A Framework for Performability Modeling of Messaging Services in Distributed Systems*

Srinivasan Ramani
IBM Corporation,
3039 Cornwallis Drive,
RTP, NC 27709
sramani@us.ibm.com

Katerina Goševa-Popstojanova
Lane Dept. of CS and Elec. Engr.
West Virginia University
Morgantown, WV 26506-6109
katerina@csee.wvu.edu

Kishor S. Trivedi
CACC, Department of ECE
Duke University,
Durham, NC 27708-0291
kst@ee.duke.edu

Abstract

Messaging services are a useful component in distributed systems that require scalable dissemination of messages (events) from suppliers to consumers. These services decouple suppliers and consumers, and take care of client registration and message propagation, thus relieving the burden on the supplier. Recently performance models for the configurable delivery and discard policies found in messaging services have been developed, that can be used to predict response time distributions and discard probabilities under failure-free conditions. However, these messaging service models do not include the effect of failures. In a distributed system, supplier, consumer, and messaging services can fail independently leading to different consequences. In this paper we consider the expected loss rate associated with messaging services as a performability measure and derive approximate closed-form expressions for three different quality of service settings. These measures provide a quantitative framework that allows different messaging service configurations to be compared and design trade-off decisions to be made.

1 Introduction

Messaging services are a useful component in distributed systems that require scalable dissemination of messages (events) from suppliers to consumers. Examples of messaging service specifications include the CORBA® Notification Service and the Java™ Message Service (JMS). These services act as an intermediary between suppliers and consumers and take care of client registration and message propagation, relieving the burden on the supplier. Thus, the supplier and consumer are not tied up in a client-server type of interaction, but rather are "decoupled". Different distributed systems have different message delivery guarantee requirements. Some systems might require good performance (throughput) and be willing to accept best effort delivery. Other systems might require strict message delivery guarantees. To satisfy these range of requirements, messaging services offer a variety of configurable quality of service settings.

If the requirements for a system, specifically the delivery guarantees, are well defined, one can attempt to deduce the quality of service settings for the messaging service by examining the delivery semantics provided by each setting [7]. But if multiple alternatives are acceptable, a quantitative method is required so that design trade-off decisions can be made. Literature exists on the performance modeling of client-server [3] and producer-consumer [1] systems. For messaging services, the recent work [8] on performance models for the configurable delivery and discard policies found in the CORBA Notification Service is relevant. Pure performance models, however, assume failure-free operation of the components involved and compute performance measures like throughput and discard probabilities due to overflowing queues. However, failures of components within a system could lead to performance degradation. The level of degradation often depends on the failures that have occurred in the system. Also, in distributed systems employing a messaging service, the supplier, consumer, and messaging service can fail independently (due to application software, operating system, or hardware failure) leading to different degrees of message loss. Thus, the effect of partial failures [5] needs to be factored in to yield a comprehensive evaluation of the system. Pure availability models on the other hand, only consider the failures of components and compute measures like availability of a system. Thus, to get a comprehensive picture, a combination of performance and availability models is needed. This

*This work was funded in part by Telcordia Technologies as a core project in the Center for Advanced Computing and Communication (CACC).

Proceedings of the Eighth IEEE international Conference on Engineering of Complex Computer Systems (ICECCS'02)
1050-4729/02 $17.00 © 2002 IEEE

concept was first proposed by Meyer and called "performability" [4]. Although, performability models have been proposed for client-server systems [6], [2], performability models for a distributed system that uses a messaging service have not been explored in detail. In this paper, we consider the expected loss rate of messaging services as a performability measure, since it is affected by both pure performance and availability of system components. We derive approximate closed-form expressions for three different quality of service (QoS) settings - "best effort", "persistent connections", and "persistent connections and messages". We then show how the model can be used to carry out design tradeoff decisions.

The rest of the paper is organized as follows. Section 2 presents a brief background on the architecture of a typical messaging service and the internal settings commonly found in these services. In Section 3, a performability modeling framework for messaging services is developed. Next, Section 4 considers the loss rate expressions when deadline violations are not considered. In Section 5, the contribution of deadline violations to the loss rates is discussed and in Section 6, the overall loss rates (with and without deadline violations) are presented. Section 7 presents an illustration of the application of the framework developed in this paper to other scenarios and Section 8 offers concluding remarks.

2 Background on messaging services

The logical view of a messaging service with a single supplier and consumer is shown in Figure 1. The event

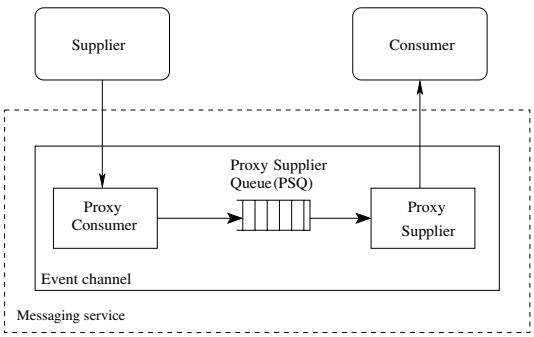

Figure 1. Logical view of a messaging service

channel is the entity within the messaging service that propagates messages from suppliers to consumers. Each supplier connects to its proxy consumer inside each event channel while each consumer connects to its proxy supplier. The proxy supplier queue (PSQ), one for each consumer, queues messages bound for the corresponding consumer. The proxies help to decouple the supplier and consumer. They also make it possible for the supplier to be unaware of the actual number of consumers connected to the channel (multiple consumers is typical in a publish/subscribe messaging service). This also helps isolate the supplier from failures of the consumer(s) so long as the messaging service itself is available. Suppliers and consumers can simultaneously connect to multiple event channels. They just need separate proxies for each channel they connect to.

As mentioned before, messaging services have a similar set of configurable features or settings. The internal settings relevant to this paper are:
EventReliability - When set to "BestEffort", the messaging service does not provide delivery guarantees. All messages inside the messaging service will be lost if the messaging service process crashes. When set to "Persistent", the messaging service maintains the internal queues and state information in persistent storage.

ConnectionReliability - When set to "BestEffort", connections between clients and the messaging service are lost when any of them crashes and is then restarted. Also, if a consumer crashes, the messaging service will discard all events queued for that consumer that have not been delivered yet. When set to "Persistent", the messaging service keeps retrying connections, ignoring temporary failure indications, in the hope that the clients will come back up. See [7] for details of the *EventReliability* and *ConnectionReliability* settings.

OrderPolicy - Specifies event delivery order. For example, in the CORBA Notification Service, this policy can be set to FIFO (first in first out), Priority or Deadline. When the delivery order is based on priority, events are delivered in the order of their priorities, with the highest priority event being delivered first. When delivery order is based on deadline, events are delivered in the order of their deadlines, with the event having the earliest deadline being delivered first.

MaxEventsPerConsumer - The maximum number, k,of events that may be queued in the master queue for a consumer at any time.

DiscardPolicy - Order in which events are discarded when overflow conditions arise in queues. Could be FIFO, LIFO (last in first out), Priority, or Deadline order. When the discard order is based on priority, events are discarded in the order of their priorities, with the lowest priority events being discarded before higher priority events. When discard order is based on deadline, events are discarded in the order of earliest deadline first.

Messaging services may be configured for the following quality of service settings: (1) Best effort (BE): *EventReliability* and *ConnectionReliability* are set to "BestEffort", (2) Persistent connections (PC): *EventReliability* is set to "BestEffort" and *ConnectionReliability* is set to "Persistent", and (3) Persistent connections and messages (PCM):

EventReliability and *ConnectionReliability* are set to "Persistent".

3 Performability modeling framework for messaging services

In this section, we start developing our performability modeling framework for messaging services, by explaining the two constituent models - an availability model and a performance model. We will use the expected loss rate associated with messaging services as a performability measure to evaluate different messaging service configurations and QoS settings. Message losses could occur due to queue overflows and deadline violations (which are performance related losses), as well as messaging service or consumer process crashes (which are availability related losses). Therefore, loss rate is a good performability measure.

3.1 Availability model

The availabilities of the supplier, messaging service and the consumer denoted by A_S, A_M, and A_C respectively, have to be computed from detailed lower-level availability models. When steady state availability measures are required, it is convenient to reduce the models to two-state availability models [10] and derive equivalent failure (γ_{eq}) and repair rates (τ_{eq}). The two-state equivalent models also simplify the derivation of closed-form solutions using a hierarchical approach such as the ones used in this paper.

To illustrate, let us consider that the supplier, messaging service, and consumer are running on separate nodes in a network and that their availabilities are modeled by the continuous time Markov chain (CTMC) shown in Figure 2(a). The availability model in Figure 2(a) considers two types of

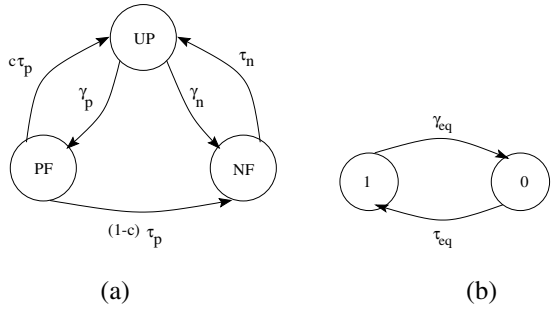

Figure 2. CTMC model of failure behavior - (a) Full model, (b) 2-state equivalent

failures; the process crashes with rate γ_p and the node hosting the process crashes with rate γ_n. If the process crashes, an attempt is made to restart the process. Restart completes at rate τ_p and succeeds with probability c. If the restart is unsuccessful (with probability $1-c$), node repair that completes with rate τ_n is attempted. Node repair is also needed when the node crashes. The steady state probabilities can be derived as:

$$\pi_{UP} = \left[1 + \frac{\gamma_p}{\tau_p} + \frac{1}{\tau_n}(\gamma_n + (1-c)\gamma_p)\right]^{-1},$$
$$\pi_{PF} = \frac{\gamma_p}{\tau_p}\pi_{UP}, \qquad \pi_{NF} = \frac{1}{\tau_n}(\gamma_n + (1-c)\gamma_p)\pi_{UP}.$$

The availability is given by $A = \pi_{UP}$.

The two-state equivalent availability model for Figure 2(a) is shown in Figure 2(b). The equivalent failure and repair rates, γ_{eq} and τ_{eq} and hence availability A can be derived as:

$$\gamma_{eq} = \frac{\pi_{UP}(\gamma_p + \gamma_n)}{\pi_{UP}} = \gamma_p + \gamma_n,$$
$$\tau_{eq} = \frac{\pi_{PF}\, c\tau_p + \pi_{NF}\,\tau_n}{\pi_{PF} + \pi_{NF}}, \qquad A = \frac{\tau_{eq}}{\gamma_{eq} + \tau_{eq}}. \tag{1}$$

In this paper, we use the superscripts (S), (M), and (C) when referring to the failure and repair rates of the supplier, messaging service, and consumer, respectively. An overall availability model for the whole system would consist of an 8-state CTMC representing all combinations in which the supplier, consumer, and messaging service can fail and be repaired. This model is shown in Figure 3. Each state is

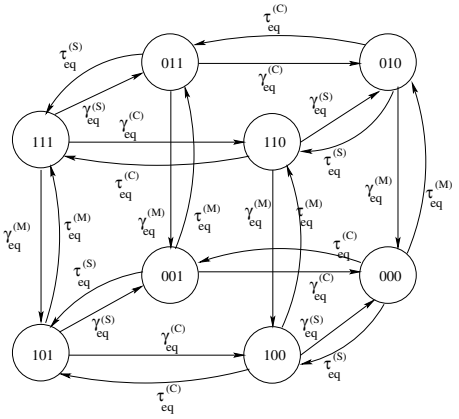

Figure 3. 8-state availability model

labeled by a 3-tuple (SMC) indicating the up/down status of the supplier (S), messaging service (M), and consumer (C), respectively. 1 indicates that the component is up, and 0 indicates that the component is down. Since we have assumed independence, it is easy to see that the steady state probabilities of all states of Figure 3 are as shown in Table 1.

Table 1. Steady state probabilities for the 8-state CTMC

State	Steady state probability
111	$A_S A_M A_C$
011	$(1-A_S)A_M A_C$
101	$A_S(1-A_M)A_C$
110	$A_S A_M(1-A_C)$
001	$(1-A_S)(1-A_M)A_C$
010	$(1-A_S)A_M(1-A_C)$
100	$A_S(1-A_M)(1-A_C)$
000	$(1-A_S)(1-A_M)(1-A_C)$

3.2 Performance model

We start with a simple performance model that does not consider deadline violations. Since the failures are rare, the queue within the messaging service can be approximated very well by its steady state queue length distribution. Let $\bar{N} = \sum_{i=0}^{k} i * p_i$ denote the expected number of messages in the queue at steady state, where p_i is the steady state probability that the queue has i messages. p_k is the probability of the queue being full [10]. For the performance models in this paper, we assume that message arrivals constitute a Poisson process with parameter λ and that the message consumption times are exponentially distributed with rate μ. Thus the queue is an $M/M/1/k$ queue, where k is the maximum number of messages that can be held in the PSQ.

We now proceed to develop the performability model. In Section 4, we combine the availability model of Section 3.1 with the simple $M/M/1/k$ queue performance model. In Sections 5 and 6, we will enhance the performability model to include the effect of deadline violations by using response time distributions.

4 Loss rates without considering deadline violations

We now derive the loss rate equations without considering deadline violations. Different applications can have different interpretations on what constitutes loss. The equations for the expected loss rates given below consider all possible contributions to message loss. For an application on hand and the interpretation of what constitutes a loss, the relevant terms can then be picked to compute the expected loss rate. Section 4.1, will illustrate this for an application that uses a reliable messaging configuration [7].

In practical situations the failure rates of the supplier, messaging service and consumer are very low in comparison with their repair rates, and especially in comparison with the message arrival and delivery rates. This observation allows us to ignore multiple failures since we can assume that a failed supplier, messaging service or consumer will be repaired before another failure occurs. Thus it is enough to consider only states 111, 011, 110, and 101 of Table 1. Considering all possible contributions to message loss, the expected loss rates can now be written down using state enumeration for the three quality of service settings:

- Best effort (BE)

$$\begin{aligned} LR_{BE} &= \text{loss rate when messaging svc. crashes} \\ &\quad + \text{loss rate when messaging svc. is down} \\ &\quad + \text{loss rate when cons.crashes} \\ &\quad + \text{loss rate when cons. is down} \\ &\quad + \text{loss rate due to discards} \\ &= A_S A_M A_C \gamma_{eq}^{(M)} \bar{N} + A_S(1-A_M)A_C\lambda \\ &\quad + A_S A_M A_C \gamma_{eq}^{(C)} \bar{N} + A_S A_M(1-A_C)\lambda \\ &\quad + A_S A_M A_C \lambda p_k . \end{aligned} \quad (2)$$

- Persistent connections (PC)

$$\begin{aligned} LR_{PC} &= \text{loss rate when messaging svc. crashes} \\ &\quad + \text{loss rate when messaging svc. is down} \\ &\quad + \text{loss rate when cons. is down} \\ &\quad + \text{loss rate due to discards} \\ &= A_S A_M A_C \gamma_{eq}^{(M)} \bar{N} + A_S(1-A_M)A_C\lambda \\ &\quad + A_S A_M(1-A_C)\lambda \\ &\quad \sum_{i=0}^{k} \left(\frac{\lambda}{\lambda+\tau_{eq}^{(C)}}\right)^{k-i} p_i + A_S A_M A_C \lambda p_k . \end{aligned} \quad (3)$$

- Persistent connections and messages (PCM)

$$\begin{aligned} LR_{PCM} &= \text{loss rate when messaging svc. is down} \\ &\quad + \text{loss rate when cons. is down} \\ &\quad + \text{loss rate due to discards} \\ &= A_S(1-A_M)A_C\lambda + A_S A_M(1-A_C)\lambda \\ &\quad \sum_{i=0}^{k} \left(\frac{\lambda}{\lambda+\tau_{eq}^{(C)}}\right)^{k-i} p_i + A_S A_M A_C \lambda p_k . \end{aligned} \quad (4)$$

In the above equations, we have considered that when the messaging service is down, since the supplier cannot pass on messages to the messaging service, the loss rate is λ. With the BE quality of service setting, the messaging service will not queue messages if the consumer is down. Hence the loss rate is again λ. However, if the consumer is down and the PC or PCM quality of service setting is used, the messaging service continues to queue messages for the consumer. In this case messages are lost only if the queue fills up before the consumer recovers. This happens with the probability $\left(\frac{\lambda}{\lambda+\tau_{eq}^{(C)}}\right)^{k-i}$, given that the queue had i messages when the consumer failed [2]. When the supplier, messaging service and consumer are all up, with rate $\gamma_{eq}^{(M)}$ the messaging service can fail, leading to the loss of the entire queue ($\bar{N}$ messages on average) when BE or PC quality of service setting is used. In addition, with the BE quality of service setting, all the messages in the queue are also lost when the consumer fails (at rate $\gamma_{eq}^{(C)}$). In all three quality of service settings, a message is discarded when an incoming message sees a full queue (with probability p_k). Note

Table 2. Summary of reward rates

State	Reward rate		
	BE	PC	PCM
111	λp_k	λp_k	λp_k
011	0	0	0
110	λ	$\lambda \sum_{i=0}^{k} \left(\frac{\lambda}{\lambda + \tau_{eq}^{(C)}} \right)^{k-i} \cdot p_i$	$\lambda \sum_{i=0}^{k} \left(\frac{\lambda}{\lambda + \tau_{eq}^{(C)}} \right)^{k-i} \cdot p_i$
101	λ	λ	λ

Table 3. Summary of impulse rewards

Transition	Impulse reward		
	BE	PC	PCM
$111 \rightarrow 101$	N	N	0
$111 \rightarrow 110$	N	0	0

that Equations 2, 3, and 4 can be obtained by considering the states 111, 011, 110, and 101 in Figure 3 as a Markov reward model [9] with reward rates and impulse rewards associated with failure transitions as summarized in Tables 2 and 3, respectively.

For illustration we take the failure and repair rates to be the same for the supplier, messaging service and the consumer. We consider that the availability model in Figure 2 is used, and assume that on the average processes fail once in 10 days, and nodes fail once in 20 days. Average process restart time is 1 minute and node repair time is 30 minutes. The probability $c = 0.99$. This gives $\gamma_{eq} = 1.7361e-06$, $\tau_{eq} = 0.0015$, and $A = 0.9988$.

Figure 4 shows the effect of varying the queue length k for different input rates, taking the delivery rate to the consumer, μ to be 430 messages/sec. In all plots the loss rates for PC and PCM quality of service settings are practically identical. This is because the relative contribution of the term $A_S A_M A_C \gamma_{eq}^{(M)} \bar{N}$ to the loss rate is negligible. With a low input rate ($\lambda = 50$), the probability of the queue being full at steady state is very small. Hence the main reason for loss in this case is when the consumer is down. With PC and PCM quality of service settings, as the queue size is increased the probability of the queue being able to absorb incoming messages when the consumer is unavailable increases. Hence the loss rates for PC and PCM are lower than for the BE quality of service setting (although not significantly). As the input rate is increased ($\lambda = 150$), the losses due to discards in a full queue start to dominate. Eventually for high input rates ($\lambda = 300, 550$) the losses are almost entirely due to discards when messages arrive at a full queue, and all three quality of service settings have the same loss rate. It is also interesting to note that for a given consumer delivery rate (430 messages/sec. in this case), when the loss rate due to discards when the queue is full dominate the overall loss rate, it is not useful to increase the queue size beyond a point (around 25 in this case). This is because the probability of a queue being full (p_k) does not drop appreciably as the queue length is increased beyond that point. This optimal queue size, of course, also depends on the distribution (and hence variance) of message inter-arrival and delivery times. We can extend the models in this paper to consider correlated arrivals using a Markov Modulated Poisson Process (MMPP), and service times with variances lower than those associated with exponential distributions using r-stage Erlang approximations.

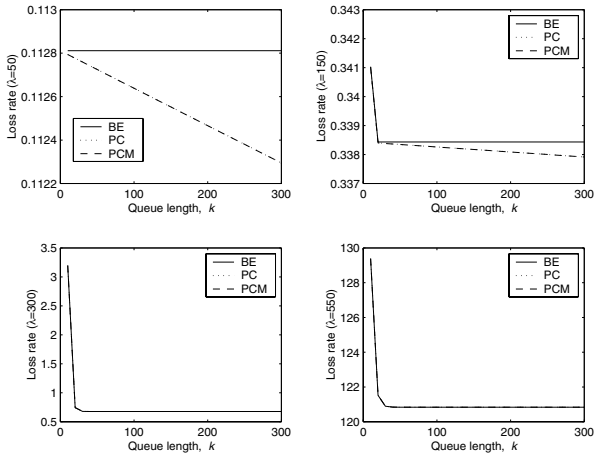

Figure 4. Effect of varying queue length k

4.1 Loss rates for the reliable messaging configuration

To illustrate how the loss rate equations can be used for different interpretations of loss, consider an application in which the supplier retries delivery of a message if it receives an exception indicating that the messaging service is unavailable, or that the queue inside the messaging service is full. For example, the CORBA Notification Service can be made to return the "IMP_LIMIT" exception when a supplier attempts to deliver to a full queue by setting the RejectNewEvents setting to "true".

With the above situation in mind, if the messages that the supplier retries are not counted toward the loss rate, the equations in Section 4 can be modified as follows:

- Best effort (BE)

$$\begin{aligned} LR_{BE}^{RM} &= \text{loss rate when messaging svc. crashes} \\ &\quad + \text{loss rate when cons. crashes} \\ &\quad + \text{loss rate when cons. is down} \\ &= A_S A_M A_C \gamma_{eq}^{(M)} \bar{N} + A_S A_M A_C \gamma_{eq}^{(C)} \bar{N} \\ &\quad + A_S A_M (1 - A_C) \lambda. \end{aligned} \quad (5)$$

- Persistent connections (PC)

$$\begin{aligned} LR_{PC}^{RM} &= \text{loss rate when messaging svc. crashes} \\ &= A_S A_M A_C \gamma_{eq}^{(M)} \bar{N}. \end{aligned} \quad (6)$$

- Persistent connections and messages (PCM)

$$LR^{RM}_{PCM} = 0. \quad (7)$$

The expected loss rate is insignificant for the PC setting in comparison with the BE setting. The losses for the BE setting are primarily due to the incoming messages that are discarded by the messaging service when the consumer is unavailable (the term $A_S A_M (1 - A_C)\lambda$ in Equation 5 dominates).

5 Losses due to deadline violations

When messages have deadlines associated with them, in addition to the losses accounted for in Section 4, losses due to deadline violations also need to be considered. The contributions to the overall loss rate from deadline violations may be classified as - (1) deadline violations at steady state when the supplier, messaging service, and consumer are up, and (2) deadline violations on failure of the supplier, messaging service, or consumer. Note that response time distributions, rather than mean response times, are required to compute the probability of deadline violations.

In order to evaluate the losses due to deadline violations, we need a performance model that solves for the probability of deadline violations under a particular delivery and discard policy configuration of the messaging service. For illustration, we will use the performance models presented in [8]. These models give the unconditional distributions of time to delivery (also called the response time distribution), $F_C(t)$, and time to discard, $F_D(t)$, for a message entering the PSQ, under the configured delivery and discard policy settings. As an example, Figure 5 shows the results from these models with delivery and discard policies set to FifoOrder. The distribution of the time to delivery or the

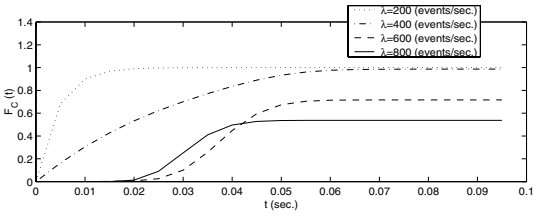

Figure 5. $F_C(t)$ **with** *OrderPolicy* **= FifoOrder,** *DiscardPolicy* **= FifoOrder**

response time distribution, $F_C(t)$ is a defective distribution with the defect at $t = \infty$ equal to the probability of discard, π_D. Also, $F_D(t) = 1 - F_C(t)$. In Figure 5, the delivery rate to the consumer was 430 messages/sec. Therefore, as the input rate λ is increased considerably beyond 430 events/sec., the number of discards happening at the front of the queue increases. Therefore, an incoming message moves toward the front of the queue quicker. Thus a message that does get delivered, gets delivered quicker, but there is a higher probability that it will get discarded (note that the curves for $\lambda = 600$ and 800 cross). We next show how the response time distributions can be used to obtain the loss rates due to deadline violations in messaging services.

5.1 Deadline violations at steady state with no failure

In this section, for notational simplicity, we shall use $F_R(t) = \frac{F_C(t)}{F_C(\infty)} = \frac{F_C(t)}{1-\pi_D}$ to denote the *conditional* response time distribution of a message given that it is successfully delivered (a non-defective distribution).

5.1.1 Delivery and discard without considering priority

When the delivery policy is not based on priority, the expected loss rate due to deadline violations (DV) at steady state can be written as:

$$LR_{DV} = A_S A_M A_C [1 - F_R(d)]\lambda(1 - p_k), \quad (8)$$

where $\lambda(1 - p_k) = \mu(1 - p_0)$ is the throughput at the consumer at steady state, and d is the deadline associated with each message. Also note that if new messages are rejected when the queue is full, $F_R(t) = F_C(t)$, since $F_C(\infty) = 1 - \pi_D = 1$. The response time distribution is non-defective in this case.

5.1.2 Delivery or discard based on priority

When delivery or discard is done based on priority, messages of different priorities have different throughputs as observed at the consumer and also have different response time distributions. Hence:

$$LR_{DV} = \begin{cases} A_S A_M A_C \sum_{i=1}^{P} [1 - F_{R_i}(d)]\lambda p_i^{(pr)}(1 - p_k), \\ \text{if new messages rejected when queue is full,} \\ A_S A_M A_C \sum_{i=1}^{P} [1 - F_{R_i}(d)]\lambda p_i^{(pr)} F_{C_i}(\infty), \\ \text{if discards within queue are allowed,} \end{cases} \quad (9)$$

where $F_{R_i}(t) = \frac{F_{C_i}(t)}{F_{C_i}(\infty)}$ is the *conditional* response time distribution of a successfully delivered message of priority i, $F_{C_i}(t)$ is the response time distribution for a message of priority i, $p_i^{(pr)}$ is the probability that an incoming message is of priority i, and $F_{C_i}(\infty)$ is the probability that a message of priority i is eventually delivered (does not get discarded). Note that when discards are allowed from within the queue, an incoming message is always accommodated in the queue (by discarding a message in the queue if the queue is full). Once in the queue, this new message could suffer a discard in accordance with the discard policy, before it is delivered

to the consumer. Again, if new messages are rejected when the queue is full, $F_{R_i}(t) = F_{C_i}(t)$.

5.2 Deadline violations on failure of supplier, consumer, or messaging service

We now consider the losses due to deadline violations in the presence of failures, with each of the three quality of service settings.

5.2.1 Delivery and discard without considering priority

Best effort (BE)
When the supplier goes down, the messaging service will still be able to deliver the messages in the queue to the consumer. These messages ($\bar{N}$ on average) would have a different response time distribution if, for example, delivery is done on priority order or discard is done on priority or FIFO order. This is because now there are no new arrivals into the queue. Denoting the *conditional* response time distribution of a successfully delivered message under this scenario by $F_{R,sf}(t)$, the expected loss rate due to deadline violations of these messages can be written as:

$$\begin{aligned} LR_{DV,sf} &= A_S A_M A_C \gamma_{eq}^{(S)} [1 - F_{R,sf}(d)] \\ & \quad min\left(\bar{N}, \frac{\mu}{\tau_{eq}^{(S)}}\right), \end{aligned} \tag{10}$$

where $min\left(\bar{N}, \frac{\mu}{\tau_{eq}^{(S)}}\right)$ is the expected number of messages delivered before the supplier is restarted.
Persistent connections (PC)
When connections are persistent, in addition to $LR_{DV,sf}$ of Equation 10, the loss rate (from deadline violations) of messages due to consumer failure needs to be considered. When the consumer is down, the queue starts to fill up and discards can occur in accordance with the discard policy in effect. When the consumer is restarted, the messages that are in the queue will have a different *conditional* response time distribution, $F_{R,cf}(t)$. The expected loss rate due to deadline violations on consumer failure with the PC setting can be written as:

$$\begin{aligned} LR_{DV,cf} &= A_S A_M A_C \gamma_{eq}^{(C)} [1 - F_{R,cf}(d)] \\ & \quad min\left(\bar{N} + \frac{\lambda}{\tau_{eq}^{(C)}}, k\right), \end{aligned} \tag{11}$$

where $min\left(\bar{N} + \frac{\lambda}{\tau_{eq}^{(C)}}, k\right)$ is the expected number of messages in the queue when the consumer is restarted (k is the maximum queue size). $\frac{\lambda}{\tau_{eq}^{(C)}}$ is the expected number of messages that arrive when the consumer is down.

Persistent connections and messages (PCM)
With PCM quality of service setting also, the terms $LR_{DV,sf}$ and $LR_{DV,cf}$ need to be considered. In addition, the messages that are in the queue at the instant the messaging service crashes suffer a delay equal to the repair time of the messaging service. These messages have a higher response time and hence have a greater likelihood of violating their deadline. Denoting the *conditional* response time distribution for these messages by $F_{R,msf}(t)$, the loss rate due to deadline violations on messaging service failure with the PCM setting can be written as:

$$LR_{DV,msf} = A_S A_M A_C \gamma_{eq}^{(M)} [1 - F_{R,msf}(d)] \bar{N}. \tag{12}$$

Note that the distributions $F_{R,sf}(t)$, $F_{R,cf}(t)$, and $F_{R,msf}(t)$ account for the recovery time distributions in addition to the (altered) queuing delays. They should also reflect the effect of the discard policy in effect. For example, $F_{R,cf}(t)$ will not be the same if discards inside the queue are done in FIFO order and LIFO order.

5.2.2 Delivery or discard based on priority

The loss rates for this case will be of the same magnitude as those without considering priority. In the next section we first discuss the magnitude of these loss rates before deciding if they are significant enough to compute.

6 Overall loss rates with deadline violations

Considering that messages have deadlines, the overall expected loss rate can now be written as:

$$\begin{aligned} LR_{BE,deadline} &= LR_{BE} + LR_{DV} + LR_{DV,sf} \\ LR_{PC,deadline} &= LR_{PC} + LR_{DV} + LR_{DV,sf} \\ & \quad + LR_{DV,cf} \\ LR_{PCM,deadline} &= LR_{PCM} + LR_{DV} + LR_{DV,sf} \\ & \quad + LR_{DV,cf} + LR_{DV,msf}. \end{aligned} \tag{13}$$

Although one can derive the $F_{R,sf}(t)$, $F_{R,cf}(t)$, and $F_{R,msf}(t)$ distributions exactly, the contribution of the loss rate due to deadline violations in the presence of failures will be insignificant in most cases. Therefore, we can consider the following approximations:

Scenario I

If the loss rates when the messaging service is down, consumer is down, or due to discards (see Section 4) are considered while evaluating the loss rate for an application, then, as seen in Figure 4(a), the contribution of terms such as $A_S A_M A_C \gamma_{eq}^{(M)} \bar{N}$ to the overall expected loss rate is insignificant. When we consider deadline violations on

failures, the terms $LR_{DV,sf}$, $LR_{DV,cf}$, and $LR_{DV,msf}$ (of Section 5.2) are again insignificant. Thus we can use the following approximations for the overall expected loss rates:

$$LR_{BE|PC|PCM,deadline} \approx LR_{BE|PC|PCM} + LR_{DV} \quad (14)$$

Scenario II

Consider that the loss rates when the messaging service is down, consumer is down, and due to discards are not considered (see, for example, the interpretation in Section 4.1). In this case, from Section 4.1 we can conclude that the contribution of $LR_{DV,sf}$ to the expected loss rate for the BE quality of service setting will be insignificant (since $LR_{DV,sf}$ is comparable to $A_S A_M A_C \gamma_{eq}^{(M)} \bar{N}$). However, with PC and PCM settings, the expected loss rates due to deadline violations on failure, given by $LR_{DV,sf}, LR_{DV,cf}$, and $LR_{DV,msf}$ are comparable to LR_{PC} and LR_{PCM}, respectively (LR_{PC}^{RM} and LR_{PCM}^{RM} in Section 4.1). We can however consider the following cases:

Case 1

One can argue that if the deadlines are small enough to make the violations sensitive to queuing delays, the expected loss rates due to deadline violations will be dominated by the term LR_{DV}. Also, with such tight deadlines, one is more likely to be interested in the steady state loss rate when all components (supplier, messaging service, and consumer) are working. Hence for the application in Section 4.1 we can write:

$$\begin{aligned} LR_{BE|PC,deadline}^{RM} &\approx LR_{BE|PC}^{RM} + LR_{DV} \\ LR_{PCM,deadline}^{RM} &\approx LR_{PCM}^{RM} + LR_{DV} = LR_{DV} \end{aligned} \quad (15)$$

Case 2

Another scenario is that the deadlines are higher than the queuing delays under failure-free operation, so that LR_{DV} is negligible. Then we are more concerned with studying if the component that failed (consumer or messaging service) can recover fast enough so that the messages can meet their deadlines. Now, the expected loss rate due to deadline violations is sensitive to the repair time distributions of the consumer and the messaging service. In this case we can write:

$$\begin{aligned} LR_{BE,deadline}^{RM} &\approx LR_{BE}^{RM} + LR_{DV,sf} \approx LR_{BE}^{RM} \\ LR_{PC,deadline}^{RM} &\approx LR_{PC}^{RM} + LR_{DV,cf} \\ LR_{PCM,deadline}^{RM} &\approx LR_{PCM}^{RM} + LR_{DV,cf} + LR_{DV,msf} \\ &= LR_{DV,cf} + LR_{DV,msf}. \end{aligned} \quad (16)$$

We can ignore $LR_{DV,sf}$ above, because with a deadline that is greater than the queuing delays, $LR_{DV,sf} \approx 0$.

$LR_{DV,cf}$ and $LR_{DV,msf}$ can be derived as follows:

$$LR_{DV,cf} = \begin{cases} A_S A_M A_C \gamma_{eq}^{(C)} [1 - F_{R \otimes C_r}(d)] \\ min\left(\bar{N} + \frac{\lambda}{\tau_{eq}^{(C)}}, k\right), \text{no priority} \\ A_S A_M A_C \gamma_{eq}^{(C)} \\ \sum_{i=1}^{P} [1 - F_{R_i \otimes C_r}(d)] p_i^{(pr)} \\ min\left(\bar{N} + \frac{\lambda}{\tau_{eq}^{(C)}}, k\right), \\ \text{with priority} \end{cases} \quad (17)$$

$$LR_{DV,msf} = \begin{cases} A_S A_M A_C \gamma_{eq}^{(M)} [1 - F_{R \otimes M_r}(d)] \bar{N}, \\ \text{no priority} \\ A_S A_M A_C \gamma_{eq}^{(M)} \\ \sum_{i=1}^{P} [1 - F_{R_i \otimes M_r}(d)] p_i^{(pr)} \bar{N}, \\ \text{with priority} \end{cases} \quad (18)$$

where $F_{R \otimes C_r}(t) = F_R(t) \otimes F_{C_r}(t)$ is the distribution resulting from the convolution of the density function of the *conditional* response time given successful delivery $\left(\frac{dF_R(t)}{dt}\right)$ with the density function of the repair time of the consumer $\left(\frac{dF_{C_r}(t)}{dt}\right)$, and $F_{R \otimes M_r}(t) = F_R(t) \otimes F_{M_r}(t)$ is the distribution resulting from the convolution of the density function of the *conditional* response time given successful delivery $\left(\frac{dF_R(t)}{dt}\right)$ with the density of the repair time of the messaging service $\left(\frac{dF_{M_r}(t)}{dt}\right)$. Similarly, when delivery or discard is based on priority, we consider the convolutions for each priority class i and uncondition on the messages being of priority i. Note that Equation 17 assumes that the $\frac{\lambda}{\tau_{eq}^{(C)}}$ messages that arrive during the time that the consumer is down see the same recovery time. Although Equation 17 is an approximation in this sense, it is a good one when the input rate λ is much higher than the consumer repair rate. At worst, it is a pessimistic estimate if λ and the consumer repair rate are comparable, since it says that all the $\frac{\lambda}{\tau_{eq}^{(C)}}$ messages can violate the deadline with the same probability, while in reality, only a fraction of these messages might have violated the deadline.

6.1 Numerical illustration

This section illustrates the effect of deadline violations on overall expected loss rates.

Scenario I

Let us first consider that losses when the messaging service is down, consumer is down, and losses due to discards are considered (Scenario I of Section 6). Then, to estimate the

overall expected loss rate including deadline violations, we need to use Equation 14. Figure 6 shows the contributions to the overall expected loss rate as a function of the deadline d, when delivery and discard order are FifoOrder, the input rate is $\lambda = 400$ messages/sec., and the queue size is 25.

In Figure 6, the loss rates for the "best effort" (BE) quality of service setting are shown. The LR_{BE} plot is obtained using Equation 2. LR_{BE} is, of course, independent of deadline d. $F_R(d)$ is obtained by dividing $F_C(d)$ (from Figure 5(b)) by $F_C(\infty) = 1 - \pi_D$, the probability of successful delivery without discard. Recall that $F_C(t)$ is the response time distribution (defective) and $F_R(t)$ is the *conditional* response time distribution for successfully delivered messages, for the particular delivery and discard policies in effect (both FifoOrder in this illustration). LR_{DV} is obtained from Equation 8 and is the contribution of deadline violations to the overall expected loss rate. The overall expected loss rate including deadline violations for this case is given by $LR_{BE,deadline}$ (Equation 14). In this illustration, the overall loss rate is dominated by deadline violations for deadlines less than 0.06 sec. For higher deadlines, deadline violations are greatly reduced and the losses are mainly due to discards occurring when the queue is full.

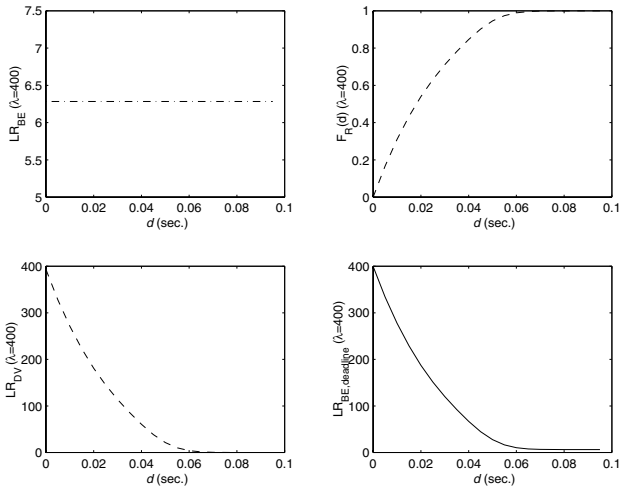

Figure 6. Expected loss rates with *OrderPolicy*=**FifoOrder,** *DiscardPolicy*=**FifoOrder;** $k = 25, \mu = 430$ **messages/sec.**

When delivery or discard is done based on message priorities, Equation 9 should be used to estimate LR_{DV}, to account for the different response time distributions and successful delivery probabilities for the different priority classes.

Scenario II

If the losses when the messaging service is down, consumer is down and those due to discards are not considered (Scenario II of Section 6), as discussed earlier, we can consider two cases. But since the procedure to obtain results for Equation 15 is similar to that described for Scenario I above, only evaluation of Equation 16 is considered now. The methods for evaluating Equation 17 and Equation 18 are similar. We will now specifically illustrate how the convolution term in Equation 18 can be evaluated.

Derivation of $F_{R \otimes M_r}(t)$ for the reliable messaging configuration:

For simplicity of illustration, let us consider that neither delivery nor discard is based on priorities. With the reliable messaging configuration, there are no discards from inside the queue. An incoming message will not be admitted into the queue if it is full (probability p_k, where k is the queue size). If a new incoming message is admitted into the queue and finds j messages in front of it, its *conditional* response time distribution given successful delivery (also the response time distribution in this case) is given by a $(j+1)$–stage Erlang distribution [10]. Therefore, conditioning on the fact that the message is admitted into the queue,

$$F_R(t) = \frac{\sum_{j=0}^{k-1} p_j \left[1 - \sum_{i=0}^{j} \frac{(\mu t)^i e^{-\mu t}}{i!}\right]}{1 - p_k}.$$

Next, the repair time distribution, $F_{M_r(t)}$ of the messaging service needs to be derived. If the availability model for the messaging service is given by Figure 2(a), the repair time distribution can be written down as follows (with the superscript (M) for the messaging service):

$$\begin{aligned} F_{M_r}(t) \quad = \quad & \frac{\gamma_n^{(M)}}{\gamma_n^{(M)} + \gamma_p^{(M)}}(1 - e^{-\tau_n^{(M)} t}) + \frac{\gamma_p^{(M)}}{\gamma_n^{(M)} + \gamma_p^{(M)}} \\ & \left[(1 - e^{-\tau_p^{(M)} t})c + \left(1 - \frac{\tau_n^{(M)}}{\tau_n^{(M)} - \tau_p^{(M)}}\right.\right. \\ & \left.\left. e^{-\tau_p^{(M)} t} + \frac{\tau_p^{(M)}}{\tau_n^{(M)} - \tau_p^{(M)}} e^{-\tau_n^{(M)} t}\right)(1 - c)\right]. \end{aligned}$$

7 Applying the framework to other scenarios

The use of a hierarchical approach makes the framework presented in this paper flexible. As an example, the architecture for reliable messaging using the CORBA Notification Service described in [7], includes a "ping"daemon that monitors the Notification Service and restarts it if it has crashed. When such a daemon is involved, the polling interval is a design parameter as well when optimizing for high availability and minimizing deadline violations. For

this case, the availability model in Figure 7 [2] (where it was described as the availability model for a server with *cold replication*) can be used. After failure of the messag-

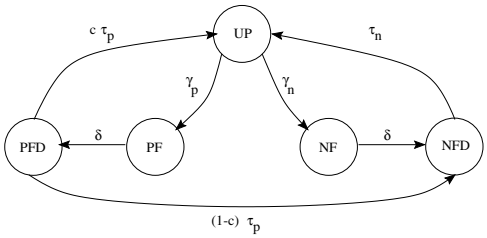

Figure 7. Availability model with polling for automatic crash detection

ing service due to process or node crash, the failure needs to be detected by a "ping" daemon (states PFD and NFD), before repair can be initiated. When cold replication is used, in addition to automatic detection of crashes, a spare node is also available. So the repair rate on node failure, τ_n, is higher. This is because the messaging service can be failed over to the spare node without waiting for time-consuming repairs of the primary node. If the polling interval is T, the rate δ can be approximated as $\delta = 2/T$ [2]. The new availability A_M, $\gamma_{eq}^{(M)}$, $\tau_{eq}^{(M)}$ and the repair time distribution $F_{M_r}(t)$ of the messaging service can now be computed and used in the hierarchical framework developed in this paper, without necessitating any changes to the models of the supplier or consumer, or to the loss rate equations.

8 Summary and concluding remarks

In this paper, we considered the expected loss rates associated with messaging services as a performability measure and derived closed-form expressions for these loss rate under different quality of service settings. We provided a hierarchical framework while formulating the performability model so that details such as application software and node failures can be incorporated into the underlying availability models without affecting the upper level closed-form loss rate equations. The quantitative framework provided in this paper thus make design decisions and tradeoffs possible, thereby aiding the engineering of complex, messaging service-oriented distributed systems.

References

[1] G. Ciardo, J. K. Muppala, and K. S. Trivedi. Analyzing concurrent and fault-tolerant software using stochastic reward nets. *Journal of Parallel and Distributed Computing*, 15:255–269, 1992.

[2] S. Garg, Y. Huang, C. M. R. Kintala, K. S. Trivedi, and S. Yajnik. Performance and reliability evaluation of passive replication schemes in application level fault tolerance. In *Proc. of the 29th International Symposium on Fault-Tolerant Computing (FTCS '99)*, pages 322–329, Madison, WI, USA, June 15-18 1999.

[3] O. C. Ibe, H. C. Choi, and K. S. Trivedi. Performance evaluation of client-server systems. *IEEE Transactions on Parallel and Distributed Systems*, 4(11):1217–1229, November 1993.

[4] J. F. Meyer. On evaluating the performability of degradable computer systems. *IEEE Transactions on Computers*, 29(8):720–731, 1980.

[5] L. E. Moser, P. M. Melliar-Smith, and P. Narasimhan. A fault tolerance framework for CORBA. In *Proc. of International Symposium on Fault-Tolerant Computing (FTCS-29)*, pages 150–157, Madison, Wisconsin, June 15-18 1999.

[6] A. Puliafito, S. Riccobene, and M. Scarpa. Modelling of client-server systems. In *Proc. of the Third International Workshop on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS '95)*, pages 340–344, Durham, NC, USA, January 18-20 1995.

[7] S. Ramani, B. Dasarathy, and K. S. Trivedi. Reliable messaging using the CORBA Notification service. In *Proc. of 3rd International Symposium on Distributed Objects and Applications (DOA '01)*, pages 229–238, Rome, Italy, September 18-20 2001.

[8] S. Ramani, K. S. Trivedi, and B. Dasarathy. Performance analysis of the CORBA Notification service. In *Proc. of 20th IEEE Symposium on Reliable Distributed Systems (SRDS '01)*, pages 227–236, New Orleans, USA, October 28-31 2001.

[9] R. M. Smith and K. S. Trivedi. The analysis of computer systems using markov reward processes. *Stochastic Analysis of Computer and Communication Systems, H. Takagi (ed.)*, pages 589–629, 1990. Elsevier Science Publishers B.V. (North-Holland).

[10] K. S. Trivedi. *Probability & Statistics with Reliability, Queuing and Computer Science Applications*. John Wiley & Sons, New York, second edition, 2001.

Acknowledgements

CORBA is a registered trademark of the Object Management Group, Inc. in the United States and other countries.

Java is a trademark of Sun Microsystems, Inc. in the United States and other countries.