In compliance with the
Canadian Privacy Legislation
some supporting forms
may have been removed from
this dissertation.

While these forms may be included
in the document page count,
their removal does not represent
any loss of content from the dissertation.

High Performance Middleware for Mobile Wireless Publish/Subscribe Systems

by
Umar Farooq, B.Sc. (EE)

A thesis submitted to the Faculty of Graduate Studies and Research in partial
fulfillment of the requirements for the degree of

Master of Applied Science

Ottawa-Carleton Institute for Electrical Engineering
Faculty of Engineering
Department of Systems and Computer Engineering
Carleton University
Ottawa, Ontario, Canada, K1S 5B6
August 19, 2003

© Copyright 2003, Umar Farooq

National Library
of Canada

Bibliothèque nationale
du Canada

Acquisitions and
Bibliographic Services

Acquisitions et
services bibliographiques

395 Wellington Street
Ottawa ON K1A 0N4
Canada

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence

ISBN: 0-612-88872-X

Our file Notre référence

ISBN: 0-612-88872-X

The author has granted a non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of this thesis in microform, paper or electronic formats.

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de cette thèse sous la forme de microfiche/film, de reproduction sur papier ou sur format électronique.

The author retains ownership of the copyright in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

L'auteur conserve la propriété du droit d'auteur qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

Canada

Abstract

Publish/subscribe middlewares are becoming popular for distributed applications because of their flexible and scalable nature. Anonymous and loosely-coupled communication between publisher and subscriber, along with the inherently asynchronous nature of these systems, help them adapt quickly to changing environments, making them a good choice for mobile networks. Although there has been a lot of work on publish/subscribe systems in the context of fixed networks, there is very little research on the behavior of these systems in the context of mobile networks.

This thesis investigates the extension of publish/subscribe middleware to mobile cellular networks with the focus on their performance in such networks. As a first step, the thesis characterizes a popular implementation of publish/subscribe system for the mobile domain, studying and analyzing the effect of different mobility parameters, which to the best of our knowledge is the first experimental research on the performance behavior of publish/subscribe systems in a mobile wireless domain. As a second step the thesis presents the design and evaluation of middleware level handoffs, a well known solution to extend publish/subscribe systems to a mobile domain, and identifies the performance concerns of such extensions. The results show that such handoff protocols involving two brokers are impractical from a performance perspective under highly dynamic and unreliable mobile wireless settings. As a final step the thesis introduces a novel technique called *semi-durable subscriptions* to overcome the challenges faced by current implementations and traditional solutions in a mobile wireless environment and through rigorous experimentation prove its quality and efficacy in such settings. The thesis presents the design and implementation of its prototype and its evaluation in different mobility environments. Through a comparison of the performance of semi-durable subscriptions technique with other alternatives the thesis provides an experimental validation that it leads to significant benefit in performance. Semi-durable subscriptions provide a rich system parameter space and the thesis demonstrates how it can be controlled to configure a system according to the desired characteristics and/or the needs of the environment. The thesis concludes with a discussion as to how the concept of semi-durable subscriptions can be extended to deal with regionalism.

Acknowledgements

First, I would like to express my sincere thanks and gratitude to my advisors Prof. Shikharesh Majumdar and Dr. Eric Parsons for their continuous support, encouragement and help throughout the thesis. I really appreciate their invaluable guidance and time during each phase of the thesis.

I would also like to express my deep appreciation and thanks to my parents, brother and sister for their patience and strong support. I am thankful to my friends in RADS Lab especially Ming, Toqeer, Jing and Yingjing whose friendship helped during the tough and to some extent monotonous periods of long and rigorous experimentation.

Table of Contents

Chapter 1

Introduction	1
1.1 Background	1
1.2 Motivation for the Thesis	3
1.3 Goals of the Thesis	3
1.4 Contributions of the Thesis	4
1.5 Thesis Outline	5

Chapter 2

Publish/Subscribe System Overview	6
2.1 Publish/Subscribe System	6
2.1.1 Components	6
2.1.2 Communication Paradigm	7
2.1.3 Subscription Types	8
2.1.3.1 Subject Based Subscriptions	8
2.1.3.2 Content Based Subscriptions	8
2.1.4 Architectures	10
2.1.4.1 Centralized Architecture	10
2.1.4.2 Distributed Architecture	11
2.1.5 Design Challenges	11
2.1.5.1 Broker Interconnection Topology	11
2.1.5.2 Routing Algorithm and Processing Strategy	14
2.1.5.3 QoS	14
2.2 Survey of Publish/Subscribe Systems	15
2.2.1 CORBA Notification Service	15
2.2.2 Elvin	16
2.2.3 Herald	16
2.2.4 Hermes	16
2.2.5 Java Message Service	16
2.2.6 JEDI	17
2.2.7 SIENA	17
2.2.8 STEAM	17
2.3 Overview of JMS	19
2.3.1 JMS Architecture	19
2.3.2 JMS Programming Model	20
2.3.3 Types of Subscriptions	22
2.3.3.1 Nondurable Subscriptions	22
2.3.3.2 Durable Subscriptions	23
2.3.4 Filtering with Message Selectors	23
2.3.5 Message Consumption Models in JMS	23

2.3.6 QoS Features	24
2.3.7 JMS Implementations	25
2.4 Publish/Subscribe in a Mobile Environment	26
2.4.1 Suitability	26
2.4.2 Publish/Subscribe Extension for Mobile Domain	26
2.4.3 Publish/Subscribe in Mobile Cellular Networks	28
2.4.4 Publish/Subscribe in Mobile Ad Hoc Networks	29
2.4.5 Challenges and Performance Issues for Publish/Subscribe in Mobile Wireless Environments	31

Chapter 3

Performance Measurement and Instrumentation	33
3.1 Experimental Setup	33
3.1.1 Basic Setup	33
3.1.2 Publisher	34
3.1.2.1 System Models and Publishing Rate	35
3.1.3 Subscribers	37
3.1.3.1 Subscription Pattern	37
3.1.3.2 Mobility Model	37
3.1.3.3 Number of Subscribers	38
3.1.4 Message Filters	39
3.1.5 Wireless Channels	39
3.2 Performance Measures	40
3.2.1 Publisher's Throughput	40
3.2.2 Subscribers' Throughput	40
3.2.3 Percentage Message Loss	41
3.2.4 Traffic on the Wireless Channels	41
3.3 Parameters	42
3.3.1 Parameters Independent of Mobile Wireless Environments	42
3.3.1.1 Number of Publishers/Publishing Rate	42
3.3.1.2 Number of Subscribers	43
3.3.1.3 Number of Topics/Selector Ranges	43
3.3.1.4 Subscription Type	44
3.3.1.5 Message Size	44
3.3.2 Mobile Wireless Environment Dependent Parameters	44
3.3.2.1 Bandwidth	44
3.3.2.2 Number of Connection/Disconnection Requests per Second	45
3.3.2.3 Mean of Signal Down Interval	46
3.3.2.4 Co-efficient of Variation of Signal down Intervals	46
3.3.2.5 Total Time in Signal Down State	47
3.4 Test Conditions	47

Chapter 4	
Characterization of JMS in Mobile Wireless	
Environments	50
4.1 Objectives	50
4.2 Performance of Nondurable Subscriptions	51
4.2.1 Effect of Bandwidth and Mobility	51
4.2.1.1 Conclusions	56
4.2.2 Effect of Frequency of Handoffs	56
4.2.2.1 Conclusions	61
4.2.3 Effect of Variability in Signal Down Intervals	62
4.2.3.1 Conclusions	65
4.2.4 Effect of Message Size	66
4.2.4.1 Conclusions	69
4.3 Performance of Durable Subscriptions	70
4.3.1 Challenges for Durable Subscription in Mobile Environments	70
4.3.1.1 Message Loss	71
4.3.1.2 Overheads of Inactive Subscriptions	71
4.3.1.3 Duplicate Messages	72
4.3.2 Extending Durable Subscriptions for Mobile Environments	72
4.3.2.1 Clustering	73
4.3.2.2 Mobile IP	74
4.3.2.3 Middleware Level Handoffs	75

Chapter 5	
Middleware Level Handoffs for Durable Subscriptions:	
Design Implementation and Evaluation	76
5.1 Objectives	76
5.2 Design and Implementation of Middleware Level Handoffs	77
5.2.1 Motivation	77
5.2.2 Design Issues	77
5.2.3 Design and Implementation	78
5.3 Evaluation of Middleware Level Handoffs	80
5.3.1 Effect of Mobility	80
5.3.1.1 Conclusions	83
5.3.2 Comparison with Nondurable Subscriptions	83
5.3.3 Effect of Mobility Parameters	85
5.3.4 Race Conditions in Durable Subscriptions with Middleware Level Handoffs	86

Chapter 6	
Semi-Durable Subscriptions: A Technique to Achieve High Performance in Mobile Wireless Publish/Subscribe Systems	87
6.1 Objectives	87
6.2 Introduction to Semi-Durable Subscriptions	88
6.2.1 Motivation	88
6.2.2 Challenges for the Proposed Technique	88
6.2.3 Solution to the Design Challenges: Semi-Durable Subscriptions	90
6.2.3.1 Review of the Challenges and the Respective Solutions	92
6.3 Design and Implementation of Semi-Durable Subscriptions	93
6.3.1 Design Components	94
6.3.1.1 Broker	94
6.3.1.2 Semi-Durable Subscriptions Manager	94
6.3.1.3 Buffer Subscriptions Manager	94
6.3.1.4 Semi-Durable Buffer Manager	95
6.3.1.5 Handoff Manager	95
6.3.1.6 Semi-Durable Buffer	95
6.3.2 Operation of Semi-Durable Subscriptions	96
6.3.3 Implementation Challenges and Solutions	100
6.3.3.1 Buffer Consistency	100
6.3.3.2 Searching Algorithm	100
6.3.4 Testing	101
6.4 System Parameters for Semi-Durable Subscriptions	102
6.4.1 Buffer Size	102
6.4.2 Internal Organization of the Buffer	102
6.4.3 Message Overwriting Policies	103
6.4.3.1 Least Recently Published Overwriting Policy	103
6.4.3.2 Invalid Messages Overwriting Policy	103
6.4.3.3 Policies with Priorities	104
6.4.3.4 Least Information Loss Overwriting Policy	104
6.5 Performance Comparison of Semi-Durable Subscriptions with Nondurable and Durable Subscriptions	105
6.5.1 Maximum Sustainable Throughput in the Server Bottlenecked Region	105
6.5.2 Maximum Sustainable Throughput in the Bandwidth Bottlenecked Region	108
6.5.3 Comparison at given Publishing Rate	111
6.6 Effect of Input Parameters on the Performance of Semi-Durable Subscriptions	113
6.6.1 Effect of Frequency of Handoffs	113
6.6.1.1 Conclusions	116
6.6.2 Effect of Variability in Signal Down Intervals	117
6.6.2.1 Conclusions	119
6.6.3 Effect of Message Size	119

6.6.3.1 Conclusions	122
6.7 Effect of Buffer Size on the Performance of Semi-Durable Subscriptions	122
6.7.1 Effect of Publishing Rate on Message Loss	123
6.7.2 Tradeoff between System Throughput and Message Loss	126
6.7.3 Conclusions	130
6.8 Extending Semi-Durable Subscriptions to Deal with Regionalism	131
 Chapter 7	
Conclusions	133
7.1 Summary	133
7.2 Future Research	136
 References	139
 Appendix A	145
A.1 Effect of Bandwidth and Mobility on Nondurable Subscriptions	145
A.2 Effect of Mobility on Durable Subscriptions	146
A.3 Maximum Sustainable Throughput of the Three Subscription Types in the Server Bottlenecked Region	147
A.4 Maximum Sustainable Throughput of Nondurable and Semi-Durable Subscriptions in the Bandwidth Bottlenecked Region	149
A.5 Performance of Nondurable and Semi-Durable Subscriptions at a Given Publishing Rate in the Bandwidth Bottlenecked Region	150

List of Figures

Figure 2.1:	Hierarchical Topology of Brokers	12
Figure 2.2:	Peer-to-Peer Topology of Brokers	13
Figure 2.3:	JMS API Architecture	20
Figure 2.4:	JMS Programming Model	21
Figure 2.5:	Publish/Subscribe in Mobile Cellular Networks	29
Figure 3.1:	Logical View of Experimental Setup	34
Figure 3.2:	Schematics of the Open Model Publisher	36
Figure 3.3:	State Diagram of Subscribers' Mobility Model	38
Figure 4.1:	Effect of Bandwidth and Mobility on Nondurable Subscriptions	53
Figure 4.2:	Effect of Frequency of Handoffs on Nondurable Subscriptions	60
Figure 4.3:	Effect of C_V of Signal Down Intervals on Nondurable Subscriptions	64
Figure 4.4:	Effect of Message Size on Nondurable Subscriptions	68
Figure 5.1:	Middleware Level Handoffs for Durable Subscriptions	78
Figure 5.2:	Effect of Mobility on Durable Subscriptions	81
Figure 5.3:	Comparison of Nondurable and Durable Subscriptions	84
Figure 6.1:	Components of Semi-durable Subscriptions	96
Figure 6.2:	Handoff in Semi-durable Subscriptions	96
Figure 6.3:	Operation of Semi-durable Subscriptions: Collaboration Diagram	97
Figure 6.4:	Maximum Sustainable Throughput of the Three Subscription Types in the Server Bottlenecked Region	107
Figure 6.5:	Maximum Sustainable Throughput of Nondurable and Semi-durable Subscriptions in the Bandwidth Bottlenecked Region	110
Figure 6.6:	Performance of Three Subscription Types at given Publishing Rate in the Region where Bandwidth is Not the Bottleneck	112
Figure 6.7:	Effect of Frequency of Handoffs on Semi-durable Subscriptions	116
Figure 6.8:	Effect of C_V of Signal Down Intervals on Semi-durable Subscriptions	118

Figure 6.9:	Effect of Message Size on Semi-durable Subscriptions	121
Figure 6.10:	Effect of Publishing Rate on Message Loss for Different Buffer Sizes ($t_{SD} = 12$ seconds)	124
Figure 6.11:	Effect of Publishing Rate on Message Loss for Different Buffer Sizes ($t_{SD} = 24$ seconds)	125
Figure 6.12:	Tradeoff between Throughput and Message Loss ($t_{SD} = 12$ seconds)	128
Figure 6.13:	Tradeoff between Throughput and Message Loss ($t_{SD} = 24$ seconds)	129
Figure A.1:	Effect of Mobility and Bandwidth on Nondurable Subscriptions	146
Figure A.2:	Effect of Mobility on Durable Subscriptions	147
Figure A.3:	Maximum Sustainable Throughput of the Three Subscription Types in the Server Bottlenecked Region	148
Figure A.4:	Maximum Sustainable Throughput of the Nondurable and Semi-durable Subscriptions in the Bandwidth Bottlenecked Region	150
Figure A.5:	Performance of Nondurable and Semi-Durable Subscriptions at a given Publishing Rate in the Bandwidth Bottlenecked Region	151

List of Tables

Table 2.1:	Survey of Publish/Subscribe Systems	18
Table 3.1:	Summary of Workload Parameters	48
Table 4.1:	Input Parameters for Investigating the Effect of Bandwidth and Mobility on Nondurable Subscriptions	52
Table 4.2:	Input Parameters for Investigating the Effect of Frequency of Handoffs on Nondurable Subscriptions	58
Table 4.3:	Effect of Frequency of Handoffs on Nondurable Subscriptions	59
Table 4.4:	Input Parameters for Investigating the Effect of C_V of Signal Down Intervals on Nondurable Subscriptions	63
Table 4.5:	Effect of C_V of Signal Down Intervals on Nondurable Subscriptions	63
Table 4.6:	Input Parameters for Investigating the Effect of Message Size on Nondurable Subscriptions	66
Table 4.7:	Effect of Message Size on Nondurable Subscriptions	66
Table 5.1:	Input Parameters for Investigating the Effect of Mobility on Durable Subscriptions	80
Table 6.1:	Input Parameters for Comparing the Performance of Semi-durable Subscriptions with that of Nondurable and Durable Subscriptions	105
Table 6.2:	Values of Input Parameters for Evaluation of Semi-durable Subscriptions	113
Table 6.3:	Effect of Frequency of Handoffs on Semi-durable Subscriptions	114
Table 6.4:	Effect of C_V of Signal Down Intervals on Semi-durable Subscriptions	117
Table 6.5:	Effect of Message Size on Semi-durable Subscriptions	120
Table 6.6:	Input Parameters for Investigating the Effect of Buffer Size on Semi-durable Subscriptions	122
Table 6.7:	Effect of Publishing Rate on Message Loss for Different Buffer Sizes ($t_{SD} = 12$ seconds)	123

Table 6.8:	Effect of Publishing Rate on Message Loss for Different Buffer Sizes ($t_{SD} = 24$ seconds)	124
Table 6.9:	Tradeoff between Throughput and Message Loss ($t_{SD} = 12$ seconds)	127
Table 6.10:	Tradeoff between Throughput and Message Loss ($t_{SD} = 24$ seconds)	128

Glossary of Terms and Symbols

μ_P	Publishing Rate on a Certain Topic/Selector Range
API	Application Programming Interface
COM	Components Object Model
COMET	Concurrent Object Modeling and Architectural Design Method
CORBA	Common Object Request Broker Architecture
CPU	Central Processing Unit
C_V	Co-efficient of Variation of Signal Down Intervals
ES	Event Service
f_{GH}	Frequency of Graceful Handoffs
f_H	Frequency of Non-graceful Handoffs
GNU	Gnu Not UNIX
GPRS	General Packet Radio Services
ICMP	Internet Control Message Protocol
IP	Internet Protocol
IPv6	Internet Protocol Version 6
J2EE	Java 2 Enterprise Edition
JMS	Java Message Service
JNDI	Java Naming and Directory Interface
JVM	Java Virtual Machine
LAN	Local Area Network
L_{ND}	Percentage Message Loss in Nondurable Subscriptions
MOM	Message Oriented Middleware
N	Average Number of Subscribers per Broker
PDA	Portable Digital Assistant
QoS	Quality of Service
RMI	Remote Method Invocation
S	Length of Message Body
T	Total Time of Simulation
t_c	Mean duration between any Two Consecutive Requests Made by Each Subscriber to the Broker
TCP	Transmission Control Protocol
t_{CS}	Mean Duration of Each Connect State Interval
t_{SD}	Mean Signal Down Interval
T_{SD}	Total Time in Signal Down State during the Simulation
UDP	User Datagram Protocol
X_P	Maximum Publisher Throughput
X_S	Subscriber Throughput

Chapter 1

Introduction

1.1 Background

The field of mobile wireless computing is as broad as that of traditional computing. There has been an ever increasing demand from consumers to be able to access all the services accessible from desktops through devices such as cell phones and portable digital assistants (PDAs) without any interruptions and performance degradation. However, user mobility and weakly connected wireless environments make it challenging for the developers to provide services seamlessly to the mobile user. Under these conditions, the concept of middleware, which is well established in fixed networks (herein called *fixed domain*) becomes even more important in the context of mobile networks (herein called *mobile domain*). Middleware in a mobile domain is not meant to deal with the prevalent heterogeneous infrastructures only. In order to ensure seamless provisioning of services to the users the middleware in the mobile domain needs to provide some services additional to those it provides in fixed networks e.g. mobility management, temporary disconnections management, QoS assurance over wireless channels and location dependent services provision.

While middleware in the fixed domain can place high computation loads on the clients' resource rich devices, use synchronous communication in permanent connection environments and can exploit static context to achieve performance, a middleware in the mobile domain deals with limited computing capacity devices which frequently get

disconnected from the network and access network from various attachment points. Hence the traditional middleware technologies designed for fixed networks cannot provide the desired behavior in mobile wireless settings. Most popular fixed domain middleware such as CORBA [46], [52] and COM [53] and mechanisms such as Java RMI [51] face challenges such as software size and protocol suitability [11]. For example, they rely mostly on remote method invocations, a synchronous paradigm that couples the two parties in space and time during the whole length of communication. In unreliable mobile settings such coupling cannot be ensured as the mobile client may get temporarily disconnected from the network due to poor wireless channels or it may move from one part of the network to the other. Relying on solutions such as Mobile IP to manage mobility cannot support location and context awareness. Location awareness is essential to provide emerging location dependent mobile services while context awareness enables the system to adapt to the current system environment such as the current wireless channel conditions.

Event based middleware on the other hand use asynchronous communication. It can also support publish/subscribe paradigm where publishers can publish events without having to care for the consumers of messages and the subscribers can receive those events whenever they are ready irrespective of when and who published it. In other words, publish/subscribe middleware decouples the communicating parties in space, flow and time. This decoupling nature of publish/subscribe middleware allows for disconnected operations and user mobility, making them suitable for the mobile domain. Moreover, it makes publish/subscribe middleware highly scalable. This has been observed in several contexts such as [7], [12], [16], [18] and [54]. This scalable feature of

publish/subscribe systems along with their inherently anonymous, asynchronous and loosely coupled nature makes them a good choice for the mobile domain.

1.2 Motivation for the Thesis

Due to the flexible and scalable nature of event based publish/subscribe systems, in recent years there has been a tremendous growth in interest in these systems in the research community. They are now considered a preferred alternative for achieving scalability and overcoming the problems related to protocol suitability in synchronous middlewares, especially for dynamic distributed applications. Unfortunately, while there has been extensive research on these systems for the fixed domain there is very little research on the behavior of these systems in a mobile domain. But as already discussed, by removing all explicit dependencies between the interacting parties, publish/subscribe systems make a very good choice for mobile applications. The thesis is thus concerned with their extension to the mobile wireless domain. The extension should not only make them work in a mobile domain but should also ensure high performance under high frequency of handoffs and error prone wireless channels. For this extension, the thesis has used one of the most widely accepted messaging middleware standards: Java Message Service, JMS [59].

1.3 Goals of the Thesis

The first goal of the thesis is to characterize a popular JMS implementation for the mobile domain, studying the effect of different parameters and mobility environments on its performance. This establishes whether or not a seamless integration of the fixed and mobile domains is possible for publish/subscribe systems. It also provides insights into

the dynamics of the publish/subscribe system, identify the challenges it faces in mobile environments and help in proposing solutions for them.

The second goal of the thesis is the design, prototype implementation and evaluation of middleware level handoffs which can guarantee that no events will be lost when a user moves from one access point to the other. The third goal of the thesis is to propose a novel technique that can satisfy the needs of mobile customers providing them with high performance as well as low message loss in mobile wireless settings.

1.4 Contributions of the Thesis

To the best of our knowledge, this thesis is the first experimental research on the behavior of publish/subscribe in the mobile wireless domain. The main contributions of the thesis are:

- The characterization of publish/subscribe systems in the mobile wireless domain.
- The study and analysis of effects of mobility parameters on publish/subscribe performance.
- The design and implementation of a prototype of Middleware Handoffs for publish/subscribe systems.
- The evaluation of Middleware Level Handoffs, their comparison with other alternatives and identification of the performance concerns.
- The demonstration of the performance concerns of all the proposals available for extending publish/subscribe to the mobile domain and identification of the basic reason for the limitations of these proposals.
- The proposal of a novel technique called semi-durable subscriptions to overcome the challenges faced by current implementations and traditional solutions.

- The design and implementation of a prototype of semi-durable subscriptions.
- The comparison of semi-durable subscriptions technique with other alternatives and the experimental validation that it leads to significant benefit in performance.
- The investigation of the effect of different workload and mobility parameters on semi-durable subscriptions performance
- The identification and study of the effect of system parameters on semi-durable subscriptions and a demonstration of how they can be controlled to configure a system according to the desired characteristics and/or the needs of the environment.

1.5 Thesis Outline

The thesis comprises seven chapters. Chapter 2 provides an overview of publish/subscribe systems, their possible extension to mobile domain and the current state of the art. Chapter 3 gives details about the experimental setup, performance measures and parameters. JMS is characterized for the mobile wireless domain in Chapter 4 while Chapter 5 presents the design, implementation and evaluation of middleware level handoffs. Chapter 6 introduces semi-durable subscriptions and presents their design, implementation and evaluation. Chapter 7 concludes this thesis and discusses directions for future research.

Chapter 2

Publish/Subscribe System Overview

This chapter gives an overview of publish/subscribe systems and the challenges they face. It also gives a brief survey of representative publish/subscribe systems to motivate the selection of Java Message Service (JMS) [59] as a platform for this research. Various features of JMS including its architecture and programming model have been briefly described. Finally, the extension of publish/subscribe systems to the mobile wireless domain and the corresponding challenges are discussed.

2.1 Publish/Subscribe System

2.1.1 Components

A publish/subscribe system consists of *publishers* that are the sources of information called *events* or *messages*, and *subscribers* that show their interest in receiving certain types of events or messages by submitting their *subscriptions* to the system. Whenever a message is published, the middleware delivers it to all the subscribers whose subscription criterion matches the information in the message. The basic model of a publish/subscribe system thus depends on a neutral mediator, which is responsible for an efficient and reliable delivery of all the events to interested subscribers. This mediator, also called message broker or broker, also provides storage and management of subscriptions and messages.

2.1.2 Communication Paradigm

The publish/subscribe paradigm is different from the traditional point-to-point communication models in a number of ways. In publish/subscribe systems the communication between the end points is anonymous, asynchronous and loosely coupled. In other words, publish/subscribe systems decouple publishers and subscribers in space, time and flow [23]. This decoupling of publishers from subscribers is discussed in more detail.

- **Space Decoupling:** Publishers and subscribers are decoupled in space as they do not need to know each other nor do they have any references to each other. Publishers publish messages to the broker without having any knowledge of the interested subscribers while subscribers receive messages from the broker without a need of knowing the originator of the message or the number of publishers participating in the interaction.
- **Time Decoupling:** The interaction between the publisher and subscriber is decoupled in time, as a subscriber needs not to be connected when the message it was interested in was published. If the subscriber is not connected, the broker can store the message for the interested subscriber and deliver it as it reconnects. On the other hand, the publisher, the originator of the message, may not be connected when the subscriber actually receives the event.
- **Flow Decoupling:** Similarly publishers and subscribers are decoupled in flow, as the publisher does not block after publishing a message. It may publish another message or do some other activity irrespective of whether the subscriber has received the message or not.

This decoupling of publishers and subscribers in time, space and flow makes publish/subscribe systems highly scalable by removing all explicit dependencies between the interacting parties. It also helps the system to adapt quickly to the dynamic environment. As a result, publish/subscribe systems are becoming increasingly popular for dynamic distributed applications characterized by multiple subscriptions for the same information. They have numerous applications and are currently being deployed for a large variety of applications such as stock-information delivery, auction systems and air-traffic control, to name a few.

2.1.3 Subscription Types

There are two types of subscriptions in publish/subscribe systems:

2.1.3.1 Subject Based Subscriptions

In publish/subscribe systems, the subscriber can subscribe to a certain topic, thus receiving all the messages published on that topic. This is generally referred to as *subject based* or *channel based* subscription. Advantages of subject subscriptions are the ease and simplicity of its implementation whereby each subject can be assigned to a multicast group for example.

2.1.3.2 Content Based Subscriptions

The publish/subscribe systems are now evolving towards what are called *content-based* subscriptions in which the user has added flexibility of choosing a certain subset of messages published on a particular topic, by submitting its filtering criteria as a part of its subscription request [49]. For example, a subscriber application using subject based subscription can subscribe to a subject *stock*, in which case it will receive all message

published on this subject. On the other hand, while using content based subscriptions the subscriber can not only specify the type of stock it is interested in but also the price range and volume using for instance the following *event schema* [type: string, price: dollar, volume: integer]. The subscriber can thus have a content based subscription such as (type = RIM & price < 25 & volume >200). In addition to this, by using different event schemas and predicates, other scenarios are also possible with content based subscriptions. For example, the subscriber can specify its interest in receiving information on a particular stock only when its price increases or decreases by a certain amount.

While content based subscriptions give the subscriber added flexibility of choosing a subset of messages it is interested in, its scalable implementation is still an open issue. In subject based subscriptions each subject can be assigned to a multicast group but in content based subscriptions such techniques cannot be readily applied. Opyrchal has studied the problem of mapping from matching events to multicast groups in detail [49]. He has simulated a number of algorithms for exploiting IP Multicast in content-based subscriptions and has outlined the advantages and disadvantages of each algorithm. Banavar has also proposed a novel technique for multicasting among the brokers in a content based system and has shown how using his technique content based systems can be deployed in large and geographically distributed settings [5].

On the other hand, Carzaniga has presented a new algorithm for content based forwarding which can make quick decisions when there are a large number of predicates and a high volume of messages [15]. In another paper Carzaniga has formulated a general model for content based addressing and routing and have shown that traditional unicast and multicast addressing and routing can be considered as instances of this more

generalized model [17]. In another paper Carzaniga formulates the foundational concepts of content based networking and has shown how these can be related to the corresponding concepts used in traditional networking [14].

One of the other challenges for content based subscriptions is efficient filtering of messages at the broker i.e. matching the incoming events to event subscriptions. Campailla has presented an efficient and scalable matching approach using Binary Decision Diagrams [8]. Aguilera has proposed another approach and has shown that his approach has sub-linear time complexity in the number of subscriptions and has a linear space complexity [1]. Fiege has described the design and implementation of an event based system that supports scopes and event mappings [31].

2.1.4 Architectures

A publish/subscribe system may have a centralized or a distributed architecture [18]. These are discussed in the following subsections.

2.1.4.1 Centralized Architecture

A centralized architecture has only one message broker. All publishers publish their events to this broker while all subscribers subscribe to this broker for events they are interested in. The centralized architecture is thus simple and as all the publishers and subscribers connect to the same broker, the job of the broker is just to match and dispatch events to the interested subscribers.

In a centralized architecture the broker represents the single point of failure. Also as the number of subscribers and/or publishers increases the broker may become the performance bottleneck, limiting the scalability of the system. Thus, although a

centralized architecture may be used for small scale deployments (more of a LAN like environment), it cannot be used for large wide-area deployments.

2.1.4.2 Distributed Architecture

A distributed architecture consists of a number of brokers each having a subset of publishers and subscribers. As in a distributed architecture, all publishers and subscribers are not connected to the same broker, in order to ensure that the message published at one broker is received by all the subscribers in the system, the brokers need to be interconnected to each other. This gives rise to the design issues such as broker interconnection topology, routing algorithm and processing strategies. These design issues are discussed in detail in the following section (Section 2.1.5).

2.1.5 Design Challenges

A distributed architecture of a publish/subscribe system poses the following design challenges [18].

2.1.5.1 Broker Interconnection Topology

Which topology should be used for interconnecting the brokers? There are several possibilities, a number of which are presented here.

- **Hierarchical Topology:** In a hierarchical topology the brokers are connected in a hierarchical fashion as shown in Figure 2.1. Each broker in a hierarchy has a *master* broker and several *client* brokers. The broker at the top of hierarchy has no master broker and is called a *root* broker. The advantage of using a hierarchical topology is the *grouping* of subscriptions as we move upward in the hierarchy, thus simplifying the filtering algorithm. Also as

opposed to other architectures, there may be a substantial bandwidth saving on the links between the brokers because in a hierarchical topology the message is duplicated only in the lower layers. Thus a message for which there are thousands of subscribers may not be duplicated until very low in the hierarchy. This is because the subscriptions of all the clients at each level are grouped together and thus multiple subscriptions for the same message appear as a single subscription at a higher level. The disadvantage of a hierarchical topology is the overloading of brokers high up in the hierarchy. Also such brokers represent critical points of failure in the network.

Figure 2.1: Hierarchical Topology of Brokers

- **Peer-to-peer Topology:** In a peer-to-peer topology shown in Figure 2.2, the brokers are interconnected as peers allowing the bidirectional flow of messages and subscriptions. There may be a single link between the two

brokers in a peer-to-peer topology (also called acyclic peer-to-peer topology) or multiple redundant links for fault tolerance. A peer-to-peer topology is flexible and does not have any critical points of failures but it faces challenges that include devising of efficient technique for routing messages among brokers.

Figure 2.2: Peer-to-Peer Topology of Brokers

- **Hybrid Topology:** A combination of the above two topologies is also possible consisting of either a hierarchy of several peer-to-peer topologies or a peer-to-peer topology of several hierarchies.

Each topology has its advantages and disadvantages and it should be carefully selected according to the requirements and desired characteristics of the system.

2.1.5.2 Routing Algorithm and Processing Strategy

Which algorithm should be used to efficiently route messages between the interconnected brokers such that a message published at one broker is delivered to all the subscribers in the system irrespective of the brokers the subscribers are connected to? Also, where in the network should the data be processed to optimize message traffic? As discussed in Section 2.1.2 for content based subscriptions simple multicasting can not be used. The problem becomes quite complex in distributed architectures having peer-to-peer and hybrid topologies. Several algorithms have been proposed (as discussed in Section 2.1.2) each having its advantages and disadvantages. The choice of the algorithm depends on the nature of the problem.

The distributed architecture along with its previously mentioned design issues has been studied in several contexts. Carzaniga has presented a design and evaluation of Wide Area Notification Service SIENA [18]. He has discussed SIENA's interface to applications and have evaluated several topologies, routing algorithms and processing strategies to optimize performance. In another paper Carzaniga has given a set of requirements for a benchmark suite for distributed publish/subscribe services [13]. Papers such as [5], [15], [17], [19], [20], [40] and [49] investigate techniques and algorithms for efficient dissemination of information in content based distributed publish/subscribe systems.

2.1.5.3 QoS

How to provide the desired QoS is an important question. Araujo has described IndiQoS architecture that supports QoS provisioning in publish/subscribe systems [2] [3] [4]. His approach allows the publishers to specify QoS profiles while publishing the

messages, and subscribers to specify QoS constraints as a part of their subscription request. The broker in the system is responsible for providing the desired QoS based on these profiles and constraints, using the underlying resources such as data networks and operating system.

Publish/subscribe systems at present however provide only limited QoS features such as ordered/unordered delivery, best effort/reliable delivery. Zhao has proposed a new high level scheme for specifying QoS parameters [65]. He has introduced an event stream interpretation language that allows the subscribers to accurately specify tolerable distortions of message delivery and at the same time gives publish/subscribe middleware flexibility to use more efficient message delivery protocols.

2.2 Survey of Publish/Subscribe Systems

This section presents a brief overview of representative publish/subscribe systems [10]. The goal is to motivate the selection of a suitable platform for this research. The public availability of the target platform has been one of the major factors in this selection. Following is a list of representative publish/subscribe systems given in an alphabetical order.

2.2.1 CORBA Notification Service

CORBA Notification Service [47] is a specification for a publish/subscribe service by Object Management Group. It is an extension of channel based publish/subscribe service ES (Event Service) in CORBA, having content based subscription mechanisms. It consists of *event suppliers* that act as publishers of events,

event consumers that act as subscribers and *event channels* that are responsible for sending events to the interested subscribers.

2.2.2 Elvin

Elvin [56] developed by Distributed Systems Technology Center (DSTC) at the University of Queensland, supports clustering of brokers; each cluster appears as a single access point to the clients and has some load balancing features. Clusters can further be grouped on a larger scale into a federation. Elvin also features quenching that helps save bandwidth and other system resources by limiting the publisher's output to only those events for which there is at least one subscriber.

2.2.3 Herald

Herald [7], a Microsoft Research Project consists of *publishers*, *subscribers* and *rendezvous points*. Rendezvous points act as access points for publishers and subscribers.

2.2.4 Hermes

Hermes [50] developed in the Computer Laboratory of the University of Cambridge consists of *event clients* and *event brokers*. Event clients can act both as publishers and subscribers while event brokers act as service access points for event clients. Event brokers can be interconnected in an arbitrary topology and a dynamic routing strategy to route subscriptions and messages can be used.

2.2.5 Java Message Service

Java Message Service [59] (JMS) developed by Sun and several partner companies, is a Java API that allows applications to create, send, receive and read

messages. It supports both point-to-point messaging and the publish/subscribe approach. JMS is discussed in detail in Section 2.3.

2.2.6 JEDI

JEDI [22], another publish/subscribe system, was developed at CEFRIEL and Politecnico di Milano. It consists of *active objects* and an *event dispatcher*. Active objects represent publishers and subscribers while the event dispatcher is responsible for sending published events to all interested subscribers. The event dispatcher can also be implemented as a distributed set of dispatching servers. One of the important characteristics of JEDI is that it offers some support for client mobility.

2.2.7 SIENA

SIENA [18] developed at University of Colorado is a distributed publish/subscribe system that aims at maximizing expressiveness of the subscription filtering language as well as the scalability of the implementation. It consists of network of *servers* acting as access points to *clients*, both publishers and subscribers. The information is dynamically routed between the servers which may be connected in a hierarchical topology or an acyclic peer-to-peer topology.

2.2.8 STEAM

STEAM [41], [42] is a publish/subscribe system developed at Trinity College of Dublin for mobile ad hoc networks. It supports both subject based and content based subscription mechanisms and consists of distributed components allowing clients to subscribe by contacting a local network service interface.

There are several other publish/subscribe systems such as TIBCO [63] and REBECA [64].

Table 2.1: Survey of Publish/Subscribe Systems

Publish/Subscribe System	Developed/Specified By:	General Availability
CORBA Notification Service	Obj. Management Group	Available
Elvin	DSTC, Uni. of Queensland	Available
Herald	Microsoft	Not Available
Hermes	University of Cambridge	Not Available
JMS	SUN and other partners	Available
JEDI	CEFRIEL, Pol. di Milano	Available
SIENA	University of Colorado	Available
STEAM	Trinity College of Dublin	Not Available

Table 2.1 shows the general availability of the publish/subscribe systems discussed above. Based on the above survey, JMS was selected as the base publish/subscribe system for this research. One of the reasons is its general availability, as many implementations of JMS are easily accessible for download and use. Also JMS defines a common set of interfaces and semantics to allow programs written in JAVA to communicate with other messaging implementations [60]. Other attractive features of JMS include:

- Support for both point-to-point and publish/subscribe messaging domains.
- Light weight clients' libraries are available to be used on mobile devices [57].
- Support for message reliability with the use of *message persistence*.

- Prevention of message loss during disconnections with the use of *durable subscriptions*.
- Support for message priorities.
- Support for transactions.
- Mature implementations based on long-standing message oriented middleware (MOM) solutions.
- Ability to be implemented as a distributed publish/subscribe system.
- Compatibility with other J2EE features such as session and entity beans.

2.3 Overview of JMS

Java Message Service [59] (JMS) developed by Sun and several partner companies, is a Java API that allows applications to create, send, receive and read messages. It supports both point-to-point messaging and publish/subscribe approach. Communication in JMS, in addition to being loosely coupled is asynchronous and reliable.

2.3.1 JMS Architecture

A JMS application consists of [60]:

- **JMS Provider:** that implements the JMS interfaces and provides administration and control tools.
- **JMS Clients:** are the programs written in JAVA that are either the producers of messages or consumers.
- **Messages:** are the objects that carry information from producers to consumers.

- **Administered Objects:** are created by Administrator and are preconfigured to be used by clients. Examples include destinations and connection factories.
- **Native Clients:** are those programs which instead of using JMS API use a messaging product's native API. These are the applications which were created before the release of the JMS API and were modified after JMS became available. As a result they include both the JMS and native API.

The relationship between various components is depicted in Figure 2.3 [60]. Here administrative tools are used to bind destinations and connection factories into a Java Naming and Directory Interface (JNDI) namespace so that JMS Clients can form a connection to these administered objects after looking them up in the namespace.

Figure 2.3: JMS API Architecture (from [60])

2.3.2 JMS Programming Model

The JMS Programming model is shown in Figure 2.4 [60]. Various components are discussed as follows [43], [60], [61]:

Figure 2.4: JMS Programming Model (from [60])

Connection Factory is an object used by the client to make a connection to the JMS provider. It contains connection configuration parameters set by the administrator. Destination objects represent the target of the messages the publishers produce and the source of the messages subscribers consume. In a publish/subscribe domain it specifies the *topic*. Connection objects represent the connection of the client with the JMS provider which may be the open TCP socket between the JMS Client and JMS provider's server daemon. Connection objects create session objects which are single-threaded contexts for message production and consumption. Session objects create message producers and message consumers. Message Producer objects send messages to a destination while consumer objects consume them. A Message consumer registers its interests with the JMS provider so that whenever a message of interest to the consumer is produced it is delivered to it.

Messages are the objects that carry information from the producers to the consumers. JMS defines six types of messages TextMessage, MapMessage,

BytesMessage, StreamMessage, ObjectMessage and Message. Each JMS message consists of three parts: header, properties and body.

i) Header: Header of a message has a number of predefined fields which are used both by JMS client and JMS provider for message identification and routing. One of the important header fields is JMSMessageID which is used to uniquely identify a message.

ii) Message Properties: This part of the message is for user defined properties. Properties in addition to those defined in the message header fields can either be used for providing compatibility with other messaging systems or for defining extra selection filters with the help of message selectors.

iii) Message Body: This part of the message carries the actual message.

2.3.3 Types of Subscriptions

JMS defines two types of subscriptions nondurable and durable.

2.3.3.1 Nondurable Subscriptions

Nondurable subscriptions provide low levels of reliability and are used for applications that can afford to miss messages. The JMS provider does not retain records of inactive nondurable subscribers. Hence, if a nondurable subscriber gets disconnected from the network, it loses all the messages published during the time it was disconnected. They, however, give high throughput as there are no overheads involved in storing messages for inactive subscribers. Examples of nondurable subscriptions include newsgroups, stock information delivery and auction systems.

2.3.3.2 Durable Subscriptions

Durable subscriptions offer high levels of reliability and are used for applications that can not afford to miss a message. Durable subscriptions ensure the delivery of all messages to the subscriber even if it is disconnected from the network for a certain period of time. The JMS provider stores all the messages for the *inactive* durable subscribers and delivers them as they reconnect to the network.

Durable subscriptions, however, impose overheads on the system in terms of maintaining and managing the list of inactive subscribers, storing messages for them and delivering the messages as they reconnect. Also, the processing time required by the message filtering algorithm increases with the increase in number of subscribers. Hence the throughput of durable subscriptions decreases with an increase in the number of inactive subscribers. In spite of these overheads durable subscriptions are used for critical applications such as air traffic control and temperature control for a reactor.

2.3.4 Filtering with Message Selectors

If a subscriber is interested in receiving only a subset of messages published on a certain topic, JMS allows the subscriber to submit its selection criterion as a part of its subscription request. This is done with the help of Message Selectors. A Message Selector is a String that contains the expression whose syntax is based on a subset of the SQL92 conditional expression [60]. The JMS provider then sends only those messages to the consumer whose header and properties match the consumer's selection criterion.

2.3.5 Message Consumption Models in JMS

JMS defines two types of models for message consumption.

a) Synchronous

In the synchronous case the consumer of the message fetches the messages from the destination by explicitly calling the `receive( )` method. This method blocks itself until the arrival of the message or it timeouts if the message does not arrive within a certain *timeout* period that can be passed as a parameter to this method.

b) Asynchronous

In the asynchronous case the consumer of the message registers itself with a `MessageListener`. Whenever the message for a consumer arrives, the JMS provider sends this message to the consumer by calling its `MessageListener's onMessage ( )` method.

2.3.6 QoS Features

JMS offers several quality of service features. They are briefly described.

a) Support for Disconnected Operations

JMS allows the retrieval of the messages published during the time the subscriber is disconnected from the network. This is done with the help of durable subscriptions discussed in Section 2.3.3.2.

b) Message Persistence

JMS defines two delivery modes for messages `NON_PERSISTENT` and `PERSISTENT`. In the `NON_PERSISTENT` mode messages may be lost in case of a JMS provider failure but in the `PERSISTENT` mode, the JMS provider takes special care and guarantees that a message will not be lost if the JMS provider fails. Such messages are logged to stable storage when sent.

c) Message Priorities

JMS defines message priorities to ensure that urgent messages are delivered first. There are ten priority levels from 0(lowest) to 9(highest) with default priority of 4. The JMS provider tries to deliver messages with higher priorities earlier than those with lower priorities. However, message delivery may not always be in exact order of message priorities.

d) Support for Transactions

JMS supports the concept of transactions. A series of operations that can be grouped together into an atomic unit of work is called a transaction. A transaction is a committed if all the operations complete successfully. However, if any of the operations in a transaction fails the transaction is rolled back and can be attempted from the beginning. Session interface in the JMS API provides the necessary `commit()` and `rollback()` methods for transactions.

2.3.7 JMS Implementations

Several commercial and public domain JMS implementations are available. Some implementations are more complete and functional than others. Among the implementations available publicly are FiornaMQ [32], JBossMQ [39], Joram [48] and OpenJMS [26] while IBM MQSeries [36] and SonicMQ [58] are the examples of commercial implementations.

2.4 Publish/Subscribe in a Mobile Environment

2.4.1 Suitability

There are many characteristics of a publish/subscribe system that make them ideal for a mobile environment. Anonymous and loosely coupled communication between publishers and subscribers along with the inherently asynchronous nature of these systems help them adapt quickly to a dynamic mobile environment. Decoupling in space allows the subscriber to move from one location to the other without *informing* the publisher while decoupling in time allows for the disconnected operations of the subscriber. This decoupling in space and time makes publish/subscribe systems a good choice for mobile and weakly connected environments as offered by mobile wireless networks. Scalable and flexible nature of these systems makes them more suitable for such networks. Thus middleware based on a publish/subscribe paradigm can be considered as one of the best options for mobile wireless scenarios. There is hence a pressing need to extend publish/subscribe systems to mobile wireless environments.

2.4.2 Publish/Subscribe Extension for the Mobile Domain

Publish/subscribe extensions for mobile wireless environments can be divided into two classes. The first class includes extensions for those applications which are based on mobile code technology called mobile agents. Mobile agents may execute on different hosts during their life cycle and hence their mobility needs to be managed if they are accessing a publish/subscribe system. The second class includes those applications which run on mobile hosts such as laptops, cellular phones and PDAs. As a mobile host moves from one location to another it may change its access point to a

publish/subscribe system. It may also get temporarily disconnected from the network due to poor wireless channels. A publish/subscribe system will thus need to handle this mobility and temporary disconnections. Although both classes are interesting and challenging, this thesis focuses on the second class (i.e. extending publish/subscribe system for mobile wireless users).

Although a lot of work has been done on publish/subscribe systems in the fixed domain, there is comparatively little research on the behavior of these systems in a wireless mobile environment. Research papers such as [21] and [29] have studied mobility management in publish/subscribe systems. Their research is more focused on general requirements dealing with design issues, data definitions for publications and query language for subscriptions. The nature of their work is thus broader and more theoretical than the work this thesis focuses on.

Other researchers have studied the specific problem of extending publish/subscribe systems to mobile wireless environments which is the focus of this thesis. Huang and Garcia-Molina have discussed how publish/subscribe systems can be extended to operate in a mobile environment, giving a basic model for the deployment of these systems on mobile networks and identifying some of the problems that would arise during such a deployment [34]. One of the issues in addressing those problems is to find a suitable evaluation environment. Caporuscio and others report their experience in attempting to evaluate the performance of their SIENA publish/subscribe service in a GPRS based environment using the available emulation environments [9]. Although Huang [34] and Caporuscio [9] address the issues in the deployment and evaluation of

these systems over wireless networks, they have not given any concrete results on the performance behavior of these systems in such environments.

Concurrent to this thesis, there has been research to support mobility with SIENA [10] and REBECA [30] [64]. However, unlike other research, the thesis is more focused on the performance aspects of publish/subscribe extensions to mobile domain. The goal of this thesis is to extend publish/subscribe systems to a mobile domain in a way that can ensure high performance under high frequency of handoffs and error prone wireless channels.

2.4.3 Publish/Subscribe in Mobile Cellular Networks

As discussed in Section 2.1.4, scalable implementations of publish/subscribe systems require them to be deployed as a distributed set of interconnected brokers. Figure 2.5 shows the distributed architecture for a publish/subscribe system in a mobile cellular environment [6], [24], [55]. A client which can act as a publisher or a subscriber or both runs on a mobile host such a cellular phone, PDA or laptop. It connects to one of the message brokers in the system over the wireless link in the cellular network. The message broker may be located somewhere in the cellular network infrastructure or it may be at a fixed node outside the network in which case the client connects to the broker through the wireless gateway.

As a client moves from one part of the network to the other or from one network to the other, it may need to disconnect from one broker and connect to the other in its new location. During this *handoff*, the client may get temporarily disconnected from the network. Also, due to weak connections in wireless environments, the client may get occasionally disconnected from the network even if it remains in the same cell, thus

hindering its ability to deliver/receive messages from the system. When a client acts as a publisher, it may not be able to publish messages during such intervals but as soon as it reconnects to any of the message brokers, it can publish some or all of the messages. However, when a client acts a subscriber it may lose messages published during the time it was disconnected from the network or as it moves from one message broker to the other. Thus client mobility and weakly connected environments in mobile cellular networks poses an additional problem of occasional message loss, generally absent in fixed networks. Limited bandwidth in wireless networks may also prevent or delay the delivery of messages to the subscriber.

Figure 2.5: Publish/Subscribe in Mobile Cellular Networks

2.4.4 Publish/Subscribe in Mobile Ad Hoc Networks

Mobile ad hoc networks are the networks that are formed by transitory association of mobile nodes. They do not require any fixed support infrastructure and are typically based on short-range wireless technology. In mobile ad hoc networks, in addition to the

publishers and subscribers, message brokers would also be running on mobile hosts. Also unlike mobile cellular networks, some nodes in an ad hoc network may occasionally get isolated from the rest of the network. This makes mobility management in ad hoc networks much more challenging.

Each node acting as a publisher or a subscriber may be required to act as a broker as well in mobile ad hoc networks. Traditionally, a broker in content based publish/subscribe networks requires a tremendous amount of computing (e.g. subscription management, event filtering and dissemination, message preservation for inactive subscribers and routing events to other brokers in the network). Additionally in mobile networks, brokers have to manage mobility. Assigning all these tasks to devices with limited computing and limited battery power such as PDAs in ad hoc networks does not seem to be a feasible solution. Hence altogether new solutions for content based publish/subscribe system in mobile ad hoc networks need to be sought. Thus extending publish/subscribe system to mobile ad hoc networks is essentially a different problem from extending publish/subscribe systems to mobile networks having fixed infrastructure components (e.g. cellular networks). Although both problems are challenging, this thesis deals only with the latter.

There has been some research for extending publish/subscribe systems to mobile ad hoc networks. Huang and Garcia-Molina have studied effective publish/subscribe tree construction for routing information from sources to interested subscribers [35]. They have proposed a new greedy algorithm called SHOPPARENT for optimal and fully distributed tree construction. Also Meier and Cahill have designed a new event based middleware called STEAM specifically for mobile ad hoc networks [41], [42].

2.4.5 Challenges and Performance Issues for Publish/Subscribe Systems in Mobile Wireless Environments

As discussed above publish/subscribe systems face additional challenges in mobile wireless environments that are generally absent in fixed networks. The major challenges are:

- **Managing clients' mobility:** Clients (both publishers and subscribers) in mobile environments may change their access points to the publish/subscribe system. The system should be able to handle this mobility ensuring that all messages are delivered to all interested subscribers independent of their access points. This will require an implementation of a handoff procedure that takes place whenever a subscriber changes its access point. The handoff procedure should ensure that all active subscriptions of the subscribers are successfully transferred to the new broker and the messages published during the handoff procedure are delivered to the subscriber.
- **Managing temporary disconnections:** Clients in wireless environments may get temporarily disconnected from the network due to poor wireless channels. Publish/subscribe systems should handle these temporary disconnections ensuring that at least no important messages are lost by the subscribers. This requires the system to store messages for the disconnected subscribers and deliver them as they reconnect.
- **Managing environments with limited bandwidth:** Clients in wireless environments may have limited bandwidths available to them due to loaded error prone wireless channels. Publish/subscribe systems should efficiently manage such limited bandwidth scenarios. This may require the broker to do message filtering (sending

only *important* messages), message simplification (converting a multimedia message to a text message) and message compression for the low bandwidth subscribers.

- **Performance Concerns:** Brokers in publish/subscribe system need to perform a number of tasks such as subscription management, event filtering, event dissemination as well as messages routing and processing. In order to provide efficient and reliable services to the client in a mobile environment the broker needs to handle mobility, temporary disconnections and limited bandwidth environments. These additional tasks can put a substantial load on the broker in a dynamic mobile environment with frequent disconnections and connections. The broker may thus become the performance bottleneck limiting the scalability of the system.

Broker performance is thus a serious concern in mobile wireless environments and techniques that not only effectively handle mobile wireless clients but also ensure high performance warrant investigation. This precisely is the focus of this thesis.

Chapter 3

Performance Measurement and Instrumentation

This chapter discusses the experimental setup used in this research. It provides details about the various components in the setup including publishers, subscribers and subscribers' mobility model. It also motivates the selection of performance metrics and parameters used in this thesis. Finally it presents an overview of the test conditions.

3.1 Experimental Setup

3.1.1 Basic Setup

Figure 3.1 illustrates the evaluation setup consisting of five nodes, one node running a stationary publisher, two nodes acting as message brokers and the remaining two nodes running cellular mobile subscribers.

Although many implementations of JMS are available, in this thesis, SonicMQ [58] is used. SonicMQ is considered a leading JMS implementation in the industry, in terms of its performance, reliability and scalability. The SonicMQ Evaluation Edition used in the setup supports encryption, dynamic routing and server clustering. During the course of the experiments, two SonicMQ message brokers were run, each on a separate Intel based Pentium 4 machine running RedHat Linux version 7.2. A single publisher running SonicMQ client's library on a separate node publishes messages on a certain topic with different message selector values, to the two brokers. Subscribers on two different machines run SonicMQ client's library and subscribes to the message brokers

for the messages in a certain selector range. Wireless channels were emulated between the subscribers and the brokers with the help of NISTNet [44].

Figure 3.1: Logical View of Experimental Setup

3.1.2 Publisher

A single stationary publisher used as a traffic generator publishes messages to the message brokers. In real environments however, there may be more than one publisher and also the publishers themselves can be mobile. But in any case all the messages published will be delivered to the corresponding message brokers for dissemination. The message brokers will forward these messages to all other message brokers in the system, which have subscribers for the published messages. The publisher in this setup thus represents a collection of different sources (publishers and brokers) from which the message brokers in the setup receive messages for dissemination. In other words, the publisher in this thesis acts as a synthetic message/load generator rather than a real world publisher.

3.1.2.1 System Models and Publishing Rate

Both open and closed system models of the publisher are used in the experimentation.

a) Closed Model

A closed model publisher publishes messages to the two message brokers as fast as the system can accept (Think Time for publisher $Z = 0$). This is sufficient to determine the maximum publishing rate which the system can sustain. The following extract shows the pseudo code used by the Publisher to publish messages.

```
...
while(keepPublishing){
    try{
        TextMessage textMessage = session.createTextMessage();
        textMessage.setText("....");
        ...
        for(int i=0; i<numConnections; i++){
            publisher[i].publish( textMessage, deliveryMode,
                                priority, timetolive );
        }
    }
}
...
```

It shows that the publisher tries to publish each message to the two message brokers in a loop. This is necessary to ensure that the corresponding messages received by the two message brokers have the same message selector value. However, as a result of the loop an overload on one of the brokers can temporarily prevent the messages from being published to the other broker as well. In real environments such a dependency may not exist and publisher may be able to publish messages to a certain broker irrespective of other brokers in the system. There is more than one way to prevent such a dependency in the experimentation; the solution considered involves the use of an open-model publisher.

b) Open Model

Figure 3.2 shows the schematics of the open-model publisher.

Figure 3.2: Schematics of the Open Model Publisher

It shows that open model publisher has a separate queue (MessageBuffer) for each broker. The MessageCreator creates messages with a certain exponentially distributed inter-arrival time and puts them in a separate queue for each broker. Two separate threads acquire messages from one of the queues and publish them to the corresponding broker, ensuring that each broker receives every message published. The publishing rate of the publisher is increased in steps until a stage is reached beyond which, increasing the publisher rate would make the system collapse due to saturation. This is adequate for determining the highest sustainable system throughputs for the system.

However, since much of the research on the performance of publish/subscribe systems (such as [37]) use a closed model publisher, the thesis in addition to the open model publisher has also used a closed model publisher. The results obtained with the closed model publisher are given in the Appendix A.

3.1.3 Subscribers

Subscribers represent clients' applications running on mobile hosts. They are the message consumers and act as sinks of messages in this setup. They were run on two stationary machines and their mobility was simulated through a simulator written in Java. Initially each subscriber subscribes to one of the message brokers for its subscriptions but its access point and hence the broker changes many times during the simulation due to mobility.

3.1.3.1 Subscription Pattern

To keep the evaluation simple, each message published in this setup has only one selector value ranging from 0 to 99. These selector values are uniformly distributed. Each subscriber subscribes for messages in a particular selector range (1/10th of the total range). Although this assumption may appear simplistic, it is sufficient to demonstrate different interests of the subscribers in the system and is adequate for investigating the performance issues that this thesis focuses on.

3.1.3.2 Mobility Model

For simulating subscribers' mobility, a program is written in Java which makes each subscriber go through a mobility model illustrated in the state diagram of Figure 3.3. In the *connect* state, the subscriber is in a particular cell, connected to one of the message brokers and receiving messages. A subscriber remains in the connect state for an exponentially distributed residence time with a mean t_{cs} , after which it either *handoffs* from one cell to the other or goes to *signal down state*. Signal down state simulates non-graceful handoffs and signal down intervals occurring due to poor wireless channels. It

remains in the signal down state for an exponentially distributed time with the mean t_{SD} and then with a certain probability either goes back to the same cell or moves to the new cell. The latter assumption holds true because most often channel losses occur near the boundary of the two cells. Hyper-exponentially distributed signal down intervals are also used for some of the experiments. The subscriber in the *handoff state* either reconnects to the same broker or connects to the other broker thus simulating that it has moved out of the location covered by the previous message broker. In either case, after the handoff, the subscriber goes back to the *connect* state.

Figure 3.3: State Diagram of Subscribers' Mobility Model

Although this is not a detailed mobility model keeping track of subscribers' position, it is sufficient to simulate the mobility characteristics of terminals required for this experimentation, including handoffs and signal down intervals.

3.1.3.3 Number of Subscribers

In the experiments, one hundred subscribers were run on each of the two machines. In the beginning, half of the subscribers are connected to one message broker and the other half to the other but this number changes many times during the course of

the simulation due to user mobility. Although one hundred subscribers per broker may seem to be much smaller than the actual number of subscribers connected to one broker in the future cellular publish/subscribe systems, each subscriber in the simulation receives many times more messages per second than is expected in the real world environment. Hence each subscriber in this setup simulates a load of many real subscribers on the broker. At the same time short mean residence time in a cell is used to generate high connection/disconnection requests per second.

3.1.4 Message Filters

As discussed in Section 3.1.3, each message in the setup of this thesis has a certain message selector value while the subscribers subscribe for different message selector ranges, so messages need to be filtered for every subscriber, somewhere in the system. SonicMQ has the option of either enabling this message filtering at the broker or at the subscribers. As it does not make sense to send all the messages on error prone wireless channels with limited bandwidths and let the limited computing capacity devices (like the palmtops) perform the message selection, it was enabled at the broker instead of at the client.

3.1.5 Wireless Channels

As subscribers in the setup represent applications running on mobile hosts in a cellular network, all the communications between the brokers and the subscribers will take place through wireless channels. Thus all the messages sent from the brokers to the subscribers will be conditioned to weakly connected wireless environments characterized by limited bandwidth, higher packet loss, packet duplication and reordering. All these

characteristics were emulated through the use of the NIST Net network emulator [44]. NIST Net is implemented as a kernel module extension to the Linux operating system and can be used to emulate a number of performance scenarios such as packet delay, congestion, bandwidth limitation, packet loss, packet duplication and packet reordering. All these parameters were set in accordance with the most commonly used values of these parameters for mobile cellular networks [24], [25], [55], [62].

3.2 Performance Measures

The following measures are used as performance indicators.

3.2.1 Maximum Publisher Throughput

Maximum Publisher Throughput (X_P) is the maximum number of messages per second that the publisher is able to deliver to the system without making the system collapse. In other words, it is the highest publishing rate that the system can sustain without terminating due to buffer overflow. This performance measure thus indicates the message handling capacity of the system.

It is measured directly by using a counter in the Publisher that keeps track of the number of messages published. For the closed model publisher, it is obtained by dividing the total number of messages published by the Publisher by the total time of the experiment in seconds. In the open model publisher it is the maximum publishing rate that the system is able to sustain.

3.2.2 Subscriber Throughput

Subscriber Throughput (X_S) is the total number of messages received per second by all the subscribers in the system divided by message fanout. As there can be more than

one subscriber interested in the same message, message fanout indicates the extent to which each message is duplicated by the broker. The total throughput of the subscribers is divided by message fanout for direct comparison with maximum publisher throughput.

It is measured by using a counter in each subscriber's thread to keep track of the number of messages received by that subscriber. At the end of the simulation, throughput of every subscriber is added up and then divided by message fanout and total duration of the experiment in seconds to get subscriber throughput. It indicates the message delivering capacity of the system and differs to maximum publisher throughput by message loss per second.

3.2.3 Percentage Message Loss

Percentage message loss is the percentage of the messages lost by the subscribers. Messages may be lost due to subscribers' mobility and temporary disconnections. In the cases where an additional mechanism was used to prevent message loss during handoffs and temporary disconnections, message loss may still occur due to reasons specific to the additional mechanism used.

It is measured by taking the difference between the maximum publisher throughput and subscriber throughput and then dividing by 100. This measure indicates the message loss prevention capacity of the system. The lower the percentage message loss the higher the message loss prevention capacity of the system and vice versa.

3.2.4 Traffic on the Wireless Channels

Traffic on the wireless channels is the number of bytes per second traveling in a given direction on the links between the brokers and the subscribers. It is measured to

indicate the traffic overhead generated by each subscription mechanism in wireless mobile environments. It is of interest because wireless bandwidth is scarce and hence precious.

It was measured using a public domain GNU based tool called Network Top (ntop) [45]. ntop is a traffic measurement and monitoring tool that supports different management activities such as security violation detection and network optimization. It can measure traffic flowing in either direction between the machine on which ntop is running and other machines with which this machine communicates. It can also characterize the traffic as TCP, UDP, and ICMP.

3.3 Parameters

The following are the workload parameters of interest.

3.3.1 Parameters Independent of Mobile Wireless Environments

3.3.1.1 Number of Publishers/Publishing Rate

In real environments there may be varying numbers of publishers publishing messages at varying rates. The total number of publishers and their publishing rates thus become input parameters to the evaluation environment. This setup however only used a single publisher which acts as a message generator and represents the collection of all message sources (publishers and brokers). The assumption is realistic because even in real environments different publishers may be connected to different brokers and the brokers themselves forward the messages to other brokers in the system. The total input to the broker would thus consist of output of a number of publishers and/or brokers; the publisher in this setup precisely represents this scenario.

As maximum sustainable throughput of the system is important in the context of this research, publishing rate of the publisher was increased in steps until the maximum rate the system can sustain is reached.

3.3.1.2 Number of Subscribers

In real environments there may be different number of subscribers in the system at different times. The total number of subscribers is thus another input parameter. However, since this research investigates the maximum sustainable throughput, it is the total demand of subscriptions of the subscribers which matters the most. The setup thus used a fixed number of subscribers but kept their subscription demands much higher than real subscribers' and at the same time scaled down their mean cell residence time to generate a higher connection/disconnection requests per second. During the course of the experiments, the total subscription demand per broker changes many times as subscribers move from one broker to the other. This is sufficient for investigating the performance issues that the research focuses on.

3.3.1.3 Number of Topics/Selector Ranges

Different subscribers will have different interests in a real publish/subscribe system. Their subscription pattern is thus an input parameter. A simple subscription pattern adequate for the nature of this research is used. It is described in subsection 3.1.3.1. Essentially, it consists of having each subscriber subscribe to one tenth of the total range of the message selector values.

3.3.1.4 Subscription Type

As discussed in Section 2.3.3, JMS defines two types of subscriptions: nondurable and durable. Different types may suit different requirements/environments of the subscribers. The thesis evaluates both of these types in detail.

3.3.1.5 Message Size

Messages may have different sizes in a publish/subscribe system, making it an input parameter for the investigation. In JMS, the total length of a message after the removal of IP and TCP headers is the sum of the lengths of the message header, the message properties and the message body. As all the messages have almost the same sized header and properties, the size of the message body (S) was varied as an input parameter and system performance was evaluated for different values of S.

As we can expect small messages (after potential message compression and/or simplification) to be sent over error prone wireless channels with scarce bandwidth in real environments, this thesis uses a small value equal to 50 Bytes as a default value of S.

3.3.2 Mobile Wireless Environment Dependent Parameters

3.3.2.1 Bandwidth

Total wireless bandwidth available is an important system parameter which may affect the overall system performance significantly. During preliminary experiments, it was found that when bandwidth is varied as an input parameter, the system always behaves in either of the two modes. In each system mode there is only a slight effect of changing the bandwidth and the overall behavior remains the same. However, the behavior of two modes differs significantly from each other. Actually in one of the

modes, bandwidth becomes the performance bottleneck while in the other mode the server becomes the performance bottleneck. Thus most of the experiments were done for three bandwidth/mobility environments: fixed wireline, mobile with 10MB total bandwidth and mobile with 220kB total bandwidth. The total bandwidth of 10MB ensures that it is the server that is the bottleneck (and not the bandwidth), representative of what this thesis term the *server bottlenecked region*. In the 220kB total bandwidth environment the bandwidth becomes the bottleneck, representative of what this thesis term the *bandwidth bottlenecked region*.

It is to be noted that the thesis expresses bandwidth in Bytes/second instead of bits/second, the commonly used unit of bandwidth. This is because the NIST Net network emulator used in the experiments takes a value in Bytes/second for setting the required bandwidth. For the sake of brevity the unit of bandwidth is referred to as Bytes instead of Bytes/second.

3.3.2.2 Number of Connection/Disconnection Requests per Second

The total number of connection/disconnection requests that each subscriber makes to the brokers per second can affect performance and is hence an important input parameter. This total number of connection/disconnection requests per second can be divided into those occurring during graceful handoffs of the subscriber from one broker to the other (f_{GH}) and those occurring because of signal down intervals and non-graceful handoffs (f_H). If a Markov chain is made from the mobility model of this thesis with each state of the Markov chain representing a particular state in the mobility model, we can determine the mean duration t_C between any two consecutive requests made by each subscriber to the broker.

The following equation shows the relationship between t_C , f_{GH} and f_H .

$$t_C = 1 / (f_{GH} + f_H) \quad (3.1)$$

Equation 3.1 shows that changing t_C changes both. However, it was found that more theoretical insights could be gained by treating them separately. Hence this thesis calls the frequency of connection/disconnection requests occurring due to signal down intervals and non-graceful handoffs as Frequency of Handoffs, f_H . The frequency of graceful handoffs of the subscriber from one broker to the other is called Frequency of Graceful Handoffs, f_{GH} . Both f_H and f_{GH} are varied by varying t_C .

As discussed in Section 3.1.3.3 and 3.3.1.2, the thesis uses short cell residence times to generate higher connection/disconnection requests per second. The default value of t_C calculated by solving the Markov chain discussed above is 154.32 seconds.

3.3.2.3 Mean of Signal down Interval

Mean signal down interval t_{SD} is the mean length of the interval during which no signal is available to the subscriber. It is an important input parameter especially for the cases where messages are stored for disconnected subscribers. It was thus varied to study its effect on different performance measures.

The default value of t_{SD} used in this research is 12 seconds. Such a high value would stress test the mechanisms that store the messages for disconnected subscribers. This is consistent with other studies such as [10].

3.3.2.4 Co-efficient of Variation of Signal down Intervals

Probability distribution of signal down intervals can affect overall system performance. So in addition to the exponentially distributed signal down intervals this

thesis also used bi-phase hyper-exponentially distributed intervals. Co-efficient of variation C_V of signal down intervals is varied as an input parameter.

The default distribution of signal down intervals in this research is exponential with $C_V = 1$.

3.3.2.5 Total Time in Signal Down State

Although total time in signal down state (T_{SD}) is not an independent parameter and can be given by

$$T_{SD} = f_H * t_{SD} * T \quad (3.2)$$

where T is the total time of the simulation. It was found that varying T_{SD}/T as an input parameter gives additional insights into the dynamics of the system. For example, for studying the effect of f_H , T_{SD}/T was varied keeping t_{SD} constant.

The default value of T_{SD}/T obtained by solving the Markov chain with default values of other parameters is 0.0742.

Table 3.1 summarizes the workload parameters of the thesis.

3.4 Test Conditions

a) Test Duration

Each test was run long enough to handle transients resulting from the initial conditions to ensure that the system has reached a steady state and the effect of transients has diminished. Tests with higher co-efficient of variation generally take longer to achieve a steady state.

Table 3.1: Summary of Workload Parameters

Category	Parameter	Default Value	Levels
Publisher Related	Publisher's System Model	Open	Closed, Open
	Publishing Rate	-	Increased in steps until the maximum rate the system can sustain is reached
Subscriber/ Subscription Pattern Related	Number of Subscribers	100 per broker	100 per broker
	Subscription Pattern	Message selector range from 0 to 99. Each subscriber subscribes to 1/10 th of the total range	Message selector range from 0 to 99. Each subscriber subscribes to 1/10 th of the total range
	Subscription Type	-	Nondurable, Durable
Message Length Related	Size of the Message Body (S)	50 Bytes	50,100,200,500 Bytes
Bandwidth/ Mobility Environment Related	Mobility Environment	Mobile	Fixed, Mobile
	Total Bandwidth	10 MB	220 kB, 10 MB
Subscribers' Mobility Related	Proportion of the Total Time in Signal Down State (T_{SD}/T)	0.0742	0, 0.002, 0.01, 0.025, 0.05, 0.0742, 0.075, 0.1, 0.15
	Mean Duration between consecutive connection requests (t_c)	154.32 seconds	Different levels corresponding to each level of T_{SD}
	Frequency of Handoffs (f_H)	0.00617 per second	Different levels corresponding to each level of T_{SD}
	Frequency of Graceful Handoffs (f_{GH})	0.000309 per second	Different levels corresponding to each level of T_{SD}
	Mean Signal Down Interval (t_{SD})	12 seconds	12, 24 seconds
	Probability Distribution of Signal Down Intervals	Exponential	Exponential, Hyper-Exponential
	Co-efficient of Variation of Signal Down Intervals (C_V)	1	1, 1.5, 2, 2.5

b) Repeatability

Each test was run multiple times to ensure repeatability. As tests were run for duration long enough to handle transients, readings from multiple tests matched closely.

c) CPU Utilizations of Machines

It was ensured in all the tests that CPU Utilization of neither the publisher's machine nor any of the subscribers' machines exceeded 50%. This was necessary to prevent publisher's and subscribers' performance affecting the overall system performance being measured. During each test, the machines running the brokers did not run anything else. The experiments ran uninterrupted by the other users in a dedicated network.

d) Log Purging

After each run all logs at the brokers were purged. In case of durable subscriptions all durable subscriptions of all the subscribers were deleted after every run. This ensured that past history of the broker did not affect performance and that each test received the same initial testing conditions.

e) JMS Related Setting

Each client was run on a separate JMS connection, as it may be located on a separate terminal/application in real environments.

NON_PERSISTENT messages with auto acknowledgements were used. Subscriber received messages asynchronously by running a Message Listener (discussed in Section 2.3.5) which was notified each time a new message for the subscriber arrived.

Chapter 4

Characterization of JMS in Mobile Wireless Environments

This chapter characterizes JMS in mobile wireless environments and studies the effect of different parameters on system performance. JMS defines two types of subscriptions, nondurable and durable; this chapter studies each of them in detail. The effect of different parameters on performance measures is demonstrated with the help of tables and graphs.

4.1 Objectives

The objectives of this chapter are outlined:

- To answer “Is seamless integration of fixed and mobile domains possible for publish/subscribe systems?”
- To be able to answer “What changes do the existing middleware require for mobile wireless environments?”
- To get insights into the dynamics of the system in mobile wireless environments.
- To study the effect of different mobility parameters on the performance of publish/subscribe system.
- To find out the performance challenges the existing middleware face in mobile cellular networks and to be able to propose solutions/alternatives.

As discussed in previous chapters, JMS defines two types of subscriptions, namely, nondurable and durable. This chapter studies their performance in detail for mobile wireless environments.

4.2 Performance of Nondurable Subscriptions

Nondurable subscriptions are less reliable and can be used only for the applications that can afford to miss messages. If a nondurable subscriber disconnects from the network for some reason, all the messages published during the time it was disconnected are lost. Nondurable subscriptions, however, give high throughput as there are no overheads involved in storing messages for the inactive subscribers.

This section studies the effect of various parameters on the performance of nondurable subscriptions in mobile wireless environments.

4.2.1 Effect of Bandwidth and Mobility

The effect of bandwidth and mobility on the performance of nondurable subscriptions is studied using three environments: a fixed wireline environment, a mobile environment with 10MB total bandwidth and a mobile environment with 220kB total bandwidth. With the workload and mobility parameters used for the mobile environment a total bandwidth of 10MB ensures that it is the server that becomes the bottleneck while in the 220kB environment the bandwidth becomes the bottleneck. As discussed in Section 3.1.2.1 both closed and open system models of publishers are used. The results with the closed model are given in Appendix A.1 while the results with the open model are given in this subsection. Default values of workload parameters presented in Table 3.1 are used for the fixed parameters in these experiments and are listed in Table 4.1.

Table 4.1: Input Parameters for Investigating the Effect of Bandwidth and Mobility on Nondurable Subscriptions

Parameter	Value
Publisher Model, Publishing Rate	Open, Maximum Sustainable
Length of the Message Body (S)	50 Bytes
Mean Length of Signal Down Interval (t_{SD})	12 seconds
Probability Distribution of Signal Down Intervals, C_v	Exponential, 1
T_{SD} / T	0.0742

(a)

(b)

Figure 4.1: Effect of Bandwidth and Mobility on Nondurable Subscriptions
(a) Throughput of Subscribers (b) Maximum Throughput of Publisher (c)
Proportion of Messages Lost

The results are shown in Figure 4.1. Figure 4.1(a) shows the subscriber throughput X_S for different environments. The graph shows that the maximum X_S is achieved for the fixed wireline case. In the mobile case when the server is the bottleneck X_S is close to the fixed wireline case. The small drop is due to the connection (and disconnection) requests that each subscriber makes as it moves from one broker to the other or as it goes from the connect state to the signal down state. Each connection/disconnection request gives rise to an overhead resulting in a slight drop in throughput. Surprisingly, the maximum publisher throughput X_P in the mobile region where the server is the bottleneck is higher than the fixed wireline case (Figure 4.1(b)). The reason is that since some of the subscribers are in the signal down state, the average number of subscribers on each broker is less than 100. Thus as the subscribers' load on the broker decreases it is able to accept more messages from the publisher. However, this acceptance of more messages (and message selection) puts an extra overhead on the broker as compared to the fixed wireline case. This is one of the reasons the subscriber throughput X_S in the mobile environment is slightly lower than the fixed wireline system.

Thus in the mobile server bottlenecked region, a higher load due to higher message acceptance and selection rate and the additional connections/disconnections overhead tend to reduce X_S . On the other hand, the decrease in the number of active subscribers tends to increase X_S . The value of X_S observed on the system is determined by both of these counteracting factors.

In the mobile bandwidth bottlenecked region, the subscriber throughput is limited by the bandwidth as the physical channel can carry only a limited number of messages per second. Traffic on the wireless channels was also measured. The results show that although the wireless channel was fully utilized carrying almost 220 kilobytes of data per second, the number of messages received by the subscriber is smaller than the total message carrying capacity of the channel. The total message carrying capacity of the channel is measured by dividing the total bandwidth with the total size of the message (including the lower layers' headers), allowing sufficient bandwidth for control message exchange. Thus it seems that when bandwidth becomes the bottleneck and thus the messages reach the subscribers after a long delay, since the system does not receive an acknowledgment from the subscriber in a *short* time it resends the same message number of times before it assumes that the subscriber is temporarily disconnected. This wastes a substantial amount of bandwidth besides increasing the size of the queue on the wireless channels and hence the corresponding delays in the receipt of the messages by the subscribers and receipt of acknowledgements by the broker.

Figure 4.1(c) shows the proportion of messages lost. In the fixed wireline case no messages are lost. But in the mobile case as the subscribers are not always connected to the broker, they may lose messages whenever they disconnect from the network. Hence

in the mobile server bottlenecked region, almost 7.5% of messages are lost which is equal to the percentage of the total time the subscribers are disconnected from the network, confirmed theoretically by solving the markov chain resulting from the mobility model. However, in the bandwidth bottlenecked region message loss is significantly higher than that. The reason for extra loss is the flooding of wireless channels. The broker sends a message to a subscriber and waits for the acknowledgement. But as the bandwidth is the bottleneck there is a substantial queue of messages on the wireless channels. As a result, the message reaches the subscriber after a long delay. During this period, as the broker does not receive an acknowledgement from the subscriber, after some time it considers the subscriber as temporarily disconnected from the network (or assumes the network link to be dead) and hence discards all the next incoming messages for that subscriber. When it finally gets back the acknowledgement it again starts sending message to that subscriber. Moreover, sometimes the broker sends messages when the subscriber is connected, but during the message transmission delay on the wireless channels, the subscriber disconnects and loses all the messages in transit.

The results obtained with the closed model publisher (given in Appendix A.1) are similar to those obtained with an open model publisher for each of the three test environments. However, for a given environment the closed model produces significantly lower throughputs in comparison to the open model. The reason is that, since the closed model publisher publishes to the two brokers in a loop, an overload on one of the brokers can prevent messages from being published to the other broker. The maximum publisher throughput X_P thus decreases which in turn decreases the subscriber throughput X_S . This has also been discussed in detail in Section 3.1.2.1.

4.2.1.1 Conclusions

The discussion presented in the previous subsection shows that in the mobile case when the server is the bottleneck, current implementation of JMS provides a subscribers throughput that is close to that in the fixed wireline networks but it suffers from bursts of message losses as the subscribers lose all messages published during the time they were disconnected from the network. This message loss in bursts may not be acceptable to a number of applications. Nondurable subscriptions are not supposed to deliver messages published during the time the subscriber was disconnected from the network, but there should at least be a way to prevent message loss occurring due to user mobility and due to temporary disconnections in weakly connected wireless environments.

Current implementations give poor performance in low bandwidth environments. They give message throughput lower than the actual message carrying capacity of the channels while the message loss is order of magnitude higher than that in higher bandwidth environments. One of the reasons it seems is that the system does not take into account potential large queues on wireless channels and resend the message without waiting for a long enough period of time necessary for low bandwidth wireless environments. A longer waiting interval would significantly reduce duplicate traffic on wireless channels and would result in an increase in system throughput and decrease in the percentage message loss.

4.2.2 Effect of Frequency of Handoffs

Experiments were conducted to find out the effect of frequency of handoffs (f_H), on the performance of nondurable subscriptions. But before describing the experimental results, let us briefly review the effect of f_H on performance measures. Let μ_p be the rate

at which messages are being published per second on a particular topic and selector range. If a subscriber interested in receiving messages in that selector range disconnects from the broker for an interval of t seconds, then the average number of messages lost by the subscriber during that interval will be $\mu_p * t$. If T_{SD} is the total time the subscriber is disconnected from the network (both due to handoffs and poor wireless channels) during the course of the experiment, then the total number of messages lost by the subscriber will be $\mu_p * T_{SD}$. Now if T is the total time of the simulation then the percentage message loss for nondurable subscriptions L_{ND} can be given as:

$$\begin{aligned} L_{ND} &= \mu_p * T_{SD} * 100 / (\mu_p * T) \\ \Rightarrow L_{ND} &= (T_{SD} / T) * 100 \end{aligned} \quad (4.1)$$

$$\text{Now } T_{SD} = f_H * T * t_{SD}$$

Thus we have,

$$L_{ND} = f_H * t_{SD} * 100 \quad (4.2)$$

This shows that the message loss percentage in nondurable subscriptions is directly proportional to both mobility parameters f_H and t_{SD} . Subscriber throughput X_S for nondurable subscriptions is inversely proportional to f_H as each connection/disconnection request generates extra load on the server.

To experimentally study the effect of f_H , the proportion of total time in signal down state T_{SD}/T is varied keeping t_{SD} constant. Default values of workload parameters presented in Table 3.1 are used for the fixed parameters in these experiments and are listed in Table 4.2. Results are shown in Table 4.3 and Figure 4.2. In these experiments, the maximum publisher throughput X_P shown in Figure 4.2(a) depends on three factors: connections/disconnections overhead, average number of subscribers connected to the

broker and the maximum instantaneous number of subscribers connected to the broker. The higher the connection/disconnection requests per second the lower the X_P . Similarly, X_P decreases with the increase in both the average and the maximum instantaneous number of subscribers connected to any one of the brokers. Although when the number of subscribers on one of the brokers increases, the number of subscribers on the other broker may decrease, the broker having higher load of subscribers limits the maximum sustainable publisher throughput of the system. This is because the broker with the higher load of subscribers cannot handle as many messages per second as the broker with the lower load of subscribers. It thus fails and causes the publisher's buffer to overflow.

Table 4.2: Input Parameters for Investigating the Effect of Frequency of Handoffs on Nondurable Subscriptions

Parameter	Value
Publisher Model, Publishing Rate	Open, Maximum Sustainable
Length of the Message Body (S)	50 Bytes
Mobility, Bandwidth Environment	Mobile, Server Bottlenecked
Mean Length of Signal Down Interval (t_{SD})	12 seconds
Probability Distribution of Signal Down Intervals, C_v	Exponential, 1

Thus Figure 4.2(a) shows that when T_{SD}/T is increased from zero (fixed subscribers) X_P decreases sharply at first. One of the reasons is the overhead due to connections/ disconnections. Also for non-zero T_{SD}/T (and hence f_H) the subscribers move from one broker to the other and hence during the course of the simulation the maximum instantaneous number of subscribers on a particular broker may temporarily become greater than the total number of subscribers connected per broker (the number of

subscribers per broker when f_H was zero). The effect of both of these factors, connections/disconnections overhead and higher maximum instantaneous number of subscribers, decreases X_P . This explains the initial dip in the X_P graph.

Table 4.3: Effect of Frequency of Handoffs on Nondurable Subscriptions
 $(f_H = ((T_{SD}/T)*100)*(1/(100*t_{SD}))); t_{SD} = 12 \text{ seconds})$

$(T_{SD}/T)*100$	Max. Publisher Throughput (X_P) (messages/sec)	Subscriber Throughput (X_S) (messages/sec)	Percentage Message Loss
0.0	340.38	340.38	0
0.2	280.14	279.54	0.215
1.0	303.35	300.19	1.04
2.5	312.03	303.96	2.59
5.0	327.99	310.25	5.41
7.5	352.65	325.4	7.73
10.0	369.52	331.07	10.40
15.0	388.98	328.43	15.73

(a)

Figure 4.2: Effect of Frequency of Handoffs on Nondurable Subscriptions
 $(f_H = ((T_{SD}/T) \cdot 100) \cdot (1/(100 \cdot t_{SD}))); t_{SD} = 12 \text{ seconds})$
(a) Maximum Throughput of Publisher (b) Throughput of Subscribers (c) Proportion of Messages Lost

However, as T_{SD}/T increases further, the average number of subscribers per broker decreases which decreases the load on the broker. The average number of subscribers N connected per broker during the experiments is given by

$$N = (1 - T_{SD}/T) \cdot \text{Total Number of Subscribers per broker} \quad (4.3)$$

Hence when T_{SD}/T is small, average number of subscribers per broker is almost equal to the total number of subscribers per broker. But as T_{SD}/T increases, the average number of subscribers per broker decreases. At higher values of T_{SD}/T , the effect of decrease in the

average number of subscribers per broker becomes more dominant than any of the other factors and thus X_P tends to increase with T_{SD}/T . Hence in Figure 4.2(a) after an initial dip, X_P keeps on increasing with the increase in T_{SD}/T .

The subscriber throughput X_S shown in Figure 4.2(b) follows the same pattern until a certain point, after which it almost becomes constant and finally tends to drop a little. The reason is that, for small values of T_{SD}/T , X_S is almost equal to X_P as percentage message loss is small; hence both curves look the same. With the increase in T_{SD}/T the average number of subscribers per broker decreases and this decrease in load on the broker is reflected as an increase in both X_P and X_S . But as T_{SD}/T increases further, message losses increases (as the subscribers are now disconnected from the network for a longer period of time), which increases the difference between X_P and X_S . Thus to deliver the same number of messages to the subscriber, the broker needs to accept more messages from the publisher and perform message selection. This increases the load on the broker. Thus, the benefit gained by the decreasing number of subscribers is offset by this extra load and hence the subscriber throughput X_S does not increase further. It eventually tends to decrease slowly as the publisher's load increases further.

The percentage message loss graph, given in Figure 4.2(c), shows that for nondurable subscriptions, percentage message loss increases linearly with a slope of 1, with the increase in T_{SD}/T , thus matching the theoretical prediction from Equation 4.1.

4.2.2.1 Conclusions

The above discussion shows that frequency of handoffs has a profound effect on the performance of nondurable subscriptions. When frequency of handoffs is varied as a parameter, the overall subscriber throughput is determined by factors such as

connections/disconnections overheads, effective number of subscribers connected to the broker and the increased load of message acceptance and selection as the percentage message loss increases.

The percentage message loss increases with the increase in f_H due to the corresponding increase in the proportion of the time the subscribers are disconnected.

4.2.3 Effect of Variability in Signal Down Intervals

The effect of variability in signal down intervals on the performance of nondurable subscriptions is studied using a bi-phase hyper exponentially distributed signal down intervals. The co-efficient of variation C_V of the distribution is varied as an input parameter. Default values of workload parameters presented in Table 3.1 are used for the fixed parameters in these experiments and are listed in Table 4.4. Although the Markov chain analysis breaks down for hyper-exponentially distributed signal down intervals, the value of T_{SD} measured experimentally during the experiments was found to be very close to the one predicted by the Markov chain analysis. The results obtained are shown in Table 4.5 and Figure 4.3.

Figure 4.3(a) and Figure 4.3(b) show that as C_V increases from 1, both the maximum publisher throughput X_P and subscriber throughput X_S drop a little. The reason is that as C_V increases, lengths of signal down intervals vary from the mean significantly. The result is the presence of signal down intervals some of which have very short durations and the others have very long durations. Due to this, the instantaneous number of connection/disconnection requests as well as the instantaneous number of subscribers connected per broker differ significantly from their average values. At one time the broker may be serving a number of subscribers much larger than the mean while at other

times it may be serving a much smaller number of subscribers. As the maximum sustainable throughput of the broker decreases with the increase in the number of subscribers, instants at which the broker has large number of subscribers connected to it, limits the maximum throughput. The result is a slight decrease in throughput with the increase in C_V . However, this decrease in throughput with C_V is not linear and the curve levels off for higher C_V values.

Table 4.4: Input Parameters for Investigating the Effect of C_V of Signal Down Intervals on Nondurable Subscriptions

Parameter	Value
Publisher Model, Publishing Rate	Open, Maximum Sustainable
Length of the Message Body (S)	50 Bytes
Mobility, Bandwidth Environment	Mobile, Server Bottlenecked
Mean Length of Signal Down Interval (t_{SD})	12 seconds
T_{SD} / T	0.0742

Table 4.5: Effect of C_V of Signal Down Intervals on Nondurable Subscriptions

C_V	Max. Publisher Throughput (X_P) (messages/sec)	Subscriber Throughput (X_S) (messages/sec)	Percentage Message Loss
1.0	344.87	319.18	7.45
1.5	337.23	311.03	7.77
2.0	334.15	308.02	7.82
2.5	334.46	308.20	7.85

Figure 4.3: Effect of C_v of Signal Down Intervals on Nondurable Subscriptions
(a) Maximum Throughput of Publisher (b) Throughput of Subscribers (c) Proportion of Messages Lost

One may think that the instants at which the broker has a smaller number of subscribers connected to it may increase X_p . However, since an open model of the publisher was used the maximum sustainable publishing rate is limited by the instants when the broker has the maximum load on it. During the instants when the broker has lesser load of subscribers, it is utilized less than its maximum capacity.

The percentage message loss graph in Figure 4.3(c) shows that the percentage message loss increases slightly with the increase in C_v . The reason is that as C_v increases at times the broker has a number of connection/disconnection requests to handle while at other times it does not have any. At times when the broker has large volume of connection requests to handle, it takes longer than average for the broker to start sending messages to the incoming subscriber. The subscriber thus also loses messages published during this additional delay. This additional delay was measured experimentally during the simulation and the results agreed with the expectation. Like throughput, the difference in message loss percentage between two successive points on the graph decreases as C_v increases.

4.2.3.1 Conclusions

The probability distribution of signal down intervals seems to affect the system throughput and percentage message loss slightly. The throughput decreases while percentage message loss increases as the C_v of signal down intervals increases. However, the effect diminishes for higher values of C_v .

4.2.4 Effect of Message Size

To study the effect of message size on the performance of nondurable subscriptions, the size of the message body (S) is varied keeping the size of message header and properties constant. In other words, the size of the contents of the message is used as a parameter. Default values of workload parameters presented in Table 3.1 are used for the fixed parameters in these experiments and are listed in Table 4.6. The results obtained are given in Table 4.7 and Figure 4.4.

Table 4.6: Input Parameters for Investigating the Effect of Message Size on Nondurable Subscriptions

Parameter	Value
Publisher Model, Publishing Rate	Open, Maximum Sustainable
Mobility, Bandwidth Environment	Mobile, Server Bottlenecked
Mean Length of Signal Down Interval (t_{SD})	12 seconds
Probability Distribution of Signal Down Intervals, C_v	Exponential, 1
T_{SD} / T	0.0742

Table 4.7: Effect of Message Size on Nondurable Subscriptions

Message Size (S) (Bytes)	Max. Publisher Throughput (X_P) (messages/sec)	Subscriber Throughput (X_S) (messages/sec)	Percentage Message Loss	Information Delivery Rate (kB/sec)
50	344.87	319.18	7.45	15.58
100	339.13	313.66	7.51	30.63
200	331.35	306.53	7.49	59.87
500	320.25	296.30	7.48	144.68

(a)

(b)

(c)

Figure 4.4: Effect of Message Size on Nondurable Subscriptions
(a) Maximum Throughput of Publisher (b) Throughput of Subscribers (c)
Proportion of Messages Lost (d) Maximum Delivery Rate of Information

Figure 4.4(a) and 4.4(b) show that the system throughput decreases with the increase in message size. The reason is that the broker has to do more work on larger messages as compared to smaller ones. This includes receiving the message from the publisher, storing it in a buffer, delivering it to all interested subscribers and then reclaiming the space from the buffer in memory. Also the size of heap memory increases with the increase in message size and the broker may need to do garbage collection more often than with small messages. The effect of all these activities reduces both the maximum sustainable publisher and subscriber throughput. Moreover, for large messages there is more contention for resources on the network and also the message carrying capacity of the channel decreases with the increase in message size. Thus the network too plays its role in bringing down the system throughputs with the increase in message size. Figure 4.4(c) shows that the message size has no effect on message loss percentage, as it depends solely on the proportion of time the user is disconnected from the network (provided the bandwidth is not the bottleneck).

Figure 4.4(d) shows the effect of message size on the maximum sustainable information delivery rate of the broker. Information delivery is given by

$$\text{Information delivery rate} = S \text{ (in kB)} * X_S \quad (4.4)$$

It reflects the amount of application level data delivered by the broker per second. The figure shows that information delivery rate increases almost linearly with the increase in message size. However, every 100% increase in message size increases the information delivery rate by 93%.

4.2.4.1 Conclusions

In the server bottlenecked region, increasing the message size decreases the system throughput as the broker needs to do more work on large messages. However, the information delivering capacity of the broker increases with the increase in message size suggesting that if possible information should be batched together and then sent as a larger message. In the bandwidth bottlenecked region too, using large messages would increase the information delivering capacity. This is because sending more message contents in the same message saves bandwidth by reducing the overhead of the JMS and lower layer headers of several smaller messages. However, as discussed in Section 4.2.1 the JMS implementation perform extremely poorly in low bandwidth environments; until it is modified to deal with low bandwidth users more efficiently, increasing the message size in bandwidth bottlenecked region will have a worse effect on overall system performance. In the experiments conducted with larger message sizes in low bandwidth regions the resending of messages by the broker (as discussed in detail in Section 4.2.1) practically brought down the subscriber throughput to almost zero and the message loss to almost 90%.

4.3 Performance of Durable Subscriptions

Durable subscriptions are highly reliable and are used for applications that cannot afford to miss messages. Even if a durable subscriber is disconnected from the broker, it does not lose messages, as the broker preserves all the messages published during the time it was disconnected. As soon as the subscriber returns, the broker sends all the stored messages. However, as discussed in Section 2.3.3.2, as now the broker has to store messages for the inactive subscribers and send them as they reconnect, there is a substantial overhead imposed on the broker. As a result the throughput of durable subscriptions is very low in comparison to nondurable subscriptions. The system throughput degrades rapidly with increasing number of inactive subscribers and/or connections/disconnections. Reports such as [37] show the effect of different parameters on the performance of durable subscriptions. If the results are compared to the corresponding results for nondurable subscriptions, it shows that system throughput of durable subscriptions can be as low as 25% of that for nondurable subscriptions. The low throughput of durable subscriptions suggests that even in the fixed networks, durable subscriptions should be restricted to applications that cannot afford to miss a message, and for small number of subscriptions.

4.3.1 Challenges for Durable Subscriptions in Mobile Environments

In the absence of any support service, durable subscriptions would face many challenges in a mobile environment in addition to the fact that frequent connection/disconnections would degrade their performance severely. Major problems for durable subscriptions in a mobile environment are discussed.

4.3.1.1 Message Loss

First of all, in the absence of any additional mechanism, mobile subscribers will lose messages even with durable subscriptions. Consider for example, a mobile durable subscriber which subscribes for some messages at Broker A, then disconnects from the network for some reason and comes back at a new broker, Broker B. Broker A will keep storing all the messages for the subscriber, but as Broker B does not have any prior knowledge of subscriber's durable subscriptions, as the subscriber connects to Broker B, it will not be delivered the messages published during the time it was disconnected from the network. Thus, the subscriber will end up losing messages even with durable subscriptions.

4.3.1.2 Overheads of Inactive Subscriptions

In the above scenario, as Broker A has no information that the subscriber has now subscribed to some other broker, it will keep storing messages for that subscriber costing the broker a substantial performance overhead. In dynamic mobile environments, there can be many subscribers which subscribe at a certain broker but lose their connection due to poor wireless channels and emerge at some new broker without canceling their subscriptions at the previous broker. Each broker will thus end up having a number of inactive subscribers that may never reconnect to that broker. As storing messages for inactive subscribers imposes a lot of performance overhead, it was observed during preliminary experiments that with the increasing number of inactive subscribers in a mobile environment the performance of broker become increasingly worse to the point of failures.

4.3.1.3 Duplicate Messages

Now suppose that the subscriber in the above scenario, after receiving certain number of messages from Broker B (and possibly from some other brokers) comes back to the domain of Broker A and reconnects to it. Since Broker A has no information about the messages the subscriber has received from other brokers, it will send all the messages it started storing from the time the subscriber disconnected from it. This will not only cause the flooding of wireless channels with messages, but also the subscriber will end up receiving duplicate messages. Duplicate messages can be a serious problem for applications which depend on certain events count. For example, consider a scenario in which a message is generated each time a goal is scored. An application keeping track of the number of goals scored by counting the number of messages it received may end up counting a single goal scored twice or more in a mobile environment with durable subscriptions.

Sometimes, the order of messages becomes important too. For example, in case of stock information delivery, the latest rate of a certain stock is the one sent in the latest message on that stock. In our example, the mobile subscriber may wrongly assume the latest rate of a certain stock as the one given in the duplicate message (which can be hours old) it received from Broker A.

4.3.2 Extending Durable Subscriptions for Mobile Environments

As discussed in the Section 4.3.1, durable subscriptions in their current form are not practical for mobile users. Solutions that can be considered to alleviate the problems durable subscriptions face in the mobile domain are presented next.

4.3.2.1 Clustering

One of the ways to support durable subscriptions in mobile environments is to use clusters of brokers. Many implementations of JMS (such as SonicMQ) support broker clustering. A cluster is a group of interconnected brokers in which each broker can directly communicate with the other brokers in the cluster. Each cluster of brokers has a configuration server that manages the subscriptions of the cluster as a whole. Thus, if a user loses its connection to a broker or the broker fails, all messages for the user can be redirected to another broker in the cluster. Hence if a mobile user subscribes for durable subscriptions at one of the brokers in the cluster, disconnects and then reconnects after some time to some other broker in the same cluster, the configuration sever can redirect all the messages published during the time the subscriber was disconnected, to the new broker. The subscriber will thus be delivered all the messages it missed by the new broker.

The cluster of brokers hence can support durable subscription in mobile cellular networks. However, the scope of clustering is limited to a small scale, as the whole world cannot be clustered together. This is because the configuration server becomes the performance bottleneck as the number of brokers and subscribers in a cluster increases. This decreases the maximum sustainable throughput of the system. In addition to the decrease in throughput, there might be some compatibility, commercial and proprietary issues involved in broker clustering. Hence, although broker clustering can be used to support durable subscriptions on a smaller scale (more of a LAN like environment) for mobile users, it does not seem to be practical on a global scale.

4.3.2.2 Mobile IP

Another way of supporting durable subscriptions in a mobile environment is making user movement transparent to the broker using Mobile IP (or IPv6). The user subscribes to only one broker and this broker instead of forwarding messages directly to the user, forwards it to the user's home agent. The home agent then forwards the message to the foreign agent in the user's current location, which delivers it to the user. Alternatively, the broker can be informed whenever the user's IP address changes and the broker can then redirect all the messages to the user's new IP address.

The drawback of such a solution is the traffic overhead involved due to triangular routing of messages. The triangular routing of messages can not only congest the network at various points but will also result in an additional delay in the delivery of every message. Moreover, such a solution cannot support emerging location aware services, where the message sent to the user can depend on its current location. For example, suppose a subscriber has durable subscriptions for *Traffic Updates*. It will be more interested in receiving the traffic conditions in its current or neighboring locations than the traffic updates of its home location where the broker to which it has durable subscription for *Traffic Updates*, is located. On the other hand, if the subscriber has the same subscription to one of the brokers in its current location, it would be able to receive traffic updates of its neighboring locations, assuming that each broker receives messages about traffic updates only of its neighboring locations. It does not make sense to send traffic updates of all locations to all the brokers in the world.

4.3.2.3 Middleware Level Handoffs

Durable subscriptions can also be supported in the mobile environment if there is a middleware level handoff procedure that takes place whenever the subscriber moves from one broker to the other. The handoff protocol between the two brokers should be capable of delivering all the missed messages to the subscriber. The new broker to whom the subscriber connects should also inform the previous broker to stop storing messages for the subscriber.

The advantage of such a procedure is that it can be used on a global scale as well as it can support location aware services. Moreover, there is no triangular routing of messages. Middleware level handoffs are studied in detail in Chapter 5 to confirm their efficacy in mobile environments.

Chapter 5

Middleware Level Handoffs for Durable Subscriptions: Design, Implementation and Evaluation

This chapter describes the design, prototype implementation and evaluation of middleware level handoffs for durable subscriptions. Performance of durable subscriptions with middleware level handoffs is also compared to that of nondurable subscriptions.

5.1 Objectives

Major goals of this chapter can be outlined as follows:

- To explore different design possibilities for middleware level handoffs and motivate the selection of one of them.
- To describe a prototype implementation of middleware level handoffs.
- To present the evaluation of middleware level handoffs so as to determine their suitability for mobile wireless environments.
- To compare their performance with nondurable subscriptions.
- To understand the challenges middleware level handoffs face in mobile wireless environments and propose solutions for them.

5.2 Design and Implementation of Middleware Level Handoffs

5.2.1 Motivation

Chapter 4 gives an evaluation of the current implementation of JMS for mobile wireless domain. The evaluation shows that nondurable subscriptions suffer from bursts of message losses in mobile wireless environments. In order to prevent message loss in mobile wireless environments durable subscriptions need to be used instead of nondurable ones. However, in their current form, durable subscriptions in mobile environments face challenges like message loss, message duplication and severe performance degradation. Several methods can be considered to extend durable subscriptions to the mobile domain (discussed in Section 4.3.2) of which middleware level handoffs seems to be the most suitable.

5.2.2 Design Issues

While designing middleware level handoffs for mobile durable subscribers several issues should be taken into account in order for the design to work reliably and efficiently. First of all, the design of middleware level handoffs should be able to deal with non-graceful conditions where the subscriber disconnects before the actual handoff between the two involved brokers takes place. This is necessary in unreliable wireless environments where a subscriber can suddenly lose signal. Secondly, the design of middleware level handoffs should ensure ordered and duplicates free delivery of messages. The procedure should be implemented at the middleware level and all the implementation details should be hidden from the application. This would ensure that a

client's application that is developed for a fixed environment need not be modified for the mobile domain, making the integration of fixed and mobile domains seamless.

5.2.3 Design and Implementation

Several designs of middleware level handoffs are possible. The design used in this thesis takes into account the issues discussed in Section 5.2.2. It works for non-graceful conditions and gives ordered and free from duplicates delivery of messages. As the source code of the JMS implementation was not available, the prototype implementation of middleware level handoffs required some minor modifications in the subscribers' applications. However, this is not the condition imposed by the design. If one were to have the source code of the JMS implementation, the details could be hidden from the applications.

Figure 5.1: Middleware Level Handoffs for Durable Subscriptions

Figure 5.1 shows the middleware level handoff protocol used in this thesis. An additional piece of software called *Handoff Manager* is run with each broker to manage users' mobility from one broker to the other. The subscribers keep track of the last

broker to which they subscribed for durable subscriptions in a log. When a subscriber disconnects from the network and then reconnects after some time, instead of contacting the broker directly it first connects to the Handoff Manager. The sequence of interactions that takes place is depicted in Figure 5.1. The description of the operations follows.

In step 1.1, the subscriber passes information such as its client id, its durable subscriptions, address of the last message broker it subscribed for durable subscriptions, as a part of its subscription request. The Handoff Manager on behalf of the subscriber connects to the broker where the subscriber last subscribed for its durable subscriptions and receives all the messages saving them in a queue (Steps 1.2 to 1.5). After it has received all the messages it unsubscribes the subscriber from that broker so that the broker no longer stores messages for that subscriber. At the same time the Handoff Manager subscribes to its local broker for the same subscriptions and save all the messages published during the time the subscriber is connected to it in a separate queue (Step 1.2a and 1.2a.1). In the end (Step 2.1), the Handoff Manager merges the two queues removing duplicates and then delivers the resulting messages to the subscriber. Finally the subscriber disconnects from the Handoff Manager and starts receiving messages from the broker as usual (Step 2.2).

In this design, saving the address of the broker in a subscriber's log ensures that the handoff procedure works even if the subscriber disconnects from the network before the actual handoff takes place. While merging the two queues the Handoff Manager removes duplicates and arranges the messages in their publishing order. This ensures ordered and free from duplicates delivery of messages.

5.3 Evaluation of Middleware Level Handoffs

5.3.1 Effect of Mobility

The performance of durable subscriptions with middleware level handoffs is evaluated for mobile wireless domains (server bottlenecked region) and is compared to the performance of durable subscriptions in fixed networks. As discussed in Section 3.1.2.1, both closed and open system models of the publisher are used. The results with the closed model publisher are given in Appendix A.2 while the results with the open model publisher are discussed in this section. Default values of workload parameters presented in Table 3.1 are used for the fixed parameters in these experiments and are listed in Table 5.1.

Table 5.1: Input Parameters for Investigating the Effect of Mobility on Durable Subscriptions

Parameter	Value
Publisher Model, Publishing Rate	Open, Maximum Sustainable
Length of the Message Body (S)	50 Bytes
Mean Length of Signal Down Interval (t_{SD})	12 seconds
Probability Distribution of Signal Down Intervals, C_v	Exponential, 1
T_{SD} / T	0.0742

(a)

Figure 5.2: Effect of Mobility on Durable Subscriptions
(a) Maximum Throughput of Publisher (b) Throughput of Subscribers (c)
Proportion of Messages Lost

Results obtained are shown in Figure 5.2. The throughput graphs (Figures 5.2(a) and 5.2(b)) clearly show that durable subscriptions perform poorly in mobile cellular networks compared to fixed networks. The subscriber throughput X_S in mobile wireless environments is 11% of that in fixed networks. The reason is the overhead incurred in keeping track of and storing messages for the inactive subscribers as well as the overhead of delivering the stored messages during handoffs. Since most of these overheads are incurred by the broker, other designs of middleware level handoffs are also expected to produce a low performance in comparison to nondurable subscriptions. Integrating the

Handoff Manager with the broker into a single entity may give slightly better performance but the overall behavior would remain the same. Actually, in the fixed networks, the number of connection/disconnection requests is much lower than that in mobile wireless environments. High frequency of connection/disconnection requests and the resulting higher number of inactive subscriptions and handoffs makes durable subscriptions almost impractical in mobile networks from a performance point of view.

The surprising result is the message loss for durable subscriptions in the mobile environment (Figure 5.2(c)) as durable subscriptions are supposed to be loss free. However, this contradiction is a result of the way in which the Handoff Manager was implemented. The Handoff Manager uses the synchronous `recv(long timeout)` method, a part of the standard JMS API, to retrieve all the stored messages from the broker the user was previously connected to. The timeout parameter is the time in milliseconds the calling function should wait before it times out in case it doesn't receive the message during that interval. When the broker is not overloaded even a small value of timeout ensures that the broker has delivered all the stored messages and there are no other messages to be sent. However, if the broker is heavily utilized, sometimes it can take a long time before serving the Handoff Manager's request. In this case, if a small timeout is used, the `recv()` method may timeout while there are more stored messages to be sent and hence some of the messages are lost. On the other hand, setting this parameter to a large value may make the calling loop infinite as the broker may have delivered all the stored messages but the following calls to `recv()` method get the message(s) published during the timeout interval. Having the source code of JMS

implementation, such problems can easily be handled and we can then expect the message loss for durable subscriptions with middleware level handoffs to be zero.

The results obtained with the closed model publisher, given in Appendix A.2, show that even with the closed model of the publisher the relationship between the results obtained for two different mobility environments is similar to that between the results obtained with an open model publisher. However, as seen in Section 4.2.1, the closed model publisher gives lower X_P and X_S than an open model.

Durable subscriptions were not studied for the bandwidth bottlenecked regions as the broker throughput itself is too low to make even a small bandwidth the bottleneck.

5.3.1.1 Conclusions

Although durable subscriptions with middleware level handoffs can successfully bring down the message loss occurring in mobile wireless environments their throughput in such environments is much lower than in fixed networks. Low throughput of durable subscriptions in mobile environments can be attributed to tight coupling between the subscriber and the broker. In fixed networks where the subscriber usually connects to the same broker, a tight coupling may not degrade performance, but in highly mobile environments this introduces a substantial overhead. Each time a subscriber moves from one broker to the other, it needs to be uncoupled from the previous broker. Durable subscriptions thus make a poor choice for mobile environments.

5.3.2 Comparison with Nondurable Subscriptions

This section compares the performance of durable subscriptions with handoffs with the performance of nondurable subscriptions, in mobile wireless environments.

Figure 5.3: Comparison of Nondurable and Durable Subscriptions
(a) Maximum Throughput of Publisher (b) Throughput of Subscribers (c) Proportion of Messages Lost

Results, obtained in Section 4.2.1 and 5.3.1, are compared in Figure 5.3. Figure 5.3(b) shows that in fixed networks, durable subscriber throughput X_S is 44% of that of nondurable subscribers. Keeping in view the ability of durable subscriptions to prevent message losses due to subscribers' disconnections, it may be worth the 56% reduction in throughput in some situations. However, when X_S of the two types is compared for mobile users, durable throughput is drastically low. It is merely 5% of that for nondurable subscriptions. This suggests that although durable subscriptions with middleware level handoffs may be used for applications that cannot afford to miss a single message, they certainly are not the right choice as a general purpose subscription mechanism for mobile wireless users.

Figure 5.3(c) shows that durable subscriptions with middleware level handoffs bring down the message loss from approximately 7.5% to approximately 2%. Moreover, as discussed earlier in Section 5.3.1, integrating the implementation of middleware level handoffs with the broker will bring the message loss for durable subscriptions to 0.

5.3.3 Effect of Mobility Parameters

Keeping in view the performance of durable subscriptions in mobile environments, it was not worth while to study in detail the effect of different mobility parameters on their performance. The throughput is too low to generate any interesting and useful results.

Results of some preliminary experimentation shows that as the number of subscribers and/or the frequency of handoffs increases, performance of durable subscriptions degrades almost exponentially, until the capacity of the broker is exceeded. Long signal down intervals also affect the performance adversely.

5.3.4 Race Conditions in Durable Subscriptions with Middleware Level Handoffs

In addition to severe performance degradation durable subscription with middleware level handoffs faces challenges such as race conditions in mobile environments. Race conditions are discussed briefly.

Due to the geographically distributed nature of brokers, it is possible that network congestion or any other similar phenomenon makes messages reach certain brokers later than others. Consider a broker A which receives messages later than another broker B. If a subscriber handoffs from broker A to B, the Handoff Manager will first deliver all the messages stored by the broker A and then the subscribers would receive messages from broker B. It is possible that some messages were published between the last message forwarded by broker A and the first message sent by broker B. The subscriber would thus lose those very messages. Theoretically, race conditions thus prevent durable subscriptions from guaranteeing loss free delivery of messages in a mobile environment. Practically, however the message loss may be rare.

Chapter 6

Semi-Durable Subscriptions: A Technique to Achieve High Performance in Mobile Wireless Publish/Subscribe Systems

This chapter introduces a new type of subscription mechanism called semi-durable subscriptions to counter the problems faced by nondurable and durable subscriptions in mobile environments. The chapter discusses its design, implementation and performance comparison with other subscription mechanisms. The effect of different mobility parameters on its performance has been studied. This work has been published as [28] and submitted for publication as [27].

6.1 Objectives

The goals of this chapter are outlined:

- To introduce a novel technique, semi-durable subscriptions, to tackle the challenges imposed by mobile wireless environments.
- To present the design and implementation of its prototype.
- To compare its performance with other subscription mechanisms and hence answer whether it makes a good alternative for mobile wireless environments.
- To study the effect of different mobility parameters on its performance.
- To explore the system parameter space for semi-durable subscriptions and show how it can be controlled to configure the system to manage given environments and/or to get the desired characteristics.

6.2 Introduction to Semi-Durable Subscriptions

6.2.1 Motivation

The discussion in Chapter 4 shows both nondurable and durable subscriptions are not suitable for mobile wireless domain. Nondurable subscriptions cannot prevent message loss occurring due to temporary disconnections of the subscriber during handoffs and due to poor wireless channels, whereas durable subscriptions in their current form cannot be deployed in mobile environments. Chapter 5 shows that although middleware level handoffs with durable subscriptions can prevent message loss, their subscriber throughput in mobile environments can be as low as 5% of that with nondurable subscriptions. Such a high cost-to-benefit ratio is unavoidable for the sensitive applications such as air traffic control that cannot afford to miss a single message. However, there are many applications where occasional message loss is tolerable (e.g. newsgroups, auction systems or even stock information delivery). Relying on durable subscriptions for disseminating information for such applications is too expensive from a performance perspective. On the other hand, nondurable subscriptions cannot give the desired behavior either, as frequently occurring message losses in bursts are not tolerable. Under these conditions, we need a new technique which is the mid way between the two extremes. It may incur occasional message loss but should give high performance. Moreover, it should be possible to configure the system any where between the two extremes (no message loss or high performance).

6.2.2 Challenges for the Proposed Technique

The new technique must have the following characteristics to fulfill its goals.

1. The new technique should prevent message loss occurring during temporary disconnections of the subscribers in mobile wireless environments. But at the same time its performance cost should not be too high.
2. The poor performance of durable subscriptions can be attributed to the tight coupling between the broker and the subscriber. The new technique thus should ensure that no such tight coupling exists between the broker and the subscribers and the subscribers are free to disconnect any time due to any reason without a need to uncouple themselves from the broker.
3. The new technique should not only work under graceful conditions but should also allow for non-graceful termination of connections between the subscribers and the broker without incurring message loss and/or substantial performance overheads
4. The new technique should support emerging location aware services. It should not incur any network overheads such as triangular routing.
5. It should ensure ordered and free from duplicates delivery of messages. It should also eliminate message loss due to race conditions.
6. It should provide rich system parameters space so that the system can be configured according to the desired characteristics and/or to manage particular needs of the mobility environment.
7. All the details of the new technique should be transparent to the applications so that the same application can be used for both fixed and mobile subscribers, making seamless integration of fixed and mobile domains possible.

6.2.3 Solution to the Design Challenges: Semi-Durable Subscriptions

In order to tackle the challenges mentioned above, this thesis introduces semi-durable subscriptions. All the mobility extensions of publish/subscribe systems discussed in Chapter 2 involve uncoupling and retrieval of messages from the previous broker the user was connected to in one form or the other (such as in middleware level handoffs) and hence incur substantial performance overheads in dynamic environments. The semi-durable subscriptions technique on the other hand tends to recover the messages lost from the local buffer at the new broker. This assumes that the messages missed by the subscriber are available at the new broker. Thus, in semi-durable subscriptions each broker buffers all the messages published irrespective of its current active subscriptions. On the arrival of a connect request the broker sends all the messages missed by the subscriber from its buffer. The Message ID of the last message the subscriber received is transmitted as a part of the subscription request and the broker sends messages published after the message with this ID.

Even if we assume that not all the messages are published to all the brokers in the system, semi-durable subscriptions can be extended to deal with this phenomenon of regionalism. It is discussed in detail in Section 6.8. However, research papers such as [49] show that in highly mobile environments, due to the practical limit on the number of multicast addresses and the corresponding overheads, the most feasible strategy is to broadcast all messages to all the brokers in the publish/subscribe system. Thus if a broker buffers messages irrespective of its current active subscriptions, messages will be available to be sent in case a new subscriber moves from another broker to this one.

The advantage of this message buffering at each broker is that it maintains the loose coupling between the subscriber and the broker. The broker thus does not incur overheads in maintaining and managing the list of its inactive subscribers. Moreover, the subscriber can now disconnect from the broker without a need to be uncoupled and the handoff of subscriber now involves only one broker.

The disadvantage is that that unlike middleware level handoffs the subscriber can now occasionally lose messages. This is because each broker can buffer only limited number of messages due to the practical limits on the size of the buffer. If we assume that each broker stores only the most recently published messages then if the subscriber connects after an interval long enough to overwrite the contents of the semi-durable buffer it will end up losing messages (as some of the messages it missed have been overwritten). However, it is worth mentioning here that semi-durable subscriptions are meant to deal with message losses occurring due to temporary disconnections of the subscribers in mobile wireless environments, they are not an alternative to durable subscriptions. As the results discussed in the later sections of this chapter show, a moderate buffer size can recover almost all of the messages missed by the subscriber due to temporary disconnections in a mobile environment even if we assume a very high publishing rate and fairly long disconnections of the subscribers.

Semi-durable subscriptions are thus a compromise between message reliability and system performance. Unlike durable subscriptions, they cannot guarantee that a message will not be lost but as opposed to nondurable subscriptions they prevent heavy message losses due to user mobility and weak connectivity. Their throughput is expected to be slightly lower than nondurable subscriptions but much higher than durable

subscriptions in a mobile network. Thus taking the good features of the two extremes, semi-durable subscriptions provide a mid way, suitable for mobile wireless users.

One point worth mentioning here is that even if we assume that there is an infinite buffer at each broker and all the messages are buffered, semi-durable subscriptions differ from durable subscriptions in a number of ways. For example, while storing messages there is no concept of messages being stored for a certain subscriber. The message stored may be delivered to more than one subscriber or may not be delivered at all. Also, no particular broker is now storing messages for a certain subscriber unlike durable subscriptions. The subscriber can receive messages from any broker in the system. This flexibility in choosing the source of messages in semi-durable subscriptions is extremely important in getting performance in a mobile domain. Also as a broker does not store messages on a subscriber by subscriber basis, its performance does not depend on the number of inactive subscribers it has.

When the number of subscribers becomes very large, issues like scalability becomes important. The scalability of the system based on semi-durable subscriptions depends on system resources and message fanout. For a given set of system resources, message fanout determines the scalability of the system. As a single message stored in the buffer in semi-durable subscriptions can be used for a number of subscribers, semi-durable subscriptions are highly scalable for large message fanout.

6.2.3.1 Review of the Challenges and the Respective Solutions

Let us now briefly discuss how the scheme of semi-durable subscriptions deals with the challenges mentioned in Section 6.2.2.

Points 1 to 3: The above discussion has already shown how semi-durable subscriptions tackle challenges mentioned in points 1 and 2. As the subscriber can now disconnect from the broker any time without a need to be uncoupled, the technique works for non-graceful conditions (point 3).

Points 4 and 5: In semi-durable subscriptions there is no triangular routing of messages. Each broker may get some messages specific to its location and since the subscriber receives messages from the broker in its current location, semi-durable subscriptions can support location aware services. Unlike, middleware level handoffs, no race conditions occur in semi-durable subscriptions. The buffer and handoff Managers in semi-durable subscriptions ensure ordered and free from duplicates delivery of messages.

Points 6 and 7: Semi-durable subscription provides a rich system parameter space to suit the needs of the environment and/or to achieve desired characteristics. The size of the semi-durable buffer, its internal organization and message overwriting policies provide enough control on system configuration allowing on one end low message loss and on the other high throughput. All the details of semi-durable subscriptions can be hidden from applications by implementing it at the middleware level, making seamless integration of fixed and mobile domains possible.

6.3 Design and Implementation of Semi-Durable Subscriptions

This section discusses the details of the design and implementation of semi-durable subscriptions.

6.3.1 Design Components

Following components take part to provide semi-durable subscriptions to mobile wireless subscribers.

6.3.1.1 Broker

Just like other subscription types, in semi-durable subscriptions the message broker is the key component. It receives messages from the publisher, manages active subscriptions and sends messages to the subscribers. After a semi-durable subscriber has successfully received missed messages from the semi-durable buffer, it connects to the broker and receives messages from the broker just like an ordinary nondurable subscriber.

6.3.1.2 Semi-Durable Subscriptions Manager

In semi-durable subscriptions, each node running the broker now also runs a Semi-Durable Subscriptions Manager. Semi-durable Subscriptions Manager is the component that manages semi-durable subscriptions. It is the main thread of such subscriptions and controls other components providing them necessary inputs. A system administrator directly interacts with this component to configure system behavior. A Semi-durable Subscriptions Manager itself interacts with Buffer Subscriptions Manager, Semi-Durable Buffer Manager and Handoff Manager. Each of these components is briefly described.

6.3.1.3 Buffer Subscriptions Manager

Buffer Subscriptions Manager is the component that subscribes to the broker for the subscriptions which need to be buffered. Although the discussion in Section 6.2

shows that semi-durable subscriptions buffer every message published, the Buffer Subscriptions Manager can control the subscriptions in cases where a certain broker does not intentionally want to provide services on certain topics or in cases where it wants to provide better services on certain topics (by using unequal buffer partitions). The Buffer Subscriptions Manager thus takes a Subscriptions Policy File (that contains information about the subscriptions to be buffered) as one of its inputs.

6.3.1.4 Semi-durable Buffer Manager

Semi-durable Buffer Manager is the component which is in direct control of the buffer. It controls the size of the buffer and its internal organization. It takes Message Overwriting Policy as one of its input to control which messages should be overwritten when the buffer gets filled. When a certain subscriber needs messages that it missed during the temporary disconnection, the Handoff Manager contacts Semi-Durable Buffer Manager to retrieve those messages.

6.3.1.5 Handoff Manager

Handoff Manager is the component to which subscriber connects each time it gets disconnected from the network whether during handoff from one broker to the other or during a signal down interval. Handoff Manager receives necessary subscription information from the subscriber and contacts Semi-durable Buffer Manager to retrieve the lost messages and delivers them to the subscriber.

6.3.1.6 Semi-durable Buffer

This is the component that actually buffers the messages. Every message written to it or read from it is controlled by Semi-Durable Buffer Manager.

Figure 6.1: Components of Semi-durable Subscriptions

Figure 6.1 shows the interaction among various components.

6.3.2 Operation of Semi-Durable Subscriptions

Handoff in semi-durable subscriptions is shown in Figure 6.2. The details of the interaction are shown in the collaboration diagram of Figure 6.3 using standard COMET notation [33]. In order to explain the details of the system operations, each step is briefly reviewed.

Figure 6.2: Handoff in Semi-durable Subscriptions

Step 1: During the first step the system initialization is performed. The system administrator provides the Subscriptions Policy File, the Buffer Organization File and the

Message Overwriting Policy File along with the desired size of the buffer as inputs to the Semi-durable Subscriptions Manager. In accordance with the Subscriptions Policy File the Buffer Subscriptions Manager subscribes to the broker that in turn notifies the publish/subscribe system of its desired subscriptions and starts receiving messages on them. At the same time Semi-durable Buffer Manager organizes the size and internal partitions of the buffer according to the inputs from the administrator. The Buffer Subscriptions Manager forwards every message to Semi-durable Buffer Manager which writes it to the buffer. When the buffer gets filled the Semi-Durable Buffer Manager uses Message Overwriting Policy to decide which messages to overwrite.

Figure 6.3: Operation of Semi-durable Subscriptions: Collaboration Diagram

Step 2: Every subscriber in semi-durable subscriptions receives messages from the broker just as a normal nondurable subscriber. Hence if it gets disconnected from the

broker for some reason it does not need to be uncoupled from the broker. The broker does not maintain the list of its inactive subscribers or store messages for them.

Step 3: Every subscriber keeps track of the ID of the last message it received in a log file. This could be the JMS Message ID or any other uniquely identifiable ID of the message used by the system. When the subscriber reconnects to the network after a disconnection, irrespective of whether it connects to the same broker or to the new broker (if it has come to a new location), instead of contacting the broker directly, it first contacts the Handoff Manager. It passes its subscription information to the Handoff Manager along with the ID of the last message it received on a certain topic. The Handoff Manager in turn contacts the Semi-durable Buffer Manager to retrieve the lost messages. The Semi-durable Buffer Manager tries to locate the message with that ID in its buffer.

Step 3A: If the Semi-durable Buffer Manager finds the message it sends all the messages matching subscriber's subscription criterion published after that message to the Handoff Manager. The Handoff Manager in turn delivers them to the subscriber. The subscriber then goes to step 4.

Step 3B: If the Semi-durable Buffer Manager does not find the message, there are two possibilities; either the new broker has not yet received that message due to the network delay or the message has been overwritten. If the Semi-durable Buffer Manager finds a message published after the message with the ID supplied by the subscriber, it means that the message has been overwritten. Otherwise, the message has not yet reached the broker. In case the message has not yet reached the broker, the system waits for the arrival of the message before disconnecting the subscriber from the Handoff Manager (to prevent

duplicate messages delivery). In case the message has been overwritten, Semi-Durable Buffer Manager delivers all the messages available in the buffer, matching subscriber's criterion (and published after the message with the ID supplied by the subscriber) to the Handoff Manager which sends them to the subscriber.

Step 4: A control message from the Handoff Manager is used to inform the subscriber of the number of outstanding messages. After receiving the last message, the subscriber disconnects from the Handoff Manager and subscribes to the broker just as a normal nondurable subscriber.

One point worth mentioning here is the message loss occurring in semi-durable subscriptions. In step 3B if the Semi-durable Buffer Manager finds in its buffer the messages published after the message with ID passed by the subscriber, it means that some of the messages missed by the subscriber have been overwritten and hence are not available to be recovered. This loss of messages can be reduced by running sophisticated overwriting policies, which can guarantee that the subscriber will receive at least all the *important* messages.

The prototype implementation of semi-durable subscriptions required minor modifications in the subscriber's application. If one were to have the source code of the JMS implementation the scheme could be implemented at the middleware level. This would hide all the implementation details from the applications and would make the integration of fixed and mobile domains seamless.

6.3.3 Implementation Challenges and Solutions

6.3.3.1 Buffer Consistency

There may be more than one subscriber requesting missed messages from the buffer at the same time. Also the broker needs to write messages to the buffer continuously. Under these conditions, using single threaded Semi-durable Buffer Manager (which controls messages read from and written to the buffer) would degrade performance, as Semi-durable Buffer Manager thread would become the performance bottleneck. On the other hand, using multi-threaded Semi-durable Buffer Manager cannot ensure buffer consistency as different threads may be writing to and reading from the buffer at the same time. Using semaphores to maintain buffer consistency allows only one thread to access the buffer at one time and hence degrades performance.

The problem was solved by dividing the buffer into several partitions and running multi-threaded Semi-durable Buffer manager such that each thread has access to only one partition of the buffer. This way several threads can access the buffer at the same time without affecting buffer consistency. This partitioning of the buffer should be logically designed (e.g. by assigning each topic or a particular selector range in a certain topic to a certain buffer partition). However, the subscriber may be interested in more than one selector range in which case the Handoff Manager would need to contact more than one Semi-durable Buffer Manager's thread and would need to put in order the resulting messages from multiple threads itself.

6.3.3.2 Searching Algorithm

One of the other performance concerns during implementation was the efficiency of the algorithm which locates messages of interest to the subscriber in the semi-durable

buffer. Due to the limited size of the buffer, messages are overwritten all the time and hence they are no longer in chronological order in the buffer. The algorithm first of all attempts to locate the message with the ID given to it as an input and then finds all the messages published after this message that match subscriber's criterion. It is quite possible that the message with such an ID does not exist in the buffer. The algorithm thus may take considerable time and become the performance bottleneck.

Here again it was found that partitioning of the buffer into logical partitions can help improve the performance of the searching algorithm as now the algorithm has to search through fewer messages. At the same time a good partitioning of the buffer may help reduce the time taken by the algorithm to match each message with subscriber's filtering criterion.

Investigation into the implementation issues in more details is beyond the scope of the thesis.

6.3.4 Testing

Semi-durable subscriptions' prototype implementation was tested by printing debug messages on the screen at different points in the code for each of the possible cases. The messages demonstrated that the system is functioning as per expectations. Additionally, each scenario was validated by first running it with a small number of subscribers and small number of messages and then comparing the results with those calculated manually.

6.4 System Parameters for Semi-Durable Subscriptions

The important system parameters for semi-durable subscriptions are described in this section.

6.4.1 Buffer Size

Buffer Size is an important system parameter for semi-durable subscriptions. Using large buffer size will reduce the message loss but will impose more overheads by decreasing the efficiency of the searching algorithm and increasing the task of delivering greater number of messages. Large buffer size also increases the heap memory size of the JVM and hence may cause more frequent garbage collections. Using smaller buffer sizes will improve throughput but may increase message loss.

Buffer size thus gives a direct control over system behavior and it can be set according to the desired output of the system or the needs of the mobility environment, any where between the two extremes (low message loss or high performance). Size of the buffer can also be set dynamically according to the publishing rate.

6.4.2 Internal Organization of the Buffer

As discussed in Section 6.3.3 it makes sense to divide the buffer into a number of partitions. It improves the efficiency of the searching algorithm and at the same time allows several requests to be served concurrently. However, parameters such as number of partitions, size of each partition need to be determined. Also different partitions can have the same size or they may have different sizes. The size may also be determined by active rate of publishing on each topic and hence the same partition may need to have different sizes at different times in accordance with the active publishing rate. The buffer

thus can be organized according to the needs of mobility environment hence providing control on the system behavior.

6.4.3 Message Overwriting Policies

The message overwriting policy is the most important system parameter, as this parameter decides which messages should be overwritten. A number of overwriting policies can be designed but the goal of each should be to ensure that at least all *important* messages are delivered to the subscribers. Now the *importance* of a message may have different meanings in different contexts and can give rise to different message overwriting policies some of which are discussed below.

6.4.3.1 Least Recently Published Overwriting Policy

This is the simplest of all the policies. In this policy when the buffer gets filled least recently published messages are overwritten. The justification behind this policy is the fact that generally recently published messages are more important than others in the sense that they carry the latest information. This policy puts the least performance overhead on the buffer but it may overwrite some of the messages which are not recent but still important.

6.4.3.2 Invalid Messages Overwriting Policy

In this policy the algorithm attempts to find out the messages which have become invalid after the arrival of a newer message. For example, in stock information delivery, a new rate of a certain stock invalidates the previous rates. So when the buffer gets filled the policy locates and overwrites invalid messages, irrespective of when they were published. This policy thus minimizes the information loss by overwriting information

that has become obsolete. However, it imposes an overhead in locating invalid messages and at the same time may decrease the efficiency of the searching algorithm. Moreover, if there are no invalid messages to be overwritten the policy fails and has to depend on other policies such as least recently published overwriting policy.

6.4.3.3 Policies with Priorities

This is a group of policies that uses message priority for selecting messages for being overwritten. For example, the policy may decide to give more priority to a certain topic or a group of topics and/or particular selector ranges in such a way that the messages belonging to these topics/selector ranges have the least chance of being overwritten. Each topic/selector range can be assigned a priority and the algorithm can take into account those priorities to locate the messages to be overwritten. Priority can also be based on message size. As larger messages require more space in the buffer, the algorithm may give priority to smaller messages by increasing the chances of larger messages of being selected to be overwritten.

6.4.3.4 Least Information Loss Overwriting Policy

This policy assigns an information index to every message corresponding to the degree of importance of the message. The higher the information index of the message the higher the importance and vice versa. This assignment of information index to every message can be based on a number of factors such as the time of publication, validity, topic of publication. Whenever the buffer gets filled the policy locates the message with the least information index and overwrites it. The policy thus minimizes the information

loss. However, since this policy requires more computation than other policies it gives rise to more overheads on the system.

6.5 Performance Comparison of Semi-Durable Subscriptions with Nondurable and Durable Subscriptions

This section compares the performance of semi-durable subscriptions with other subscription mechanisms. Comparison has been done for both server bottlenecked and bandwidth bottlenecked regions. Default values of workload parameters presented in Table 3.1 are used for the fixed parameters in these experiments and are listed in Table 6.1.

Table 6.1: Input Parameters for Comparing the Performance of Semi-durable Subscriptions with that of Nondurable and Durable Subscriptions

Parameter	Value
Length of the Message Body (S)	50 Bytes
Mean Length of Signal Down Interval (t_{SD})	12 seconds
Probability Distribution of Signal Down Intervals, C_v	Exponential, 1
T_{SD} / T	0.0742
Semi-durable Buffer Capacity, Partitions	30000 Messages, 10
Message Overwriting Policy	Least Recently Published

6.5.1 Maximum Sustainable Throughput in the Server Bottlenecked Region

Maximum sustainable throughput of the three subscription types was compared for the server bottlenecked region using both open and closed publisher model (as

discussed in Section 3.1.2.1). The results with closed model are given in Appendix A.3 while the results with open model are shown in Figure 6.4.

Figure 6.4(a) shows that the maximum sustainable publisher throughput X_P for semi-durable subscriptions is about 82.5% of that for nondurable subscriptions while the subscriber throughput X_S is about 90% of that of nondurable subscriptions. This loss in throughput capacity of semi-durable subscriptions is due to the overhead of running the semi-durable managers, maintaining the buffer and recovering the missed messages to the subscribers. But this loss in throughput pays off by reducing the message loss (Figure 6.4(c)) of approximately 7.5% in non-durable subscriptions to only 0.18%. The X_S of durable subscriptions, where we can expect the message loss to be zero, is merely 6% of that achieved with the semi-durable subscriptions. This strongly suggests that bringing the message loss, occurring in mobile wireless environments, to a negligible value through the use of semi-durable subscriptions is cost effective but bringing it down to zero with durable subscriptions is too expensive.

The graph in Figure 6.4(b) shows that the X_S for semi-durable and durable subscribers consists of two parts, one part corresponds to the messages received from the broker (shown in darker shade) and the other corresponds to the messages received from the Handoff Manager (shown in lighter shade). The throughput part from Handoff Manager in semi-durable subscriptions is about 7.5% of the total subscriber throughput. This in fact is the percentage of messages which are lost in nondurable subscriptions due to temporary disconnections of the subscribers. The reason for message loss in durable subscriptions has already been discussed in Section 5.3.1.

Figure 6.4: Maximum Sustainable Throughput of the Three Subscription Types in the Server Bottlenecked Region: (a) Maximum Throughput of Publisher (b) Throughput of Subscribers (c) Proportion of Messages Lost

One point worth mentioning here is that not all the loss of 0.18% occurring in semi-durable subscriptions is due to the limited size of the semi-durable buffer. A significant amount of this message loss is due to the *leaky* implementation where some messages are lost during the transfer of a client from the Handoff Manager to the broker. Integrating the broker and the semi-durable managers into a single entity can prevent this message loss.

The results obtained with the closed model publisher (given in Appendix A.3) show that semi-durable subscriptions give higher X_S than even nondurable subscriptions. The reason is that in closed model publisher an overload on one broker can prevent messages from being delivered to the other broker. In nondurable subscriptions the time the broker is not delivered any messages because of the overload on the other broker gets wasted. This is because it cannot transmit any messages as there are no new messages to be delivered. On the other hand, with semi-durable subscriptions the system is able to deliver messages even during this period through the Handoff Manager. That is why although X_P in semi-durable subscriptions is slightly lower (as there is an overhead of running semi-durable managers and buffers), X_S is higher than nondurable subscriptions. The rest of the results are similar to those obtained with the open model publisher.

6.5.2 Maximum Sustainable Throughput in the Bandwidth Bottlenecked Region

As already discussed in Chapter 5, throughput of durable subscriptions is too low to achieve a bandwidth bottleneck region, thus the maximum sustainable throughput of semi-durable subscriptions in a bandwidth bottlenecked region has been compared to nondurable subscriptions only. As discussed in Section 3.1.2.1 both open and closed

models of the publisher were used. The results with closed model are given in Appendix A.4 while the results with open model are shown in Figure 6.5.

Figure 6.5(a) shows that in the bandwidth bottlenecked region, maximum sustainable publisher throughputs X_P of both nondurable and semi-durable subscriptions are almost equal; which is conceivable as the throughput is being limited by the bandwidth which is kept the same for both subscription types. However, contrary to the intuition the subscriber throughput X_S of semi-durable subscriptions is significantly higher than that of nondurable subscriptions (Figure 6.5(b)). The reason is that the current implementations of JMS do not perform well in limited bandwidth environments and generate significant overhead traffic (discussed in detail in Section 4.2.1). The messages delivered by the broker thus consume higher bandwidth than actually required, reducing the total message carrying capacity of the channels. On the other hand, the Handoff Manager in semi-durable subscriptions is designed in such a way that it does not generate any extra traffic. Moreover, the Handoff Manager sends all the stored message in the same TCP session, unlike the newly published messages which are sent message by message by the broker. This delivery of messages in bulk saves a significant amount of bandwidth as well. Figure 6.5(b) shows that semi-durable subscriptions bring down the broker throughput (shown in darker shade) to 89.55 from 101.75 in nondurable subscriptions. In the resulting saved bandwidth of about 12 (101.75-89.55) messages per second semi-durable subscriptions are able to deliver almost 61 messages per second through the Handoff Manager (shown in lighter shade).

Figure 6.5: Maximum Sustainable Throughput of Nondurable and Semi-durable Subscriptions in the Bandwidth Bottlenecked Region
(a) Maximum Throughput of Publisher (b) Throughput of Subscribers (c) Proportion of Messages Lost

Figure 6.5(c) shows that message loss is significantly higher in limited bandwidth regions than that in the higher bandwidth regions for both nondurable and semi-durable subscriptions. The reason for higher message loss again lies in the fact that current implementations do not perform well in limited bandwidth environments (discussed in detail Section 4.2.1). Although semi-durable subscriptions significantly reduce the percentage message loss, it is still much higher than that in higher bandwidth regions. However, when this loss was studied in detail it was found that almost no message is lost due to limited buffer size or during the delivery of messages through the Handoff Manager. All the loss occurring in semi-durable subscriptions was during the period when the subscribers were connected to the broker. Section 4.2.1 describes how a broker can incorrectly assume a connected subscriber is disconnected and discards all incoming messages for it resulting in a significant message loss. In future JMS implementations that can handle limited bandwidth environments more effectively, we can expect the message loss with semi-durable subscriptions in bandwidth bottlenecked regions to be comparable to that occurring in server bottlenecked regions.

The results obtained with the closed model publisher given in Appendix A.4 are similar to those obtained with the open model publisher.

6.5.3 Comparison at given Publishing Rate

Performance of the three subscriptions mechanisms was compared at a given a publishing rate for the cases where the bandwidth becomes the bottleneck as well as for the cases where bandwidth does not become the bottleneck. The results for bandwidth bottlenecked region are similar to those in Section 6.5.2 and are shown in Appendix B5 while the results for the region where the bandwidth does not become the bottleneck are

given in this section. The results were obtained with different publishing rates equal to or lower than 285.29 messages/second (the maximum publishing rate semi-durable subscriptions can sustain with default values of the parameters). The results with the publishing rate of 285.29 are shown in Figure 6.6. The discussion is equally valid for all publishing rates lower than this message publishing rate.

Figure 6.6: Performance of Three Subscription Types at given Publishing Rate in the Region where Bandwidth is Not the Bottleneck
(a) Throughput of Subscribers (b) Proportion of Messages Lost

The graph in Figure 6.6(a) shows that at a certain publisher's rate (lower than or equal to the maximum rate semi-durable subscriptions can sustain), the subscriber

throughput X_S of semi-durable subscriptions is more than that achieved with non-durable subscriptions. In this case it is almost 7.5% higher. This corresponds to the messages delivered by the Handoff Manager in semi-durable subscriptions and lost with nondurable subscriptions. The X_S for durable subscriptions is zero, as they cannot withstand such a high publishing rate.

6.6 Effect of Input Parameters on the Performance of Semi-Durable Subscriptions

This section investigates the effect of different input parameters on the performance of semi-durable subscriptions. For this evaluation the default input parameters (discussed in Section 3.3) are shown in Table 6.2.

Table 6.2: Values of Input Parameters for Evaluation of Semi-durable Subscriptions

Parameter	Value
Publisher Model, Publishing Rate	Open, Maximum Sustainable
Mobility, Bandwidth Environment	Mobile, Server Bottlenecked
Mean Length of Signal Down Interval (t_{SD})	12 seconds
Semi-durable Buffer Capacity, Partitions	30000 Messages, 10
Message Overwriting Policy	Least Recently Published

The results for each set of experiments also include the corresponding graph for nondurable subscriptions for comparison purposes.

6.6.1 Effect of Frequency of Handoffs

This section studies the effect of frequency of handoffs on the performance of semi-durable subscriptions. For these experiments an exponential distribution of signal

down intervals is used. The length of message body is 50 Bytes. The results are shown in Table 6.3 and Figure 6.7.

Table 6.3: Effect of Frequency of Handoffs on Semi-durable Subscriptions
 $(f_H = ((T_{SD}/T)*100)*(1/(100*t_{SD})); t_{SD} = 12 \text{ seconds})$

$(T_{SD}/T)*100$	Max. Publisher Throughput (X_P) (messages/sec)	Subscriber Throughput (X_S) (messages/sec)	Percentage Message Loss
0.0	315.02	315.02	0
0.2	245.41	245.408	0.0008
1.0	252.73	252.72	0.0025
2.5	264.39	264.24	0.056
5.0	281.27	280.96	0.11
7.5	285.48	284.97	0.18
10.0	285.48	284.71	0.27
15.0	285.48	284.51	0.34

Results for nondurable subscriptions have already been discussed in Section 4.2.2. For semi-durable subscriptions, Figure 6.7(a) shows that as the frequency of handoffs is increased from zero (fixed subscribers), the maximum publisher throughput X_P decreases sharply at first. The reasons are the connections/disconnections and handoffs overheads which are imposed on the system each time a subscriber disconnects and/or reconnects to the system. Also for nonzero f_H the number of subscribers per broker may temporarily become greater than the average number of subscribers per broker. The increased load of subscribers on the broker is another cause of a lower maximum sustainable throughput. As the frequency of handoffs and hence T_{SD}/T increases, total time of disconnection of subscribers increases which decreases the average active subscribers per broker. This

decrease in load of subscribers on the broker increases X_p . The behavior of semi-durable subscriptions is the same as of nondurable subscriptions until about T_{SD}/T equal to 0.05. However, as the f_H increases further, unlike nondurable subscriptions, the X_p for semi-durable subscriptions does not increase further and more or less remains the same. The reason is that in semi-durable subscriptions the connections/disconnections overhead is much greater than in nondurable subscriptions, as every subscriber reconnecting to the broker first connects to the Handoff Manager to recover the missed messages. Hence as T_{SD}/T and thus f_H increases, this overhead increases, offsetting the benefit gained in performance due to the decreasing number of subscribers per broker.

Figure 6.7: Effect of Frequency of Handoffs on Semi-durable Subscriptions

$$(f_H = ((T_{SD}/T) \cdot 100) \cdot (1/(100 \cdot t_{SD}))); t_{SD} = 12 \text{ seconds})$$

(a) Maximum Throughput of Publisher (b) Throughput of Subscribers (c) Proportion of Messages Lost

Subscriber throughput X_S for semi-durable subscriptions in Figure 6.7(b) follows the same pattern as X_P (unlike nondurable subscriptions). The reason is that due to small message loss the X_P and X_S in semi-durable subscriptions are almost the same. Figure 6.7(c) shows that message loss percentage for semi-durable subscriptions increases with the increase in frequency of handoffs. This is because some messages are lost in semi-durable subscriptions when the subscriber is transferred from the Handoff Manager to the broker and hence with the increase in f_H the frequency of transfers increases which in turn increases the message loss. Also with increasing f_H the probability that some subscribers will disconnect from the network for a period long enough to overwrite the semi-durable buffer, increases. However, the rate of increase in message loss with f_H in semi-durable subscriptions is much lower compared to that in nondurable subscriptions.

6.6.1.1 Conclusions

These results show that for the mobility parameters used in the experiments, although semi-durable subscriptions give slightly lower maximum throughput than non-durable subscriptions, they effectively bring down the message loss to almost zero. The

difference between the message loss of the two subscription mechanisms increases with the increase in f_H without any change in the difference between the throughputs. This shows that semi-durable subscriptions are a much better choice for mobile environments than nondurable subscriptions.

6.6.2 Effect of Variability in Signal Down Intervals

The effect of variability in signal down intervals on the performance of semi-durable subscriptions is studied by using bi-phase hyper exponentially distributed signal down intervals. The co-efficient of variation C_V of the distribution is varied as an input parameter. The default values of S equal to 50 Bytes and T_{SD}/T equal to 0.0742 are used in these experiments. Although the Markov chain analysis breaks down for hyper-exponentially distributed signal down intervals, the value of T_{SD} measured experimentally during the experiments was found to be very close to the one predicted by the Markov chain analysis. The results are shown in Table 6.4 and Figure 6.8.

Table 6.4: Effect of C_V of Signal Down Intervals on Semi-durable Subscriptions

C_V	Max. Publisher Throughput (X_P) (messages/sec)	Subscriber Throughput (X_S) (messages/sec)	Percentage Message Loss
1.0	285.29	284.78	0.18
1.5	280.09	279.14	0.34
2.0	277.83	276.08	0.63
2.5	274.16	271.77	0.87

Figure 6.8(a) and 6.8(b) show that for both nondurable and semi-durable subscriptions, the maximum sustainable publisher as well as subscriber throughput decreases slightly with the increase in C_V . The reasons are the same for both subscription types and have been discussed in detail in Section 4.2.3.

Figure 6.8: Effect of C_v of Signal Down Intervals on Semi-durable Subscriptions
(a) Maximum Throughput of Publisher (b) Throughput of Subscribers (c) Proportion of Messages Lost

Figure 6.8(c) shows that message loss in semi-durable subscriptions increases significantly with the increase in C_V . The reason is that as C_V increases, length of signal down intervals varies from its mean significantly. The result is the presence of signal down intervals some of which have very long durations and the others have very short durations. As the maximum length of the signal down intervals increases, it increases the probability that the subscriber will remain disconnected for a period long enough that the buffer partition the subscriber is interested in gets overwritten. This increases the message loss. The longer the signal down intervals, the higher the message loss. The reason for increase in message loss with the increase in C_V for nondurable subscriptions has been discussed in Section 4.2.3.

6.6.2.1 Conclusions

The results show that increase in C_V of the signal down intervals has a similar effect on both nondurable and semi-durable subscriptions as far as throughputs are concerned. However, in semi-durable subscriptions increase in message loss with the increase in C_V is much more significant than that in nondurable subscriptions.

6.6.3 Effect of Message Size

The effect of message size on the performance of semi-durable subscriptions is studied using size of the message body S as a parameter. An exponential distribution is used for modeling the signal down intervals. Default value of T_{SD}/T equal to 0.0742 is used in these experiments. The results are given in Table 6.5 and Figure 6.9.

Figure 6.9(a) and 6.9(b) show that just like nondurable subscriptions, in semi-durable subscriptions both the maximum publisher throughput X_P and subscriber

throughput X_S decreases with an increase in message size. The reason is the same as for nondurable subscriptions (discussed in detail in Section 4.2.4). As expected there is no effect of message size on percentage message loss (Figure 6.9(c)).

Table 6.5: Effect of Message Size on Semi-durable Subscriptions

Message Size (S) (Bytes)	Max. Publisher Throughput (X_P) (messages/sec)	Subscriber Throughput (X_S) (messages/sec)	Percentage Message Loss	Information Delivery Rate (kB/sec)
50	285.29	284.78	0.18	13.91
100	281.28	280.8	0.17	27.42
200	277.08	276.61	0.17	54.03
500	270.58	270.12	0.17	131.89

(a)

(b)

Figure 6.9: Effect of Message Size on Semi-durable Subscriptions
(a) Maximum Throughput of Publisher (b) Throughput of Subscribers (c) Proportion of Messages Lost (d) Maximum Delivery Rate of Information

Figure 6.9(b) shows that for the same increment in size of the messages, X_S for nondurable subscriptions goes down by almost 7% while for semi-durable subscriptions it goes down by only 5%. The reason is that semi-durable subscriptions handle network contention more efficiently in comparison to nondurable subscriptions (discussed in Section 6.5.2) and hence as the message size increases they are less affected by increased contention for network resources than nondurable subscriptions. Figure 6.9(d) shows that the information delivery capacity of both nondurable and semi-durable subscriptions increases with the increase in message size.

6.6.3.1 Conclusions

System throughputs decrease slightly with the increase in message size for both nondurable and semi-durable subscriptions. However, semi-durable subscriptions are less affected by increased sizes of the messages as they handle network resources better than nondurable subscriptions. The increase in information delivering capacity with the increase in message size suggests that if possible information should be batched together and then sent as a larger message.

6.7 Effect of Buffer Size on the Performance of Semi-Durable Subscriptions

As discussed in Section 6.4.1, buffer size is an important system parameter for semi-durable subscriptions; it can be used to configure the system to get the desired behavior. It can also be used to exploit the inherent tradeoff between system throughput and message loss in semi-durable subscriptions. This section studies this tradeoff with the help of the results of a number of experiments which were conducted using the default values of fixed workload parameters presented in Table 3.1 and are listed in Table 6.6.

Table 6.6: Input Parameters for Investigating the Effect of Buffer Size on Semi-durable Subscriptions

Parameter	Value
Length of the Message Body (S)	50 Bytes
Probability Distribution of Signal Down Intervals, C_v	Exponential, 1
T_{SD} / T	0.0742
Mobility, Bandwidth Environment	Mobile, Server Bottlenecked
Number of Buffer Partitions	10
Message Overwriting Policy	Least Recently Published

6.7.1 Effect of Publishing Rate on Message Loss

This section studies the message loss occurring at different buffer sizes at a given publishing rate. The first set of results shown in Table 6.7 and Figure 6.10 is generated using t_{SD} equal to 12 sec. To study the effect of longer signal down intervals, second set of results shown in Table 6.8 and Figure 6.11 uses a t_{SD} equal to 24 sec. For the second set duration between consecutive connection requests to the broker t_C is increased to 308.64 sec (from 154.32 sec in first set) to keep T_{SD}/T equal to the default value of 0.0742.

Table 6.7: Effect of Publishing Rate on Message Loss for Different Buffer Sizes ($t_{SD} = 12$ seconds)

Publishing Rate (messages/sec)	Percentage Message Loss at Buffer Size = 10000 Messages	Percentage Message Loss at Buffer Size = 20000 Messages	Percentage Message Loss at Buffer Size = 30000 Messages	Percentage Message Loss at Buffer Size = 50000 Messages
100	0.15	0.15	0.15	0.15
150	0.18	0.16	0.16	0.16
190	0.26	0.16	0.16	0.16
265	0.56	0.17	0.17	0.17
277.62	0.62	0.18	0.17	0.17
285.29	0.66	0.19	0.18	-
287.59	0.67	0.19	-	-
293.74	0.71	-	-	-

Publishing rate is increased in steps until the maximum sustainable throughput is achieved. Since smaller buffer sizes can sustain higher publishing rates, the results given in Table 6.7 and Table 6.8 show more readings for smaller buffer sizes as compared to

larger ones. Results at few publishing rates are omitted to save space (only important ones are shown).

Figure 6.10: Effect of Publishing Rate on Message Loss for Different Buffer Sizes
($t_{SD} = 12$ seconds)

Table 6.8: Effect of Publishing Rate on Message Loss for Different Buffer Sizes
($t_{SD} = 24$ seconds)

Publishing Rate (messages/sec)	Percentage Message Loss at Buffer Size = 10000 Messages	Percentage Message Loss at Buffer Size = 20000 Messages	Percentage Message Loss at Buffer Size = 30000 Messages	Percentage Message Loss at Buffer Size = 50000 Messages
100	0.2	0.07	0.07	0.07
150	0.56	0.13	0.08	0.08
190	0.96	0.21	0.08	0.08
265	1.76	0.49	0.12	0.09
280.84	1.92	0.59	0.18	0.09
289.69	2.02	0.65	0.24	-
291.74	2.04	0.67	-	-
295.69	2.08	-	-	-

Figure 6.11: Effect of Publishing Rate on Message Loss for Different Buffer Sizes ($t_{SD} = 24$ seconds)

The above results show that as the buffer size increases, message loss for a given publishing rate decreases. However, beyond a certain buffer size the percentage message loss does not decrease with the increase in buffer size. The reason is that not all the loss occurring is due to the messages getting overwritten. A significant portion of loss occurs in this prototype implementation of semi-durable subscriptions during the transfer of subscribers from the Handoff Manager to the broker. This portion of loss increases with the increase in publishing rate. This is because increased publishing rates increase the probability that the message will be missed in the duration in which the subscriber transfers from the Handoff Manager to the broker. The portion of the loss is higher for t_{SD} equal to 12 seconds than for t_{SD} equal to 24 seconds. The reason for this is that for t_{SD} equal to 24 seconds t_C is equal to 308.64 seconds instead of 154.32 seconds (that corresponds to t_{SD} equal to 12 seconds). A higher t_C means a lower number of connections/disconnections per second and thus for t_{SD} equal to 24 seconds this portion of message loss is almost half of that for t_{SD} equal to 12 seconds.

If we subtract the portion of the message loss occurring during the transfer of subscribers (in integrated implementations it will not be present at all) from the total loss, the results show that at a given publishing rate with the increase in buffer size message loss decreases until it reaches zero. Increasing buffer size beyond this critical value just increases the overheads on the system without any reduction in message loss. Furthermore, it is well known that the increase in service time decreases the maximum sustainable throughput of a server. The critical value of buffer size depends on the publishing rate as well as on the mobility environment. For example, in Figure 6.10 ($t_{SD} = 12$ seconds) at a publishing rate of 250 messages/second a buffer size of 20000 messages is sufficient to reduce the message loss to a small value but in Figure 6.11 ($t_{SD} = 24$ seconds) a buffer size of 30000 messages or more is desirable. The buffer size thus should be carefully selected according to the given mobility environment as well as the publishing rate.

6.7.2 Tradeoff between System Throughput and Message Loss

This section studies in detail the tradeoff between system throughput and message loss in semi-durable subscriptions using buffer size as a control parameter. The results are obtained using an open publisher model operating at the maximum sustainable publishing rates. Table 6.9 and Figure 6.12 show the results for t_{SD} equal to 12 seconds while Table 6.10 and Figure 6.13 give the results for t_{SD} equal to 24 seconds. Default value of T_{SD}/T equal to 0.0742 is used in these experiments.

Figure 6.12(a) and Figure 6.13(a) show that as the buffer size increases, the maximum sustainable publisher throughput X_P decreases. The reasons are the overheads of the larger buffer sizes in terms of increased task of delivering more messages to the

subscribers, frequent garbage collection due to increased heap memory size of JVM and decrease in the efficiency of the searching algorithm. However, this decrease in X_P pays off in terms of decreasing the message loss in Figure 6.12(c) and 6.13(c). Thus when the buffer size is increased from 10000 to about 16000 messages for t_{SD} equal to 12 seconds there is a significant reduction in message loss bringing it to a very small value. On the other hand, for t_{SD} equal to 24 seconds the buffer size is needed to be increased to about 28000 messages or more to bring the message loss to a very small value. Why the message loss is not reduced to zero has already been discussed in Section 6.7.1.

Table 6.9: Tradeoff between Throughput and Message Loss ($t_{SD} = 12$ seconds)

Buffer Size (messages)	Max. Publisher Throughput (X_P) (messages/sec)	Subscriber Throughput (X_S) (messages/sec)	Percentage Message Loss
10000	293.74	291.65	0.71
15000	291.63	290.90	0.25
20000	287.59	287.04	0.19
30000	285.29	284.78	0.18
50000	277.62	277.15	0.17

(a)

Figure 6.12: Tradeoff between Throughput and Message Loss ($t_{SD} = 12$ seconds)
(a) Maximum Throughput of Publisher (b) Throughput of Subscribers (c)
Proportion of Messages Lost

Table 6.10: Tradeoff between Throughput and Message Loss ($t_{SD} = 24$ seconds)

Buffer Size (messages)	Max. Publisher Throughput (X_P) (messages/sec)	Subscriber Throughput (X_S) (messages/sec)	Percentage Message Loss
10000	295.69	289.54	2.08
20000	291.74	289.79	0.67
30000	289.69	288.99	0.24
50000	280.84	280.59	0.09

Figure 6.13: Tradeoff between Throughput and Message Loss ($t_{SD} = 24$ seconds)
(a) Maximum Throughput of Publisher (b) Throughput of Subscribers (c)
Proportion of Messages Lost

Thus for a given mobility environment a certain threshold buffer size is required to bring the message loss to an acceptable value. Increasing the buffer size beyond that value just increases the overheads on the system. The threshold value can be determined from the knee of the message loss curve. The knees are clearly visible at buffer sizes of 16000 and 28000 messages in Figure 6.12(c) and 6.13(c) respectively.

Subscriber throughput X_S curves in Figure 6.12(b) and 6.13(b) show that with the increase in buffer size, X_S does not change significantly until the threshold value of the buffer is reached. With the increase in buffer size X_P decreases which tends to decrease X_S . But at the same time the increase in buffer size decreases message loss which increases X_S . Thus the subscriber throughput X_S more or less remains constant until the threshold value of the buffer is reached. Increasing buffer size beyond the threshold value has a negative effect on X_S as it decreases X_P without decreasing the message loss.

The tradeoff between message loss and throughputs exists before the threshold value of the buffer is reached. For example if a higher maximum publisher throughput is more desirable than low message loss, the system can use a buffer size value less than the threshold value.

6.7.3 Conclusions

Buffer size is an important system parameter in semi-durable subscriptions. At a given publishing rate the size of the buffer should be determined according to the needs of the mobility environment. It makes sense to increase the size of the buffer to a threshold value necessary to bring the message loss to a tolerable value but increasing buffer size beyond this would just increase the overheads on the system.

There exists a tradeoff between maximum publisher throughput and message loss which can be controlled by buffer size. Both the maximum sustainable publisher throughput and message loss decreases with the increase in buffer size. However, there exists a certain threshold buffer value for the given mobility environment that most suits the system in a sense that not only it brings down the message loss to a very small value but it also maintains the maximum subscriber throughput. In most of the cases it will make sense to set the buffer to that threshold value.

6.8 Extending Semi-Durable Subscriptions to Deal with Regionalism

Regionalism is a phenomenon in which subscribers in different regions have different interests. To understand how it affects semi-durable subscriptions consider a semi-durable subscriber which moves from one broker to another. The broker in the subscriber's new location may or may not have subscriptions for the same topic and/or selector range. If it has the subscriptions, it can send the subscriber, from its semi-durable buffer, the messages lost during the handoff. However, if it does not have the required subscriptions, then the subscriber will lose all the messages published during the time it was disconnected. Before reviewing how such situations can be dealt with it is worth mentioning here that the existing papers on publish/subscribe systems advocate broadcasting all the messages to all the brokers and hence such situations may never arise. Research such as [49] show that due to the limited number of available IP multicast addresses and the overhead involved in frequent computing of the multicast trees, in a mobile environment all messages should be broadcast to all the brokers. Even if the broadcasts on a global scale are too expensive from network performance point of view,

regional broadcasts can be used. If we assume that messages are not broadcast then we consider the following methodology to extend semi-durable subscriptions to deal with regionalism.

In publish/subscribe systems, whenever a subscriber subscribes to the broker for some subscriptions on which the broker doesn't already have the subscriptions, the broker sends an update request to its neighboring brokers. These brokers forward this request until a tree to deliver messages on that subscription to the broker is formed. This thesis argues that as a part of this update, the broker can request the delivery of messages lost by the subscriber. The nearest broker having subscriptions for that topic and selector range can send them. The concept is similar to handoffs for durable subscriptions but here the broker does need not to contact a specific broker. Any broker having subscriptions on that topic/selector range that is located in the most suitable location (e.g. from network resources point of view) can recover those messages. Also unlike durable subscriptions the brokers do not incur an overhead in keeping track of their durable subscribers and storing messages for them. As one of the features of publish/subscribe is loose coupling between different entities, this flexibility in finding the source of messages in semi-durable subscriptions, is closer to this theme compared to durable subscriptions.

Chapter 7

Conclusions

This chapter summarizes conclusions drawn during each step of the thesis. It also gives directions for future research.

7.1 Summary

Decoupling of publishers and subscribers in space, time and flow makes publish/subscribe systems much more scalable than ordinary point-to-point models. By removing all explicit dependencies between the interacting parties, publish/subscribe systems allow for disconnected operations and mobility. These features along with their flexibility and inherently asynchronous nature make them a good choice for the mobile wireless domain. However, a lot of work still needs to be done before these systems can give high performance and reliability in such environments. Hence at least for now the seamless integration of fixed and mobile domains does not seem possible.

Although nondurable subscriptions give almost as much throughput for mobile users as for fixed users, they suffer from message loss. The message loss occurs in bursts as the user loses all the messages published during the time it was disconnected and hence it may not be acceptable to many applications. Nondurable subscriptions are not supposed to deliver messages published during the time the subscriber was disconnected from the network, but there should at least be a way to prevent message loss occurring during the temporary disconnections of the subscribers due to poor wireless channels and

mobility. Durable subscriptions, on the other hand, face a number of challenges in a mobile domain (e.g. message loss, message duplication) which should be dealt with before they can be deployed for mobile users. Middleware level handoffs can make durable subscriptions work in mobile wireless environments but they give extremely low throughput. Throughput of durable subscriptions is low compared to nondurable subscriptions even in fixed networks, but in mobile wireless environments durable subscriber throughput can become as low as 5% of that of nondurable subscribers. Paying such a high cost to prevent message loss occurring in nondurable subscriptions may be suitable for some applications, but it certainly cannot be justified for general information dissemination. The low throughput of durable subscriptions in mobile environments can be attributed to a tight coupling between the broker and the subscriber.

In limited bandwidth environments, current implementations of JMS perform poorly. The system should take into account potential queues on the wireless channels and should wait longer before resending the messages. This would not only save precious wireless bandwidth but would also increase the system throughput and would decrease the message loss.

To overcome the problems faced by nondurable and durable subscriptions this thesis introduces semi-durable subscriptions. Taking the good features of both nondurable and durable subscriptions, semi-durable subscriptions provide a mid way between the two extremes (high performance and no message loss) suitable for the mobile wireless domain. Semi-durable subscriptions ensure loose coupling between the broker and the subscribers and hence do not incur substantial performance overheads and at the same time they recover almost all of the messages missed by the subscribers during

temporary disconnections in a mobile wireless environment. They require a very nominal cost in terms of a small reduction in total throughput capacity of the system as compared to nondurable subscriptions. Bringing message loss to exactly zero with the use of durable subscriptions can be very expensive as their subscriber throughput can be as low as 6% of that for semi-durable subscriptions. Semi-durable subscriptions handle limited bandwidth environments much more efficiently than the SonicMQ implementation of JMS, giving significantly higher subscriber throughput than even nondurable subscriptions (See Section 6.5.2). Also at all publishing rates equal to or lower than the maximum rate semi-durable subscriptions can sustain semi-durable subscriptions give higher subscriber throughput than even nondurable subscriptions and at the same time reduce the message loss to a small value (See Section 6.5.3). All these features make the semi-durable subscription technique the best choice for mobile cellular networks.

Semi-durable subscriptions provide a rich parameter space to configure the system into the desired mode. Buffer size, buffer organization and message overwriting policies can be used to exploit the inherent tradeoff between message loss and throughput in semi-durable subscriptions. There exists a certain threshold buffer value for a given mobility environment that most suits the system in a sense that not only it brings down the message loss to a very small value but it also maintains the maximum subscriber throughput. In most of the cases it will make sense to set the buffer size to that threshold value. Semi-durable subscriptions can be extended to deal with regionalism.

Mobility parameters such as frequency of handoffs, mean and co-efficient of variation of signal down intervals can significantly affect the performance of publish/subscribe systems. Hence all these factors should be taken into account while

deploying publish/subscribe systems in mobile wireless networks. Message size is another important factor that can affect performance. Increasing the message size decreases the subscriber throughput but increases the information delivering capacity of the broker. Hence (where possible) batching information together before sending may improve performance.

7.2 Future Research

A number of issues that warrant investigations are presented:

- The thesis shows how buffer size in semi-durable subscriptions can be varied to set the system according to the needs of the environment. Other system parameters such as internal organization of the buffer and message overwriting policies can also be varied to study their effect on system performance. Implementing each policy and doing their relative performance comparison is another important direction for research.
- The tradeoff between system throughput and message loss in semi-durable subscriptions shows that it makes sense to dynamically set the size of the buffer. Dynamic partitioning of the buffer as well as dynamically selecting the message overwriting policy may also help improve performance in certain scenarios. Such an adaptive system based on semi-durable subscriptions requires further investigation.
- This thesis used a peer-to-peer topology of brokers. However, other topologies such as hierarchical as well as hybrid are possible (See Section 2.1.5.1). It is an important future research direction to compare the behavior of different

topologies for dynamic mobile wireless environments with high frequency of connections and disconnections.

- If using such hierarchical or hybrid topology of brokers, the relationship between semi-durable buffers at each layer of the hierarchy and their corresponding message overwriting policies make a very interesting problem to probe in. In such topologies, message buffering at the brokers in each layer of the hierarchy can be limited to active subscriptions in the region covered by the broker. The messages missed by the subscribers moving from one region to another can be recovered from the broker at a higher level in the hierarchy which covers a wider region.
- In bandwidth bottlenecked regions, message compression and/or message simplification (converting a multimedia message to text message) can help improve performance. The broker can also filter the messages for bandwidth bottlenecked users sending them only the *important* messages. All these tasks can be assigned to a Message Manager which can run along with the broker to manage message conversion and selection for the users with limited bandwidth. Design, prototype implementation and evaluation of such a Message Manager are important directions for future research.
- In semi-durable subscriptions (also in durable subscriptions with middleware level handoffs) if a user connects after a long interval, there may be many messages stored for it. Sending all those messages may flood the wireless channels and waste precious wireless bandwidth. This can be avoided by configuring the Handoff Manager to take into account the current available bandwidth and then decide about the number of messages which should be sent.

Prototype implementation of this concept and an investigation of its effect on the overall system behavior is an important direction for future research.

- This thesis gives an overview of how the semi-durable subscriptions can be extended to deal with regionalism. Prototype implementation and evaluation of such extensions to investigate the relative benefit in performance is an interesting direction for future research.
- As discussed in Chapter 4 and Chapter 6, current implementations of JMS perform poorly in low bandwidth environments owing to the short timeout interval. It needs further investigation to figure out the most suitable duration the system should wait before resending the message. It may also make sense to set timeout intervals dynamically by taking into account the average acknowledgement time.

References:

- [1] M. K. Aguilera, R. E. Strom, D. C. Sturman, M. Astley and T. D. Chandra, "Matching Events in a Content-Based Subscription System," in *Proceedings of the 18th ACM Symposium on Principles of Distributed Computing (PODC'99)*, Atlanta, GA, May 1999, pp. 53-61.
- [2] F. Araujo and L. Rodrigues, "On QoS-Aware Publish-Subscribe," in *Proceedings of the 1st International Workshop on Distributed Event-Based Systems (DEBS '02)* in conjunction with the 22nd International Conference on Distributed Computing Systems (ICDCS'02), Vienna, Austria, July 2002, pp. 511-515.
- [3] F. Araujo and L. Rodrigues, "Quality of Service in Indirect Communication Systems," in *Proceedings of the 4th European Research Seminar on Advances in Distributed Systems (ERSADS 2001)*, Bertinoro, Italy, May 2001.
- [4] F. Araujo, L. Rodrigues, "The IndiQoS Message Broker: an Instantiation Using RSVP," *Technical report DI/FCUL TR 02-3*, Department of Computer Science, University of Lisbon, Lisbon, Portugal, March 2002.
- [5] G. Banavar, T. D. Chandra, B. Mukherjee, J. Nagarajarao, R.E. Strom and D. C. Sturman, "An Efficient Multicast Protocol for Content-based Publish/Subscribe Systems," in *Proceedings of the 19th IEEE International Conference on Distributed Computing Systems (ICDCS'99)*, Austin, TX, May 1999, pp. 262-272.
- [6] G. Brasche and B. Walke, "Concept, Services and Protocols for the New GSM Phase 2+ General Packet Radio Service," *IEEE Communications Magazine*, Vol. 35, No. 8, August 1997, pp. 94-104.
- [7] L. F. Cabrera, M. B. Jones, and M. Theimer, "Herald: Achieving a Global Event Notification Service," in *Proceedings of the 8th Workshop on Hot Topics in Operating Systems (HoTOS-VIII'01)*, Elmau, Germany, May 2001, pp. 87-94.
- [8] A. Campailla, S. Chaki, E. Clarke, S. Jha, and H. Veith, "Efficient Filtering in Publish-Subscribe Systems Using Binary Decision Diagrams," in *Proceedings of the 23rd International Conference on Software Engineering (ICSE 2001)*, Toronto, Canada, May 2001, pp. 443-452.
- [9] M. Caporuscio, A. Carzaniga and A. L. Wolf, "An Experience in Evaluating Publish/Subscribe Services in a Wireless Network," in *Proceedings of the 3rd International Workshop on Software and Performance* in conjunction with *International Symposium on Software Testing and Analysis (ISSTA 2002)*, Rome, Italy, July 2002, pp. 128-133.

- [10] M. Caporuscio, A. Carzaniga and A. L. Wolf, "Design and Evaluation of a Support Service for Mobile, Wireless Publish/Subscribe Applications," *Technical Report CU-CS-944-03*, Department of Computer Science, University of Colorado, Boulder, CO, January 2003.
- [11] L. Capra, W. Emmerich and C. Mascolo, "Middleware for Mobile Computing (A Survey)," in *Advanced Lectures on Networking - Networking 2002 Tutorials*, Pisa, Italy, May 2002, pp. 20-58.
- [12] A. Carzaniga, "Architectures for an Event Notification Service Scalable to Wide-Area Networks," *PhD Thesis*, Politecnico di Milano, Milano, Italy, December 1998. Available from <http://www.cs.colorado.edu/~carzanig/papers/index.html>
- [13] A. Carzaniga and A. L. Wolf, "A Benchmark Suite for Distributed Publish/Subscribe Systems," *Technical Report CU-CS-927-02*, Department of Computer Science, University of Colorado, Boulder, CO, April 2002.
- [14] A. Carzaniga and A.L. Wolf, "Content-Based Networking: A New Communication Infrastructure," in *Proceedings of NSF Workshop on an Infrastructure for Mobile and Wireless Systems* in conjunction with the 10th IEEE International Conference on Computer Communications and Networks (ICCCN 2000), Scottsdale, AZ, October 2001, pp. 59-68.
- [15] A. Carzaniga and A. L. Wolf, "Fast Forwarding for Content-Based Networking," *Technical Report CU-CS-922-01*, Department of Computer Science, University of Colorado, Boulder, CO, November 2001. Revised, September 2002.
- [16] A. Carzaniga, D. S. Rosenblum and A. L. Wolf, "Achieving Expressiveness and Scalability in an Internet-Scale Event Notification Service," in *Proceedings of the 19th ACM Symposium on Principles of Distributed Computing (PODC 2000)*, Portland, OR, July 2000, pp. 219-227.
- [17] A. Carzaniga, D. S. Rosenblum and A. L. Wolf, "Content-Based Addressing and Routing: A General Model and its Application," *Technical Report CU-CS-902-00*, Department of Computer Science, University of Colorado, Boulder, CO, January 2000.
- [18] A. Carzaniga, D. S. Rosenblum and A. L. Wolf, "Design and Evaluation of a Wide-Area Event Notification Service," *ACM Transactions on Computer Systems*, Vol. 19, No. 3, August 2001, pp. 332-383.
- [19] A. Carzaniga, D. S. Rosenblum and A. L. Wolf, "Interfaces and Algorithms for a Wide-Area Event Notification Service," *Technical Report CU-CS-888-99*, Department of Computer Science, University of Colorado, Boulder, CO, October 1999. Revised May 2000.

- [20] A. Carzaniga, E. Di Nitto, D. S. Rosenblum and A. L. Wolf, "Issues in Supporting Event-Based Architectural Styles," in *Proceedings of the 3rd International Software Architecture Workshop (ISAW3)*, Orlando, FL, November 1998, pp. 17-20.
- [21] G. Cugola and E. Di Nitto, "Using a Publish/Subscribe Middleware to Support Mobile Computing," in *Proceedings of Workshop on Middleware for Mobile Computing*, Heidelberg, Germany, November 2001.
- [22] G. Cugola, E. Di Nitto and A. Fuggetta, "The JEDI Event-Based Infrastructure and its Application to the Development of the OPSS WFMS," *IEEE Transactions on Software Engineering*, Vol. 27, No. 9, September 2001, pp. 827-850.
- [23] P. T. Eugster, P. Felber, R. Guerraoui and A. M. Kermarrec, "The Many Faces of Publish/Subscribe," *Technical report DSC ID: 2000104*, Swiss Federal Institute of Technology (EPFL), Lausanne, Switzerland, January 2001.
- [24] European Telecommunications Standards Institute, "Digital Cellular Telecommunications System (Phase 2+); General Packet Radio Service (GPRS); Service Description; Stage 2," *Technical Specifications ETSI TS 101 344 v7.9.0*, September 2002. Available from <http://www.etsi.org>
- [25] European Telecommunications Standards Institute, "Universal Mobile Telecommunications System (UMTS); UTRAN Overall Description," *Technical Specifications ETSI TS 125 401 v5.3.0*, June 2002. Available from <http://www.etsi.org>
- [26] Exolab Group, *OpenJMS*, <http://openjms.sourceforge.net/license.html>
- [27] U. Farooq, S. Majumdar and E. W. Parsons, "High Performance Middleware for Mobile Wireless Publish/Subscribe Systems," submitted to *The 24th International Conference on Distributed Computing Systems (ICDCS 2004)*, Tokyo, Japan, March 2004.
- [28] U. Farooq, S. Majumdar and E. W. Parsons, "Semi-Durable Subscriptions: A Technique to Achieve High Performance in Mobile Wireless Publish/Subscribe Systems," *Technical Report*, Systems and Computer Engineering Department, Carleton University, Ottawa, Canada, July 2003.
- [29] P. Fenkam, E. Kirda, S. Dustdar, H. Gall and G. Reif, "Evaluation of a Publish/Subscribe System for Collaborative and Mobile Working," in *Proceedings of the 11th IEEE International Workshop on Enabling Technologies (WETICE'02)*, Pittsburgh, PA, June 2002, pp. 23-28.
- [30] L. Fiege, F. C. Gartner, O. Kasten and A. Zeidler, "Supporting Mobility in Content-Based Publish/Subscribe Middleware," in *Proceedings of ACM/IFIP/USENIX International Middleware Conference (Middleware 2003)*, Rio de Janeiro, Brazil, June 2003, pp. 103-134.

- [31] L. Fiege, G. Muhl and F. C. Gartner, "A Modular Approach to Build Structured Event-Based Systems," in *Proceedings of the 2002 ACM Symposium on Applied Computing (SAC'02)*, Madrid, Spain, March 2002, pp. 385-392.
- [32] Fiorano Software Inc., *FioranoMQ*, <http://www.fiorano.com/products/fmq>
- [33] H. Gomaa, *Designing Concurrent, Distributed and Real-Time Applications with UML*, Addison Wesley, 2000.
- [34] Y. Huang and H. Garcia-Molina, "Publish/Subscribe in a Mobile Environment," in *Proceedings of the 2nd ACM International Workshop on Data Engineering for Wireless and Mobile Access (MobiDE'01)*, Santa Barbara, CA, May 2001, pp. 27-34.
- [35] Y. Huang and H. Garcia-Molina, "Publish/Subscribe Tree Construction in Wireless Ad hoc Networks," in *Proceedings of the 4th International Conference on Mobile Data Management (MDM 2003)*, Melbourne, Australia, January 2003, pp. 122-140.
- [36] IBM Inc., *MQSERIES*, <http://www-4.ibm.com/software/ts/mqseries/>
- [37] Jahming Technologies, "High Performance Messaging with JMS: A Benchmark Comparison of SonicMQ 4.0 vs. IBM MQSeries 5.2," *Technical Report*, December 2001. Available from http://www.sonicsoftware.com/products/enterprise_messaging/whitepapers/index.ssp
- [38] R. Jain, *The Art of Computer Systems Performance Analysis*, John Wiley & Sons Inc., 1991.
- [39] JBoss Group, *JBoss*, <http://www.jboss.org>.
- [40] G. Muhl, L. Fiege, F. C. Gartner and A. P. Buchmann, "Evaluating Advanced Routing Algorithms for Content-Based Publish/Subscribe Systems," in *Proceedings of the 10th IEEE International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS 2002)*, Fort Worth, TX, USA, October 2002, pp. 167-178.
- [41] R. Meier and V. Cahill, "Steam: Event-Based Middleware for Wireless Ad hoc Networks," in *Proceedings of the 1st International Workshop on Distributed Event-Based Systems (DEBS '02)* in conjunction with the 22nd International Conference on Distributed Computing Systems (ICDCS'02), Vienna, Austria, July 2002, pp. 639-644.
- [42] R. Meier and V. Cahill, "Taxonomy of Distributed Event-Based Programming Systems," in *Proceedings of the 1st International Workshop on Distributed Event-Based Systems (DEBS '02)* in conjunction with the 22nd International Conference on Distributed Computing Systems (ICDCS'02), Vienna, Austria, July 2002, pp. 585-588.

- [43] R. Monson-Haefel, D. A. Chappell and M. Loukides, *Java Message Service*, O'Reilly & Associates, 2000.
- [44] National Institute of Standards and Technology, *NIST Network Emulation Tool*, <http://snad.ncsl.nist.gov/itg/nistnet/index.html>
- [45] Network Top, *NTOP*, <http://www.ntop.org/ntop.html>
- [46] Object Management Group, *Common Object Request Broker Architecture (CORBA/IIOP) v 3.0.2*, <http://www.omg.org>
- [47] Object Management Group, *CORBAServices: Common Object Service Specification*, <http://www.omg.org>
- [48] Object Web, *Joram*, <http://www.objectweb.org/joram>
- [49] L. Opyrchal, M. Astley, J. S. Auerbach, G. Banavar, R. E. Strom and D. C. Sturman, "Exploiting IP Multicast in Content-Based Publish/Subscribe Systems," in *Proceedings of IFIP/ACM International Conference on Distributed Systems Platforms and Open Distributed Processing (Middleware 2000)*, New York, NY, April 2000, pp. 185-207.
- [50] P. Pietzuch and J. Bacon, "Hermes: A Distributed Event-Based Middleware Architecture," in *Proceedings of the 1st International Workshop on Distributed Event-Based Systems (DEBS '02)* in conjunction with the 22nd International Conference on Distributed Computing Systems (ICDCS'02), Vienna, Austria, July 2002, pp. 611-618.
- [51] E. Pitt and K. McNiff, *Java.RMI: The Remote Method Invocation Guide*, Addison Wesley, 2001.
- [52] A. Pope, *The CORBA Reference Guide: Understanding the Common Object Request Broker Architecture*, Addison-Wesley, 1998.
- [53] D. Rogerson, *Inside COM*, Microsoft Press, 1997.
- [54] D. S. Rosenblum and A. Wolf, "A Design Framework for Internet-Scale Event Observation and Notification," in *Proceedings of the 6th European Software Engineering Conference* in conjunction with the 5th International Symposium on the Foundations of Software Engineering (SIGSOFT'97), Zurich, Switzerland, September 1997, pp. 344-360.
- [55] J. Schiller, *Mobile Communications*, Addison Wesley Professional, 1999.
- [56] B. Segall and D. Arnold, "Elvin has Left the Building: A Publish/Subscribe Notification Service with Quenching," in *Proceedings of AUUG97 Conference*, Brisbane, Australia, September 1997, pp. 243-255.

- [57] Softwired Inc., *iBus Mobile*, <http://www.softwiredinc.com/products/mobile/mobile.html>
- [58] Sonic Software Inc., *SONICMQ*, <http://www.sonicsoftware.com>
- [59] Sun Microsystems Inc., *Java Message Service*, <http://java.sun.com/products/jms>
- [60] Sun Microsystems Inc., "Java Message Service Tutorial," *Tutorial*. Available from <http://java.sun.com/products/jms/tutorial/index.html>
- [61] Sun Microsystems Inc., "Hierarchy for Package javax.jms," *JMS API Specifications v1.0.2*. Available from <http://java.sun.com/products/jms/javadoc-102a/javax/jms/package-tree.html>
- [62] The International Engineering Consortium, "UMTS Protocol and Protocol Testing". *WebProForum Tutorial*. Available from <http://www.iec.org>
- [63] TIBCO Software Inc., *TIB/RENDEZVOUS*, http://www.tibco.com/solutions/products/active_enterprise/rv/default.jsp
- [64] A. Zeidler and L. Fiege, "Mobility Support with REBECA," in *Proceedings of the 1st International Workshop on Mobile Computing Middleware(MCM'03)* in conjunction with the 23rd International Conference on Distributed Computing Systems (ICDCS'03), Providence, RI, May 2003, pp. 354-361.
- [65] Y. Zhao and R. Strom, "Exploiting Event Stream Interpretation in Publish/Subscribe Systems," in *Proceedings of the 20th ACM Symposium on Principles of Distributed Computing (PODC 2001)*, Newport, RI, August 2001, pp. 219-228.

Appendix A

A.1 Effect of Bandwidth and Mobility on Nondurable Subscriptions

This section gives the results on the effect of bandwidth and mobility on nondurable subscriptions, obtained with a closed model publisher. Other than the publishing model, all input parameters for these experiments are the same as given in Table 4.1. The results are shown in Figure A.1.

Figure A.1: Effect of Mobility and Bandwidth on Nondurable Subscriptions
(a) Throughput of Subscribers (b) Maximum Throughput of Publisher (c)
Proportion of Messages Lost

A.2 Effect of Mobility on Durable Subscriptions

This section lists the results on the effect of mobility on durable subscriptions obtained with a closed model publisher. All parameters used in these experiments are the same as given in Table 5.1 (except the publisher's system model). The results are shown in Figure A.2.

Figure A.2: Effect of Mobility on Durable Subscriptions
(a) Maximum Throughput of Publisher (b) Throughput of Subscribers (c)
Proportion of Messages Lost

A.3 Maximum Sustainable Throughput of the Three Subscription Types in the Server Bottlenecked Region

This section compares the maximum sustainable throughput of the three subscription types with a closed model publisher. The parameters used for these experiments are given in Table 6.1. The results are shown in Figure A.3.

Figure A.3: Maximum Sustainable Throughput of the Three Subscription Types in the Server Bottlenecked Region: (a) Maximum Throughput of Publisher (b) Throughput of Subscribers (c) Proportion of Messages Lost

A.4 Maximum Sustainable Throughput of Nondurable and Semi-Durable Subscriptions in the Bandwidth Bottlenecked Region

This section compares the maximum sustainable throughput of nondurable and durable subscriptions in the bandwidth bottlenecked region using a closed model publisher. The results are shown in Figure A.4.

Figure A.4: Maximum Sustainable Throughput of the Nondurable and Semi-durable Subscriptions in the Bandwidth Bottlenecked Region: (a) Maximum Throughput of Publisher (b) Throughput of Subscribers (c) Proportion of Messages Lost

A.5 Performance of Nondurable and Semi-Durable Subscriptions at a Given Publishing Rate in the Bandwidth Bottlenecked Region

This section compares the performance of three subscription types at a given publishing rate of 187.76 (the maximum rate semi-durable subscriptions can sustain with the default values of the parameters in the 220kB total bandwidth environment). The rest of the parameters are the same as in Table 6.1. The results are given in Figure A.5.

Figure A.5: Performance of Nondurable and Semi-Durable Subscriptions at a given Publishing Rate in the Bandwidth Bottlenecked Region
(a) Throughput of Subscribers (b) Proportion of Messages Lost