pre-existing messages are dropped from low priority classes based on the timestamp of newly arrived messages and the derived optimal timeframe in order to achieve efficient system performance.

III. PROPOSED QUEUING SCHEME

Our objective is to propose an efficient scheduler for the PS in heavy traffic situation. In our model, we deal with the RPID messages as input for the PS. We consider that the RPIDs carry presence information only and that the size of a RPID message is below the maximum size of an IP packet. We also assume the messages are elastic due to the high arrival rate of the messages. This way, a flow of messages can be represented as a packet stream. The scheduler model is shown in Fig. (3) and the corresponding flow chart is provided in Fig. (4). The scheme is created to support multiple traffic classes. The PS assigns the publishing presence information a hierarchy of priorities. Our tendency in creating this queuing algorithm was to take into consideration the high publishing rate and the number of watchers associated with a particular publisher (presentity).

The classifier differentiates traffic into classes based on arrival rate of messages from the presentities. A class selector separates arriving messages intro different queues for every class according to weight of each arriving message. We define the weight of a presentity message as the number of watchers watching that particular presentity. Lower arrival rate gets higher priority and the higher priority classes are placed at the top in sequence (see Fig. (3)) i.e., class 1 has lower arrival rate than that of class 2, class 2 has lower arrival rate than that of class 3 and so on.

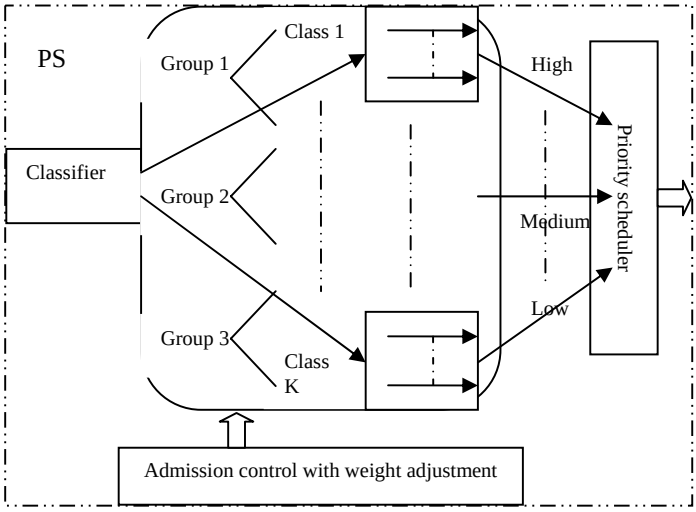

Fig. 3 Proposed queuing scheme

Having all flows of each class the same arrival rate, are further classified according to their weight. The lower the weight (number of watchers watching a presentity), the higher the priority for that message. Thus, the topmost flow in a class has the highest priority while the bottommost flow has the lowest priority. A flow may have messages from multiple presentities that have same arrival rate and same weight. The messages will be inserted in that case in the First Come First Served (FCFS) manner in the flow. A class will be able to use the empty space of other classes in the buffer in case its own buffer is full. With these assumptions, the flows belonging to class 1 will be first served until the buffer for this class is emptied and so forth. However, to avoid monopolization of the bandwidth by the higher priority flows, we should limit the maximal capacity that can be allocated to them. This can be accomplished by an admission control mechanism. Since

the higher priority classes have lower arrival rate, they may be grouped together (group 1) to be serviced in heavy traffic situation. Similarly, medium priority classes may be grouped into group 2 and lower priorities into group 3. The range of class-grouping will depend on the network scenarios and the frequency of state changes by the IMS presentities. The adaptation of group size may be followed from the work specified in [26]. In this work, we are particularly interested in the lower priority classes for which huge number of NotifyPresUp messages need to be generated by the PS destined to the end IMS terminals. The PS will keep a time stamp for every RPID message arriving. The PS will also generate an optimized/threshold stay time for each class which will be discussed later. The threshold time means minimum stay time of messages at the PS. This threshold time will be used to compare the time stamp difference between two arriving message of the same flow in the queue to drop a pre-existing message from the PS to reduce load and to save number of message generation. If the time stamp difference is greater than the threshold time, then the RPID is not dropped and a NotifyPresUp is generated for that RPID.

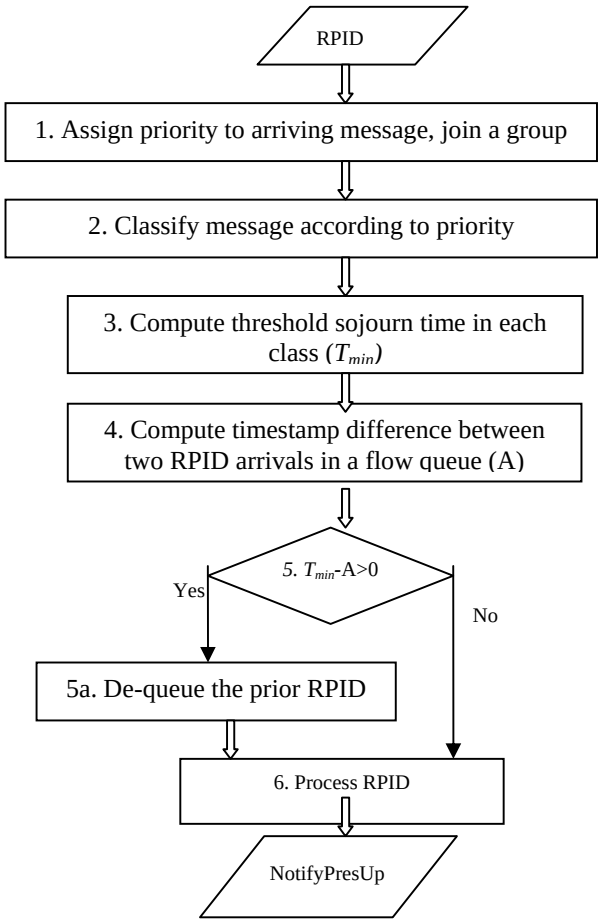

Fig. 4- flow chart for WCBQ

We furnish the admission control functionalities in the next section. For the theoretical derivation of the admission control expressions we consider the PS as an M/G/1 system where as for the sake of implementation and performance analysis we consider the PS as the M/M/1 system, a special case of M/G/1. Justification has been provided as to the application of the M/M/1 system with the classified service time of RPID messages at the PS.

IV. ADMISSION CONTROL MECHANISMS

Let us use B for bandwidth of the outgoing wireless link. The weights assigned to flows in a class j are $w_{ji,}$ $i=1,2,\ldots,N$, where N is the number of flows in the class. The relative throughput of each flow normalized on the link bandwidth for the class j is:

$$RT_{ji} = \frac{w_{ji}}{\sum_{i=1}^{N} w_{ji}} \tag{1}$$

When the wireless path is error-free, a flow from class j should get bandwidth share b_{ji}:

$$b_{ji} = RT_{ji} * B(A_j\%) = \frac{w_{ji}}{\sum_{i=1}^{N} w_{ji}} B(A_j\%) \tag{2}$$

Where, $A_j\%$ is the amount of bandwidth allocated to class j.

The admission controller may use the following in heavy traffic situation to compute the maximum number of publishing information accessible in the PS queue. Let z_j be the number of arrivals of a class j, S is the total size of the PS buffer and l is the average size of the RPID document. Thus, the maximum number of messages is defined as:

$$X = \left\lfloor \frac{S - \sum_{j} z_j l p_{ts_j}}{l} \right\rfloor \tag{3}$$

Where, p_{ts_j} is the probability that the PS completes servicing a message of class j in time slot, ts with mean service rate, μ_j. Then,

$$p_{ts_j} = 1 - e^{-\mu_j(ts)} \tag{4}$$

This probability behavior is depicted in Fig. (5).

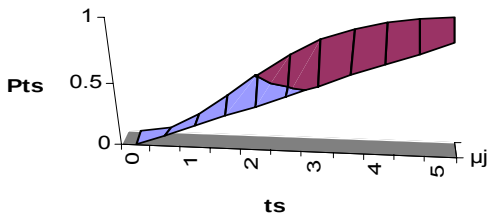

Fig. 5 behavior of Pts

Bounds of tail distribution can be used to develop efficient

admission control mechanisms. Let, there *K* classes of publishing information and *K* independent Poisson processes with arrival rates $\lambda_1,....,\lambda_K$ in the system. The arrival rates are considered to be equivalent to the steady state probability of presentity movement which is defined later in Eq. (37). By the law of superposition of independent Poisson processes: $\lambda = \sum_{j=1}^{K} \eta_j \lambda_j$ Where, η_j is the number of different presentities publishing information to the PS with the same rate of λ_j in class *j*.

The traffic intensity for a class *j* is defied, $\rho_j = \frac{\eta_j \lambda_j}{\mu_j}$.

For the stability condition i.e., ρ <1, we may define the mean waiting time at the PS by applying the Polloczek-Kinchin formula for an M/G/1 system (see [27]):

$$E[W] = \frac{\lambda \overline{\sigma}^2}{2(1-\rho)} \tag{5}$$

Where,

$$\rho = \sum_{j=1}^{K} N_j \rho_j \tag{6}$$

and where, $\overline{\sigma}^2$ is the second-order moment of the service time of an arbitrary message. Let, $\overline{\sigma}_j^2$ be the second-order moment of messages of class *j=1,2,…,K*. By the law of total probability and from Bayes' formula [15]:

$$\overline{\sigma}^2 = \sum_{j=1}^{K} \frac{\eta_j N_j \lambda_j}{\lambda} \overline{\sigma}_j^2 \tag{7}$$

Hence,

$$E[W] = \frac{\sum_{j=1}^{K} \eta_j N_j \lambda_j \overline{\sigma}_j^2}{2(1-\sum_{j=1}^{K} N_j \rho_j)} \tag{8}$$

A. Blocking Probability

The presentities publish their state via the RPID document to the PS (RPID document travels through the presentities P-CSCF and the home networks S-CSCF) and PS acknowledges with a 200 (OK) response. Before the PS places the Publish message from the presentity to an appropriate class of queues, it has to decide whether to send a 200 (OK) message back to the presentity or to discard the message. Upon arrival of a new message, the PS checks for errors and determines whether there is room for this message in the buffer. Depending on outcome of these tests, a 200 (OK) is sent to the transmitting presentity or the packet is discarded at once. In case of the 200 (OK) not being sent from the PS (alternatively the PS might send an error message with code range 500-599), the presentity retransmits the RPID after a suitable interval of time which we call the timeout. We shall assume that the PS cannot contain more than E_K non-acknowledged messages. Under these conditions, the buffer overflow can be modeled for a PS as follows.

Let a PS is composed of

a. K classes of queues denoted as class *1,2,…,K;*

b. *K* timeout boxes denoted as stations $1', 2', \ldots, K'$;

c. *K* 200 (OK) boxes denoted as stations *1",2",….,K".*

With the above conditions, the PS will behave like the combination of several FCFS (First Come First Sever) and IS (Infinite Server) stations. In this situation, the results of BCMP network can be applied to achieve the steady state probabilities. The transmission-retransmission process between PS and presentities can be captured by the timeout and 200 (OK) boxes. More precisely, it is assumed that with probability q_j the attempted transmission over class $j = 1,2,....,K$ fails, either through blocking or through message error. We model this event as having the message enter the timeout box where it resides for a random interval of time. The probability of successful transmission over class *j* i.e., $1-q_j$ is modeled as having the packet enter the 200 (OK) box for a random period of time.

Thus, the probability of a message is retransmitted exactly *n* times over class *j* before success is:

$$(1-q_j)(q_j)^n \tag{9}$$

Therefore, the mean number of transmission for a message over class *j* is:

$$\frac{1}{1-q_j} \tag{10}$$

The total message arrival rate for all classes, λ has been defined before. Let, $p_j, j = 1,2,...,K$, be the probability of a message destined to class *j* in the queue, β be the mean service rate of the arrived messages and δ_j be the mean service rate for the messages in class *j* (processing rate only). The constant service rate β includes the checking whether the buffer is full and a message contains error. We also denote the arrival rate at box j' and j'' as λ_j' and λ_j'' respectively; and mean service rate in timeout box and in 200 (OK) box, as μ_j' and μ_j'' respectively.

Assuming

$\underline{n} = (n_0, n_1, n_2, ..., n_K, n_1', n_2', ..., n_K', n_1'', ..., n_K'')$ with $|\underline{n}| \le E_K$ where n_0 is the arrived messages in the PS and n_j, n_j', n_j'' are the number of messages in j, j', j'' respectively in state $\underline{n}$, the traffic equations read as follows:

$$\lambda_j = \lambda p_j + \lambda_j^{/}$$
$$\lambda_j^{/} = \lambda_j q_j \quad (11)$$
$$\lambda_j^{//} = \lambda_j (1-q_j)$$

We find

$$\lambda_j = \frac{\lambda p_j}{1-q_j}$$
$$\lambda_j^{/} = \lambda p_j \frac{q_j}{1-q_j} \quad (12)$$
$$\lambda_j^{//} = \lambda p_j$$

Let $\pi_K^{WCBQ}(\underline{n})$ be the stationary probability of being in state $\underline{n} \in S_K$ where

$$S_K = \{(n_0, n_1, n_2, \ldots n_K, n_1^{/}, n_2^{/}, \ldots n_K^{/}, n_1^{//}, \ldots n_K^{//}) : |\underline{n}| \le E_K\}. \quad (13)$$

Then,

$$\pi_K^{WCBQ}(\underline{n}) = \frac{1}{G_K}\left(\frac{\lambda}{\beta}\right)^{n_0} \prod_{j=1}^{K}\left\{\left(\frac{\lambda_j}{\delta_j}\right)^{n_j} \frac{1}{n_j^{/}!}\left(\frac{\lambda_j^{/}}{\mu_j^{/}}\right)^{n_j^{/}} \frac{1}{n_j^{//}!}\left(\frac{\lambda_j^{//}}{\mu_j^{//}}\right)^{n_j^{//}}\right\} \quad (14)$$

for all $\underline{n} \in S_K$ and $\pi_K^{WCBQ}(\underline{n}) = 0$ for $\underline{n} \notin S_K$ where G_K is a normalizing constant chosen such that $\sum_{\underline{n} \in S_K} \pi_n^{WCBQ}(\underline{n}) = 1.$

Therefore the total average blocking probability at a PS for the grouped weighted class based queuing is

$$B_{PS}^{WCBQ} = \sum_{\underline{n} \in S_K, |\underline{n}| = E_K} \pi_K^{WCBQ}(\underline{n}). \quad (15)$$

Under the assumptions of the model, the steady state distribution for the First Come First Served (FCFS) admission control of a PS is given by:

$$\pi_K^{FCFS}(\underline{n}) = \pi(0) \prod_{j=1}^{K} \frac{\rho_j^{n_j}}{n_j!} \quad (16)$$

Where

$$\pi(\underline{0}) = \frac{1}{\sum_{\underline{n} \in F_K} \prod_{j=1}^{K} \frac{\rho_j^{n_j}}{n_j!}} \quad (17)$$

Where $F_K \equiv \{\underline{n} \mid \underline{n} \in S_K \text{ but } n_i \notin S_K \text{ for some } i\}$

The service rate here is $\mu_j = \beta\mu_j^{/} + \beta\delta_j\mu_j^{//}$ since the arriving messages are either put into the timeout boxes to discard or processed to generate NofifyPresUp. Therefore, the probability of blocking for FCFS admission control is obtained by:

$$B_{PS}^{FCFS} = \sum_{\underline{n} \in F_K} \pi_K^{FCFS}(\underline{n}). \quad (18)$$

For the event throttle model, the throttle mechanism applies on the buffer of the PS to reduce the number of messages generated. A watcher may use a throttle mechanism during its subscription to the PS for the minimum time period between two notifications of its presentities. The PS is allowed to use a throttling policy in which the minimum time period of notifications is longer than the one given by the watcher. This throttle mechanism can be applied with both FCFS and WCBQ method. The difference will be in the point of servicing the messages. In such cases, the average blocking probability can be obtained from Eq. (15) and Eq. (18) with the modification of the mean processing rate, δ_j as the size of generated NotifyPresUp will vary. It is difficult to standardize such processing rate for event throttling mechanisms since the watcher list is not identical for every watcher. From the earlier discussion of event throttling mechanism, it is obvious that extra processing will be required due to state matching and batching processes which will increase blocking.

B. Efficient Dropping of Buffer Messages

As mentioned earlier that our concern is with the flows that have heavy weights associated in the lower priority classes. These types of flows will force the PS to generate huge amount of messages in quick succession. Since, our classifier and weight adjustment scheme will schedule the higher arrival-rate jobs that have heavy weights associated at the end of the buffer; we propose a mechanism that will drop these kinds of pre-existing jobs from the buffer with the arrival of new jobs from the same source/presentity or from the same message flow in order to ameliorate the message generation process for the PS. A minimum time frame is the key within which if a message at the same flow arrives, the pre-existing message from the buffer will be dropped. The mechanism may be applied periodically during heavy traffic in the network. Note that, we use the term job and message interchangeably.

We may use the square root dimensioning method to compute the minimum time frame. Let, T_{ji} be the mean sojourn time encountered by a job at flow *i* of class *j* and C_{ji} the capacity allotted for flow *i.* If the stability condition $\eta_{ji}\lambda_{ji} < \mu_j C_{ji}$ for *i=1,2,..,N* holds, then (ignoring the constant η for each class for the simplicity of derivation):

$$T_{ji} = \frac{1}{\mu_j C_{ji} - \lambda_{ji}} \quad \forall i = 1,2,..,N \quad (19)$$

Note that, here μ_j is the mean service rate that includes both timeout or 200 (OK) message servicing as well for class *j*, i.e., $\mu_j = \beta\mu_j^{/} + \beta\delta_j\mu_j^{//}$.

Thus, the average sojourn time of messages at class *j* is (assuming $\rho_{ji} = \frac{\eta_{ji}\lambda_{ji}}{\mu_j}$):

$$T_j = \sum_{i=1}^{N} \frac{\lambda_{ji}}{\lambda_j} \left(\frac{1}{\mu_j C_{ji} - \lambda_{ji}}\right) = \frac{1}{\lambda_j} \sum_{i=1}^{N} \frac{\rho_{ji}}{C_{ji} - \rho_{ji}} \qquad (20)$$

The stability condition now reads $\rho_{ji} < C_{ji}$ for *i=1,..,N*. Therefore, our threshold time frame problem reduces to:

Minimize: T_j

With respect to: $\{C_{ji}\}_{i=1}^{N}$

Under the constraint: $D_j = \sum_{i=1}^{N} C_{ji}$

Where, D_j is the total capacity of the PS allocated to class *j*.

Applying the Lagrange multiplier technique, we define:

$$f(C_{j1},...,C_{jN}) = \frac{1}{\lambda_j} \sum_{i=1}^{N} \frac{\rho_{ji}}{C_{ji} - \rho_{ji}} + \beta\left(\sum_{i=1}^{N} C_{ji} - D_j\right). \qquad (21)$$

Solving the equation $\partial f(C_{j1},...,C_{jN}) / \partial C_{ji} = 0$ for every *i=1,2..,N;* we obtain:

$$C_{ji} = \rho_{ji} + \sqrt{\frac{\rho_{ji}}{\lambda_j \beta}} \quad i=1,2,..,N. \qquad (22)$$

From the constraint $\sum_{i=1}^{N} C_{ji} = D_j$ and Eq. (22) we get:

$$\frac{1}{\sqrt{\lambda_j \beta}} = \frac{D_j - \sum_{i=1}^{N} \rho_{ji}}{\sum_{i=1}^{N} \sqrt{\rho_{ji}}} \qquad (23)$$

Introducing the value of Eq. (23) into Eq. (22) yields:

$$C_{ji} = \rho_{ji} + \frac{\sqrt{\rho_{ji}}}{\sum_{i=1}^{N} \sqrt{\rho_{ji}}} \left(D_j - \sum_{i=1}^{N} \rho_{ji}\right) \qquad (24)$$

Therefore the minimum timeframe or sojourn time in the class *j* becomes,

$$T_{\min} = \frac{\sum_{i=1}^{N} \rho_{ji}}{\lambda_j \left(D_j - \sum_{i=1}^{N} \rho_{ji}\right)} \qquad (25)$$

It can be observed that, in order to make the threshold time fair for all classes, η_j has to be the average number of presentities present for each class in the above derivation i.e., total number of presentities in a presence system divided by the total number of classes in a PS. Eq. (25) may be used in the heavily weighted flows of the low priority classes in a round robin fashion during heavy traffic for a period of time. The idea here is to drop a pre-existing job from the buffer in order to reduce generating huge number of messages for the presentities with the same arrival rate in a very short span of time. The time stamp difference between the two job-arrivals of a same flow is compared with T_{min} and if that is less than T_{min}, then a prior job from the buffer will be dropped by the classifier. If a class load is greater than or equal to assigned capacity i.e., $(D_j - \sum_{i=1}^{N} \rho_{ji}) \le 0$ then the threshold becomes negative or infinite. In that case, every two or three etc. alternate pre-existing message from a same presentity or a flow may be dropped. In practical, the accurate notifications for the very rapid state changes of presentities may not be significant for their watchers. The dropping will slow down the end devices which are low in battery capacity/memory receiving NotifyPresUp in very quick succession as well as message generation for the PS.

Alternatively, we may want to keep the dropping rate of the messages to a certain rate. Since the arrivals are Poisson process, the probability of *r* message arrives at a flow in a known period *T* in class *j* is

$$\Pr[R = r \mid T = t] = \frac{(\lambda_j t)^r}{r!} e^{-\lambda_j t} \qquad (26)$$

Therefore, the probability of *r* arrivals to a flow for a length of time can be determined by the *Laplace* transformation as follows:

$$p_r(r) = \int_{t=0}^{\infty} \left\{\frac{(\lambda_j t)^r}{r!} e^{-\lambda_j t}\right\} e^{-st} dt$$
$$p_r(r) = \frac{\lambda_j{}^r}{r!} \left[(-1)^r \frac{d^r}{ds^r} \left(\frac{1}{s + \lambda_j}\right) \right] \Big|_{s=\lambda_j} \qquad (27)$$

From above Eq. (26) and Eq. (27), a PS can compute the probability of specific number of message arrivals for each of the classes in the PS within time T_{min}. The steps to perform dropping of pre-existing messages by the PS requires *O(1)* time whereas the queuing may take place in linear time which is practical. This queuing delay is very low compared to the processing delay of heavily weighted messages; not to mention the bandwidth consumption at the outgoing link.

V. Performance Analysis

We consider 3 groups of messages in the PS with multiple classes in each group. The priorities in the classes are assigned by the number of watchers associated. The blocking probability by each group of messages is taken as the performance measure. We simulate till the maximum number of unacknowledged messages in a group. We applied the dropping method over all classes of group 3 once. The numerical analysis was performed with Java programming language. Later we will show more simulation comparison with Opnet modeler. The average RPID message length is considered to be fixed for simplicity. Thus the common distribution of service times can be considered exponential as

in M/M/1 model. The parameters used are shown in Table 1. 22 channels is a realistic capacity in current wireless system. Bandwidth is allocated in quota of a channel which is a fixed transmission speed. Message flows in our problem request different transmission speeds or the unit of bandwidth (for instance, this may be a 9.6 Kbps channel), and the total number of channels available is the system capacity. As mentioned earlier that allocation of bandwidth should be defined for different groups in order to avoid monopolization of prioritization. A flow using *x* channels can transmit at *x* times the nominal rate of a single channel. The flow effective bandwidth has been furnished in a later section of this chapter. We used the fixed point technique to compute the blocking probabilities. The state change was captured as the number of messages getting serviced and departing from the PS or being dropped to cause a transition from state a_1 to a_2. We define this state transition probability as:

$$P[a_1 -> a_2] = \prod_{j=1}^{K} \frac{n_{a_1}!}{n_{a_2}!(n_{a_1} - n_{a_2})!} p_{ts_j}^{(n_{a_1} - n_{a_2})} (1 - p_{ts_j})^{n_{a_2}} \quad (28)$$

Where, n_{a_1} and n_{a_2} are total number of messages in state a_1 and a_2 respectively and *K* is the number of classes in a group. p_{ts_j} is computed from Eq. (4). The slot duration was kept 1.

Therefore, the average blocking probability of Eq. (15) and Eq. (18) for a group is computed as follows:

$$Block = \sum_{\underline{n}} \pi_K(\underline{n}) \times \sum_{\underline{n}} P[a_1 -> a_2] \quad (29)$$

Where, $\underline{n} \in S_K$ for WCBQ and $\underline{n} \in F_K$ for FCFS.

For FCFS, the messages in each group were served in first come first served manner and were considered not to be classified.

TABLE I
PARAMETERS FOR BLOCKING PERFORMANCE WITH VARYING LOAD

Parameter	Group 1	Group 2	Group 3
Capacity (channels)	4	6	12
Total arrival rate, λ (arr./per min)	1-20	21-50	51-100
Weight (associated watcher/message, random)	1-24	25-49	50-100
Priority (1-3)	High (1)	Medium (2)	Low (3)
q_j (Probability of attempted transmission fails)	.005	.01	.015
E_K (Maximum no of unacknowledged messages)	10	35	75
β (msg/min)	50	50	50
μ' (msg/min)	5	5	5
μ'' (msg/min)	50	50	50

Since, an IMS terminal can have 100 watchers in its list; the maximum randomly generated associated watcher limit was kept 100 for an arriving message (in group 3). The processing load, (λ/δ) was varied from 4 to 10 for the three groups to compute the respective blocking probability. The processing load depends on processing a RPID document and generating NotifyPresUp messages for a presentity's state change. The blocking probability for the three groups as a function of the processing load is shown in Fig. (6), Fig. (7) and in Fig. (8).

The FCFS scheme provides an improvement at the earlier stage in blocking performance for the low priority group where the arrival rates and the associated watchers are low (Fig. (6)). The performance of both the medium priority groups (FCFS and WCBQ) were recorded the same (Fig. (7)) at preliminary stage where as WCBQ supersedes FCFS gradually for higher load. However, performance degradation is experienced by high priority group of messages for FSFC scheme. Our proposed WCBQ provides intelligent contention resolution for group 3 messages. The main reason for the performance gains can be summarized as follows:

WCBQ scheduling discriminates against the arrival rate and associated weight of the arrived messages in the sense of dropping pre-existing messages based on minimum sojourn time. Since, the system under consideration is non-pre-emptive, when the higher arrival rated messages arrives, they are buffered until capacity is free. The channels are utilized as such low arrival rated messages use them when the higher arrival rated messages are not availing them. The average channel utilization for three groups of WCBQ is depicted in Fig. (9) for growing load with WCBQ scheduler. We see that the heavily weighted flows i.e., flows of group 3 utilize the PS capacity earliest which is obvious.

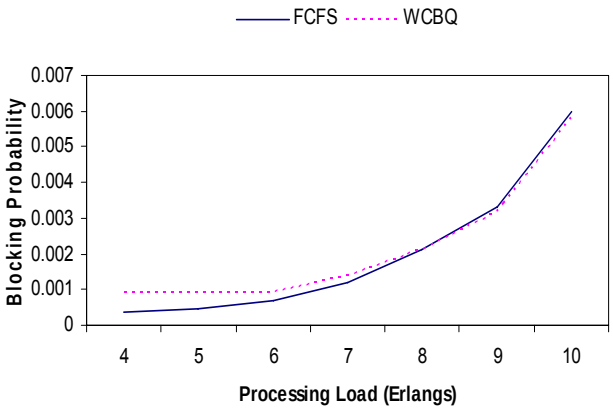

Fig. 6 blocking performance of group 1

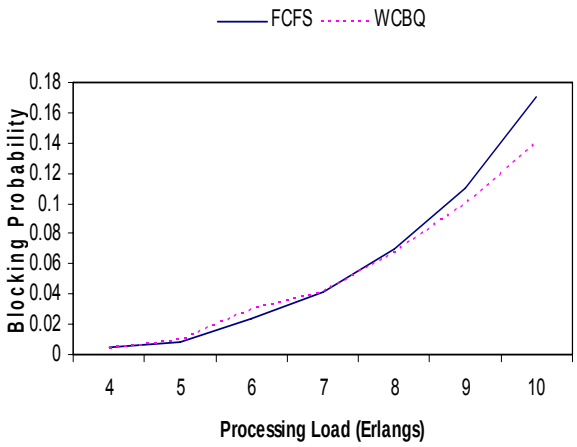

Fig. 7 blocking performance of group 2

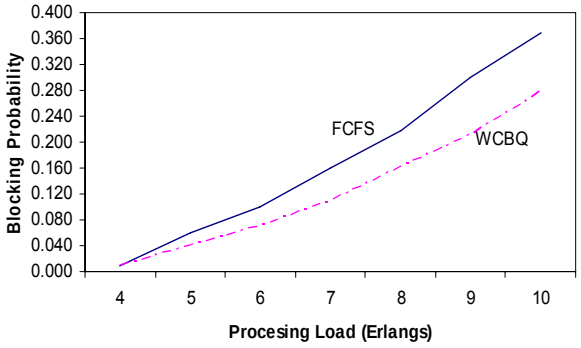

Fig. 8 blocking performance of group 3

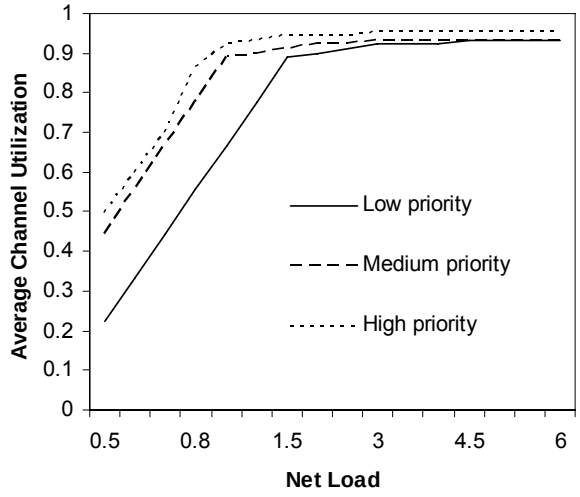

Fig. 9 probability of servicing messages

Here, it is obvious that the partial state event throttling notification will perform even poorly since, the PS will have to find the state difference from the last full state notification, and to batch the state changes to merge them though this will reduce the number of out going messages.

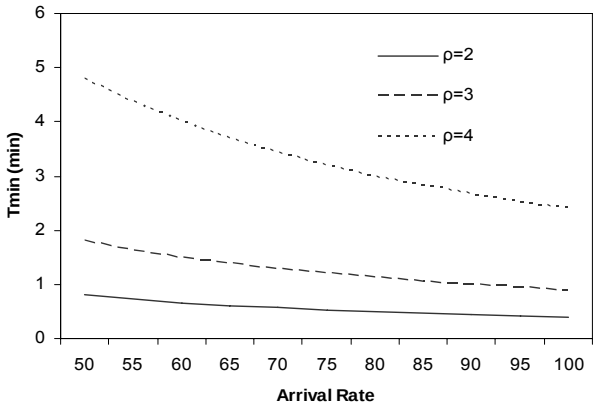

Fig. 10 minimum sojourn time for group 3

Next we performed the experiment for the minimum timeframe of messages of group 3 with different net load (see Fig. (10)). The timeframe goes down with growing arrival rates meaning the wait time reduces for large number of arrivals. The obvious reason for decreasing timeframe is that the traffic intensity was kept fixed.

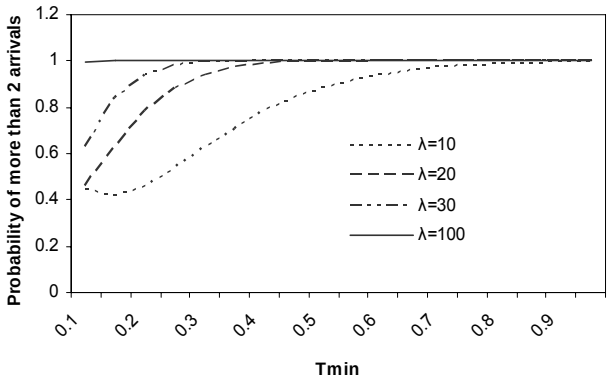

Fig. 11 probability of more than 2 arrivals

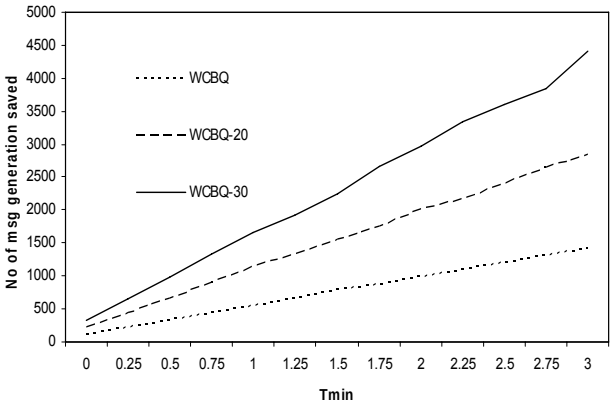

Fig. 12 no of message generation saved

Since, WCBQ will drop a pre-existing message based of another arrival; we find the probability of more than two arrivals with the same time frame. It is obvious (Fig. (11)) that higher arrival rates have almost unit probability in such cases.

Based on our simulations in Fig. (10) and Fig. (11), we performed experiment shown in Fig. (12) to find out the number of message generation saved for the PS by dropping pre-existing heavily weighted messages using our algorithm. Again, the simulation was performed over the group 3 messages for WCBQ, WCBQ-20 and WCBQ-30. We considered WCBQ with different throttle requirement for group 3 messages; the messages were held 20 seconds (WCBQ-20) and 30 seconds (WCBQ-30) for WCBQ before processing them. The pre-existing messages were de-queued once from the buffer-array according to the comparison of arrival time stamps and the threshold minimum time. The number of message generation saved means that the number of messages needed to be generated for the number of dropped messages from PS. If a NotifyPresUp message needs to be traversed via routers, then the PS may use the multicast mechanism with which generation of multiple messages are saved by the PS. However, we consider here the worst case scenario where the PS has to generate a NotifyPresUp message for each of the associated watchers of a presentity in WCBQ. The weights were randomly generated. The more the arrival rate, the more the message generation saved by the PS. Moreover, we see that the more the throttling time, the more gain in saving generating messages. This is easy to perceive from the fact that throttling mechanisms will generate only one NotifyPresUp message for a watcher within the throttled timeframe.

The results achieved in the above set of simulation will vary according to the input model of Table 1. However, we believe our WCBQ will exhibit parallel behavior to the results presented above and will definitely reduce load for a PS.

VI. Effective Bandwidth

One of the important parameters for any admission control system is the effective bandwidth. We provide theoretical expressions for our WCBQ system in this section as it is difficult to capture the exact behavior. The effective bandwidth for FCFS admission control, B_{eff_FCFS} can be obtained from [20] if the messages at PS behave as elastic traffic. For a given average allocated rate of bandwidth *b* for a class, it is defined as in [20]:

$$B_{eff\_FCFS} = K.h \tag{30}$$

Where,

$$h = \left(\frac{1}{b} + \frac{1}{\lambda_j l} \right)^{-1} \tag{31}$$

K is the number of classes/sources, l is the file size and λ_j is the arrival rate of a class *j*.

It is difficult to estimate the total bandwidth savings by using the throttle mechanism over a subscription, since such estimates will vary depending on the usage scenarios. However, it is easy to see that given a subscription where several full state notification would have normally been sent in any given throttle interval, a throttled subscription would only send a single notification during the same interval, yielding bandwidth savings of several times the notification size. With partial-state notifications, drawing estimates is further complicated by the fact that the states of consecutive updates may or may not overlap. However, even in the worst case scenario, where each partial update is to a different part of the full state, a throttled notification merging all of these *n* partial states together should at a maximum be the size of a full-state update. In this case, the bandwidth savings are approximately *n* times the size of the NotifyPresUp header. It can be observed that, the available compression schemes may be applied simultaneously with the WCBQ or throttle mechanisms for compound bandwidth savings.

When the message flows are elastic, the effective bandwidth required by an additive flow in a class *j* can be written as:

$$\sum_{j=1}^{K} N_j \alpha_j + \alpha_j \leq 1 \tag{32}$$

With

$$\alpha_j = \frac{\lambda_j a(1 - \phi_j(-(\log c)/a))}{\log c} \tag{33}$$

Where

$$a > 0, c \in (0,1), \phi_j(\theta) = \int_0^{\infty} \exp(\theta y) dG_j(y). \tag{34}$$

The detail of the above expressions with parameters are provided in the Appendix which is the application of Kingman's [14] result into the G/G/1 system [27] that is merged into the M/G/1[15] system to be applicable to a flow

of our WCBQ.

VII. TRANSITION PROBABILITIES

The transition probabilities of presentity's states can be computed from a simple Markov model. The arrival rates are considered to be equivalent to the steady state probability of presentities. Let the number of states for a presentity to change be arbitrary. The activity elements of a presentity can hop among any state from its initial state which can be modeled as a pure birth process. However, the probabilities of coming back to its initial state are equivalent. The scenario is depicted in Fig. (13). We assume that state zero is the initial position of a presentity which may be thought of its actual anchoring position. The other states may represent the presentity's state change to busy, idle, not available etc. These state changes reflect the different values of the activity elements for instance; class, content-type, place-type, privacy, relationship, sphere etc. in the RPID (Rich Presence Information Data Format) extension. We also assume that the presentity's initial state is saturated so that upon completion of one state change, it will enter to another statically identical state instantaneously. Let, *p* the rate that denotes the presentity's state change and *q* denotes the state transition rate at which the presentity changes state from *m* to state *0*.

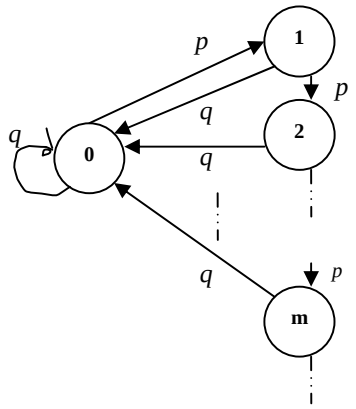

Fig. 13 Markov chain for a presentity's state change

Let, v_0 and v_m be the equilibrium probability of state *0* and *m* respectively.

$$v_m = \left(\frac{p}{p+q}\right)^m v_0 \tag{35}$$

By the law of total probability,

$$v_0 = \frac{q}{p+q} \tag{36}$$

Substituting Eq. (36) into Eq. (35),

$$v_m = \left(\frac{p+q-q}{p+q}\right)^m v_0$$
$$\Rightarrow v_m = (1-v_0)^m v_0 \tag{37}$$

The steady state probability for different transition probabilities is shown in Fig. (14).

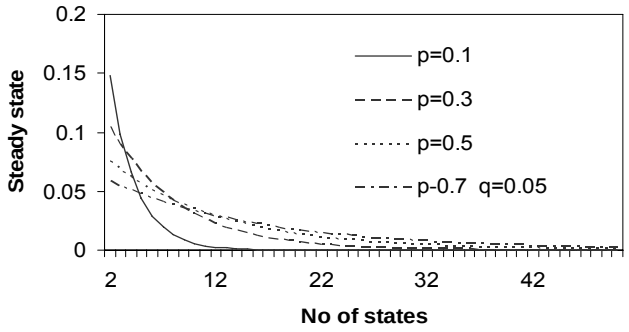

Fig. 14 steady state probability of a presentity's states

VIII. COST CONSUMPTION FOR PS

In this section, let us represent the cost of an IMS presence system in terms of number of messages generated by the Presence Server to provide the presence service that includes: a) The PS has to generate 200OK message to acknowledge receipt of RPID message from each presentity (see Fig. (2)); b) The PS has to generate NotifyPresUp message to notify each associated watcher of a presentity about the presentity's state change (see Fig. (2)). In summary, if a presentity changes state, the PS generates one 200OK message and number of NotifyPresUp messages upon receipt of a RPID message. Let *H* be the total number of IMS presentities observed by the IMS watchers via a PS in the system. Thus, assuming no RPID is corrupted, the total cost of presentities movement for FCFS is:

$$C_{FCFS} = \sum_{d=1}^{H} v_{md}(1+M_d) \tag{38}$$

Where, M_d is the number of NotifyPresUp messages generated by the PS for a state change of the d^{th} presentity to notify its watchers. 1 is added due to the 200OK message generation upon receipt of a RPID. v_{md} is the steady state probability of the state change of the d^{th} presentity which is presented in Eq. (37).

Thus, the total cost of a presence system at steady state in a real-time interval *T*, in the long run on the average for FCFS is:

$$C(t)_{FCFS} = \int_0^T C_{FCFS}\,dt \approx T\sum_{d=1}^{H} v_{md}(1+M_d) \tag{39}$$

Let, $U_d = f(T_{\min}) = f(\eta_j, v_{md}, \mu_j, D_j)$ is the rate in

the PS, the RPID messages are dropped for the d^{th} presentity. Thus, the total cost for our dropping algorithm WCBQ in a real-time interval T is:

$$C(t)_{WCBQ} \approx T\sum_{d=1}^{H} v_{md}(1+M_d) - T\sum_{d=1}^{H} U_d M_d, \quad U \geq 0 \qquad (40)$$

$$C(t)_{WCBQ} = T\sum_{d=1}^{H} \{v_{md}(1+M_d) - U_d M_d\}, \quad U \geq 0 \qquad (41)$$

U is zero if the dropping is not applied to a class.

IX. Simulation for Cost Consumption

We generated a simulation environment with Opnet Modeler. The environment considers that a PS was serving 1000 IMS terminals which were watching each other and generating RPID. Every terminal has a watch list that indicates the terminals it is watching. We kept all the watcher list size to be maximum i.e. 100 to accomplish the justification of our work. Since, the number of watcher associated with each terminal was fixed (100) we needed not to classify further in a class according to the weight. Every terminal also has a list of watcher watching it with the associated arrival rate. Here, arrival rate represents the class since classes are distinguished by message arrival rate i.e., the presentities message generation rate to the PS. We assumed that the group 3 messages were arriving from 51 message/minute to 100 messages/minute i.e., there are 50 classes in the PS from class number 51-100 in group 3. We applied our message dropping mechanism to these 50 classes. The total channel capacity, D of the PS for these 50 classes was kept to be 25 assuming a channel can service 2 message flows i.e., 2 classes in this instance (since the number of associated watchers is equal for each input message and thus not required to classify further as weight under a class). This means a class j will require 0.5 of channel. Unlike the previous simulation model, we vary the traffic intensity ρ_j in a class j randomly where $0 < \rho_j < 0.5$. The simulation was run for 50 minutes. We assumed no message was corrupted and all the messages were acknowledged properly upon arrival at the PS. Both WCBQ and throttled WCBQ were compared with FCFS.

The following figure (Fig. (15)) shows the number of terminals generating messages to the PS inside a class at the class rate. This is actually the computation of η_j for each class j which is required to compute the number of associated watchers and dropped messages. We see that the plots of η_j are random since the arrival rate/message generation rate was generated randomly for each of 1000 terminals.

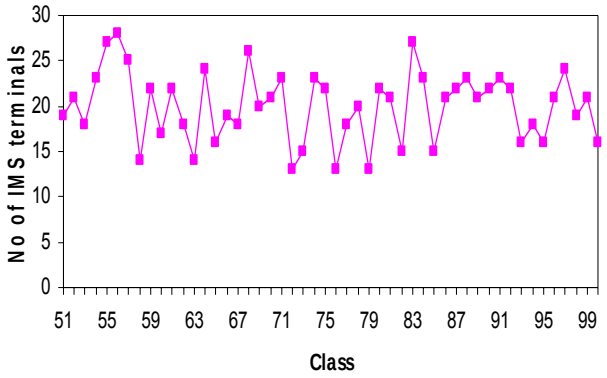

Fig. 15 number of terminals watching at the class rate

Fig. (16) represents the average number of messages dropped in every minute in a class on the average. We see a sharp growth of dropping at the later classes as the later classes are served later and the threshold time is less for these classes. The spikes represent the random behavior of the parameters of the simulation.

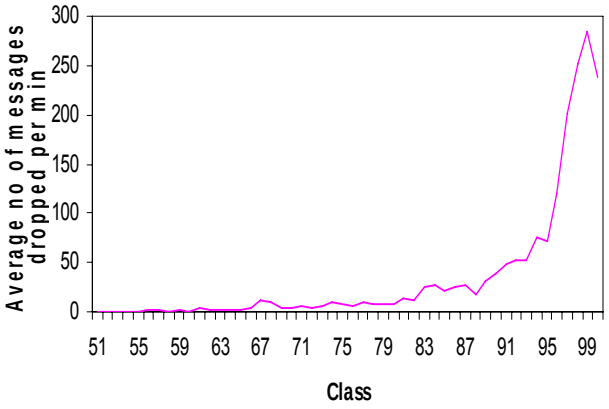

Fig. 16 messages dropped per minute on the average

The total number of messages dropped in the simulation lifetime is provided in the following figure (Fig. (17)). Again, we see that the later classes produce high number of message-drop. This graph is the consequence of Fig. (16).

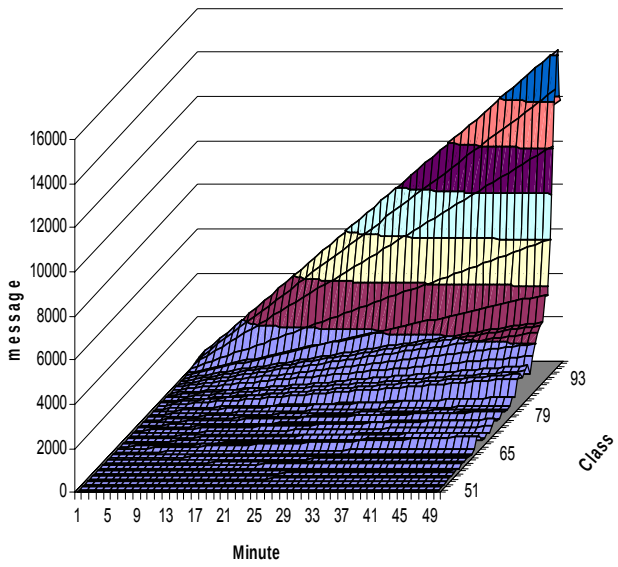

Fig. 17 cumulative message drop during simulation period

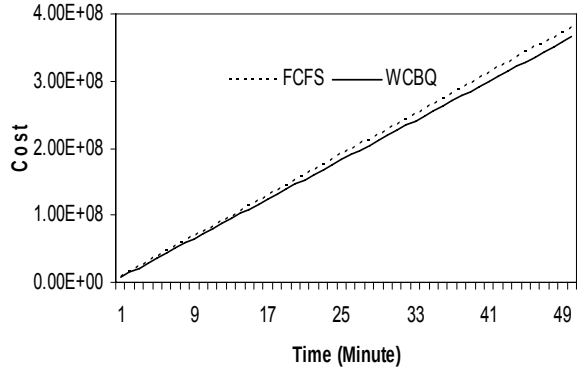

Fig. 18 comparison of message generation cost

Fig. (18) illustrates the cost comparison of WCBQ and FCFS. According to the cost computation expressions, the cost represents number of messages generated by the PS due to the arriving RPID messages at the PS. Since the every terminal is watching to its maximum capacity i.e.100 watchers and since a 200OK message needs to be generated for every RPID arrival, the total number of messages generated by the PS for each arriving RPID is 101 (100 NotifyPresUp and 1 200OK message) for FCFS. For WCBQ, 100 messages were saved due to every RPID drop. The PS still needed to generate one 200OK message to acknowledge the generating terminal for every dropped RPID. Thus, WCBQ cost was computed by subtracting 'number of dropped messages * 100' from the corresponding FCFS cost. As mentioned earlier that the message size were less than IP packet. We used the Process module of Opnet modeler to initiate the message arrivals, Queue module to set the service behavior and Sink module to count messages and dispose the message to free up memory. The built in process module 'acb_fifo' of Opnet modeler was used to emulate FCFS system in an infinite buffer environment. We considered that the arriving message sizes are equal in a class and thus the service rate is same for an individual class. The cost difference was found 1.48E+07 in the final minute of simulation.

Next, we compared the performance of throttled WCBQ-20 and WCBQ-30 with WCBQ and FCFS. Since with the throttled mechanism, the minimum elapsed time between two consecutive NotifyPresUP messages destined to one terminal is 20 seconds for WCBQ-20 and 30 seconds for WCBQ-30, it means that each of 1000 IMS terminals in the simulation receives three (60 seconds / 20 seconds) for WCBQ-20 or two (60 seconds / 30 seconds) for WCBQ-30 NotifyPresUp messages in every minute depending on the RPID generation rate of the terminals they are watching. For this, the message generation rate of the corresponding terminals of every node was calculated to determine whether message was needed to be generated after every minimum throttle period. Since our simulation was performed for the heavily arrival rated messages, we found that every terminal was destined to receive a NotifyPresUp after the minimum throttle period of 20 and 30 seconds. The watcher list was traversed for every node to find out how many number of terminals was watching a node with what rate. The following figure (Fig. (19)) shows the average number of watchers watching at each class rate for a node on the average. We find that the number is always at least greater than one i.e., there is always more than or equal to one node/terminal who is watching a node at each class rate.

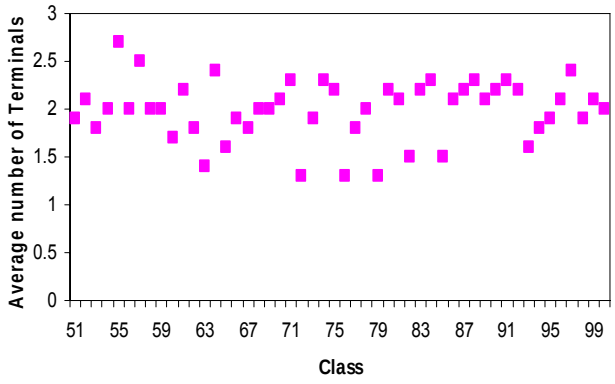

Fig. 19 average no of watchers watching a presentity at each class rate

Fig. (20)). It is to be mentioned that the cost of FCFS-20 and FCFS-30 are the same as the cost of WCBQ-20 and WCBQ-30 since our cost function only computes the number of messages generated from the PS. In practical, the throttling methods differ in generating NotifyPresUp messages in terms of size as RPIDs destined to same presentity are batched and combined to produce one NotifyPreUp. But since it is difficult

to compute such message size, we express out cost function in terms of number of message generation. For the costs in Fig. (20), we computed the number of 200OK messages generated for every arrival of RPID plus the number of throttled NotufyPresUp messages generated (i.e., 1000*3 or 1000*2) for every terminal per minute.

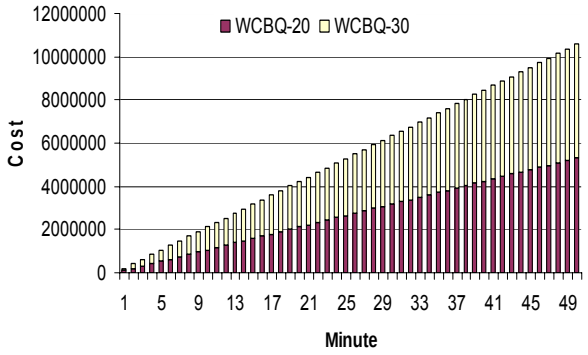

Fig. 20 cost comparison of WCBQ-20 vs WCBQ-30

Fig. (21) and Fig. (22) illustrate the number of message generation saved for WCBQ-20 and WCBQ-30 with respect to FCFS and WCBQ respectively. The cost of WCBQ-20 and WCBQ-30 were subtracted from the cost of FCFS and WCBQ to find the number of saved messages during the simulation lifetime. The computed difference during the final minute of the simulation runtime with respect to FCFS was found to be 515650000.00 and 515700000.00 for WCBQ-20 and WCBQ-30 respectively where as the computed difference with respect to WCBQ was found to be 498293800.00 and 498343800.00 for WCBQ-20 and WCBQ-30 respectively.

X. CONCLUSIONS

It is recognized that a framework for reducing load is essential in IMS for large scale of traffic to facilitate presence service to the terminals. Obviously, there has to be a compromise between the amount of information sent, the frequency of the notification, and the bandwidth used. Our WCBQ is a preliminary work of a novel queuing mechanism to provide class differentiation and to reduce the load in the IMS presence server during heavy traffic. The grouping was done to assign priority on the arrived messages. In our test-bed, the dropping application was limited to group 3 only in order to balance the real time view and the notification rate. The tasks of admission controller for the PS are demonstrated. The optimized dropping time frame has been developed based on Lagrange multiplier technique. The PS benefits significantly from the algorithm in terms of reducing the number of message generations. The channel allocation and mean service rate will affect the performance of the PS. Nonetheless, the dropping mechanism definitely reduces load from the PS when the message arrival rate is high and the number of watchers watching presentities are large. The determination of group size and the application rate of our WCBQ are to be configured by the IMS presence service providers.

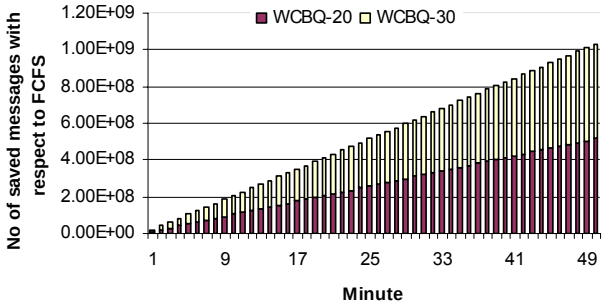

Fig. 21 no of message generation saved compared to FCFS

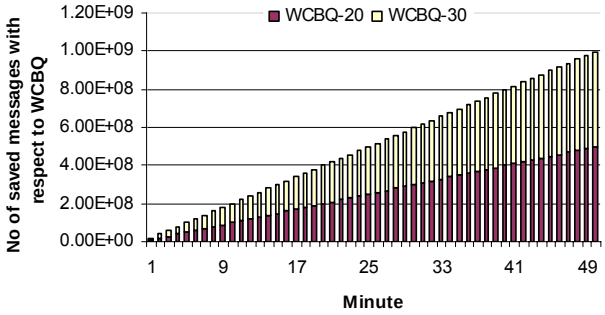

Fig. 22 no of message generation saved compared to WCBQ

Simulation was performed to show the behavior of the model in terms of cost functions. Moreover we demonstrated results of throttle mechanisms with both WCBQ and FCFS. WCBQ with throttle mechanism performs better in terms of generating messages; though their blocking probabilities are expected to be high.

The IETF engineers are still working on some optimal solution in the presence service. Sending less information in presence documents may lead to IMS users not getting a good experience with presence systems used from wireless terminals. Sending presence information less periodically will lead to an inaccurate presence view of the presentities. Above all, the best results perhaps can be achieved by applying the above mentioned techniques simultaneously. Work is underway on defining such hybrid admission controller to maximize revenue for service providers.

APPENDIX

We define, $\phi(\theta) = E[\exp(\theta(s_n - \tau_n)]$, The Laplace

transformation of the random variable $s_n - \tau_n$, where s_n is the service time of the *nth* message and τ_n is the time between arrivals of messages *n* and *n+1* for the GI/GI/1 system.

In a GI/GI/1 queue we assume that:

1. $(s_n)_n$ is a sequence of independent random variables with the common cumulative distribution function (c.d.f.) *G(x)*, namely, $P(s_n \le x) = G(x)$ for all $n \ge 1, x \ge 0;$

2. $(\tau_n)_n$ is a sequence of independent random variables with the common c.d.f. *F(x)*, namely, $P(\tau_n \le x) = F(x) = 1 - \exp(-\lambda x)$ for all $n \ge 1, x \ge 0.$ λ has been defined before with the superposition of all independent Poisson processes.

We also assume that there exists $m > 0$ such that $\phi(m) < \infty$. From Kingman's result [14] we know that if $\theta > 0$ such that $\phi(\theta) \le 1$, then

$$P(W_n \ge x) \le e^{-\theta x} \quad \forall x > 0, n \ge 1 \qquad \text{(A.1)}$$

And $$P(W \ge x) \le e^{-\theta x} \quad \forall x > 0 \qquad \text{(A.2)}$$

Where, W_n is the waiting time in queue of the *nth* message.

The above results may be used in our multiclass M/G/1 WCBQ if we can reduce our queue to G/G/1 queue.

With the probability λ_k / λ, the *nth* message will be a message of class *k*, Let X_k be the time that elapses between the arrival of the *nth* message and the first arrival of a message of class *k*. Since the arrival process of each class is Poisson, and therefore memoryless, we know that X_k is distributed according to an exponential random variable with rate λ_k.

Therefore,

P((n+1)st message is of class k) = $P(X_k < \min_{j \ne k} X_j))$

$$= \int_0^\infty P(x < \min_{j \ne k} X_j) \lambda_k e^{-\lambda_j x} dx$$

$$= \int_0^\infty \prod_{j \ne k} P(x < X_j) \lambda_k e^{-\lambda_j x} dx$$

$$= \lambda_k \int_0^\infty \prod_j e^{-\lambda_j x} dx$$

$$= \lambda_k \int_0^\infty e^{-\lambda x} dx = \frac{\lambda_k}{\lambda}$$

Let us now determine the c.d.f. *G(x)* of the service time of an arbitrary message.

G(x)=P(service time of new message $\le x$)

$$= \sum_k P(\text{service time of new message} \le x \text{ and new message of type } k)$$

$$= \sum_k P(\text{service time of new message} \le x \mid \text{new message of type } k) \frac{\lambda_k}{\lambda}$$

From the above and from the law of total probability and Bayes' formula,

$$G(x) = \sum_k \frac{\lambda_k}{\lambda} G_k(x)$$

In particular, the mean service time $1/\mu$ of our multiclass M/G/1 WCBQ is given by

$$\frac{1}{\mu} = \int_0^\infty x dG(x) = \sum_k \frac{\lambda_k}{\lambda} \int_0^\infty x dG_k(x) = \sum_k \frac{\rho_k}{\lambda}$$

With $\rho_k = \frac{\lambda_k}{\mu_k}$.

Thus, for stability condition of $\sum_k \rho_k < 1$,

$$\phi(\theta) = E[e^{\theta(s_n - \tau_n)}]$$
$$= E[e^{\theta s_n}] E[e^{-\theta \tau_n}]$$
$$= \left(\frac{\lambda}{\lambda + \theta}\right) E[e^{\theta s_n}]$$

Now, $E[e^{\theta s_n}] = \int_0^\infty e^{\theta y} dG(y) = \sum_k \frac{\lambda_k}{\lambda} \int_0^\infty e^{\theta y} dG_k(y)$

Therefore, $\phi(\theta) = \sum_k \frac{\lambda_k}{\lambda + \theta} \int_0^\infty e^{\theta y} dG_k(y)$

Thus, $P(W \ge a) \le c$ if $\phi(-(\log c)/a) \le 1$ which is of the form

$$\sum_k \frac{\lambda_k}{\lambda - (\log c)/a} \int_0^\infty e^{-(\log c) y / a} dG_k(y) \le 1.$$

REFERENCES

[1] 3GPP TSG SSA, IP Multimedia Subsystem (IMS) – Stage 2 (release 7), TS 23.228 v7.3.0, 2006-03.

[2] 3GPP. Presence service; Architecture and functional description; TS 23.141, v7.0.0, 2005-09.

[3] H. Sugano, S. Fujimoto, G. Klyne, A. Bateman, W. Carr, and J. Peterson, Presence Information Data Format (PIDF). Internet-Draft RFC 3863, *Internet Engineering Task Force* 2004.

[4] H. Schulzrinne, V. Gurbani, P. Kyzivat and J. Rosenberg, "RPID – Rich Presence Extensions to the Presence Information Data Format (PIDF)." RFC 4480, *Internet Engineering Task Force*, Jul 2006.

[5] A. B. Roach, J. Rosenberg, B. Campbell, "A Session Initiation Protocol (SIP) Event Notification Extension for Resource Lists." RFC 4662, *Internet Engineering Task Force*, Aug 2006.
[6] A. W. Berger, W. Whitt, "Effective bandwidths with priorities." *IEEE/ACM Transactions on Networking*; Vol: 6 (4), 1998, pp: 447-460.
[7] S. Jordan, K. Jogi, C. Shi, I. Sidhu, "The variation of optimal bandwidth and buffer allocation with the number of sources." *IEEE/ACM Transactions on Networking*; Vol: 12 (6), 2004, pp: 1093-1104.
[8] D. Ayyagari, A. Ephremides, "Admission control with priorities: approaches for multi-rate wireless system." *Mobile Networks and Applications*; Vol: 4 (3), 1999, pp: 209-218.
[9] T. Janevski, B. Spasenovski, "QoS provisioning for wireless IP networks with multiple classes through flexible fair queuing." *Global Telecommunications Conference*, GLOBECOM '00. IEEE 2000; Vol: 3 (27), pp:1677 – 1701.
[10] A. Roach, "Session Initiation Protocol (SIP)-Specific Event Notification." RFC 3265, *Internet Engineering Task Force* 2002.
[11] G. Camarillo, "Compressing the Session Initiation Protocol (SIP)." RFC 3486, *Internet Engineering Task Force*, February 2003.
[12] H. Hannu, J. Christoffersson, S. Forsgren, K.-C. Leung, Z. Liu, and R. Price, "Signalling Compression (SigComp) – Extended Operations." RFC 3321, *Internet Engineering Task Force*, January 2003.
[13] H. Hannu, J. Rosenberg, C. Bormann, J. Christoffersson, Z. Liu, and R. Price, "Signalling Compression (SigComp)." RFC 3320, *Internet Engineering Task Force*, January 2003
[14] J. F. C. Kingman JFC. "Inequalities in the Theory of Queues." *Journal of Royal Statistical Society*; Vol: 32 (B), 1970, pp: 102-110.
[15] K. S. Trivedi, Probability & Statistics with Reliability, Queuing, and computer Science Applications, New Delhi, Prentice' Hall of India, 2003.
[16] P. Marbach, "Analysis of a static pricing scheme for priority services." *IEEE/ACM Transactions on Networking* (TON); Vol: 12 (2), 2004, pp: 312 – 325.
[17] J. R. Moorman, J. W. Lockwood, "Multiclass priority fair queuing for hybrid wired/wireless quality of service support." *Proceedings of the 2nd ACM international workshop on Wireless mobile multimedia*, 1999, pp: 43 – 50.
[18] "Traffic Management Specification v4.0." ATM Forum Document AF-TM-0056.000, 1996.
[19] C. C. Beard, V. S. Frost, "Prioritized Resource Allocation for Stressed Networks." *IEEE/ACM Transactions on Networking*, 2001; Vol: 9 (5), pp: 618-633.
[20] A. W. Berger and Y. Kogan, "Dimensioning Bandwidth for Elastic Traffic in High-Speed Data Networks." *IEEE/ACM Transactions on Networking*; Vol: 8 (5), 2000, pp: 643-654.
[21] H. K. Telio, E. Leppanen, M. Lonnfors, J. C. Requena, "Functional Description of Event Notification Filtering." RFC 4660, *Internet Engineering Task Force*, 2006.
[22] A. Niemi, "Session Initiation Protocol (SIP) Event Notification Extension for Notification Throttling." Internet-Draft draft-niemi-sipping-event-throttle-04, *Internet Engineering Task Force* 2006; Work In progress.
[23] M. Ghaderi and R. Boutaba, "Call Admission Control for Voice/Data Integration in Broadband Wireless Networks." *IEEE Transactions on Mobile Computing* 2006; Vol: 5 (3), pp: 193-207.
[24] S. Floyd and V. Jacobson, "Link-sharing and resource management models for packet networks," *IEEE/ACM Transactions on Networking* 1995; Vol: 3(4), pp: 365-386.
[25] F. Risso, "Decoupling Bandwidth and Dlelay Properties in Class Based Queuing," *Computers and Communications*, Proceedings. Sixth IEEE Symposium on, 3-5 July 2001, pp: 524 – 531.
[26] A. Striegel and G. Manimaran, "Dynamic class-based queue management for scalable media servers," *Real-Time Technology and Applications Symposium*, RTAS, Proceedings. Sixth IEEE 31 May-2 June 2000, pp: 228 – 236.
[27] L. Kleinrock, Queuing Systems, Vol. 2, Computer Applications, John Wiley & Sons, New York, 1976.

