

Data Broadcasting for Wireless Databases

Agustinus Borgy Waluyo

This thesis is presented in total fulfilment of
the requirements of the degree of
Doctor of Philosophy

Faculty of Information Technology

Monash University

October 2005

Abstract

This thesis studies data broadcasting for wireless databases. Its main objective is to ameliorate the performance of database queries in a wireless broadcast environment. Specifically, it is concerned with optimising *query access time*, *client tuning time* and *power consumption* when obtaining on air broadcast database items.

Query access time refers to the response time from the time a query is issued until the time that query results are received. In this thesis, a method to minimise query access time for retrieving broadcast database items is presented. Optimisation is achieved by: (i) determining the optimum number of broadcast channels, and (ii) formulating data broadcast ordering and scheduling schemes. Modifications of these schemes with replication and the effect on performances are also studied.

In wireless computing, it is necessary to ensure power conservation of the clients through the reduction of client tuning time which also indicates its energy consumption. In light of this issue, this thesis presents index broadcasting schemes for a multiple channel environment, which have the ability to provide accurate information for a client to tune in at the appropriate time for the required data. As a

result, mobile clients are able to conserve energy by switching into power saving mode or doze mode and back to active mode when the data is about to be broadcast. These schemes are classified into two categories based on the type of queries involved, namely: indexing strategies for both Traditional Queries, and Location-Dependent Queries. Their performance is also presented in this thesis.

A hybrid model combining the models for optimising query access time, tuning time and power consumption is presented. The combined model represents the broadcast-based information system that can be utilised by wireless telecommunication providers to offer scalable, effective and efficient data broadcast delivery services to mobile clients. The prototype system has been built in a real wireless environment and it is used to represent an experimental test bed of the combined model and to measure and study the effectiveness and efficiency of the schemes. The evaluation of this model includes a comparison with conventional as well as existing schemes. The performance results suggest the superiority of this model in every aspect of the evaluation. The prototype system is also used to validate the simulation results and to ensure the correctness of simulations when larger parameters are applied. The outcome of this investigation can be employed for data broadcast delivery services which are tailored to serve traditional as well as location dependent queries to mobile clients.

Declaration

This thesis contains no material which has been accepted for the award of any other degree or diploma in any university. The material presented in this thesis is the product of the author's own independent research under the supervision of *Professor Bala Srinivasan* and *Dr David Taniar*.

A fair amount of the materials presented in this thesis has been published in various refereed journals and conference proceedings.

Agustinus Borgy Waluyo

October 2005

List of Publications

1. **Waluyo, A.B.**, Srinivasan, B. and Taniar, D., "Research in Mobile Database Query Optimization and Processing", *Mobile Information Systems*, 1(4), 2005 (in press).
2. **Waluyo, A.B.**, Srinivasan, B., and Taniar, D., "Efficient Broadcast Indexing Scheme for Location-Dependent Queries in Multi Channels Wireless Environment", *Journal of Interconnection Networks - Special Issue on P2P Systems and Distributed Applications*, 6(3): 303-322, 2005.
3. **Waluyo, A.B.**, Goh, G., Srinivasan, B., and Taniar, D., "On-Building Data Broadcast System in a Wireless Environment", *International Journal of Business Data Communications and Networking*, Idea Group Publishing, USA 1(4), 14-36, 2005.
4. **Waluyo, A.B.**, Srinivasan, B., and Taniar, D., "Indexing Schemes for Multi Channel Data Broadcasting in Mobile Databases", *International Journal of Wireless and Mobile Computing*, 1(6), 2005.
5. **Waluyo, A.B.**, Srinivasan, B., and Taniar, D., "Research on Location-Dependent Queries in Mobile Databases", *International Journal on Computer Systems: Science and Engineering*, 20(3): 77-93, 2005.
6. **Waluyo, A.B.**, Srinivasan, B., and Taniar, D., "Data Dissemination in Mobile Databases", *Encyclopedia of Information Science and Technology*, Idea Group Publishing, USA, Jan 2005.
7. **Waluyo, A.B.**, Srinivasan, B., and Taniar, D., "Global Indexing Scheme for Multi Indexing Channels in a Mobile Computing Environment", *Radiomatics - Journal on Communication Engineering*, 1(1): 18-29, 2004.

8. **Waluyo, A.B.**, Srinivasan, B., Taniar, D. and Rahayu W., "Incorporating Global Index with Data Placement Scheme for Multi Channels Mobile Broadcast Environment ", In *Proceedings of the Embedded and Ubiquitous Computing (EUC'05), Lecture Notes in Computer Science*, Springer Verlag, 2005 (in press).
9. **Waluyo, A.B.**, Srinivasan, B., and Taniar, D., " Global Indexing Scheme for Location-Dependent Queries in Multi Channels Mobile Broadcast Environment", In *Proceedings of the 19th International Conference on Advanced Information Networking and Applications (AINA'05)*, IEEE Computer Society Press, vol 1, pp.1011-1016, 2005 (**Awarded as Best Paper**).
10. **Waluyo, A.B.**, Goh G., Taniar, D., and Srinivasan, B. "On Building a Data Broadcasting System for Mobile Databases ", In *Proceedings of the 19th International Conference on Advanced Information Networking and Applications (AINA'05)*, IEEE Computer Society Press, Vol 1, pp.538-543, 2005.
11. **Waluyo, A.B.**, Goh G., Srinivasan, B., and Taniar, D., "Applying Multi-Channelling and Indexing Scheme for Efficient Data Retrieval in Wireless Broadcast Environment ", In *Proceedings of the second International Conference on Intelligent Sensing and Information Processing (ICISIP'05)*, IEEE Computer Society Press, pp.32-37, 2005.
12. **Waluyo, A.B.**, Srinivasan, B., and Taniar, D., " Data Items Ordering Scheme for Multi Channel Data Broadcasting in a Mobile Computing Environment", In *Proceedings of the International Conference on Advances in Mobile Multimedia (MoMM'04)*", ISBN 3-85403-184-X, pp.371-380, 2004
13. **Waluyo, A.B.**, Srinivasan, B., and Taniar, D., "Location Dependent Queries in Mobile Databases ", In *Proceedings of the International Conference on Information and Knowledge Engineering (IKE'04)*, ISBN 1-932415-27-0, pp. 362-370, 2004.
14. **Waluyo, A.B.**, Srinivasan, B., and Taniar, D., "A Taxonomy of Broadcast Indexing Schemes for Multi Channel Data Dissemination in Mobile Databases", In *Proceedings of the 18th International Conference on Advanced Information Networking and Applications (AINA 2004)*, Vol. 1, IEEE Computer Society Press, pp. 213-218, 2004.

15. **Waluyo A.B.**, Hsieh R., Taniar D., Rahayu W., and Srinivasan B., "Utilising Push and Pull Mechanism In Wireless E-Health Environment", In *Proceedings of the IEEE International Conference on e-Technology, e-Commerce and e-Service (EEE-04)*, IEEE Computer Society Press, pp. 271 - 274, 2004.
16. **Waluyo, A.B.**, Srinivasan, B., and Taniar, D., "Allocation of Data Items for Multi Channel Data Broadcast in Mobile Databases", In *Proceedings of the Embedded and Ubiquitous Computing (EUC'04), Lecture Notes in Computer Science*, vol. 3207, Springer-Verlag, pp. 409-418, 2004.
17. **Waluyo, A.B.**, Srinivasan, B., and Taniar, D., "Optimising Query Access Time over Broadcast Channel in Mobile Databases", In *Proceedings of the Embedded and Ubiquitous Computing (EUC'04), Lecture Notes in Computer Science*, vol. 3207, Springer-Verlag, pp. 439-449, 2004.
18. **Waluyo, A.B.**, Srinivasan, B., and Taniar, D., "Optimal Broadcast Channel for Data Dissemination in Mobile Database Environment", In *Proceedings of the Advanced Parallel Processing Technologies (APPT'03), Lecture Notes in Computer Science*, vol. 2834, Springer-Verlag, pp. 655-664, 2003.
19. **Waluyo, A.B.**, Srinivasan, B., and Taniar, D., "Global Index for Multi Channel Data Dissemination in Mobile Databases", In *Proceedings of the 18'th Computer and Information Sciences (ISCIS'03), Lecture Notes in Computer Science*, vol. 2869, Springer-Verlag, pp. 210-217, 2003.
20. **Waluyo, A.B.**, Srinivasan, B., and Taniar, D., "Mobile Query and Processing in Mobile Database Environment", In *Proceedings of the International Conference on Advances in Mobile Multimedia (MoMM'03)*, ISBN: 3-85403-171-8, pp. 55-64, 2003.
21. **Waluyo, A.B.**, Srinivasan, B., and Taniar, D., "Index Dissemination for Multi Broadcast Channel in Mobile Databases", In *Proceedings of the International Conference on Advances in Mobile Multimedia (MoMM'03)*, ISBN: 3-85403-171-8, pp. 65-74, 2003.

Acknowledgment

This thesis would not have been possible without invaluable guidance, support, continual encouragement, discussion, contribution to ideas and advice from my two supervisors, Prof. Bala Srinivasan and Dr. David Taniar, to whom I am deeply indebted. I am a much better researcher because of their excellent guidance.

I would like to dedicate this thesis to my parents to whom I owe my success in life and also to the memory of my late grandfather who left us in 2004. I gratefully acknowledge the financial support throughout my candidature from a Monash Graduate Scholarship (MGS) sponsored by Monash University.

I owe special thanks to my friend Ling Tan for many hours of pleasant conversations, research discussions, and perspective on life. I am glad to have a nice officemate and friend, Guy Kijthaweessinpoon. I also thank Samar Zutshi and Bruna Pomella for correcting grammatical and spelling mistakes.

I thank my family for their encouragement and prayers. I owe very special thanks to my sister, Wenny, for her tremendous support on my living since I started my study in Melbourne. Last but not least, I thank GOD for giving me the knowledge and for letting me achieve this academic level, which is truly quite beyond my dreams.

Contents

Abstract	<i>i</i>
Declaration	<i>iii</i>
List of Publications	<i>iv</i>
Acknowledgment	<i>vii</i>
Contents	<i>viii</i>
List of Figures	<i>xi</i>
List of Tables	<i>xvi</i>
1 Introduction	1
1.1 Background	1
1.2 Motivation	5
1.3 Objectives of this Thesis	9
1.4 Scope of the Research	10
1.5 Contributions of this Thesis	12
1.6 Thesis Organization	14
2 Query Taxonomy, Optimisation and Processing in the Mobile Database Environment	18
2.1 Introduction	18
2.2 Preliminaries	20
2.2.1 Wireless Communication Network	21
2.2.2 Location Positioning System	25
2.3 Query Taxonomy in Mobile Database Environment	27
2.3.1 Location-dependent Queries	29
2.3.2 Context-dependent Queries	38
2.3.3 Location- and Context-dependent Queries	39
2.4 Query Optimisation and Processing in Mobile Database Environment	40
2.4.1 Traditional Queries	41

2.4.1.1	On-demand Mechanism	42
2.4.1.2	Data Broadcasting Mechanism (on-demand broadcast)	43
2.4.1.3	Data Broadcasting Mechanism (periodic broadcast)	49
2.4.1.4	Hybrid On-demand and Broadcasting Mechanism	61
2.4.1.5	Other Strategies	63
2.4.2	Location-Dependent Queries	64
2.4.2.1	On-demand Mechanism	65
2.4.2.2	Data Broadcasting Mechanism	76
2.5	Discussion	82
2.6	Data Broadcast Management Framework.	87
2.7	Conclusion.	89
3	Optimising Channel Utilisation	91
3.1	Introduction	91
3.2	Preliminaries	92
3.3	Determining Optimum Number of Broadcast Channel	96
3.4	Formulating Data Broadcast Ordering and Scheduling Schemes	104
3.4.1	Without Replication.	106
3.4.2	With Replication.	112
3.5	Performance Evaluation	118
3.5.1	Determining Optimum Number of Broadcast Channel	119
3.5.2	Formulating Data Broadcast Ordering and Scheduling Schemes	126
3.6	Discussion.	133
3.7	Conclusion.	134
4	Indexing Schemes for Multi Channel Data Broadcasting.	138
4.1	Introduction	138
4.2	Preliminaries	140
4.3	Index Broadcasting for Traditional Queries	145
4.3.1	Non-Replicated Scheme (NRI)	150
4.3.2	Partially-Replicated Scheme (PRI)	153
4.3.3	Fully-Replicated Scheme (FRI)	158
4.4	Index Broadcasting for Location-Dependent Queries.	159
4.4.1	Preliminaries	159
4.4.2	Global Index for Location-Dependent Queries	160
4.5	Performance Evaluation	169
4.5.1	Traditional Queries	169

4.5.2	Location-Dependent Queries	184
4.6	Index Maintenance	190
4.7	Discussion	198
4.8	Conclusion.	201
5	Hybrid Model	203
5.1	Introduction	203
5.2	Hybrid Model	205
5.3	Performance Evaluations	209
5.4	Prototype Model	210
5.5	Experimental Results	214
5.5.1	Extended Simulation	169
5.5.2	Performance Analysis of each Proposed Strategy.	184
5.6	Discussion	229
5.7	Conclusion.	230
6	Conclusion	232
6.1	Background	232
6.2	Summary of the Research Result	233
6.2.1	Minimising Query Access Time.	169
6.2.2	Minimising Client Tuning Time and Power Consumption	184
6.2.3	Optimising Query Access Time, Client Tuning Time and Power Consumption	169
6.2.4	Performance Evaluation	184
6.3	Future Research	237
	References	244
	Appendix A Simulation Model	254
	Appendix B Prototype of Data Broadcast System	264

List of Figures

1.1	Thesis framework	11
1.2	Thesis contributions	12
1.3	Thesis structure	15
2.1	Outline of this Chapter	20
2.2	Mobile computing environment	21
2.3	Query taxonomy in mobile database environment.	27
2.4	Single-cell movement (moving user seeking static object/s).	32
2.5	Multi-cell movement (moving user seeking static object/s)	32
2.6	Single-cell movement (static user seeking moving object/s)	34
2.7	Multi-cell movement (static user seeking moving object/s)	34
2.8	Single-cell movement (moving user seeking moving object/s).	35
2.9	Multi-cell movement (moving user seeking moving object/s)	35
2.10	Continuous query	37
2.11	Non-continuous query	38
2.12	Query processing in mobile databases.	40
2.13	Query processing for traditional queries	42
2.14	The architecture of on-demand mechanism	42
2.15	The architecture of on-demand broadcast mechanism	43
2.16	The architecture of periodic broadcast mechanism.	49
2.17	Multiple channels architecture	55
2.18	Data Stripping, Partitioning, and Replication Mechanism	56
2.19	A full index tree	59
2.20	Multi-level signature	61
2.21	Hybrid indexing	61

2.22	Query execution sites	64
2.23	Query processing for location-dependent queries	64
2.24	Valid scopes	66
2.25	Location-based concept hierarchy.	68
2.26	Query scheduling for multi-cell movement	71
2.27	Location tracking architecture	72
2.28	Single and multiple channels broadcast structure	79
2.29	Tasks of data management scheme with relation to the proposed method in this thesis	88
3.1	Chapter framework	93
3.2	Query access time	93
3.3	Multiple channels architecture	97
3.4	Broadcast channel partitioning: an example	98
3.5	Broadcast cycle partitioned area	99
3.6	Broadcast program – with and without replication	105
3.7	Broadcast ordering scheme –without replication	107
3.8	Stage 1 algorithm – without replication	108
3.9	The architecture of the proposed method	109
3.10	An example of the proposed method process	109
3.11	Stage 2 algorithm– without replication.	110
3.12	Final stage algorithm– without replication	111
3.13	Broadcast ordering scheme – with replication	113
3.14	Stage 1 algorithm– with replication	114
3.15	Stage 2 algorithm – with replication	115
3.16	Stage 3 algorithm – with replication	116
3.17	Final stage algorithm– with replication	117
3.18	Optimum number of broadcast items with pre-determined number of request (uniform and non-uniform request)	122
3.19	Optimum number of broadcast items with server utilisation $\rho < 1$ (uniform and non-uniform request)	124
3.20	Optimum number of broadcast items with server utilisation $\rho > 1$ (uniform and non-uniform request)	125
3.21	Conventional vs. proposed method (30 broadcast data items)	129
3.22	Conventional vs. proposed method (45 broadcast data items).	129
3.23	Conventional vs. proposed method (20 broadcast data items)	131
3.24	Conventional vs. proposed method (40 broadcast data items)	131

3.25	Proposed method (uniform vs. skewed access distribution): 20 broadcast data items	132
3.26	Proposed method (uniform vs. skewed access distribution): 40 broadcast data items.	133
4.1	Chapter framework	141
4.2	Integrated and separated index channel	144
4.3	A sample tree index	146
4.4	Tree index in broadcast channel environment	146
4.5	Index construction and restructuring stages	148
4.6	Data channel structure and index using B^+ tree.	149
4.7	Post Order Traversal - NRI	150
4.8	Post Order Traversal algorithm with index allocation scheme (NRI)	151
4.9	Non-replicated indexing scheme	152
4.10	Post Order Traversal - PRI	154
4.11	Post Order Traversal algorithm with index allocation scheme (PRI)	154
4.12	Construction and restructuring of partially-replicated indexing scheme	155
4.13	Partially-replicated indexing scheme	156
4.14	Index construction.	162
4.15	Unbalanced index model – Row	163
4.16	Unbalanced index model - Column	164
4.17	Post Order Traversal with index allocation scheme (regions based).	166
4.18	Global index model for location-dependent query with structuring process	167
4.19	Index construction – a scenario	168
4.20	Global index model with a scenario	168
4.21	NRI vs. PRI vs. FRI (single data item retrieval)	171
4.22	FRI vs. PRI vs. NRI (two data items retrieval).	172
4.23	(a) PRI vs. conventional scheme (single data item retrieval).	173
	(b) PRI vs. conventional scheme (two data items retrieval)	173
4.24	(a) PRI vs. PRI-50 (single data item retrieval)	174
	(b) PRI vs. PRI-50 (two data items retrieval)	175
4.25	(a) Global Index vs. Non-Global index (single data item retrieval): 4 index pages per sec.	177
	(b) Global Index vs. Non-Global index (single data item retrieval) with skew request distribution: 4 index pages arrival rate per sec	177

4.26	(a) Global Index vs. Non-Global index (single data item retrieval): 2 index pages per sec.	178
	(b) Global Index vs. Non-Global index with (single data item retrieval) with skew request distribution: 2 index pages arrival rate per sec. . . .	178
4.27	(a) Global Index vs. Non-Global index (two data items retrieval): 4 index pages per sec.	179
	(b) Global Index vs. Non-Global index (two data items retrieval) with skew request distribution: 4 index pages arrival rate per sec . . .	180
4.28	(a) Global Index vs. Non-Global index (two data items retrieval): 2 index pages per sec.	181
	(b) Global Index vs. Non-Global index (two data items retrieval) with skew request distribution: 2 index pages arrival rate per sec . . .	181
4.29	Tuning time: Global Index vs. Non-Global index	182
4.30	Power consumption: Global Index vs. Non-Global index.	183
4.31	Polygon valid scope	185
4.32	Square vs. Polygon valid scope	185
4.33	(a) Global index vs. Non-Global index (balanced tree): access time . . .	187
	(b) Global index vs. Non-Global index (balanced tree): tuning time . . .	187
	(c) Global index vs. Non-Global index (balanced tree): power consumption	188
4.34	(a) Global index vs. Non-Global index (unbalanced tree): access time . . .	189
	(b) Global index vs. Non-Global index (unbalanced tree): tuning time . . .	189
	(c) Global index vs. Non-Global index (unbalanced tree): power consumption	190
4.35	Architecture of the proposed broadcast system	192
4.36	Index entry insertion.	195
4.37	Index entry deletion	197
5.1	Chapter framework	205
5.2	Architecture of a broadcast system	207
5.3	Data and index broadcasting in the hybrid model	207
5.4	Access time and tuning time calculation of the hybrid model	208
5.5	Prototype components	212
5.6	Initialisation phase	213
5.7	Client in power saving mode	213
5.8	Client receiving index information and partial disconnection	214

5.9	Client obtaining the selected share prices	214
5.10	Query access time: proposed vs. existing method (simulated)	218
5.11	Query access time: proposed vs. existing method (prototype)	218
5.12	Simulated vs. Prototype: results comparison	219
5.13	Tuning time: Global index vs. Non-Global index (prototype).	220
5.14	Power consumption: Global Index vs. Non-Global Index.	220
5.15	(a) Proposed method vs. conventional method: 30 number of broadcast data items	223
	(b) Proposed method vs. conventional method: 45 number of broadcast data items	223
5.16	(a) Proposed method vs. existing method: 30 number broadcast data items	224
	(b) Proposed method vs. existing method: 45 number broadcast data items	225
5.17	Tuning time: Global index vs. Non-Global index	226
5.18	Power consumption: Global index vs. Non-Global index	226
5.19	Proposed data broadcast management schemes	231

List of Tables

2.1	Comparison among the proposed scheme and related studies – Traditional Queries	85
2.2	Comparison among the proposed scheme and related studies – Location Dependent Queries	87
3.1	Uniform and Non-Uniform formulas	103
3.2	Parameters of concern - determining optimum number of channels . . .	119
3.3	Set of parameters for Uniform and Non-Uniform request	120
3.4	Average access time of on-demand and broadcast channel (pre-determined number of request with uniform request)	123
3.5	Average access time of on-demand and broadcast channel (Pre-determined number of request with non-uniform request)	123
3.6	Simulation and analytical performance in determining optimum number of broadcast items	126
3.7	Parameters of concern – broadcast ordering and scheduling scheme without replication	128
3.8	Parameters of concern – broadcast ordering and scheduling and ordering scheme with replication.	130
4.1	Parameters of concern – PRI, NRI, FRI	170
4.2	Parameters of concern – Global Index and Non-Global Index (Traditional Queries)	176
4.3	Parameters of concern – Location Dependent Queries	184
5.1	Parameters of concern – hybrid model	216
5.2	Simulated vs. prototype experimental results	221
5.3	Parameters of concern – extended simulation	222
5.4	Performance analysis for each proposed strategy.	228

Chapter 1

Introduction

1.1 Background

In recent years, the use of wireless technology devices has been growing at an exponential rate. According to an analysis, by 2006 there will be over 760 million mobile users connected to the Internet and over 1.7 billion users by 2007 (Drews et al, 2003). Most people are now able to access information systems located in wired networks anywhere and anytime using portable size wireless computing devices powered by batteries (e.g. notebooks, tablet PCs, personal digital assistants (PDAs) and GPRS-enabled cellular phones). These portable computing devices communicate with a central stationary server via a wireless channel and become the integrated part of the existing distributed computing environment. Subsequently, mobile clients can have access to database information systems located at the static network while they are travelling and this type of computing is known as *mobile computing* (Barbara,

1999; Myers and Beigl, 2003). Mobile computing provides *database applications* with useful aspects of wireless technology, and a subset of mobile computing that focuses on querying central database servers is referred to as mobile databases (Barbara, 1999; Malladi, et al 2002).

Mobile service providers are establishing a number of information services including weather information or weather forecast services, news, stock indices information, foreign exchange, election results, tourist services, airline schedules, location-dependent query, and route guidance, to name a few (<http://www.wapforum.org/>). In order to realise the potential of wireless information services, a number of issues and challenges need to be addressed including mobile data management (Barbara, 1999), cache management (Barbara and Imielinski, 1994), wireless network infrastructure (Bria et al, 2001), location-dependent data management (Lee et al, 2002), power management issues (Stan, and Skadron, 2003) and data broadcasting issues (Imielinski, Viswanathan and Badrinath, 1994).

Queries in a mobile environment can be classified into traditional queries, and location-dependent queries. Traditional queries are those queries that we normally invoke in traditional database systems on a non-wireless communication network environment. The only difference is that now these queries and their results are transmitted over a wireless communication network. However, queries on wireless database systems have much more variety, something which does not exist in traditional non-wireless databases, such as location-dependent queries. A query such as retrieving local tourist attractions considers the user's location, whereas a query such as retrieving taxis within a radius of 5km deals with moving objects. It is

realized that location-dependent queries will soon become common and of great interest. Consequently, providing efficient and effective mobile information services that cover both traditional queries and location-dependent queries will be highly desirable.

Despite the complexity involved in processing the mobile queries, a mobile computing environment also possesses several novel characteristics, which make it more challenging than a traditional distributed system. These characteristics include:

- *Resource constrained mobile devices*: To provide better portability and improve attractiveness, mobile devices are becoming smaller and lighter. However, such designs usually involve some trade-offs including low battery life, low computational power and smaller storage capacity. Especially with battery power, the life expectancy of a battery (e.g. nickel-cadmium, lithium ion) was estimated to increase the time of effective use by only another 15% in several years to come (Paulson, 2003). Furthermore, it should be noted that wireless data transmission requires a greater amount of power or up to 10 times as much power as the reception operation (Zaslavsky and Tari, 1998 ; Xu et al, 2002).
- *Low network bandwidth*: Mobile clients can connect to the fixed network via various wireless communication networks including wireless radio, wireless Local Area Network (LAN), wireless cellular, satellite, etc. Each of the wireless networks provides a different bandwidth capacity. However, this wireless bandwidth is too small compared with a fixed network such as ATM (Asynchronous Transfer Mode) that can provide speeds of up to 155Mbps (Elmasri and Navathe, 2003). Designing a high network utilization data access

method to provide an acceptable response time becomes an important issue in the mobile computing literature.

- *Asymmetric communication cost*: The different bandwidth capacity between the downstream communication and upstream communication has created a new environment called *Asymmetric Communication Environment*. In fact, there are two situations that can lead to communication asymmetry (Acharya, et al, 1995). One is due to the capability of physical devices. For example, servers have powerful broadcast transmitters, while mobile clients have little transmission capability. The other is due to the patterns of information flow in the applications. For instance, in a situation where the number of servers is far fewer than the number of clients, it is asymmetric because there is not enough capacity to handle simultaneous requests from multiple clients.
- *Heterogeneity of mobile devices*: Mobile telecommunication industries have developed a large variety of mobile devices such as Laptops, Tablet PC, Handheld PCs, Pocket PC, and Mobile Phones. However, these mobile devices have also various features and capabilities such as operating system, computational power, display and network capability. Consequently, this heterogeneity raises some challenges in content management and content delivery to the mobile service providers.
- *Mobility*: Wireless technology enables mobile users to move freely and independently from one place to another. A service handoff occurs when a user moves from one network service area into another. It is essential to ensure service handoffs seamlessly and transparently to the users.

- *Frequent Disconnections*: Mobile users are frequently disconnected from the network. This may be due to several reasons including signal failures, empty network coverage, and power saving. The later reason is advantageous since active mode requires thousand times more power than doze or power saving mode ((Imielinski, Viswanathan and Badrinath, 1994).). Wireless radio signals may also be weakened due to the client's further distance from the base station or the speed at which the client is moving.

In light of the complexity of query processing, as well as the retained characteristics of a mobile environment, it is essential to have an effective data delivery mechanism that is able to manage all of the above issues.

1.2 Motivation

The processing of queries in a mobile environment can be generally classified into: (i) on-demand queries, and (ii) broadcast-based queries. On-demand queries are those where the client initiates the query and sends the query to the server. The server processes the query, and sends the result back to the client. Broadcast-based queries are when the server broadcasts the data instances periodically through one or more broadcast channels. Mobile clients capture and select data items that are of interest to them or to this query (Acharya, et al, 1995; Bar-Noy , Naor , and Schieber , 2000 ;Imielinski, Viswanathan, and Badrinath, 1997).

With broadcast-based queries, a mobile client is able to retrieve information without wasting power to transmit a request to the server. Other characteristics include scalability as it supports a large number of queries; query performance is not affected

by the number of users in a cell or the request rate, and it is effective despite a high degree of overlap in the user's request.

The behaviour of the broadcast-based information system is unidirectional which means the server disseminates a set of data periodically to a multiple number of users. With this mechanism, the requests from the clients are not known *a priori*. However, this mechanism can be exploited to overcome the resource limitations in a mobile environment, as follows:

- *Power preservation*: Wireless transmission usually requires approximately 10 times the amount of power as compared with the reception operation (Zaslavsky and Tari, 1998). Hence, with broadcast-based systems, power consumption can be substantially conserved since mobile clients do not need to explicitly send data requests to information servers, but need only to listen to the relevant data broadcast channel and retrieve the desired data items.
- *Scalability*: Broadcast-based systems support a large number of queries and the query performance is not affected by the number of users in a cell as well as the request rate. Moreover, it is effective to a high degree of overlap in the user's request.
- *High bandwidth utilization*: This can be achieved since data broadcasting can be obtained by a multiple numbers of mobile clients in a single transmission on the broadcast channel. Moreover, as a mobile environment faces asymmetric communication costs, the utilisation of a downstream bandwidth to disseminate information without usage of the upstream one is of great interest.

- *Cost efficiency*: As a wireless communication system essentially applies a broadcast component to deliver data services, a broadcast-based information system can be implemented without introducing extra cost (Xu et al , 2002).

Additionally, with a broadcast-based information system, the mobile devices' heterogeneity issue can be eased considering that clients are not necessarily required to establish permanent connection with the server during querying and have the right application to listen to the relevant broadcast channel. This case is similar to the radio and television industry where consumers are required to select only the desired channel. When clients are disconnected from the network during query processing, they can simply repeat the process when they reconnect without worrying about the large amount of power consumption required to send the request back to the server as is the case with the traditional client-server application. Therefore, data broadcasting is a very promising mechanism for information delivery services in a mobile environment.

However, the main problem with broadcast services is that the performance of data broadcast degrades as the number of data items being broadcast increases. This thesis is mainly concerned with this issue and proposes techniques and algorithms to ameliorate the performance of retrieving database items over multiple broadcast channels environment. Note that multiple-channels environment is of importance due to the following reasons (Prabhara, Hua, and Oh, 2000 ; Yee and Navathe, 2003; Huang and Chen, 2004):

- *Application Scalability*: Considering an application that is running on the server and needs to be scaled to support a very large number of clients. In this situation,

additional physical channels will be required. If these channels are in non-contiguous frequencies, they need to be treated as separate channels when broadcasting data items.

- *Fault tolerance*: Considering a station that has more than one server with a transmission capability and those servers broadcast data in different non-contiguous frequencies in the same cell. If one server is down, the frequency can be allocated to the other server.
- *Heterogenous clients*: Mobile devices are manufactured by many mobile telecommunication industries. Consequently, these mobile devices have also various features and capabilities including heterogenous communication capabilities, thereby precluding the possibility of having a single high-speed transmission channel.
- *Reconfiguration of adjoining cells*: Considering when there are two adjacent cells whose servers transmit in different frequency ranges. At some point it is decided to merge the two cells and use one of the two servers to serve the new cell. In this case, the frequency range of the other server should be migrated and added to the residual server; consequently, the residual server gets multiple physical channels.
- *Network Standard*: Many network standards including FDMA-based systems partition the network bandwidth into several physical channels while individual mobile clients are designed to listen to at most one physical channel at a time. Consequently, in such network environment, multiple physical channels (even with contiguous frequency) cannot be merged into one single channel.

1.3 Objectives of this Thesis

Although the emergence of wireless technology has a very promising market, the mobile database is facing a number of resource constraints, in particular: restricted battery life, bandwidth, computational power and smaller storage. Consequently, the need to use power efficiently and effectively is a crucial issue. Moreover, it is realized that most applications in a mobile environment involve read operation rather than write operation (Huang, Sistla and Wolfson, 1994), therefore it is of utmost important to achieve high performance (latency time) while minimizing power consumption.

Wireless broadcasting technology has been applied for decades in the radio and television industries and it is very effective in coping with a massive number of clients. However, due to the sequential access of broadcast data items in a mobile environment, the increasing number of broadcast items causes mobile clients to wait for a substantial amount of time before receiving desired data item. Consequently, the advantages of broadcast strategy will be diminished.

Three factors need to be considered to address the issue of conserving battery power and maintaining short access latency and they are:

- *Access time*: Elapse time from the time a request is initiated until all data items of interest are received.
- *Tuning time*: Amount of time spent by the client to listen for the desired broadcast data item(s). Tuning time comprises of two modes: active and doze mode. Active mode is when the client listens to the channel for the desired data

item, hence, costly to power consumption, while doze mode is when clients simply turned into a power saving mode.

- *Power Consumption:* Amount of battery power required during query operation to filter out the desired data items on air. Generally, the amount of power consumption is directly related to the tuning time.

The *main objective* of this research is to investigate novel data broadcast management schemes. The proposed schemes are designed to minimise the above three factors: access time, tuning time and power consumption of mobile clients when querying data items on air considering multiple broadcast channels environment.

Specifically, this thesis will investigate a model to obtain an optimum solution for broadcast channels, taking into consideration the number of channels, number of data items requested, data ordering and replication, index structures, and data/index channel composition. The model is fundamental for optimizing access and tuning times, which will make a significant impact to power consumption.

1.4 Scope of the Research

A broadcast-based information system is illustrated in Figure 1.1. It shows how the data broadcast mechanism works in a wireless environment. A server which is located in a fixed wired network retrieves database items from data repository. These data items are disseminated over broadcast channel at regular intervals by a transmitter. A complete broadcast file is referred to as a broadcast cycle. Prior to disseminating the broadcast file, a server is able to perform a number of processing functions including

broadcast scheduling, determine the broadcast channel, addition, deleting, and re-ordering of broadcast data items, and constructing the necessary index structure.

The scope of this research falls within the data broadcast management stage as identified in Figure 1.1. Data broadcast management schemes at the server side are proposed. These schemes are concerned with strategies to deliver broadcast services in such a way that the client is able to receive services by optimising the access time, tuning time and power consumption. Subsequently, these schemes determine how clients should retrieve and process the data broadcast items on air. The result of this investigation can be utilised by wireless telecommunication provider to offer scalable, effective and efficient data broadcast delivery services to mobile clients. In this thesis, the term mobile client, mobile computer, mobile unit, mobile user and client are used interchangeably.

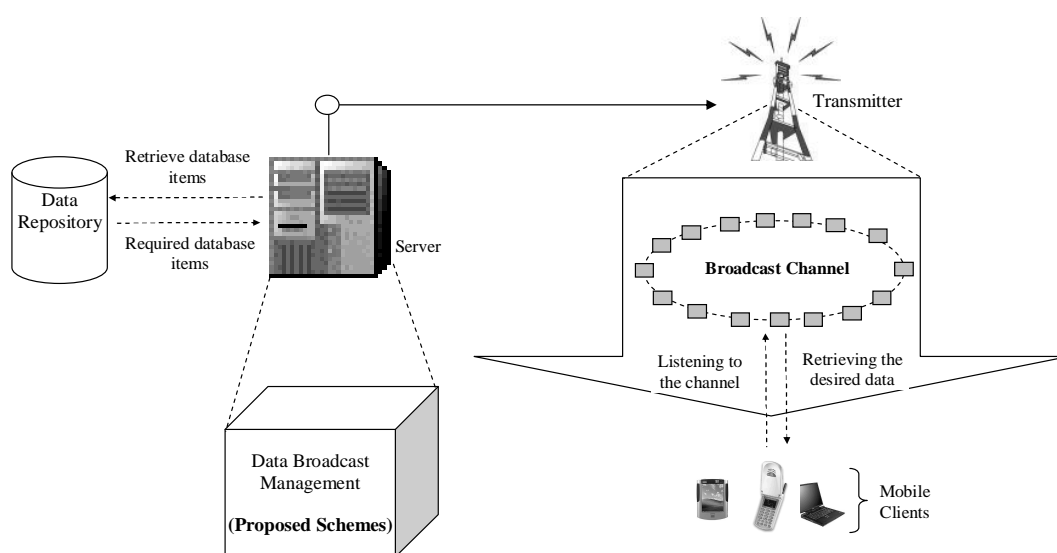


Figure 1.1. Thesis framework

1.5 Contributions of this Thesis

The specific contributions of this thesis are listed below, and the relationships between the contributions and the research scope of the thesis are shown in Figure 1.2. These contributions correspond to the three factors as described earlier, which are: minimising query access time, client's tuning time and power consumption.

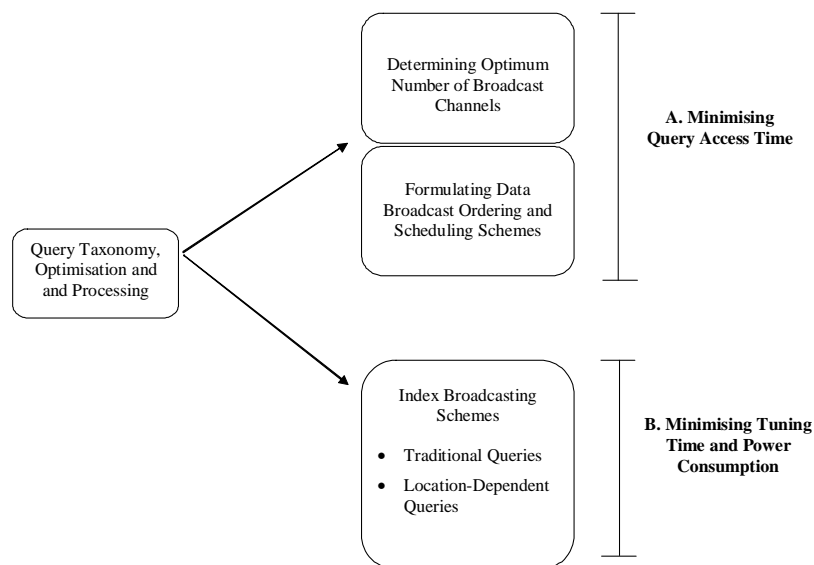


Figure 1.2. Thesis contributions

- **Query Taxonomy, Optimisation and Processing**

A comprehensive taxonomy and a classification of query optimisation and processing in a mobile database environment are provided. A comprehensive study of mobile queries gives not only an understanding of mobile user behaviors but also a direction for query processing and optimization. The main aim of such query taxonomy is to define a framework for mobile query optimization. In a mobile environment, it is necessary to take into account other aspects of mobility, including the location of the mobile clients and their movements. These queries are then location-dependent,

where the location of a mobile client becomes a parameter of the query. This issue expands considerably the complexity of query processing requirements.

- **Determining Optimum Number of Data Broadcast Channels**

Traditionally, in broadcasting technique, the sets of database items are periodically broadcast to multiple clients over a single or multiple channels. The broadcast of database items over multiple channels, an algorithm to find the optimal number of broadcast channels in a particular point in time depending on the number of users in a cell, and query access patterns will be studied. This will enable us to determine optimum channel allocation to broadcast a set of data items.

- **Formulating Data Broadcast Ordering and Scheduling Schemes**

A broadcast scheduling and ordering scheme for multi-broadcast channels is proposed. The proposed scheme is designed to keep the most requested data items close to each other in the channel. The broadcast program *with replication* and *without replication* is considered. A broadcast program with replication corresponds to the case where the data items appear with different frequencies. Generally, the most popular data items will be broadcast more often than others. On the other hand, a broadcast program is without replication when all data items are broadcast with equal frequencies or uniform frequencies.

- **Index Broadcasting Schemes**

Index broadcasting schemes are classified into two categories based on the type of queries involved namely Traditional Queries, and Location-Dependent Queries. Various indexing schemes for Traditional Queries are presented including the Global Indexing scheme for a multi-broadcast channels environment, with the objective of

minimising clients' tuning time and power consumption while maintaining a minimum query access time.

As mobility is one of the unique characteristics of a wireless environment, the kind of information requested is generally location-dependent; that is, the mobile client's location is relevant to the information requested or the information requested is based on a particular location (Lee et al, 2002). This request is generally known as location-dependent queries, and such services can be called *Location Dependent Information Services*. The proposed indexing scheme for answering traditional queries is modified to serve location-dependent queries in mobile broadcast environment and its performance is analysed.

1.6 Thesis Organization

The thesis is organized into 6 Chapters. The inter-relationships between the chapters are depicted in Figure 1.3. Chapter 2 describes a taxonomy for queries and its optimisation and processing of data management in mobile environment. Existing works on mobile data management are discussed in this chapter. This chapter also highlights the query types to be dealt with in this thesis. Moreover, this chapter outlines the achievements of the conventional methods in broadcast data management schemes, and more importantly highlight the problems which remain to be explored.

The main body of this thesis, which addresses performance amelioration issues summarised in Chapter 2, is divided into two parts: (i) *optimising query access time*, and (ii) *optimising tuning time and power consumption*. The solution to this first part

is discussed in Chapter 3 and the second part is presented in Chapter 4 and a hybrid model is provided in Chapter 5.

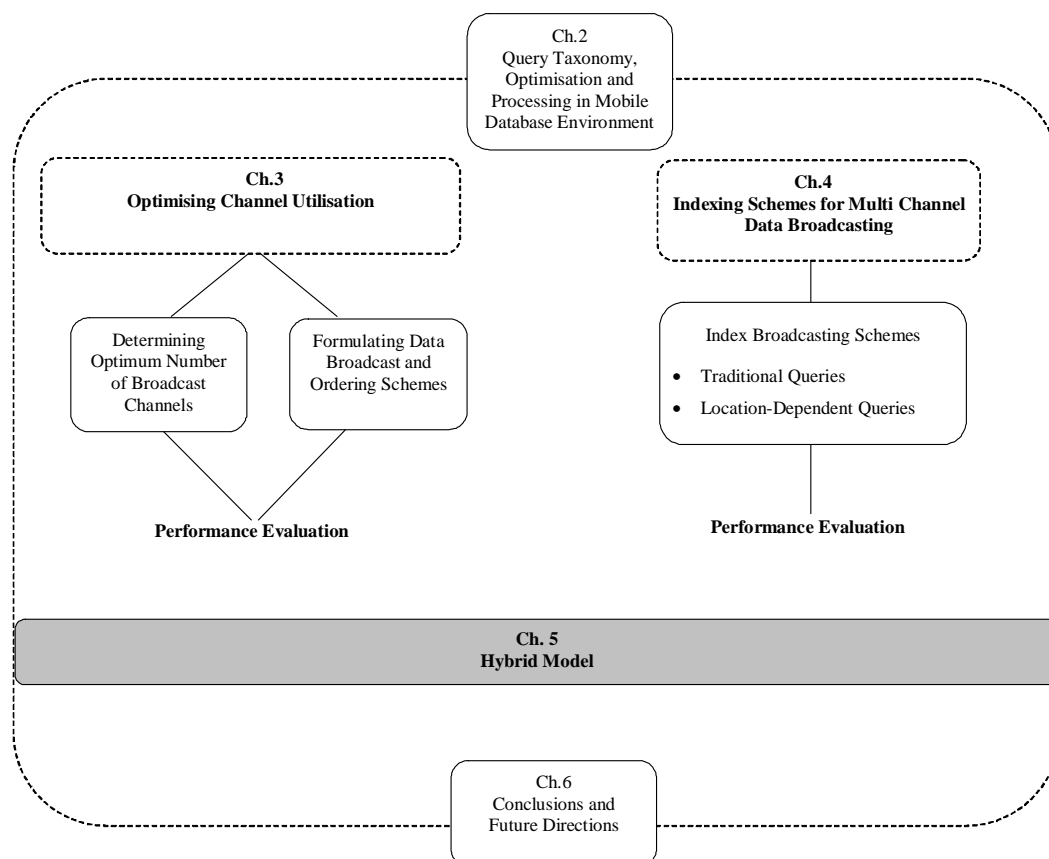


Figure 1.3. Thesis structure

Chapter 3 comprises of two parts: (i) Determining the optimum number of broadcast channels in which analytical models for calculating access time of information services over broadcast channel are provided. Since clients can normally perform a traditional client-server application by sending the request to the server over on-demand channel, the analytical models of this scheme are also presented. It performs a comparative analysis between the two models for minimising queries access time. The query access time over on-demand channel is studied to determine the optimum number of database items to be broadcast in a channel; and (ii) formulating broadcast ordering and scheduling algorithms. These algorithms are designed for a multi-

broadcast channel environment. Two algorithms are developed. One algorithm is concerned with broadcast program *with replication* and the other *without replication*. A broadcast program with replication corresponds to the case where the data items appear with different frequencies. On the other hand, a broadcast program without replication is when all data items are broadcast with equal frequencies or uniform frequencies. A broadcast program without replication is also called a flat broadcast program.

Chapter 4 studies various broadcast indexing schemes including the Global index scheme for multiple broadcast channels environment. The structure and mechanism of each scheme is presented and described thoroughly. The Indexing scheme provides accurate information for a client to tune in at the appropriate time for the required data item (Lee, Leong and Si, 2002). Furthermore, the Global indexing scheme is extended and modified to accommodate location-dependent queries in mobile broadcast environment. A complex index structure and query processing for location-dependent queries is also outlined in this chapter.

Chapter 5 presents a combined model of the previous two chapters forming a single broadcast system. This model is applied to a prototype system that has been built in a real wireless environment. The prototype system is used to represent an experimental test bed for the overall proposed data broadcast management schemes and to measure and study the effectiveness and efficiency of the schemes.

Throughout this thesis, a validation of the quantitative analysis, the simulation model and the prototype system are provided to analyse the performance of the proposed

data broadcast management schemes. A quantitative analysis is used to describe the behaviour of each proposed model by means of cost equations, and to perform quantitative analysis between different models. Comparative and sensitivity analyses are performed by simulation and the results are presented. The experimental performance evaluation is carried out by implementing the proposed models on a real set of wireless environment, whereas the simulation performance evaluation is implemented in a simulation program in which the values of several systems parameters are varied for sensitivity analysis. Varying the systems parameters in the real wireless environment is more difficult, due to the characteristics of the system structure. However, a further experimental model is valuable in demonstrating the reliability of the simulation model.

Chapter 6 gives a summary of the results achieved and an insight into future work.

Chapter 2

Query Taxonomy, Optimization and Processing in the Mobile Database Environment

2.1 Introduction

The development of wireless technology has led to *mobile computing*, a new era in data communication and processing (Barbara, 1999; Myers and Beigl, 2003; Imielinski and Viswanathan, 1994; Imielinski and Badrinath, 1994). With this technology, people can now access information any time and anywhere using a portable size wireless computer powered by battery (e.g. Palmtops and PDAs). These portable computers communicate with a central stationary server via a wireless channel. Mobile computing provides *database applications* with useful aspects of wireless technology, and a subset of mobile computing that focuses on query to a central database server is referred as mobile databases (Barbara, 1999; Malladi and Davis, 2002).

This chapter presents *query taxonomy, optimization and processing in the mobile database environment*. There are two major parts of review in this chapter and they are classified into: (i) query taxonomy and (ii) query optimization and processing in the mobile database environment.

The inherent characteristics of the mobile environment, as indicated earlier in Chapter 1, have a direct impact on data management in the mobile database environment and present a number of research challenges. Some queries in mobile databases have unique characteristics. These characteristics mainly arise from the aspect of mobility, including the location of the mobile clients and their movements. These queries are then location dependent, where the location of a mobile client becomes a parameter of the query. This issue expands considerably the complexity of query processing requirements.

The subsequent sections of this chapter are organized as follows (see Figure 2.1). A preliminary section which provides a general overview of wireless networks architecture will be presented in Section 2.2. Section 2.3 provides a framework of queries involved in mobile databases. Query optimization and processing schemes in mobile data management environment are then presented and classified in Section 2.4. These schemes are derived from existing works in the field. Subsequently, Section 2.5 of this chapter outlines the achievements of the conventional methods in broadcast data management schemes, and more importantly highlight the problems which remain outstanding. By the end of the chapter in Section 2.6, it will highlight the query types to be dealt with within this thesis.

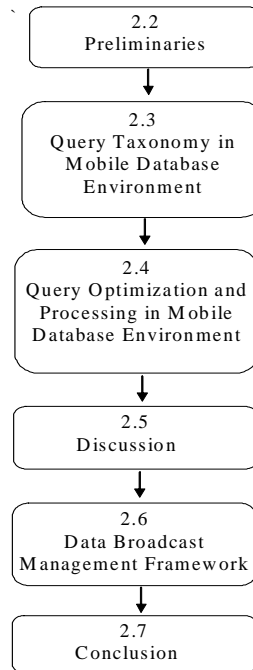


Figure 2.1. Outline of this Chapter

2.2 Preliminaries

Wireless architecture is fundamentally different from the wired environment. Consequently, the type of queries, query processing mechanisms as well as communication technology, also differ accordingly. Wireless networking infrastructure provides ubiquitous wireless communication coverage. This coverage will assist mobile users to have access to network resources via a different type of communication media and independent from the location of the user or the information being accessed.

In general, each mobile user communicates with a Mobile Base Station (MBS) in order to carry out any activities such as transaction and information retrieval. MBS has a wireless interface to establish communication with mobile clients and it serves a

large number of mobile users in a specific region called *cell*. In a mobile computing environment, each MBS is connected to a fixed network as illustrated in Figure 2.2.

Mobile clients can move between cells while being active and the intercell movement is known as a handoff process (Imielinski and Badrinath, 1994; Trivedi, Dharmaraja and Ma, 2002). Each client in a cell can connect to the fixed network via wireless radio, wireless Local Area Network (LAN), wireless cellular, or satellite. Each of the wireless networks provides a different bandwidth capacity. However, this wireless bandwidth is too small compared with the fixed network such as ATM (Asynchronous Transfer Mode) that can provide speed of up to 155Mbps (Elmasri and Navathe, 2003).

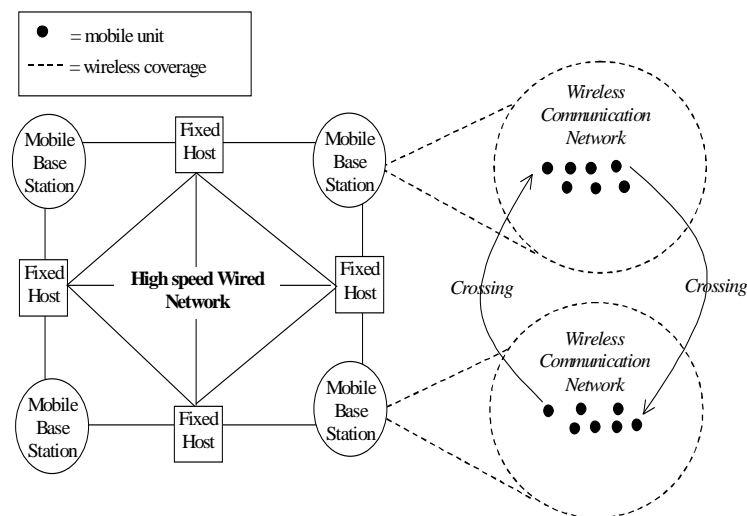


Figure 2.2. Mobile computing environment

2.2.1 Wireless Communication Networks

In a mobile environment, wireless communication networks are required to connect to the stationary host and establish communication with the central database server.

A variety of wireless communication networks is described in detail as follows:

- **Cellular Network**

Cellular network evolves from generation to generation. The first generation of cellular network (1G) is based on analog technology. The voice is transmitted using Frequency Modulation (FM). This technology includes Advance Mobile Phone System (AMPS), Total Access Communication System (TACS), and Nordic Mobile Telephone (NMT). It typically provides a considerably low data rate, which is from 1.2 to 9.6 Kbps (Pitoura, and Samaras, 1998).

The second generation (2G) includes Time Division Multiple Access (TDMA), and Code Division Multiple Access (CDMA), Global System for Mobile Communications (GSM), and Personal Digital Cellular (PDC). The transmission rate ranges from 9-14 Kbps (Pitoura, and Samaras, 1998).

The second and a half generation (2.5 G) enhances the second generation to provide a better transition to the third generation (3G). Some of these generations are High Speed Circuit-Switched Data (HSCSD), General Packet Radio Services (GPRS), and Enhanced Data Rates for Global Evolution (EDGE). The transmission rates of these three technologies are up to 38.4 Kbps, 144Kbps, and 384Kbps, consecutively (Sharma, 2001).

The third technology (3G) provides transmission rates from 144 Kbps to 2000 Kbps. Examples of this generation are the Universal Mobile Telecommunication System (UMTS), the Code Division Multiple Access 2000, and the NTT DoCoMo of Japan (Sharma, 2001).

The fourth generation (4G) has not yet been released, but promises to provide transmission ranging from 10Mbps to 150Mbps (Sharma, 2001).

- **Radio Network**

Radio Network can be classified into two categories namely, Public Packet Data Network, and Private Packet Data Network. This type of network only provides data transmission and utilizes base stations, network control, centres, and switches infrastructure for data transmission (Zaslavsky, Tari, 1998). Examples of Public Packet Data Network include Ericsson's Enhanced Digital Access Communication Systems (EDACS), Motorola's private DataTac, and ASTROs (Sharma, 2001). Some data providers who use this technology include Research in Motion (RIM), AT&T Wireless, Verizon, Palm.net, and OmniSKy (Sharma, 2001). The average data rate for this network is between 4800 bps and 19.2Kbps (Zaslavsky, Tari, 1998).

The Private Packet Data Network is a private organisation or company that established the network for its own use. One example of a company that established a private network is Federal Express (Sharma, 2001).

- **Wireless LAN Network**

Currently, of all the networks, Wireless LAN (WLANs) network provides the highest speed in communication. However, WLANs is not designed to support true mobility but is more concerned with providing a wireless interface or bridge to the wired networks (Pitoura, and Samaras, 1998). The institute of Electrical and Electronics Engineers (IEEE) 802.11 standard for wireless LANs focuses on medium-range but high data rate applications (Sharma, 2001). IEEE802.11 is currently the dominant

approach used in providing an Ethernet-like wireless networking environment (Vaughan-Nichols, 2002). There are at least three IEEE802.11 versions of technology namely IEEE802.11b, IEEE802.11g, and IEEE802.11a.

IEEE 802.11b is currently the most popular version. It uses the unlicensed 2.4 GHz frequency band. The actual data rate is from 2-4 Mbps in an indoor environment (Vaughan-Nichols, 2002) and it is capable of up to three channels of simultaneous data transfers. IEEE 802.11g can operate on two options of frequencies. One is in 2.4 GHz, which is compatible with IEEE 802.11b. The other is in 5GHz frequency that uses orthogonal frequency division multiplexing (OFDM), which offers up to 54 Mbps. IEEE 802.11a simply utilizes the 5Ghz frequency range, which also provides data rate up to 54 Mbps. However, the actual data rate in typical office situation is approximately 20 Mbps (Vaughan-Nichols, 2002).

- **Satellite**

Satellite technology normally provides long latency, wide-range, and high cost. However, it has a limited quality voice or data (Zaslavsky, Tari, 1998). The long latency is due to the long distance that incorporates propagation delay for data transmission. Sharma (2001) mentions three categories of satellite, in particular: Geostationary Earth Orbit (GEO) that rotates approximately 22.3 K miles above the earth with four of its satellites covering Earth including Low Earth Orbit (LEO) that rotates about 180-1000 miles above Earth and has 50 members surrounding Earth, and Medium Earth Orbit (MEO), which rotates ranging from 6.25k to 10 K miles with 12 members around Earth.

2.2.2 Location Positioning Systems

Wireless networks, as indicated earlier, enable mobile clients to establish a wireless connectivity with the central server which is located in the fixed network. Subsequently, clients are able to conduct on-line activities such as transaction and querying information. Some queries in the mobile environment are location-dependent, which means that the query results are obtained based on the user's current location. As mobile users frequently move from one place to another, most of the time the location information needs to be determined by the network rather than being explicitly stated and attached in the queries. This case is generally known as location-dependent information services (LDIS).

In location-dependent information services, the position of a mobile user can be determined by using one of the following systems: satellite-based system, network-based system, local-based system, and hybrid system. These four systems have different features and characteristics, which include accuracy and reliability factors.

- **Satellite-based system**

The most well-known and commercially used satellite positioning system nowadays is the global positioning system (GPS). The GPS provides location identifier in a form of a coordinate tuple (ie. longitude and latitude). Satellite technology normally provides long latency, wide-range, and high cost. Satellite technologies have been widely used to locate the position of moving objects. However, a satellite signal does not work well in the indoor situation. Thus, people find an alternative solution for indoor location tracking. Satellite technology is also utilised for Internet, cable or

telephone infrastructures especially in the rural areas where the installation of such equipment is cost prohibitive or difficult to reach. This type of location positioning system is also called geometric model (Lee, et al, 2002).

- **Network-based system**

A network-based system is a positioning system that utilizes a common telecommunication network such as a cellular network. This system requires the cell id of the object in order to locate its position. The accuracy of this system is between 50 metres to a few kilometers. To improve accuracy, the signal positioning calculation should include a greater number of base stations. The calculation can be done within the network or within the device itself. This type of operation is known as the symbolic location model (Lee, et al, 2002).

- **Local-based system**

This system relates to a short distance signal transmission, which is designed for use in indoor locations in particular. Some of these projects include Xerox ParcTab (Want et al, 1996) and the Cyberguide project (Abowd et al, 1997) that built a tracking system using infrared technology. Other alternative technology employed a radio frequency transmitter (RF) like the one developed at the Hewlett-Packard Laboratory by AT&T (Asthana et al, 1994) and the Pinger near-field tagging system (Hull, Neaves and Bedford-Roberts, 1997). A Wireless Local Area Network (WLAN) technology is included in this system. This type of system is also classified as a symbolic location model (Lee et al, 2002).

- **Hybrid system**

A hybrid system integrating both satellite and network-based systems has been made possible with the innovation of Assisted-GPS (Tsalgatidou, 2003). With this system, the GPS receiver is provided with additional data that is sent through a network. The additional data will enable the GPS receiver to improve processing speed and accuracy in determining location. It can also be used for indoor use with a restricted capability. The hybrid system has been utilized in the KDDI au network in Japan.

2.3 Query Taxonomy in Mobile Database Environment

In this section, different types of queries in mobile databases are classified. Some queries exist only in a wireless environment, while the other can be a common type of query in traditional databases. Figure 2.3 depicts the structure of this section. As can be seen from the Figure, queries in a mobile environment are classified into two classes: one is context awareness queries and the other is traditional queries in Database Management Systems (DBMS). Context awareness queries are categorized into three types, namely; Location-dependent queries, Context-dependent queries, and Hybrid queries.

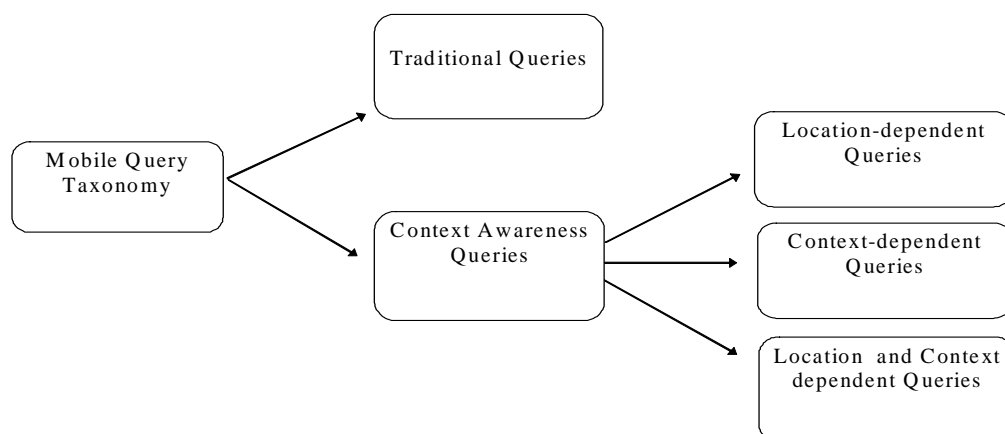


Figure 2.3. Query taxonomy in a mobile database environment

Traditional queries are common queries in traditional Database Management Systems (DBMS). These queries are typical queries that people are dealing with on a day-to-day basis in a wired network environment system. This type of query explicitly mentions the required information in the query statement, thus, the query result is based only on the actual query itself.

Examples: (i) A university student wants to retrieve his/her academic record or personal details. (ii) A traveling sales person inquires about product availability, price, etc. (iii) Travelers require information about share prices, weather information, airline schedules, and soon.

Other queries in a mobile environment involve some context awareness information. The word 'context' implies a variety of aspects. Schilit et al (1994) define the word context according to three categories namely, computing context, user context, and physical context. Computing context relates to computing resources such as network connectivity, bandwidth, printer, and workstations. User context is associated with user's needs, preferences, roles, profile, and alike. Physical context involves environment issues which include lighting, noise, traffic, temperature, and humidity. Chen and Kotz (2000) add another category of context called 'time context', which refers to time of day, week, month, year, etc. Ebling, Hunt and Lei (2001) define context as either an aspect of the physical world or conditions and activities in the virtual world. It is further described that context information can either be transient when the context associates with the environment at a point of time, or persistent when the context involves a history of transient context. In other words, it can always be assumed that context relates to who, when, where, and what.

Context awareness queries create a new class of applications in mobile computing. With context awareness, a mobile device is expected to adapt constantly to a wide range of dynamically and continually changing situations. It is important for the device to be able to aware of the situation, environment, and tasks that the mobile client is currently performing, as well as those that will be performed in the near future. The utilization of context information in an application minimizes the amount of user involvement in a service by providing related information.

Knowledge of the context of the query enables the device to pre-fetch all data that is highly related and likely to be queried in the near future. With this query, the client initiates only a single request and all related data will be retrieved implicitly. This technique prevents a client from making multiple requests that result in energy inefficiency. As an example, let us consider how to find information about restaurants in a particular area. The query will result in retrieving information about restaurants based on the user's preferences for Italian, Chinese or fast food, for instance, as well as pre-fetching maps, and traffic and weather conditions, which may be queried next.

Most applications have been focusing on location awareness rather than context awareness as a whole. Thus, mobile queries can be classified into location-dependent queries, context-dependent queries, and a combination thereof.

2.3.1 Location-dependent Queries

Location-dependent query is a class of queries that are initiated by mobile clients. In this type of query, the location of the mobile client is a parameter of the query. The

value of the location parameter is provided explicitly by the client or implicitly using a global positioning system (GPS). An example of this query is a traveler's information system that provides information to motorists about hotels, restaurant, bars and the like, depending on their current location.

Processing of this query must be based on knowledge of the user's location. For queries that require services that are ahead of the current route location, direction of motion and speed must of necessity be taken into account. Recent technology provides a new feature in automobiles, which offers navigational aids as a built-in feature. In general, each location update generates two direct costs, particularly transmission cost, which refers to the cost to inform the server of the new location, and server processing cost, which corresponds to the cost of updating the system containing the location of the mobile unit. The location parameter can be in any vehicles, such as taxis, trucks, and helicopters.

Location-dependent queries can be classified into two categories. The first category is based on user and object types, and the second is based on query types. User and object types represent the state (i.e. still or moving) of the mobile user when issuing the query and the searched object. The query types category relates to the state of the queries whether continuous or non-continuous queries. A continuous query is very much different from conventional or non-continuous queries, which are based on an instant of the database at some moment in time. With this query, clients need to send a query only once and notification of the updated information will be sent automatically as clients move to different regions. These two categories are described as follows.

a. User and Object Types

This section covers the issues and challenges in location-dependent query based on user and object types. There are three types of location-dependent query in this category: (i) moving user seeking static object/s; (ii) moving user seeking moving object/s; and (iii) static user seeking moving object/s.

- Moving User seeking Static Object/s

This type of query originates from a moving user, and the searched object is static. Examples of this query are:

- Tourists request information about nearby tourist attractions, bars, hotels, or shopping centre while traveling.
- A mobile user to obtain traffic conditions while driving home from his/her office.

This query may involve certain constraints such as a tourist wishing to know the location of any bars within a two-mile range, or hotels that cost less than \$50 per night. The common operators for constrained location-dependent queries can be found in (Seydim et al, 2001).

The conventional query processing schemes in this category can be explained as follows. The location information is found with the help of a positioning system (i.e. Satellite-based system). The location details are attached to the query whenever the client invokes a query. This information may include one or more of the following: current position, current velocity and current time. This information, together with the query, is sent to the server. As such, the location data is static.

Client mobility is the most essential property of a wireless environment. However, as a mobile client is frequently and constantly moving from one place to another, the query result may not be valid by the time the result returns to the user device. There are two situations when query result becomes invalid. First is when the query result is received by a mobile client after moving out from the area but the area is still within the same cell. This situation is illustrated in Figure 2.4. The second scenario is similar to the first one except that the mobile client is not only moving out of the area, but also out of the cell within which the query was initiated. Figure 2.5 depicts the multi-cell movement.

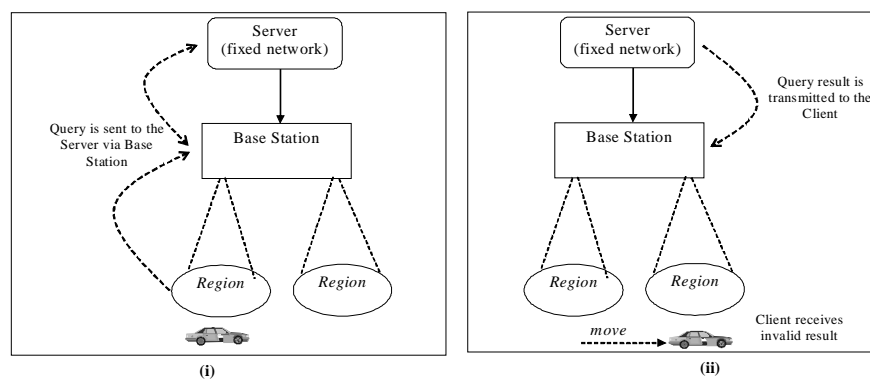


Figure 2.4. Single-cell movement (moving user seeking static object/s)

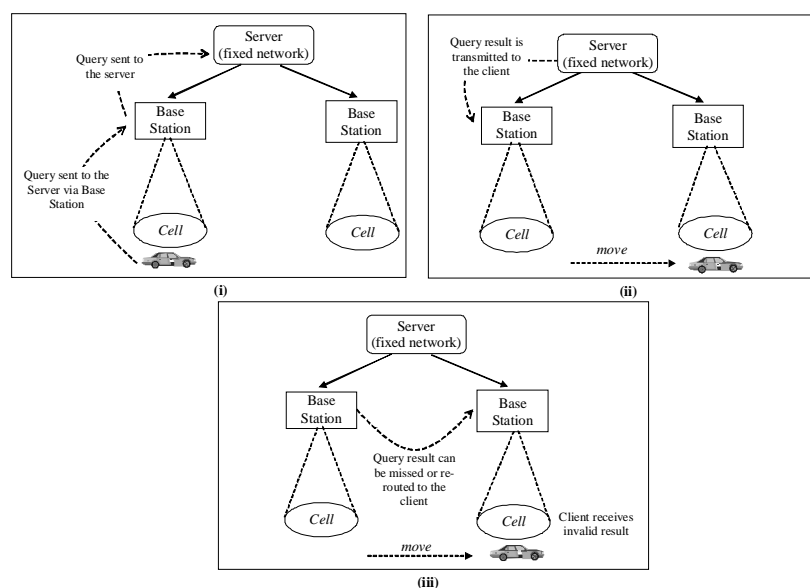


Figure 2.5. Multi-cell movement (moving user seeking static object/s)

- Static User seeking Moving Object/s

This query originates from a stationary user, and the searched object is moving.

Examples of this query are:

- Taxi dispatcher to find free taxis in certain region.
- Police officer to retrieve those patrolling cars closest to car number 'xxx'.
- A mobile user to display when the next incoming free taxis will arrive at the location where he/she is standing at.
- An army to retrieve friendly helicopters within 10 miles (battle field context).

Likewise, this type of query may include certain constraints such as:

- Police officer to prevent motorists entering within 2 miles of a dangerous location.

It can be a future query, which includes spatial and temporal elements, like:

- Traffic controller to retrieve cars that will reach a particular intersection within the next 5 minutes. These cars will be detoured.
- A soldier to find friendly tanks that will enter a given region within the next 10 minutes (battle field context).

Common query processing schemes in this scenario can be described as follows.

A satellite-based positioning system (i.e. GPS) is attached to the moving object.

The object constantly communicates with the server, and updates its location in the central database system. This case is similar to the first category except that here the searched object is constantly moving. Therefore, by the time the query result is received by the user, the result has to be valid in order to correspond to the current position of the moving object. Figure 2.6 shows this scenario when

the object is moving to a different region or area in a cell. Multiple-cell movement as illustrated in Figure 2.7 depicts the situation where the searched object is no longer in the initial cell.

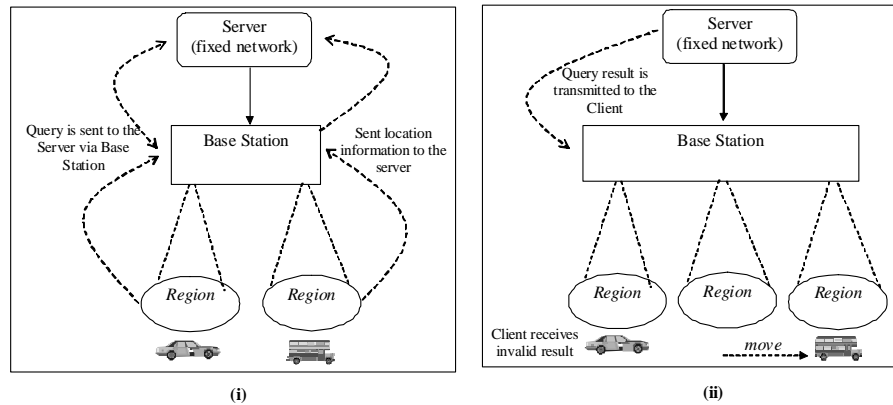


Figure 2.6. Single-cell movement (static user seeking moving object/s)

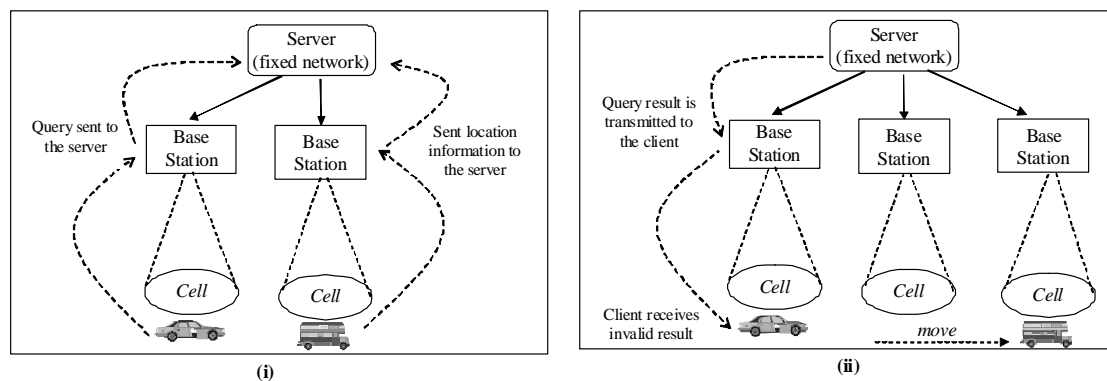


Figure 2.7. Multi-cell movement (static user seeking moving object/s)

- Moving User seeking Moving Object/s (Moving USMO)

This type of query originates from moving user, and the searched object is also moving. Examples of this query are:

- A patrolling policeperson to find out how many other patrolling officers are in the same region.
- A flying helicopter requests the time when it will intercept a friendly tank (battle field context).

Query processing schemes in this case are similar to the second category. However, this time both the user and the searched object are constantly moving. Consequently, it is more difficult to determine a correct query result since either the user or the object can easily invalidate the result. Figure 2.8 illustrates the situation where both mobile user and the searched object are moving out of their initial region or area but still within a single cell. The multiple-cell movement is depicted in Figure 2.9.

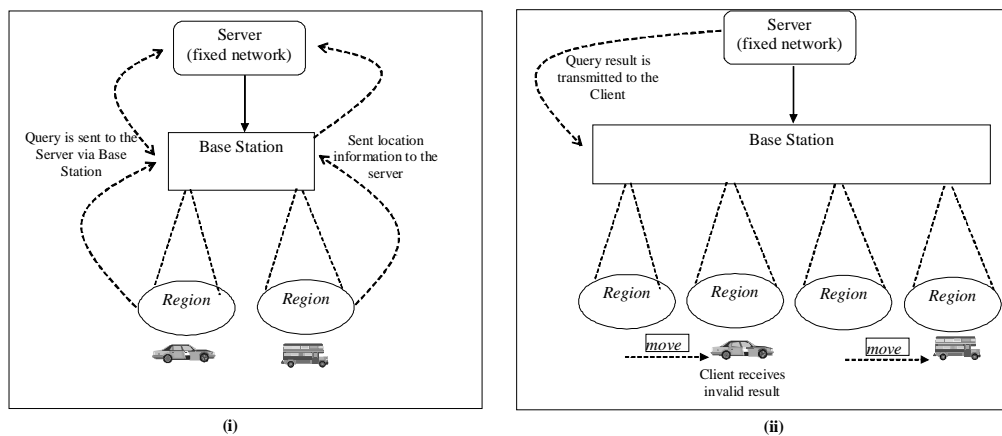


Figure 2.8. Single-cell movement (moving user seeking moving object/s)

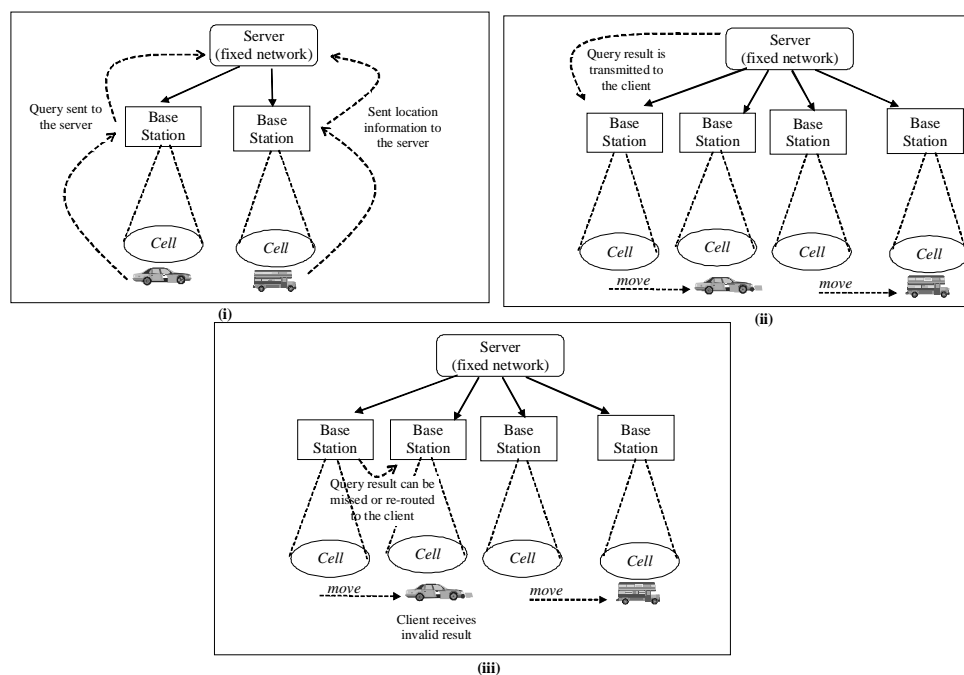


Figure 2.9. Multi-cell movement (moving user seeking moving object/s)

Location-dependent queries in a mobile environment are very much centered around the above three types of queries. The related issues will become more complex when they involve the client's disconnection. Frequent disconnection is one of the characteristics of a wireless environment; therefore, it is common for a mobile client to become disconnected from the network after sending queries due to signal distortion, the handoff process or limited battery power. When such a situation occurs, the server has to be made aware of, and determine, the most appropriate action to take in order to process the query efficiently and accurately without asking the mobile user to re-send the same query.

Location-dependent query requires location-dependent data. Location-dependent data can be categorized into *static location-dependent data* and *dynamic location-dependent data*. Static location-dependent data relates to the stationary object, when the object location is not continuously updated in the database (e.g. restaurant, hotels). Generally, the “*moving user seeking static object/s*” query type falls within this category. On the other hand, dynamic location-dependent data correspond to a moving object that requires constant update of the object's location in the database (e.g. taxis, ambulance, or even the mobile users themselves). The “*moving and static user seeking moving object/s*” query type normally corresponds to dynamic location-dependent data.

b. Query Types

Location-dependent queries can also be classified based on the query type namely; continuous query, and non-continuous query.

- Continuous query

The continuous query includes real-time monitoring of mobile objects. Real-time monitoring queries are continuous for monitoring purposes. In a continuous query setting, a query is not once-off, meaning that even after the initial query is answered, the query is still kept by the server. Hence, in the case where the user moves into a different location or when new information becomes available, new information will then need to be dispatched to the user. Example: (i) To request information about nearby tourist attractions, hotels, or shopping centre while travelling. With this class of query, clients need to send a query only once and notification of the updated information about nearby tourist attractions, hotels, or shopping centre will be sent automatically as clients move to different regions. (ii) To notify mobile clients whenever they are close to a certain situation such as dangerous zone or traffic jam by providing some form of alerts to them. Figure 2.10 shows an illustration of real-time query monitoring.

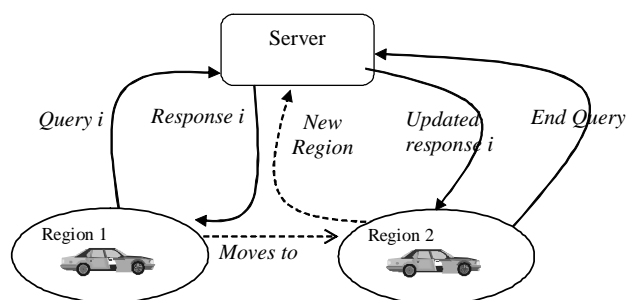


Figure 2.10. Continuous query

In this application, the system must be able to provide the accurate query results and update them in real time whenever mobile clients enter or exit the region defined by the query. This class of query can be referred as *range-monitoring*

queries (Cai and Hua, 2002). The range-monitoring queries are removed from the system only when the user explicitly ends the query.

- Non-continuous query

The non-continuous location-dependent query is different from the continuous query as the system does not manage the query, but only the location of each mobile unit in specified regions. This class of query does not have to specify when to end the query since the server does not keep the query, but location only. Figure 2.11 illustrates the non-continuous query type. Examples: retrieve nearby hospital, police station or petrol station in the area.

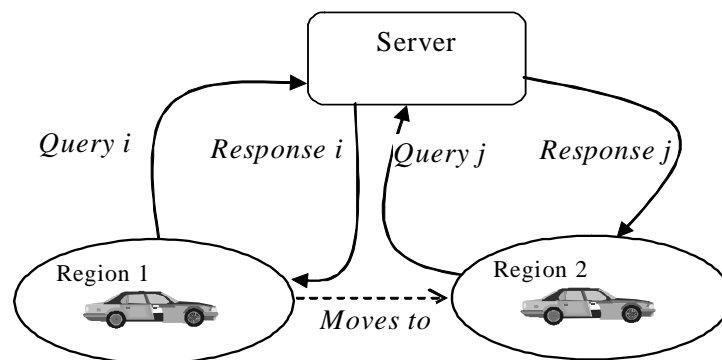


Figure 2.11. Non-continuous query

2.3.2 Context-dependent Queries

This type of query requires maintenance of an internal representation of user's needs, preferences, roles, profile, etc. With these parameters, application will be able to decide the situation and task that a user is currently performing, and adapt to changes in user needs and roles. This enables a mobile device to not only retrieve the required information, but also some other information that is highly relevant (Ebling, et al 2001; Franklin, 2001). Examples: (i) Tourist wants to see a list of restaurants in a

town. Rather than retrieving the entire list of restaurants, the query retrieves the restaurants based on the user's preferences and needs such as cuisine preference, price, occasion, etc. (ii) A business woman wants to check the closest meeting schedule. The query retrieves the time as well as the previous history of the meeting, people involved, client accounts, meal preferences, required data, etc.

2.3.3 Location- and Context-dependent Queries

Location- and context-dependent queries require the system to maintain all parameters of context-awareness queries including location parameter. Examples: (i) A traveler wants to find restaurants in the region that suits his/her taste that are within 10 minutes of current travel distance; (ii) a teenager wants to check movies in the nearby cinemas that s/he will like, and whether or not s/he will be in time for the show, given travel time and distance.

These queries are very useful for nomadic users since it not only considers the users' preferences and needs, but also the location of the user. A more sophisticated application utilizes a variety of sensors to monitor the environment as well as user's actions in order to provide assistance with the tasks being performed by the user. This application requires the ability to process data streams in real-time, and to analyze and interpret it precisely. The main issue in context-awareness query is to accurately analyze the environment and the user's intention, regardless of the source of the context information. It is a difficult challenge since there is the possibility of conflicting data, and the need to have efficient processing to provide a useful application to the user.

2.4 Query Optimization and Processing in Mobile Database Environment

In mobile databases, queries can be processed mainly using two different mechanisms: (i) the on-demand mechanism (or also called pull operation), and (ii) the data broadcasting mechanism (or also called push operation), which is illustrated in Figure 2.12.

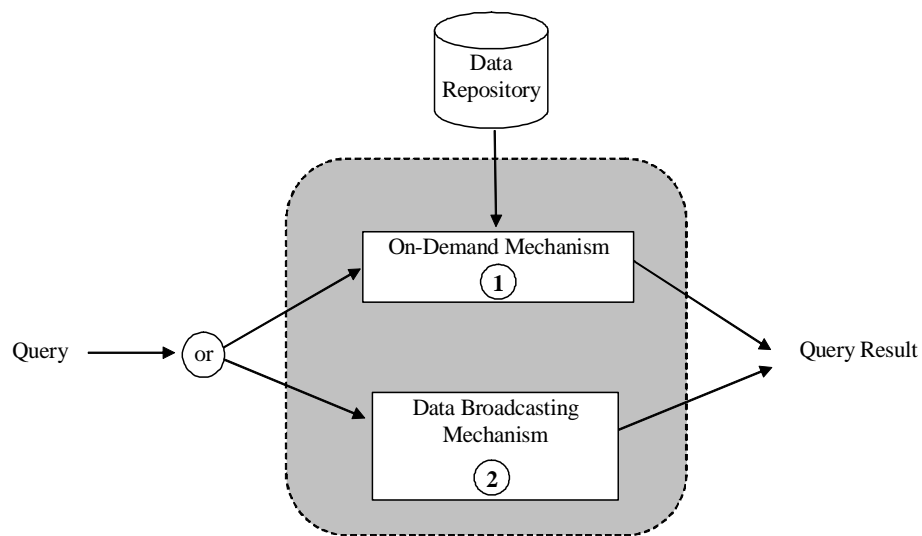


Figure 2.12. Query processing in mobile databases

The data items are obtained from the data repository at the remote central stationary server. Data management strategies in the on-demand mechanism refer to the optimization method used at the server side to serve an on-demand request or a request that is sent to the server for processing. Data broadcasting strategy relates to determining a method to disseminate the database item to mobile client so that the *response time*, *tuning time* and *power utilization* of retrieving database items are minimized.

Query optimization and processing strategies in mobile databases are very much centered around these two most important mechanisms. In this thesis, all query optimization and processing mechanisms are classified based on the type of queries involved, and they are: (i) traditional queries and (ii) location-dependent queries.

2.4.1 Traditional Queries

Traditional database queries in a wireless environment are concerned with the same types of queries as those that are generally applied in a wired network environment system. In traditional queries, the query result is not dependent on the location of mobile users and so the data involved is not location-dependent data. This query processing is classified into two categories: (i) on-demand mechanism and (ii) data broadcasting mechanism. The data broadcasting mechanism is further classified into on-demand broadcast and periodic broadcast. The structure of the query processing mechanism for traditional queries in a mobile database environment is shown in Figure 2.13.

The fundamental concept of the data broadcasting mechanism has been introduced for some time and applied in different projects, such as TeleText (Ammar and Wong, 1985; Wong, 1988), and DataCycle (Bowen, et al 1992). These projects are concerned with a fast/fixed network rather than a wireless network. Data broadcasting offers a number of advantages that can also be realised in a mobile environment (Imielinski, Viswanathan and Badrinath, 1994). As such, several new issues need to be addressed that require entirely different approaches as compared with wired network environment.

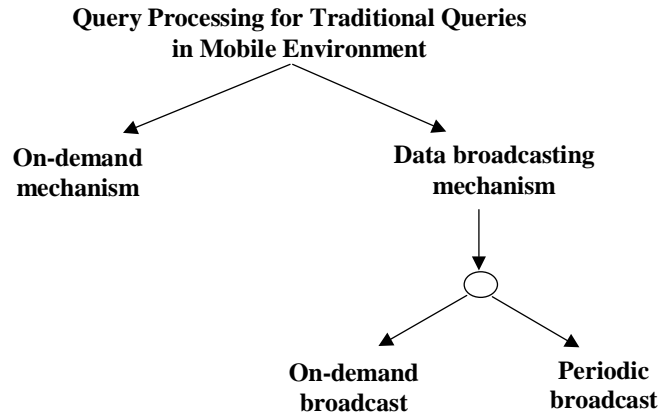


Figure 2.13. Query processing for traditional queries

2.4.1.1 On-demand Mechanism

The on-demand mechanism relates to a traditional point-to-point scheduling scenario, which is similar to client-server interaction. As illustrated in Figure 2.14, mobile clients establish a connection with the server via a dedicated link or channel. A request is sent to the server; when necessary, the request can be placed in a queue. The server picks the request, processes it, and sends the result back to the client via another dedicated channel.

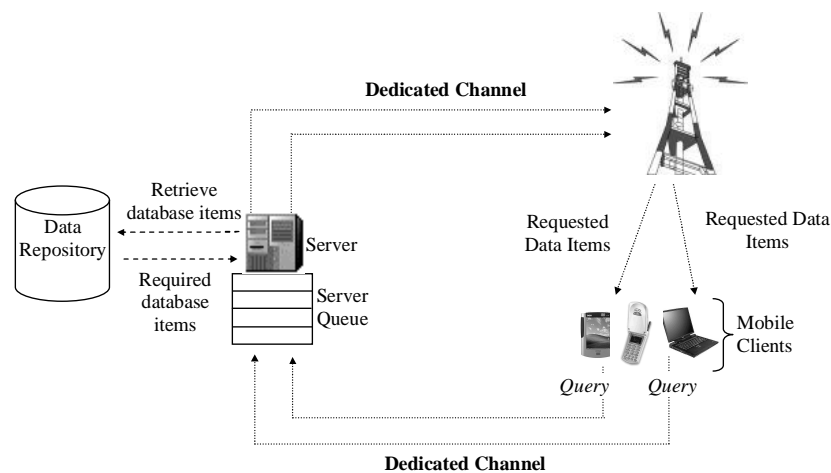


Figure 2.14. The architecture of on-demand mechanism

This scenario is not desirable in a mobile environment where resource preservation is of the utmost importance, as it satisfies only an individual client. This scenario will

show its limitation when it comes to scaling the number of users or queries. Therefore, a more scalable and robust paradigm is of much interest. Push operation, which is described in the next section, is designed to overcome this issue. Push operation is known as a scalable paradigm, where many clients can be satisfied simultaneously so that resource utilization is maximized (Yajima et al, 2001; Malladi and Davis, 2002 ; Aksoy et al, 1999). This pull operation is more appropriate when privacy is of concern since the data is not directly disseminated but requested through a dialogue (two-way communication) with the database server via a point-to-point or dedicated channel.

2.4.1.2 Data Broadcasting Mechanism (On-demand Broadcast)

Push-based (On-Demand Broadcast) is where mobile clients send queries to the server through a dedicated or point-to-point channel, and the client monitors to the broadcast channel to retrieve the query result. In this mechanism, the downlink channel (or broadcast channel) used to send the query result from the server is shared by all clients, which is the opposite of the pull operation where the channel is reserved for an individual client. This operation is shown in Figure 2.15.

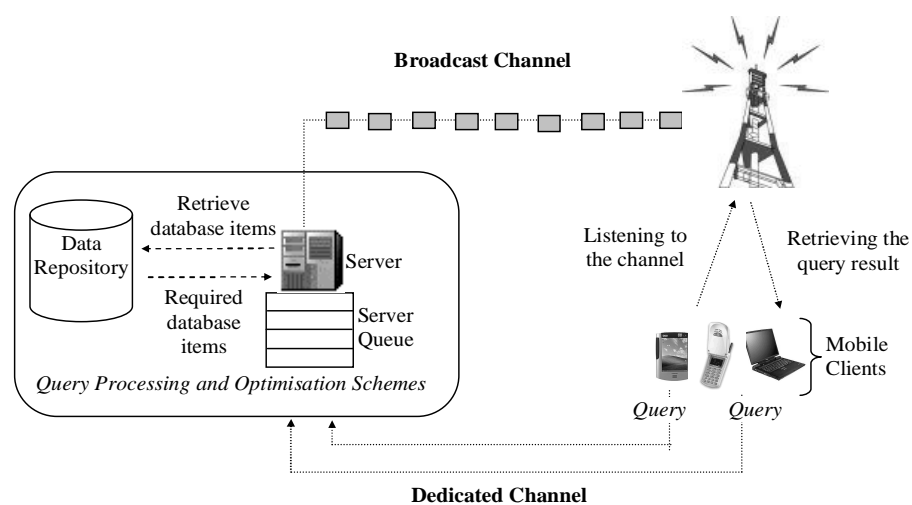


Figure 2.15. The architecture of on-demand broadcast mechanism

In a wireless environment, the number of requests initiated by clients in a cell can be very large. Thus, it is necessary for the server to apply a certain technique in order to optimize the processing of the request in the server. This section studies different strategies that enable the server to optimize the processing of queries.

Two types of algorithms are used in a wireless environment for this purpose, namely, the push-based and the disk scheduling algorithms. The push-based scheduling algorithm is utilized to determine how queries are to be served efficiently in the server, considering a number of factors such as the length of the query, the wait time and the popularity of the items. Disk scheduling is related to how data is placed on the disk so that it improves the query response time.

Push-based scheduling algorithms include $R \times W$ algorithm, *MAX*, and exhaustive $R \times W$ algorithm (Aksoy and Franklin, 1998; Acharya and Muthukrishnan, 1998; Triantafillou, Harpantidou and Paterakis, 2001). The $R \times W$ scheduling algorithm is based on the product of the number of outstanding Requests (R) and the wait time (W) of the oldest outstanding request for all data items in the broadcast server queue (Aksoy and Franklin, 1998). The data item with the highest product value is chosen for broadcast. This mechanism provides a balance treatment between hot and cold items since the highest product value is derived from either the popularity of the data item, which creates a high R -value, or the longest outstanding request (W). *MAX* is a scheduling algorithm for data requests of different sizes (Acharya and Muthukrishnan, 1998). It introduces an alternative to the response time of a request metric or stretch of a request, which is considered as a good alternative for varying data size. Acharya and Muthukrishnan, (1998) also presented the criteria for

optimizing the worst-case stretch of each request. The exhaustive $R \times W$ algorithm involves a clustering strategy to group a set of requests that require the same set of data items and then $R \times W$ algorithm is then applied (Triantafillou, Harpantidou and Paterakis, 2001). However, the algorithm assumes a fixed length for all data items, and mobile clients need to continuously listen to the channel after the request has been sent to the server in order to eliminate the chance of transmission error. A disk scheduling algorithm, such as the C -LOOK algorithm, processes each request in the cylinder position on the disk, which has been sorted previously in an ascending order (Worthington, Ganger and Patt, 1994).

The possibility of a joint algorithm comprising of the exhaustive $R \times W$ algorithm and the C -LOOK algorithm has also been investigated and analyzed in Triantafillou, Harpantidou and Paterakis (2001). This joint algorithm is classified into two schemes: one combines separate push based and disk scheduling algorithms; the other amalgamates push-based and disk scheduling algorithms into a single operation. The details of the joint algorithm are as follows:

a. Combining separate push based and disk scheduling algorithms

Two algorithms under this category are: (a) The AdoRe algorithm, and (b) The Flush algorithm. AdoRe algorithm incorporates the $R \times W$ scheduling algorithm and the disk scheduling algorithm with the designed mechanism, while the Flush algorithm employs a first come, first served approach followed by the C -LOOK algorithm.

- The AdoRe algorithm: Active Disk on Requests

This algorithm sends a number of requests (K) from the broadcast server queue to the disk scheduler queue for processing. This is done when the disk becomes

idle. In a situation where the number of requests in the broadcast server is greater than K , the $R \times W$ algorithm is applied and a group of K requests with the highest $R \times W$ value is transmitted to the disk queue. The advantage of this algorithm is the ability to keep the disk system highly utilized while improving the efficiency. The efficiency is achieved by reducing the average disk service time based on forwarding a batch of requests to be processed. For instance, a *C-LOOK* algorithm will take less time to serve a batch of requests like $K = 20$ than will the running of the $R \times W$ algorithm 20 times to pick up one request at a time and forward it to the disk. In the case of the number of requests being less than K , the *AdoRe* algorithm picks one request after another to the disk scheduler queue.

- The FLUSH algorithm

This algorithm is not concerned with the number of requests in the broadcast server queue. All requests in the broadcast server are flushed to the disk server straight after the disk completes the processing of a single request. When there are more than one request, they are queued in the *C-LOOK* wait list. Subsequently, the *C-LOOK* algorithm is applied to process each request in the disk queue.

b. Amalgamating push based and disk scheduling algorithms

Two algorithms fall into this category namely, the Oweist algorithm, and the $R \times W/S$ algorithm.

- The Oweist algorithm: Optimal Weighted Service Time

As in the *AdoRe* algorithm, a parameter K is used for the formation of groups of requests. Whenever the disk becomes idle, K or fewer requests are picked out from the broadcast queue and transmitted to the disk queue. However, this algorithm does not consider the use of the $R \times W$ algorithm, but rather, it incorporates a new mechanism to form a group of K requests or less. The algorithm maintains a graph of K requests. The edge connecting two requests r_i and r_j is associated with $R_j \times S_j$. R_j corresponding to the number of requests in the broadcast queue for item j , and S_j denotes the disk access cost to access item j . In this case, it is assumed that item i is the previous item retrieved from the disk queue. A number of factors are taken into account when calculating the disk access cost. These factors include the search time from the cylinder of item i to the cylinder of item j , the access time to find the desired item j once the disk head is on j 's cylinder, and the time to retrieve item j from the disk. This algorithm calculates all possible permutations for the K requests and picks the optimal permutation that provides the smallest total weighted cost. It is noted that when K equals 1, *Oweist* is identical to *AdoRe* with the same K .

- The $R \times W/S$ algorithm

This algorithm combines the push based scheduling algorithm ($R \times W$) and disk service time (S). It is not concerned with grouping requests because requests from the broadcast server are sent to the disk server one at a time. When the

disk is idle, the $R \times W/S$ algorithm is executed to find the request that obtains a highest value of the $\frac{R \times W}{S}$ to be transmitted to the disk server for processing.

A multiple query optimization scheme for on-demand broadcast has been investigated by Malladi (2002). In this scheme, the queries are processed in a group and common expressions among queries are analyzed. Subsequently, it allows the result to be broadcast only once. The mechanism of this scheme is as follows: mobile clients send a query to the server; the server places the query from the clients in the queue; the requests are collected in a group based on a given query time window; the server processes and evaluates each query according to common expressions; the server retrieves the data from the data repository; the mobile client is sent information about when and for how long to listen to the channel and obtain the result; the server broadcasts the result via the broadcast channel; and the mobile client tunes into the channel at a predetermined time. It was claimed that this scheme would improve the query performance by reducing waiting time as well as bandwidth utilization.

The drawback of on-demand broadcast is the increase of bandwidth when the client sends the request to the server. The query performance is overwhelmed by multiple client requests. The congested channel and server queue may severely affect the query performance and power consumption of mobile clients. The periodic broadcast mechanism in the next category prevents the possibility of congested channel bandwidth and server queue, which is caused by an increase in the number of clients, and request arrival rate.

2.4.1.3 Data Broadcasting Mechanism (Periodic Broadcast)

As illustrated in Figure 2.16 *Push operation (periodic broadcast)* refers to periodically broadcasting database items to an unbounded number of clients in the cell through one or more broadcast channels and allows a mobile client to capture and select data items for the data in which it is interested.

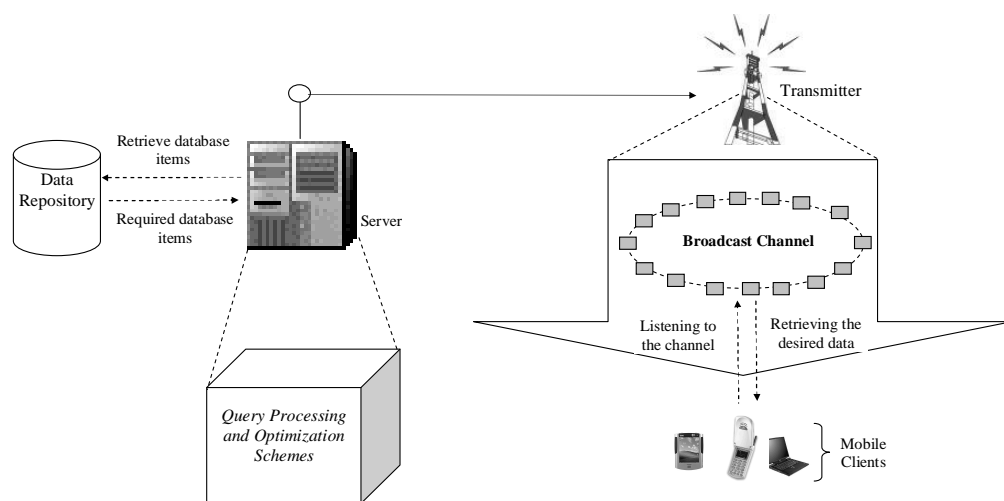


Figure 2.16. The architecture of the periodic broadcast mechanism

Access to data is sequential as clients need to wait for the desired data to arrive on the broadcast channel (Imiliensky, Viswanathan and Badrinath, 1997). The behavior of the broadcast channel is unidirectional which means the server disseminates a set of data periodically to a multiple number of users. With this mechanism, the request is not known a priori, which is different from on-demand broadcast. Furthermore, this mechanism can be utilized to overcome the resource limitations in mobile databases. In this way, a mobile client is able to retrieve information without wasting power to transmit a request to the server. Other characteristics include scalability as it supports a large number of queries; query performance is not affected by the number

of users in a cell as well as the request rate; and, it is effective even when there is a high degree of overlap in the user's request.

The main challenge in periodic broadcast is to minimize query response time and the tuning time of retrieving database items. In some cases, the response time is equal to the tuning time. As the data size increases over time, the required number of data items to be broadcast also increases accordingly. This situation may cause some mobile clients to wait for a substantial amount of time before receiving a desired data item. Consequently, the advantages of the periodic broadcast strategy will be diminished. Alternatively, they can send a request via point-to-point or dedicated channel to the server (pull operation). In this case, the server processes the query and sends the result back to the client. This situation severely affects the query response time and power consumption of mobile clients, as it involves queuing and cannot scale over the capacity of the server or network. It should be noted that the term data item corresponds to database record or tuples, and data segment contains a set of data items. A complete broadcast file is referred to as a broadcast cycle.

Data dissemination schemes can be classified into two categories: one is *minimizing query response time*, and the other is *minimizing tuning time*. These schemes are designed while considering the inherent limitations in mobile computing environment.

a. Minimizing Query Response Time

There are several periodic broadcasting schemes, which include: (i) Selection of Data Items to be Broadcast, (ii) Non-Uniform Frequency Distribution of Broadcast Data Items, (iii) Distribution of Data Items over Multiple Channels, and (iv) Organization

of Data items. These schemes aim to minimize the query response time by reducing either the waiting time for the arrival of the desired data, or both waiting and download time in certain cases.

- Selection of Data Items to be Broadcast

A selection mechanism needs to be designed to reduce the broadcast cycle length, which eventually reduces the query response time. During each broadcast cycle, additional items might be qualified as hot and some previously hot items might cool down, and therefore need to be replaced depending on the size of the cycle. A replacement algorithm is required to replace the cold data items with the new hot items. The server determines a set of appropriate database items to be broadcast, using information from queries received. Since hot items are those accessed by most clients, the access pattern of each client on the database items has to be derived from the clients back to the server. Finally, statistics will be compiled and the desired data items can be broadcast appropriately.

There are three replacement algorithms, namely *Mean Algorithm*, *Window Algorithm*, and *Exponentially Weighted Moving Average (EWMA) Algorithm* (Leong and Si, 1997). These algorithms maintain a score for each database item to estimate the access probability of the next broadcast cycle. The scores are defined by measuring the cumulative access frequencies of each database item over the length of the observation period.

Alternatively, a *log-time* algorithm that is designed based on the *packet fair queuing* can be used to determine a set of appropriate database items as well as

the time to broadcast these items (Hameed and Vaidya, 1997; Hameed and Vaidya, 1999). The log-time algorithm is also called $O(\log M)$ algorithm. The M in $O(\log M)$ denotes the number of database items in the server and so $(\log M)$ is used to find the average time required per iteration of the scheduling algorithm. This log-time algorithm can also be utilized to incorporate both single and multiple channels (Hameed and Vaidya, 1997; Hameed and Vaidya, 1999). This is also useful for handling a certain factor like transmission errors on scheduling.

However, as the size of the database increases, the number of broadcast items may also increase accordingly. This situation will lead to an increase in response time. The next scheme can be used to improve the response time of a majority of requests by manipulating the frequency of hot items to be broadcast.

- Non-Uniform Frequency Distribution of Broadcast Data Items

The different bandwidth capacity between the downstream communication and upstream communication has created a new environment called *Asymmetric Communication Environment*. In fact, there are two situations that can lead to communication asymmetry (Acharya, et al, 1995). One is due to the capability of physical devices. For example, servers have powerful broadcast transmitters, while mobile clients have little transmission capability. The other is due to the patterns of information flow in the application. For instance, in the situation where the number of servers is far fewer than the number of clients, it is

asymmetric because there is not enough capacity to handle simultaneous requests from multiple clients.

Broadcast Disk is an information system architecture, which utilizes multiple disks of different sizes and speeds on the broadcast medium. This architecture is used to address the above situations (Acharya, et al, 1995 ; Acharya, et al, 1996). The broadcast consists of chunks of data from different disks on the same broadcast channel. The chunks of each disk are evenly scattered. However, the chunks of the fast disks are broadcast more frequently than the chunks of the slow disks. This is the opposite of a *flat broadcast* where the expected delay before obtaining an item of interest is the same for all broadcasted items. With differing broadcast frequencies for different items, hot items can be broadcast more often than others. The server is assumed to have the indication of the clients' access patterns so that it can determine a broadcast strategy that will give priority to the hot items. It is designed so that the speeds of the disk can be adjusted to the configuration of the broadcast.

- Distribution of Data Items over Multiple Channels

An alternative strategy for improving query response time is to distribute the broadcast data over more than one broadcast channel. In general, data items are broadcast over a single channel with the underlying rationale that it provides the same capacity (bit rate/bandwidth) as multiple channels, and it is used to avoid problems such data broadcast organization and allocation while having more than one channel (Imielinski, Viswanathan and Badrinath, 1997). Nevertheless, the use of single channel will show its limitation when there is a very large

number of data items to be broadcast. Figure 2.17 illustrates an example when a broadcast cycle is split into two channels. Data items are broadcast repeatedly over a single channel, but none is broadcast over more than one channel.

The allocation of broadcast cycle into multiple numbers of broadcast channels will reduce the chance of long delay before obtaining the desired data items. As a simple example, assume the size of the each data items is 50 bytes. Consider a mobile client who wants to retrieve data item #8. The following two cases analyze the access time of mobile client when the data is broadcast over a single channel as compared with two channels. It is assumed that the client probes into a first data item in a channel.

Case 1: Data items (#1-10) are broadcast in a single channel. In this case, all data items are available in a single channel. The client who wants to retrieve data item #8, has to wait until the desired items are arrived. The total access time required will be $(50 \times 8) = 400$ bytes.

Case 2: Data items (#1-10) are split into two broadcast channels: channel one and channel two with 6 and 4 data items respectively. Since data items are split into two broadcast channel, the access time will also be different. In this case, client can switch to another channel and wait for the data items of interest to arrive. The total access time for the client to retrieve data item #8 would be $(50 \times 2) = 100$ bytes. Thus, when compared with the first case, this time client can have a much more efficient data access.

In another case, multiple broadcast channels are necessary since multiple low bandwidth channels cannot simply be merged into a single high bandwidth channel (Prabhakara et al, 2000). In light of this issue, Prabhakara et al (2000) presented a broadcast ordering scheme, where the hot items or the most frequently accessed data items are broadcast more often than cold items. The hot items are allocated in the fastest bandwidth in decreasing order.

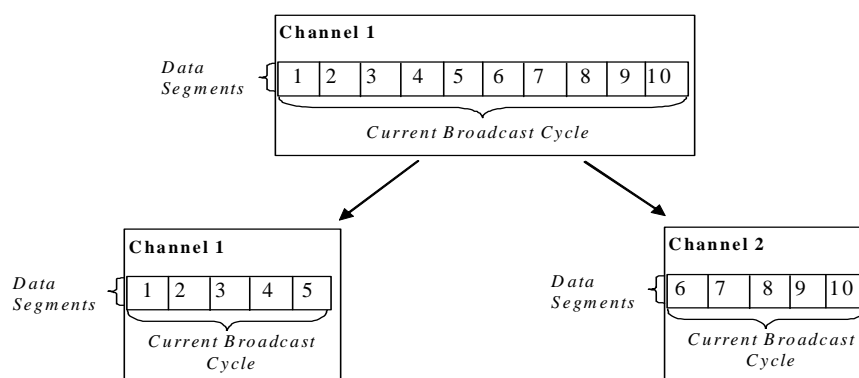


Figure 2.17. Multiple channels architecture

The number of items in the channel is specified by being given a temperature threshold value. Tran, et al (2001) enhanced the work in Prabhakara et al (2000) by considering different bandwidth posed by each channel and the size of each data items is varied. However, these approaches do not consider the relationship of one data item with the other. Therefore, these schemes are not effective for multiple data items retrieval. The next scheme describes the strategy to overcome this issue.

Another advantage of broadcasting the database items over multiple channels is to solve problems such as data distortion and data distribution (Leong and Si, 1995). The following four broadcasting mechanisms were proposed: Data

- Organization of Data Items

The previous schemes are concerned with the retrieval of a single data item. However, in most cases multiple data items may be involved. A number of applications in this category may include a situation where mobile client wants to obtain information about the prices of more than one stock item at the same time (for instance, to list the stock prices of car companies under General Motors Corp.), or when a client wants to retrieve the weather forecasts for various cities concurrently (i.e. to list the weather forecasts for all cities in a certain region). To accommodate this type of request, we need to consider the relationship between one data item and the others. Organizing data items over the broadcast channel can be applied to reduce waiting time as well as download time. The basic idea of this scheme is to allocate related data items in such a way that the client does not have to wait for a substantial amount of time for the relevant data item. The organization of broadcast data is designed to match the query access pattern from mobile clients as closely as possible. As the query initiated by each client varies, this problem is sometimes considered an NP-hard problem (Chung and Kim, 2001; Liberatore, 2002).

The difficulty is to decide the broadcast order in advance, even without much knowledge of any future query. In general, an access graph is needed to identify the optimal organization of database items over a channel. The access graph is used to represent the dependency of data items. Once the access graph is built, a certain algorithm is utilized to determine the best broadcast program. A cost model called the Semantic Ordering Model (SOM) can be used to determine the

most efficient access graph (Si and Leong, 1999). SOM is defined into two models: namely, Simple and Extended SOM. Simple SOM considers only the relationship among entity types while Extended SOM incorporates historical query access patterns. Heuristics algorithm (Hurson, Chehadeh and Hannan, 2000), randomized algorithm (Bar-Noy, Naor and Schieber, 2000), and Genetic algorithm (Huang and Chen, 2002; Huang and Chen, 2003) are some algorithms that can be used to identify the most effective organization of broadcast data items. The final broadcast program can be distributed over both single or multiple channels.

b. Minimizing Tuning Time

A broadcast indexing scheme is needed to reduce the tuning time by providing accurate information for a client to tune in at the appropriate time for the required data (Lee, Leong and Si, 2002). In this scheme, some form of directory is broadcast along with the data, the clients obtaining the index directory from the broadcast and using it in subsequent reads. The information generally also contains the exact time of the data to be broadcast. As a result, mobile clients are able to conserve energy by switching to “doze” mode and back to “active mode” when the data is about to be broadcast.

In general, the broadcast indexing technique causes a trade-off between optimizing the client tuning time and the query response time. The consequence of minimizing one of them is the increase on the other. For instance, to minimize the response time is to reduce the length of broadcast cycles. In this case, the index can be broadcast once in each cycle; however, it will be at the cost of the tuning time since the client

will have to wait for the index to arrive which happens only once in each broadcast cycle. On the other hand, when the index directory is frequently broadcast in each broadcast cycle to reduce the tuning time, the response time will be greatly affected due to the occupancy of the index in the cycle. Thus, it is necessary to find the optimal balance between these two factors.

Broadcast indexing can be *tree-indexing* based, *signature* based, or *hybrid indexing* that is the combination thereof. Similar to a traditional disk-based environment, the index tree technique can be applied to data broadcasts on wireless channels. Instead of storing the locations of disk records, the tree-index serves as a pointer to guide the client to the relevant index and each index contains the exact time of the data to be broadcast (Imielinski, Viswanathan and Badrinath, 1994). Figure 2.19 illustrates an example of an index tree. Tree-indexing based on *B+*-tree structure has been presented in (Leong and Si., 1995). This technique incorporates a separate index channel to locate the data segment in a multi-data channel. However, this technique is mainly concerned with problems such as data distortion and data distribution.

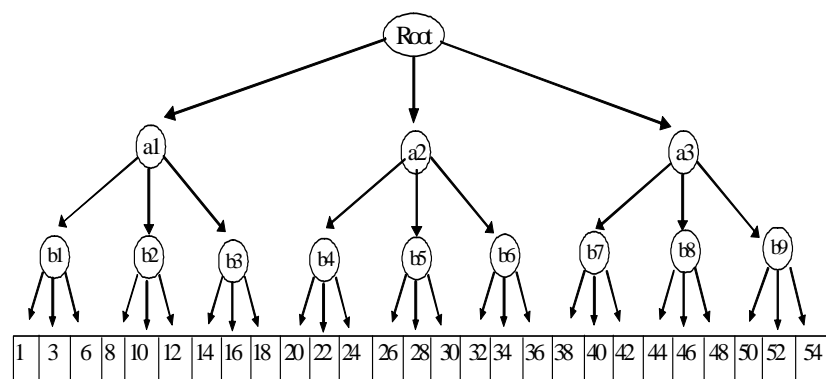


Figure 2.19. A Full Index Tree

Clustering index, non-clustering index, and multiple index methods are used to determine the index organization over a broadcast channel (Imielinski, Viswanathan

and Badrinath, 1997). Clustering index refers to clustering a record's attribute whenever attribute that belong to the records have the same value consecutively. The non-clustering index defines a method for indexing the non-clustering attributes by partitioning each non-clustered attribute in the broadcast cycle into a number of segments called *meta segments*. In multiple indexes, a second attribute is chosen to cluster the data items within the first clustered attribute.

A signature-based index algorithm is derived by hashing the attribute values into bit strings followed by combining them together to form a bit vector or signature (Lee and Lee, 1996). The signature is broadcast together with the data on every broadcast cycle. The mechanism is also applied to the query initiated by the client. To process the query, mobile clients need to tune into the broadcast channel and verify the query signature with the data signature by performing a certain mathematical operation like 'AND' operation. If the signature is not matched, the client can tune to the "doze" mode while waiting for the next signature to arrive. The main issue with this method is to determine the size of the signature as well as the number of levels of the signature. There are three different signature-based index algorithms, namely *simple signature*, *integrated signature* and *multilevel signature* (Lee and Lee, 1996).

In simple signature, the signature frame is broadcast ahead of each data frame. This makes the total number of signature frames the same as data frames. An integrated signature is applied to a group of data frames and the signature is calculated accordingly. Figure 2.20 shows a combination of these two algorithms, forming a multilevel signature. This technique is designed to interleave with data items in a single channel.

A hybrid indexing technique made up of Index tree and Signature is expected to outperform the single indexing techniques by integrating the advantages of the two techniques into a single operation (Lee, Hu and Lee, 1998; Hu, Lee and Lee, 1999). This technique is shown in Figure 2.21.

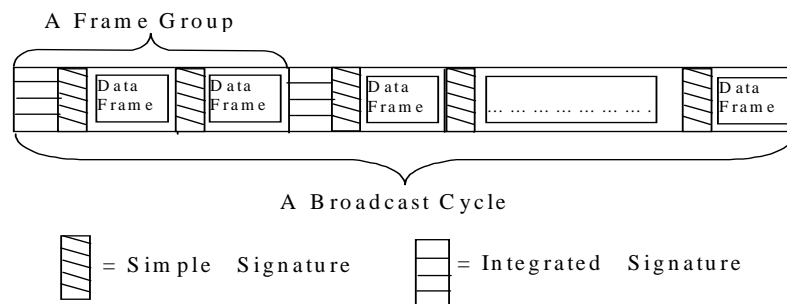


Figure 2.20. Multi-level Signature

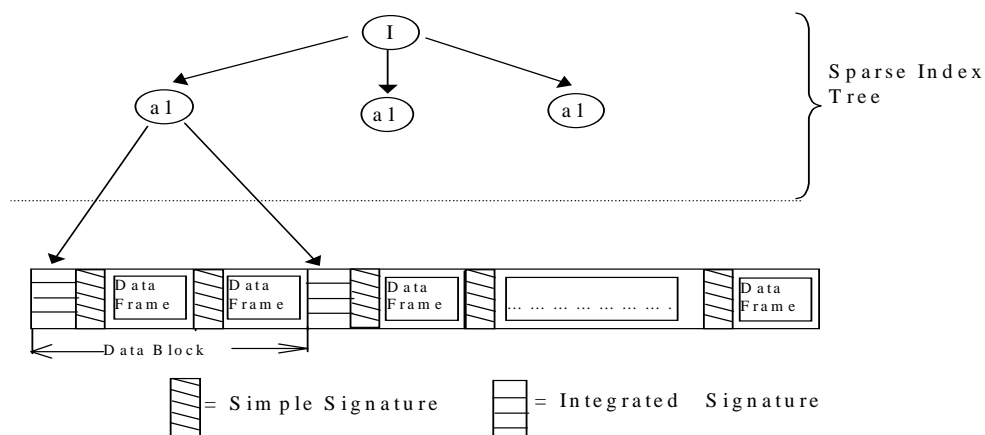


Figure 2.21. Hybrid Indexing

2.4.1.4 Hybrid On-demand and Data broadcasting Mechanism

The hybrid on-demand and periodic data broadcasting methods try to find the best channel allocation mode or the combination thereof to serve information requests. There are four possibilities of channel allocation methods namely *exclusive on-demand*, *exclusive broadcast*, *static hybrid*, and *dynamic hybrid* (Acharya et al, 1997; Lee, Hu and Lee, 1997; Hu, Lee and Lee, 1998; Hu, Lee and Lee, 1999). The exclusive on-demand method assigns all channels to on-demand or point-to-point

mode as opposed to the exclusive broadcast method which assigns all channels to broadcast mode. The static hybrid method allocates the channels into both on-demand mode and broadcast mode. Because the dynamic hybrid method is an evolution of the static hybrid, it means that the number of channels allocated for on-demand mode and broadcast mode is dynamically changed depending on the server workload and query access patterns.

The broadcast channel and the on-demand channel cause different access overheads. The access overheads required for the broadcast channel are determined by the amount of time a client has to wait for the desired data to arrive plus the time to filter the relevant data. As with the on-demand channel, the access overhead is derived from the amount of time a client needs to connect to the server. The time is linearly dependent upon the load of the server. According to the concept of the broadcast channel, the access time will not be affected either by the rate of requests or the number of users in a cell. This is opposed to the on-demand channel whose performance is fully dependent on these two factors. The hybrid method is considered a better approach overall. Exclusive on-demand channels do not perform well with the variation of the number of channels.

Although this scheme seems promising, it requires a vigorous cost calculation since it introduces a high degree of complexity into the algorithm before it can decide the best channel allocation. Furthermore, the utilization of an on-demand channel forces the client to consume a large amount of energy especially after considering the asymmetric communication cost and limited power resources as indicated earlier.

2.4.1.5 Other Strategies

Other strategies include the utilization of a certain technique to examine the most efficient plan to execute a query. Initially, the query is compiled into a sequence of candidate plans and the optimal one is taken from these candidate plans after considering the battery level of the client as well as the load of the server (Ganguly and Alonso, 1993).

Search algorithms for determining a set of candidate plans include the use of optimization techniques such as partial order dynamic programming (Ganguly, Hasan and Krishnamurthy, 1992; Ganguly, 1992), and the linear set algorithm and linear combinations algorithm (Ganguly and Alonso, 1993). Factors that are considered include the query cost model and optimization criterion to obtain the plans. The optimal plan corresponds to optimal energy utilization at a particular stage.

A similar strategy divides the query processing into three phases: translation, optimization and execution (Kottkamp and Zukunft, 1998). As shown in Figure 2.22, the optimization and execution phases are combined into a single phase to avoid excessive communication between server and client. These phases can be executed on different sites, either at the server site or the client site, depending on the resources situation. Three possible processing strategies can be applied when choosing the query execution site. The first alternative is to send the query to the server and the server processes the query all the way to the end. The second alternative is to do the translation phase at the client site to the point where the analysis indicates high energy consumption for the next phase due to its complexity. The query is subsequently sent to the server for processing until all phases are completed and the

data are retrieved. The last alternative is to do the query processing in the mobile unit from the start of processing to the end.

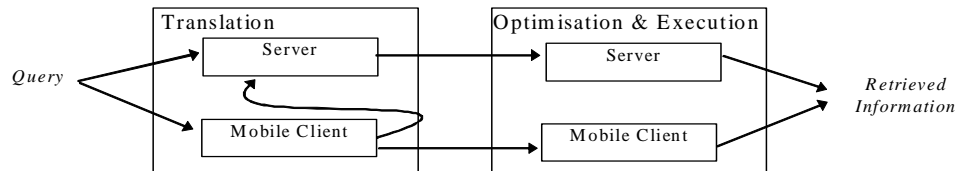


Figure 2.22. Query execution sites

2.4.2 Location-dependent Queries

Traditional database queries on a wireless environment are concerned only with conventional query processing with an absence of wires. Location-dependent queries, in contrast, depend heavily on the location criteria (of mobile clients, requested data, and broadcast data). The following discussion involves research issues and questions for each processing mechanism, namely (i) on-demand query processing, and (ii) the data broadcasting mechanism. Furthermore, each processing scheme is classified based on static and dynamic location-dependent data. Figure 2.23 depicts the query processing for location-dependent queries in mobile environment.

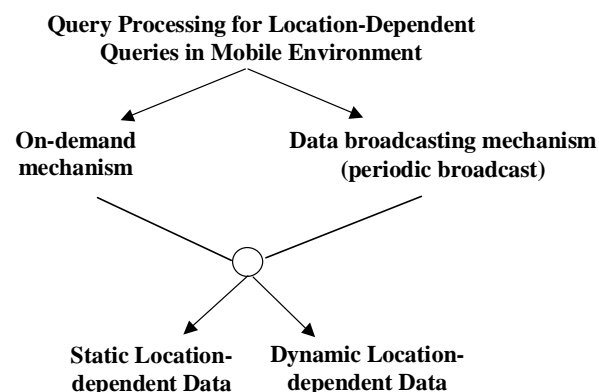


Figure 2.23. Query processing for location-dependent queries

2.4.2.1 On-demand mechanism

The on-demand mechanism relates to when a mobile user invokes a query, which is then sent for processing to the server and the results are relayed back to the mobile user. There is nothing special here until we understand the fact that the mobile user may have moved into a different location resulting in the invalidity of the query results produced by the server. The problem is promulgated by any disconnection that has occurred to the mobile client especially when the query results are already in the air, which may result in loss of information.

a. Static Location-dependent Data

Mobility is the most important aspect of a wireless environment. However, it is also necessary to ensure that mobile client obtains a valid query result while traveling or moving from one point to another. For instance, mobile client initiates a query “Find the closest bars” at current location (i.e. location A). In this case, if the query is not processed immediately, there is a chance that the mobile client is no longer in location A. Therefore, when the mobile client receives the result in location B, the results are no longer correct as they were associated with location A. There are at least five main issues in this category, which include: (i) query processing protocols, (ii) efficient query processing, (iii) query results paths, (iv) query scheduling, and (v) data placement whether it be centrally or de-centrally, replicated or non-replicated and its consistency management.

The first issue is to define the query processing protocols for on-demand static location-dependent data access. Zheng et al (2002) has introduced the concept of valid scopes. Valid scopes define the area or region within which the query result is

considered valid. Determining the valid scope concerns specifying the area/region in which the query result is considered valid with respect to the user's location (Zheng et al, 2003). The valid scopes identifier are specified based on the positioning system; for instance, valid scopes with a network-based system are identified in cellids, while satellite-based system in coordinate tuples. Prior to sending the query result to the mobile client, the result is first checked against its valid scope and the current location of mobile client to make sure that client is still within the valid scope of the answer. The valid scope information is attached to result. Thus, the mobile client is able to determine the validity of the result with respect to its position. An example of valid scope is given in Figure 2.24. As shown in the figure, there are four data items and four valid scopes: data item D1, D2, D3, and D4 is associated with valid scope VS1, VS2, VS3, and VS4, respectively.

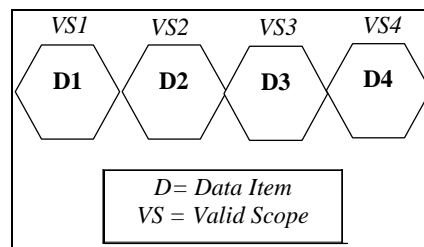


Figure 2.24. Valid scopes

An investigation of valid scope representation has been presented in (Zheng et al, 2002). Two initial schemes are considered namely the Polygonal Endpoints (PE) scheme and the Approximate Circle (AC) scheme. The PE scheme uses a polygon shape as a valid scope representation. It records all the endpoints of the shape. The drawback of this scheme is that when the polygon is large, the number of endpoints to be recorded will also be large. Consequently, it will occupy the limited space available in a client's local cache and the performance will be

degraded. The AC approach manages to overcome the drawback of PE by using circle representation to approximate the size of the polygon rather than recording all the endpoints.

The valid scope is calculated from the centre of the circle based on the radius value. Although AC scheme improves the PE performance, the valid scope may not be precise especially when the polygon's shape is stretched and narrow. This will cause the client to incorrectly identify valid data as invalid. Consequently, the cache hit ratio will decrease significantly. Caching-Efficiency-Based (CEB) scheme is designed to find the balance between AC and PE (Zheng et al, 2002). The CEB scheme analyzes the representation against the precision and the transmission cost. Generally, the more precise the valid scopes, the more bandwidth is required to send the information and vice versa.

A voronoi diagram-based indexing method is a scheme designed to efficiently locate the position of the user, process the query, and retrieve the correct result (Zheng and Lee, 2001). Based on the information that the user sent, together with the query such as current velocity, position, and time stamp to the server, the server will be able to determine the most accurate result. The query result will include a prediction such as how long the result will remain correct according the user's velocity as well as the radius within which the result will still be valid. Subsequently, the client can store the query result, the client's current location and the maximum validity radius to its cache for future use. To ensure the validity of the prediction, this method used the client's maximum speed and the shortest

distance. This is applied to cater for the possibility of the client changing the speed or direction.

The second issue relates to the efficiency of query processing. Madria et al (2000) proposed a location-based concept hierarchy to model location-dependent data. Concept hierarchy defines relationship that generalise lower data to high layer information. For instance, to generate domain knowledge of cities that belongs to a state. Domain experts may be consulted to ensure the validity and completeness of the hierarchy. Figure 2.25 depicts location-based concept hierarchy. The proposed concept hierarchy was designed as distributed directories to provide assistance to find values in the database according to the reference location of a query. A spatial index model like R^+ trees was utilised to index each location attribute in the relations. Basically, R^+ tree is an extension of B-tree for multi dimensional static objects (Beckmann et al, 1990). To process the query efficiently, data partitioning and hierarchical replication were applied. The concept hierarchies are stored at the database server. When there is more than one database server, each database server may store concept hierarchies and relevant indexes according to their location, or replicate them.

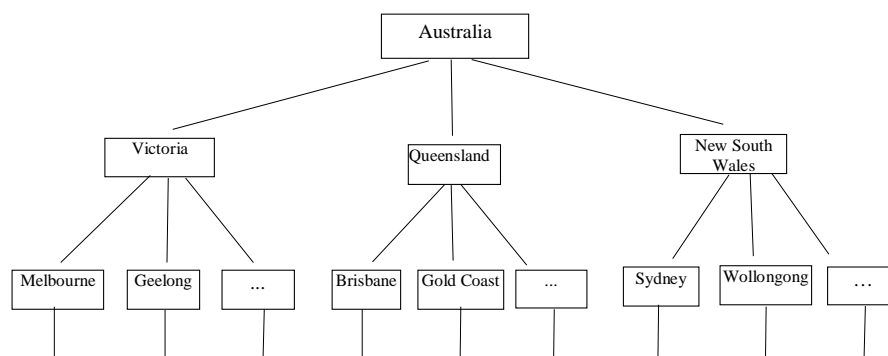


Figure 2.25. Location-based concept hierarchy

The next consideration is to determine the query results path. A query traverses from the mobile client through a wireless network to the base station, which might then traverse through several wired nodes and fixed network to the database server. The query results may travel using the same or different paths. Considering a few problems associated with the movement of mobile clients, it is then critical to investigate *path traversals using path index*. It is also important to investigate the effect of the proposed method on power utilization, particularly the impact of disconnection of mobile devices when the query results are transferred down from the server. The technique should be able to speed up the data retrieval as well as maintain energy efficiency since a number of checking may be needed before it can decide how the data obtained.

Xu et al (2002) presented D-tree spatial index structure. This index model was designed to enhance the efficiency in locating correct and valid data for location-dependent queries.

When the number of mobile clients is large, it is then necessary to determine location-dependent query scheduling. Query scheduling in mobile databases produces a certain degree of complexity not found in traditional query scheduling. In traditional query scheduling, the concern is how to minimize processing time and resources in answering all queries. In a mobile environment, it is essential to consider other aspects, such as movement frequency of mobile clients as they have to be prioritised; otherwise, when the query results are produced, they might already be out of date since the client has already moved to another region.

Zheng and Lee (2001) proposed location-dependent query scheduling for multi-cell movement. It assumed that the mobile client is able to detect the time when an initial cell is left and a new cell is entered. This handoff information is then passed on to the server. Four mechanisms are presented namely, Naïve method, Priority Method, Intelligent Method, and Hybrid Method.

The naïve method is a conventional method in which the client is required to re-submit the query to the server whenever s/he has not received the result after entering a new cell. The server will drop the query result if it finds out the client is no longer in the initial cell. A priority method is designed to provide a priority for clients, who have entered a new cell but have not received the query result or what is called handoff client. However, it will cause a delay in the processing of queries for other clients. A certain threshold can be applied to process the number of queries from the handoff client. The Intelligent method has the ability to speed up the query processing for clients who are about to leave the cell. In this method, clients send the information about the time when they expect to leave the current cell. This method is effective for clients who are just about to move out of the current cell. However, normal clients will still have the processing of their queries delayed. Hybrid method is the combination of the priority and intelligent methods. It was shown that the hybrid method provides a better performance than all the others. Figure 2.26 illustrates these schemes. The most recent approach that discusses query scheduling for multi-cell movement can be found in (Jayaputera and Taniar, 2005).

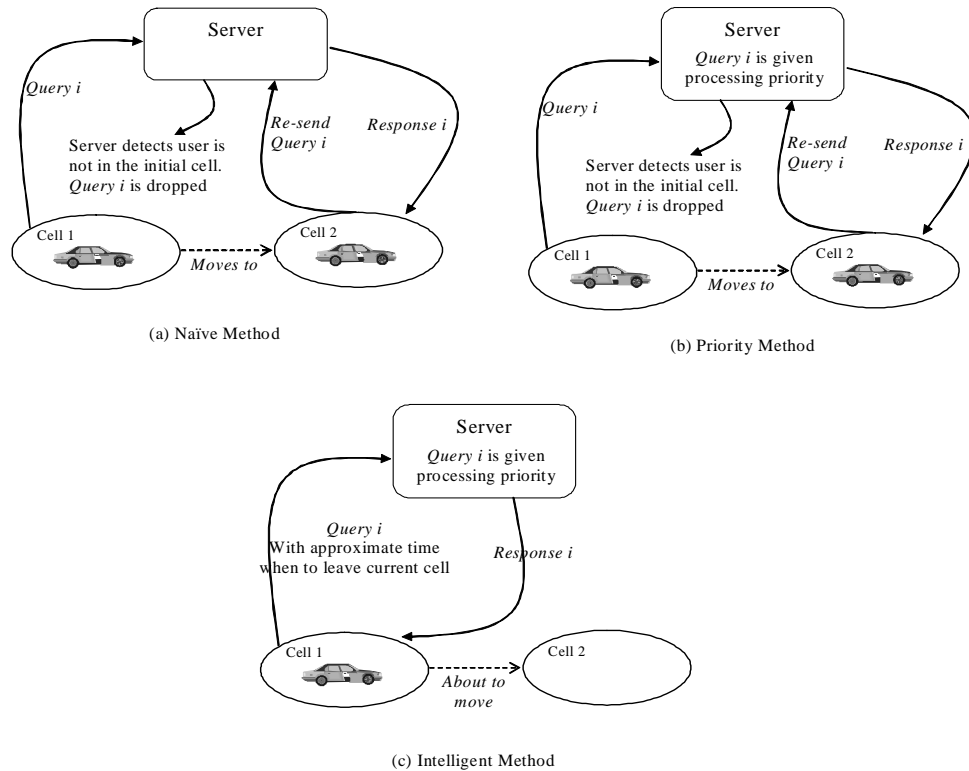


Figure 2.26. Query scheduling for multi-cell movement

Other issues, which are definitely necessary to pursue and investigate, include data placement: whether it is in a centralized repository or in a distributed environment. A centralised repository is not desirable especially when it involves a large amount of database information with an unlimited number of users accessing the data item. However, the centralised system is easier to maintain and more manageable. A distributed environment provides a more robust means of storing data items and valid scopes, as well as processing queries from different cells. It should be noted that a distributed environment typically requires a more complex mechanism and maintenance.

b. Dynamic Location-dependent Data

This type of query relates to location data that is dynamically changing. Existing literature focuses on the efficiency of the location tracking mechanism on the

server side. The main issue in this query is to determine how mobile users can be tracked efficiently and their location information stored in the database.

This case corresponds to the location tracking mechanism that is employed to monitor and update the location of mobile user. Since the number of users carrying mobile devices increases linearly with the service demand, the communication traffic for locating users also increases accordingly. Continuously updating the location of the moving object causes severe network performance and bandwidth overhead costs. Moreover, current satellite-based (i.e. GPS) technology is only able to provide or update the position of a moving object at a discrete interval such as every 5 seconds or so. This technology is not sophisticated enough to support location updates on a moving object, and continuous monitoring of the position of a moving object is critical. This situation requires an efficient strategy for location tracking and management. The fact that objects to be stored in the database are moving objects which occasionally or even frequently are out of connection, an investigation of the data structure to support this query type is equally important. Figure 2.27 depicts location tracking architecture.

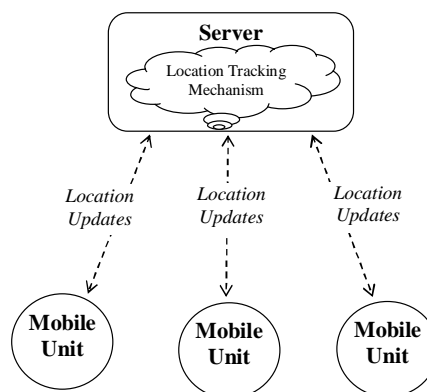


Figure 2.27. Location tracking architecture

Existing Database Management Systems (DBMS) are still not equipped to manage a continuous update of data from a large number of mobile users (Wolfson, 2002). A data model, *moving objects spatio-temporal* (MOST), which is built on top of existing DBMS has been proposed in (Sistla, et al, 1997). A query language called *Future Temporal Logic* (FTL) for the MOST model was also presented in (Sistla, et al, 1997). This query language is designed to accommodate future query, such as to retrieve cars that will intersect a region within the next 5 minutes. A similar attempt can be found in (Gutting, et al, 2000), which present spatial-temporal data type and its query language to operate as an extension to existing DBMS.

Various strategies have been developed to update the location of the mobile user while minimising transmission cost, and server processing cost, which include time function, indexing, statistical model, and historical data.

- Time Function

The MOST is designed to model the velocity of the object as a function of time $f(t)$ (Sistla, et al, 1997; Wolfson, et al 1999). It is used to provide an estimation of the location of mobile clients at different times. The answer set of continuous query, which employs the MOST model, consist of tuples $(S, begin, end)$. It indicates that object S is the result of the query, which valid from time *begin* to time *end*. Subsequently, the transmission time required to send the tuples in the answer set has to be determined. This technique avoids excessive location updates due to no explicit update being necessary unless there is a change in the parameter $f(t)$.

- Indexing

This approach is to index the locations of mobile clients using some spatial methods and update the index tree each time a mobile client moves (Kollios et al, 1999; Saltenis et al, 2000). This is used to predict the future movement of the client. Pfoser (2002) proposed an indexing technique that indexes the trajectories of the moving client rather than relying on the client's current position to detect the movement of the client.

Cai and Hua (2002) proposed a technique called *Spatial Query Management* (SQM) for continuous query. In this technique, a database map consists of many sub-domains, and each mobile client is assigned to a sub-domain as its resident domain. When the mobile client moves, it detects its position against the monitoring areas inside its resident domain. If any query boundaries are crossed, the mobile client informs the server to update the relevant query results and determine whether the client is still in its resident domain; the server then specifies a new resident domain for the client. A D-tree (Domain tree) indexing was designed to model the domain decomposition hierarchy, which allows the location update process to be cost efficient.

An indexing scheme for location predictive query or future query was reported in (Tao et al, 2003). This future query is categorised as spatio-temporal query, which requests information on a moving object that will intersect with the query window during a future time interval (Tao et al, 2003). The proposed indexing model was called TPR*-Tree or Time Parameterized R*-Tree. TPR*-Tree is

designed to overcome its predecessor, TPR-tree that was proposed by (Saltenis et al, 2000).

TPR-tree is an improved version of R-tree indexing, which is made to accommodate current and anticipated future locations of a moving object in an efficient manner. The unique feature of TPR-tree is that the tree structure consists of functions of time instead of a spatial object. Saltenis and Jansen (2002) also proposed R^{EXP}-Tree, which is concerned with expiring information of moving object, such as the data becomes invalid after a certain time parameter is passed.

- Statistical Model

Location estimation using statistical model was presented by (Roos et al, 2002). In this model, the location of the mobile user is determined based on a number of variables such as the signal power transmitted to the base station, angle of signal arrival, and propagation delay.

- Historical Data

A pre-fetching mechanism for a location-dependent query was proposed by (Kubach and Rothermel, 2000; Kubach and Rothermel, 2001). This technique predicts the information that the mobile client may need in the future, as s/he moves to a different location. Subsequently, the predicted information is transferred to the mobile client at the right moment. Historical data is used to find the user's movement patterns.

- Semantic-based model

The semantic-based model was presented as an extension of GSTD algorithm (Pfooser and Theodoridis, 2003). The GSTD algorithm is designed to generate data sets of the moving object's trajectories that were governed by a number of parameters (Theodoridis et al, 1999). The parameters include duration of the moving object at a given time and location which also involves the change of time stamps between subsequent location, shift of an object that is calculated from centre point shift, and resizing of an object which is more applicable to objects that may be condensed or dispersed such as storms, certain insect or animal populations, and soon.

(Pfooser and Theodoridis, 2003) incorporate other properties into the GSTD algorithm such as speed, heading, and agility of objects. Subsequently, groups of moving objects with similar movement characteristics are clustered. Furthermore, it also concerns the situation where the movement of objects (i.e. cars) is affected by static objects (i.e. buildings, traffic jammed, parks), which is referred as infrastructure.

At present, there is no existing work that deals with the simultaneous movement of both user and sought object. However, it is believed that issues pertaining to these types of queries can be simplified if the solutions for the previous two query types have been found.

2.4.2.2 Data Broadcasting Mechanism

The concept of a data broadcasting mechanism for location-dependent queries is the same as periodic data broadcasting in the traditional query processing mechanism.

Query processing using data broadcasting mechanism refers to query that is processed on air. However, in a location-dependent query, the utilization of data broadcasting is more complex as the client should be informed of when the relevant data will arrive in the channel while considering the current location of the client and the required object as the parameters.

a. Static Location-dependent Data

Traditionally, in the broadcasting technique, the sets of database items are periodically broadcast to multiple clients over a single or multiple channels. Since mobile users may spread into many different regions, the central database server must be intelligent enough to organize its data so that mobile clients in different regions can still efficiently retrieve the relevant data, otherwise mobile clients will waste a lot of unnecessary power during the tuning and listening processes. The problem is exacerbated when the mobile users frequently move from one region to another, requiring frequent tuning as well. Therefore, an efficient data structure for broadcast cycles is critical in saving mobile clients' battery power.

Considering the following scenario: suppose database items to be broadcast are the location of restaurants in a specific state (i.e. Vic.). Tourists are holding a PDA connected wirelessly with the central server. The tourists are able to tune in to the channel to obtain relevant data, such as only nearest restaurants in the region. When the user moves to a different region and tunes in to the incoming channel, the data obtained is adjusted according to the new location of the user. There are at least three main issues here that need to be addressed: (i) data broadcasting protocols (ii) broadcast channels utilization and (iii) efficient data broadcast query processing.

Data broadcasting commonly makes use of indexing to help a mobile client to conserve energy. In this case, the mobile client's tuning time can be reduced by providing accurate information for a client to tune in at the appropriate time for the required data.

Xu et al (2003) proposed a D-tree index for location-dependent queries in data broadcasting environment. The basic concept is to index data regions or valid scopes of data items, and broadcast the index structure, which interleaves with relevant data item over a broadcast channel. Client can tune in to the channel, check the valid scopes with current location, find out when the correct data item will be broadcast, and retrieve the data item on air. The performance of the D-tree index was evaluated against other indexing structures such as the trian-tree index (Kirkpatrick, 1993), trap-tree index (Berg et al, 1997), and R*-tree index (Beckmann and Kriegel, 1990). D-tree was found to be a better scheme compared with the others.

An advantage of data broadcast on multiple channels is that it will cope efficiently with a large amount of database items to be broadcast. Since access to data is sequential, the mobile client has to wait until the desired data arrives on the channel. Furthermore, the expected delay that occurs with a broadcasting index directory can also be overcome. However, one must be careful when determining the maximum number of channels, since a large number of channels will waste bandwidth and incur overhead costs when moving from channel to channel while tuning and accessing the data. In contrast, fewer channels will increase access time.

In location-dependent queries, multiple channels may be used to accommodate different regions. By utilizing an index directory, a mobile client will be able to determine which channel contains the relevant information, and the desired information can be retrieved efficiently. The architecture of the broadcast mechanism for a location-dependent query with single and multiple broadcast channels is illustrated in Figure 2.28. Figure 2.28 (a) shows the partition of cell x into two regions, that is, region a and b . Each cell and the related regions are normally measured in coordinate value. In this example, we simplify the relevant details. Figure 2.28 (b) demonstrates the construction of index tree based on the cell partitioned area, and the bottom leaf of the index contains the data buckets that are relevant to each region. Having obtained the index structure, the broadcast structure can then be created. Finally, the broadcast program may be disseminated into single or multiple channels as given in Figure 2.28 (c).

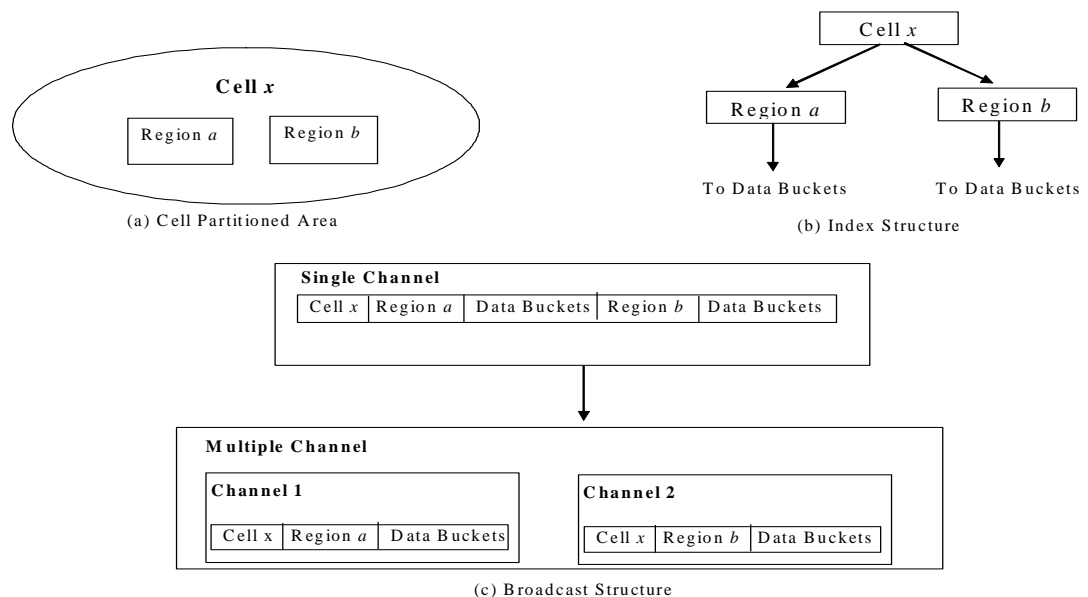


Figure 2.28. Single and multiple channel broadcast structure

Jung et al (2002) proposed a broadcasting scheme for location-dependent queries when the target location is defined for urban areas. It was claimed that it is necessary to consider the characteristics of the target area in order to obtain an efficient query processing. This scheme also broadcast an index directory which informs the order of the broadcast data items. In this case, the cell region is divided into grids and the index is created based on each grids. It is expected that the index size becomes smaller as compared with the indexing of each data item. The broadcast order was determined using a mechanism that is called sequence of space filling curve (Jagadish, 1990). This mechanism is used to cluster spatially adjacent objects and the broadcast order is constructed accordingly. By doing so, the amount of energy consumption needed by the mobile client in order to retrieve the data item can be reduced.

It is certainly believed that the on-air strategies can be improved in a lot of ways. There are several questions that have not been addressed in the existing literature including how mobile clients know when they have moved to a different region and must retune to the channel in order to update the information, and how query results are stored on air when the user is disconnected. Hence, there are plenty of open researches in this area to investigate.

b. Dynamic Location-dependent Data

Existing literature dealing with location-dependent query processing emphasised that broadcast mechanism is just suitable for scalable applications such as traffic reports, tourism information, and weather information (Lee et al, 2002). This type of application does not need the exact position of the client in order to provide the

query result. However, in certain situations, the broadcast mechanism can still be applied as shown in the following examples.

Example 1: Suppose database objects to be broadcast are the location of patrolling police cars, and the user is the police holding a PDA connected wirelessly with the central server. The police are able to tune in to the channel to obtain relevant data, such as only those police cars patrolling the region. When the police move to a different region and tune in to the incoming channel, the data obtained is adjusted according to the new location of the police.

Example 2: Using the same example as Example 1, some of the police cars on the road might be temporarily off-line or outside the region covered by the system. The police in a particular location who incidentally tune in to a broadcast cycle containing this 'missing' information might not be aware of the existence of these off-line police cars and might have misinterpreted the circumstance, even though later in the next broadcast cycles these police cars are already on-line or within the covered region. The main issues in this case include: (i) data structure to capture moving objects in the database, considering disconnection and range covered, and (ii) efficient data broadcast organization and scheduling.

Each broadcast cycle contains a set of data to be broadcast, and these data will be re-broadcast again in the subsequent cycles. When data to be broadcast contain moving objects, in some cycles, some of these objects might be out of the covered range or are temporarily disconnected due to a poor communication. These particular cycles might give misleading information to mobile users especially when they only tune to

one of these cycles and do not re-tune to the subsequent cycles which may have recovered those moving objects. The utilization of data broadcasting scheme to this query type is promising but it also involves some challenging issues to investigate.

2.5 Discussion

This thesis is concerned with the issues of query processing over the data broadcasting mechanism. Therefore, in this section, the major achievements of existing work on data broadcast management for traditional queries and location-dependent queries will be highlighted.

- **Data Broadcast Management for Traditional Queries**

A number of broadcast strategies have been described and each of these provides a different approach in optimizing the power, capacity and bandwidth usage while maintaining a reasonable query response time. The organization of data items on air to match the order of the broadcast data with the order of data required for optimal query processing offers a good solution to improve the mobile client's access time and tuning time.

The optimal organization of database items is identified according to the query access patterns and semantics of the database being broadcast. However, a vigorous cost calculation has to be used which introduces a high degree of complexity in the algorithm before it can decide the best organization. The calculation becomes more complex as the query involves more entity types. Therefore, when there are a large number of database items to be broadcast, this technique has shown its limitation.

Generally, the number of mobile users in a cell determines the length of the broadcast data. As a rule of thumb, the more number of data to be broadcast, the more requests will be served from the data broadcast and this will reduce the chance of mobile clients sending the request to the server. However, at a certain point the advantage of the broadcast data will be diminished if there is too much data in the broadcast cycle. Consequently, it will severely affect the query response time since mobile users have to experience a considerably long delay before they receive the desired data. Therefore, it is essential to decide what data to broadcast in order to serve most of the requests since the query access pattern is changed. A number of techniques have been described to overcome this problem such as Mean Time algorithm, Window algorithm and Exponentially Weighted Moving Average algorithm. These techniques provide a selection mechanism for data broadcast to maintain the number of data items to be broadcast while satisfying most of the requests.

Another technique that can be used to reduce the length of the broadcast data is to split the broadcast channel into multiple channels. The use of multiple channels to broadcast the database items is a good way to keep up with the enormous number of data items and provide a chance for the mobile clients to select the channel which broadcasts the data of interest within a shorter duration. Besides, it also helps with the transmission error and unreliability of the channel.

The hybrid allocation method, which assigns the broadcast channel to both on-demand mode and broadcast mode, offers another strategy to enhance the query performance. This technique manages to provide an optimal solution by determining the number of data to be served from on-demand channel and broadcast channel. The

dynamic hybrid method, which changes the number of channels allocated for the on-demand mode and broadcast mode dynamically by considering the load of the server and request access pattern, is considered a better approach. However, this method simply compares the access overhead of the broadcast channels, which is the waiting time to receive the desired data with the on-demand channels that corresponds to the time to queue for a connection to the server and compromise these two overheads to gain a better access time. However, the use of on-demand channels incur a number of costs such as transmission cost of the query from the mobile client to the server, processing cost in the server and the cost to send the query response back to the mobile client. The cost to transmit the query to server is even greater when the query requires related data items that belong to a number of entity types due to the uplink bandwidth limitation. Thus, the on-demand channel is better used only when the data items of interest are not broadcast.

The use of the indexing technique helps the mobile clients to search for the desired data and determine when the data will arrive. This reduces the query processing time as well as utilizing the power more efficiently. However, one should be aware of the consequences of having too many or too few broadcast directories in a broadcast cycle over the latency time and tuning time. The broadcast disk is another strategy to further improve the query performance. The combining of two or more broadcast strategies such as the use of indexing technique along with broadcast disk strategies provides a good outcome. Table 2.1 shows the comparisons of all prior related schemes and the proposed scheme in this thesis. Note that most prior studies did not

employ broadcast indexing and no previous studies consider broadcast organization while allocating index and data segment in a separate channel.

Table 2.1. Comparison among the proposed scheme and related studies –Traditional Queries

Algorithm	Number of Channels	Replication		Include Indexing Scheme	Index and Data Segment	
		With	Without		Integrated	Separated
Si and Leong (1999)	Single	No	Yes	Yes	No	Yes
Acharya et al (1995)	Single	Yes	No	No	-	-
Prabhakara et al (2000)	Multiple	No	Yes	No	-	-
Tran, at al (2001)	Multiple	No	Yes	No	-	-
Huang and Chen (2002)	Multiple	Yes	No	No	-	-
Huang and Chen (2003)	Multiple	Yes	No	No	-	-
Hurson et al (2000)	Multiple	No	Yes	No	-	-
Liberatore (2002)	Single	No	Yes	No	-	-
Lee et al (2002)	Single	No	Yes	No	-	-
Imielinski (1997)	Single	No	Yes	Yes	Yes	No
Noy et al (2000)	Single	No	Yes	No	-	-
Guanling Lee et al (2003)	Single	No	Yes	No	-	-
Proposed Scheme	Optimum	Yes	Yes	Yes	No	Yes

- **Data Broadcast Management for Location-dependent Queries**

In location-dependent queries where the mobile client's location is relevant to the information requested, or the information requested is based on a particular location, the utilization of the broadcast mechanism is very challenging. Data item organization for nomadic users' queries will be much more complex, as the calculation may become obsolete as soon as the client moves. Data to be broadcast are also naturally changed, depending on the location of the clients. A mobile client should be informed when the relevant data will arrive in the channel while considering the current location of the client as the parameter.

Since mobile users may spread into many different regions, the central database server must be intelligent enough to organize its data so that mobile clients in different regions can still efficiently retrieve the relevant data; otherwise, mobile clients will waste a lot of unnecessary power during the tuning and listening processes. The problem is exacerbated when the mobile users frequently move from one region to another, requiring frequent tuning as well. Therefore, an efficient data structure for broadcast cycles is critical in saving mobile clients' battery power.

So far, data broadcast management for location-dependent queries has not been much explored. In fact, very little research has been undertaken on this issue including (Xu, et al 2003; Xu, et al 2002; Jung, et al 2002). As such, this issue opens up many research problems, which is definitely of great interest to pursue. Table 3.2 shows the comparisons of related scheme and the proposed scheme for location-dependent queries.

Table 3.2. Comparison among the proposed scheme and related studies – Location Dependent Queries

Algorithm	Broadcast Channel	Valid Scope
Xu et al (2003)	Single	Rectangular
Proposed Scheme	Multiple	Square

2.6. Data Broadcast Management Framework

From a data broadcast processing point of view, query classification raises the following important issues.

- *What technique can be used to disseminate a vast amount of data items without any exclusion while maintaining the query access time low as compared to on-demand mechanism?*
- *How can a multiple channel system be utilised and adaptively used in a broadcast paradigm?*
- *How can complex information from the database server be efficiently broadcast considering different types of mobile users requiring different types of information?*
- *How are the data broadcast scheduled?*
- *What is a suitable index structure to determine when each data item arrives in the channel without sacrificing the access time?*
- *What is a suitable index structure to serve location-dependent queries in mobile environment?*
- *What query processing protocols are used in broadcast environment?*

The first two issues are associated with the use of the broadcast channel to provide optimum resources for disseminating a set of data items. The next two issues relate to broadcast ordering and scheduling methods. These issues are concerned with the query access time. The following two issues focus on the indexing scheme to provide efficient data items retrieval by taking into account the efficiency and the type of queries involved. The last issue considers the client's processing protocol in retrieving broadcast data items on air. This issue is determined according to the proposed solution of the earlier issues. Hence, the tasks of a data broadcast management scheme can be summarized as follows.

- Discovering data broadcasting models, ordering and scheduling algorithms for minimising query access time.
- Formulating indexing structure into efficient tuning time and power consumption.

Figure 2.29 depicts the relationship of the task of a data management scheme with the proposed method given by each chapter in this thesis.

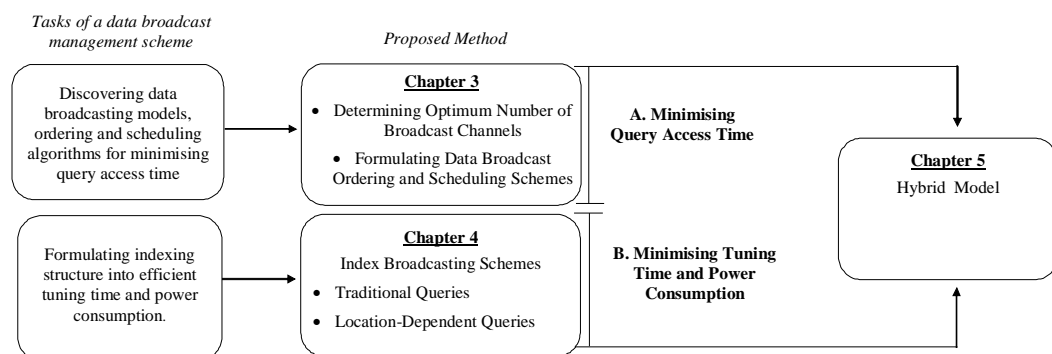


Figure 2.29. Tasks of data management scheme with relation to the proposed method in this thesis

2.7 Conclusion

Queries in a mobile environment to some extent involve a unique class of query that is not available in the traditional distributed system. This is due to the radical evolution from stationary network to wireless network that has created several novel properties, such as mobility, in particular. Queries in a mobile environment are classified into two main categories, namely: context-awareness queries and traditional queries. Context-awareness queries are further divided into three classes; context-dependent queries, location-dependent queries, and hybrid queries.

In location-dependent query, the location of the object becomes the parameter of the query, and since the object continuously moves from one place to another, it requires a complex mechanism to monitor the object location as well as ensure a valid query result. However, due to inherent constraints, such as low bandwidth, short-life battery, limited storage space, and frequent disconnection, information services to mobile clients have been much restricted.

The major contributions of this chapter are:

- Queries taxonomy in a mobile environment are classified. A classification is essential as it makes it possible to identify the types of queries for data management in a wireless environment. A classification consequently serves as a scope of the domain of data broadcast management schemes in mobile environment.

- Query optimisation and processing issues are identified. These issues need to be considered when dealing with data management in a mobile environment.
- A framework for data broadcast management scheme is defined. Generally, a data broadcast management is to provide data broadcast models, scheduling and ordering algorithms. Furthermore, the broadcast indexing model and structure will need to be investigated. This model serves a certain class of query. The index model for location-dependent queries requires a more complex mechanism and processing than traditional queries.

Chapter 3

Optimising Channel Utilisation

3.1 Introduction

An information broadcasting scheme is not new and has been applied in our surroundings for sometime including the radio programs, TV programs, and news paper to name a few. In a mobile computing environment, data broadcasting is an effective mechanism for disseminating database information to mobile clients. The ability to maintain query performance despite the high frequency of requests and large number of mobile clients has made the data broadcast mechanism a desirable technique in mobile database query processing. A number of information services that can benefit from utilising data broadcasting include weather information or weather forecast services, news, stock indices information, foreign exchange, election results, tourist services, airline schedules, location-dependent services, route guidance and so on (<http://www.wapforum.org/>). However, due to sequential access of broadcast

data items in a mobile environment, the increasing number of broadcast database items causes mobile clients to wait for a substantial amount of time before receiving their data items of interest. Consequently, the advantages of broadcast strategy will be diminished. Moreover, the longer the query access time, the longer mobile clients have to listen to the channel and examine the data items on air. This operation is not only time consuming but also very expensive from an energy point of view. Therefore, it is of importance to optimise the query access time of mobile clients in accessing broadcast data items.

The rest of this chapter is organized as follows. Section 3.2 describes the outline of this chapter in details. Section 3.3 presents the proposed scheme for determining optimum number of broadcast channels. Section 3.4 explains the formulation of broadcast ordering and scheduling scheme including one with replication and without replication. Section 3.5 presents the performance evaluation results. It is subsequently followed by discussions in Section 3.6. Finally, Section 3.7 concludes the chapter.

3.2 Preliminaries

This chapter *presents* data broadcast management schemes, which are designed to optimise broadcast channel utilisation so that a client query access time is minimised when retrieving broadcast database items. Figure 3.1 illustrates the outline of this chapter within the broadcast-based information systems.

Figure 3.2 illustrates query access time or elapsed time from the time a request is initiated until all data items of interest are received. This Figure shows the total access time to retrieve data item # 8 for example, which starts from the time the

client probes into the beginning of the data channel until the desired data item has been obtained.

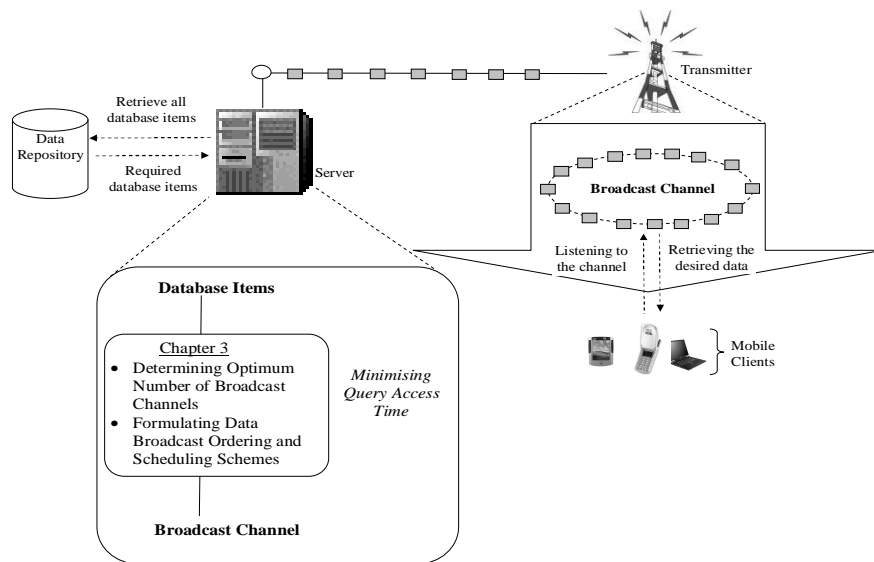
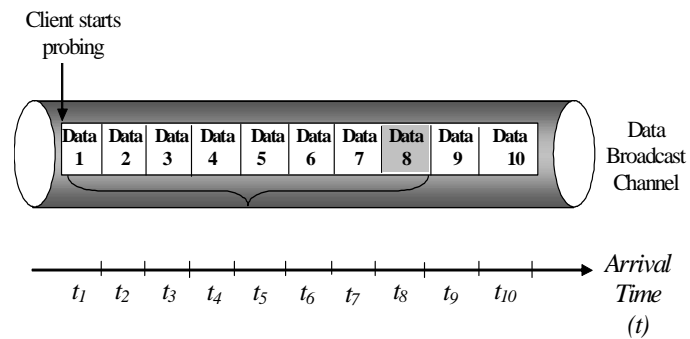


Figure 3.1. Chapter framework



Data Item of Interest = Data item #8

Total Access Time = Total Elapsed Time

$$= \sum_{x=1}^8 t_x$$

Figure 3.2. Query access time

In this chapter, a method for minimising query access time for retrieving broadcast database items is presented. The proposed method is divided into two stages: (i) to determine optimum number of broadcast channel. In this stage, analytical models to calculate query access time over broadcast channel and on-demand channel are presented. These models will be utilised to find the optimum number of items in a channel and thus the number of broadcast channels given a set of broadcast data items, and (ii) to formulate data broadcast ordering and scheduling schemes that are concerned with the access patterns of a mobile client accessing the broadcast data items, will be presented. This scheme is further classified into two categories, namely a broadcast program *with replication* and *without replication*.

a. Determining Optimum Number of Broadcast Channels

Analytical models for both query access time over a broadcast channel and on-demand channel to find the optimum number of items in a channel will be studied. The analytical models are examined to find the optimum number of broadcast items in a channel while utilising query access time over an on-demand channel as a threshold point. The optimum number indicates a point to split the broadcast cycle and allocate the data items to the new channel. When the optimum number is located, the number of broadcast channels should be increased. Subsequently, the length of the broadcast cycle is split, and broadcast over multiple channels. Several factors are taken into account, which include: request arrival rate, service rate, number of request, size of data item, size of request, number of data item to retrieve, and bandwidth. This includes any request that returns single and multiple data items.

The reason for this is to ensure that the access time of queries in retrieving data items over a broadcast channel are always superior to that of an on-demand channel by determining the optimum number of broadcast channel while considering all the advantages of a data broadcasting scheme.

It is assumed that all data items are broadcast and made available for mobile clients to retrieve whatever data items they are interested in over a wireless channel without having to send any queries to the server. This is highly desirable due to the fact that wireless data transmission usually requires a greater amount of power as compared to the reception operation (Zaslavsky and Tari, 1998; Xu et al, 2002). For example, in the case of the Hobbit chip (Argade et al, 1993), the energy consumption for sending data is about one thousand times bigger than for data receiving. It is also amplified by the fact that the life expectancy of a battery (e.g. nickel-cadmium, lithium ion) was estimated to increase the time of effective use by only another 15% (Paulson, 2003). Thus, the need to use power efficiently and effectively is a crucial issue.

b. Formulating Data Broadcast Ordering and Scheduling Schemes

Once the optimum number of channels required to broadcast a set of database items is known, it is then necessary to minimise the average query access time for clients in accessing broadcast data items. In light of this issue, a data broadcast ordering and scheduling scheme for a multi-broadcast channel environment involving data segments at the server side, which aims to minimize query access time, will be investigated. Moreover, it also considers single and multiple data items requests. The broadcast scheduling and allocation schemes are concerned with access patterns of the mobile client in accessing the broadcast data items. This scheme is further

classified into two categories, namely a broadcast program *with replication* and *without replication*. A broadcast program with replication corresponds to the case where the data items appear with different frequencies. The basic idea of a broadcast program with replication is that the most popular data items will be broadcast more often than others. Consequently, the majority of mobile clients will have a considerably shorter access time as their data of interest arrives more frequently in the channel. On the other hand, a broadcast program without replication applies when all data items are broadcast with equal frequencies or uniform frequencies, which is also called a flat broadcast program.

3.3 Determining Optimum Number of Broadcast Channels

In general, the data items are broadcast over a single channel with the underlying reason that it provides the same capacity (bit rate/bandwidth) as multiple channels, and it is used to avoid problems such data broadcast organization and allocation while having more than one channel (Imielinski, Viswanathan and Badrinath, 1997). Nevertheless, the use of a single channel will show its limitation when there is a very large number of a data item to be broadcast. In fact, the number of broadcast channels in a cell that can be simultaneously utilized is up to 200 channels (Leong and Si, 1995).

The proposed strategy splits the length of the broadcast cycle when the number of broadcast items reaches an optimum point. This process continues until the access time is above the optimal point. Figure 3.3 illustrates the situation where a broadcast cycle is split into two channels.

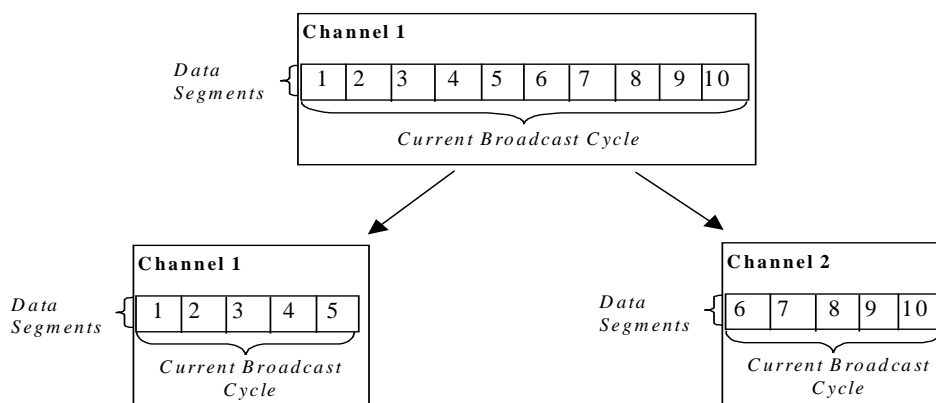


Figure 3.3. Multiple channels architecture

The allocation of a broadcast cycle to an optimal number of broadcast channels will eliminate the possibility of long delay before obtaining the desired data items. Figure 3.4 illustrates a situation where the optimum number of items in a channel is known. As a simple example, assume the size of each data item is 50 bytes. Consider a mobile client who wants to retrieve data item #8. The following two cases analyse the access time of a mobile client when the data is broadcast over a single channel as compared with two channels. It is assumed that the client probes into the first data item in a channel.

Case 1: Data items (1-10) are broadcast in a single channel. In this case, all data items are available in a single channel. The client who wants to retrieve data item #8, has to wait until the desired item arrives in the channel. The total access time required will be $(50 \times 8) = 400$ bytes.

Case 2: Data items (1-10) are split into two broadcast channels: channel one and channel two with 6 and 4 data items respectively as shown in Figure 3.4 . Since data items are split into two broadcast channels, the access time will also be different. In this case, the client can switch to another channel and wait for

the data item of interest to arrive. The total access time for client to retrieve data item #8 would be $(50 \times 2) = 100$ bytes. Thus, in comparison with the first case, this time the client can have a much more efficient data access.

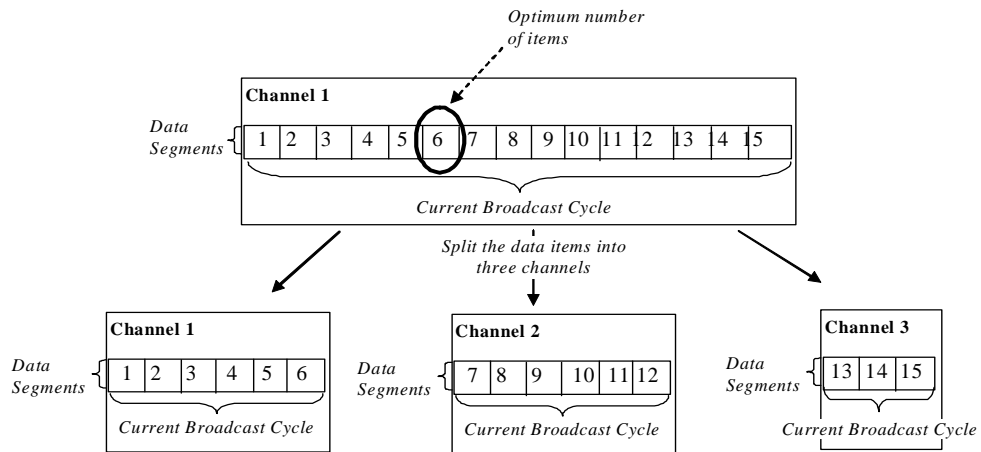


Figure 3.4. Broadcast channel partitioning: an example

The broadcast channel's data access overhead can be reduced by determining the optimum number of data items in a channel. To find the optimum number of broadcast items, analytical models to calculate average access time of the broadcast and the on-demand channel are developed. The query access time for the on-demand channel is used as a threshold point to determine the optimum number of database items to be broadcast in a channel. When the optimum number is located, the number of broadcast channels should be increased. Subsequently, the length of the broadcast cycle is split, and broadcast over multiple channels. In this context, a mobile client can initiate only a single request at a time; the next request has to wait until the first request has been completely responded to.

- **Analytical Model of Broadcast Channel**

The following scenario is used to calculate the average access time (T_B) for retrieving data using a broadcast channel:

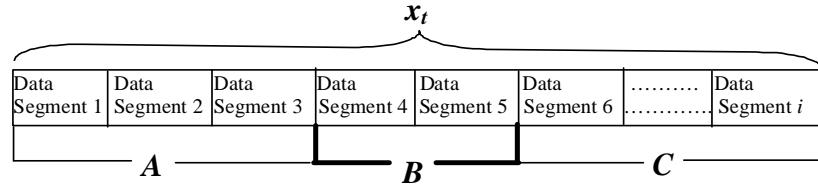


Figure 3.5. Broadcast cycle partitioned area

In Figure 3.5, the broadcast cycle is partitioned into three areas: area A contains data segments preceding data segments in area B , area B contains the number of desired data segments, and area C includes the rest of data segments. x_t is the total number of broadcast data that makes up $A+B+C$. There are three different scenarios where a mobile client probes into one of these areas. s corresponds to the size of the data item, and it is considered to be uniform. The downlink bandwidth is denoted by b_s .

Probe A: When mobile client probes into area A , the average access time is given by:

$$\frac{\sum_{i=0}^{A-1} (A-i+B) \times s}{b_s} = \frac{[A \times (A+2B+1)] \times s}{2 \times b_s} \quad (3.1)$$

Probe B: When mobile client probes into area B , then the average time is equal to the

$$\text{total length of broadcast cycle } (A+B+C) = \frac{(A+B+C) \times B \times s}{b_s} \quad (3.2)$$

Probe C: When mobile client probes into area C , the average access time can be

$$\text{calculated from: } \frac{\sum_{i=0}^{C-1} (C-i+A+B) \times s}{b_s} \text{ or equally } \frac{C \times (2A+2B+C+1) \times s}{2 \times b_s} \quad (3.3)$$

Based on the three scenarios, the average access time from equation (3.1), (3.2), and (3.3) can be calculated as follows:

$$\frac{[(A \times (A+2B+1)) + (2B \times (A+B+C)) + (C \times (2A+2B+C+1))] \times s}{2 \times x_t \times b_s} \quad (3.4)$$

Equation (4) can be rewritten as:

$$T_B \approx \frac{[(x_t - B)^2 + (2 \times B \times x_t) + (2 \times B \times (x_t - B)) + (x_t - B)] \times s}{2 \times x_t \times b_s} \quad (3.5)$$

- **Analytical Model of On-demand Channel**

As mentioned earlier, mobile clients can send their request to be processed via point-to-point or on-demand channel. Thus, the average access time of on-demand channel (T_D) is used to locate the threshold value in determining the optimal broadcast channel. The optimal point where the average access time of the broadcast channel is equal to, or less than, on-demand channel. To calculate the average access time of on-demand channel, there are three cases to be considered:

- Pre-determined Number of Request*

This case considers the situation when the numbers of requests in the server are known prior to processing and will not allow incoming requests to arrive before all requests are completed. The analytical model to calculate the access time of an on-demand channel in equation 3.6 is based on the architecture that is depicted in Figure

2.14 (chapter 2). The access time of an on-demand channel comprises of the time needed to transmit a request to the server, process the request that involves the time spent in the server queue and retrieve the relevant data, and send the result back to the client. The transmission of request to the server and to send the requested data item back to the mobile client are affected by uplink bandwidth, b_r and downlink bandwidth, b_s . Size of a request and the size of requested data item are indicated by r and s respectively, n denotes the total number of requests, i represents the request number and the server service rate is given by μ . However, in a real situation the number of requests is hardly known. Thus, the arrival rate of a request is applied, which is described in the subsequent strategy. The average access time can be calculated using the following formula:

$$T_D = \frac{r}{b_r} + \frac{\sum_{i=1}^n i \times \frac{1}{\mu}}{n} + \frac{s}{b_s} \quad (3.6)$$

b. Request Arrival Rate

The pre-determined number of requests is replaced by the arrival rate of request (λ). Thus, the average access time of on demand channel (T_D) can be calculated as follows:

If the server is lightly loaded $\rho = \left(\frac{\lambda}{\mu} \leq 1\right)$, which means there is no waiting time, then:

$$T_D = \left(\frac{r}{b_r} + \frac{1}{\mu} + \frac{s}{b_s}\right) \quad (3.7)$$

or else if the server heavily loaded $\rho = \left(\frac{\lambda}{\mu} > 1\right)$, then:

$$T_D \approx \left(\frac{r}{b_r} \right) + \frac{\sum_{i=1}^{q_{\max}} \left(i \times \frac{1}{\mu} \right) - \left[(i-1) \times \frac{1}{\lambda} \right]}{q_{\max}} + \left(\frac{s}{b_s} \right) \quad (3.8)$$

The average time in the server queue and the average processing time by the server are dynamically changed depending on the arrival rate of request (λ) and service rate (μ). The maximum number of requests in the server queue is defined by q_{\max} .

- **Uniform and Non Uniform Request**

This category involves requests that return multiple data items. However, the order of the data retrieval is made arbitrary, which means the first arrival of any data of interest will be directly retrieved.

Let $d_1, d_2 \dots d_{\max}$ specifies the number of relevant data items in a request. The number of relevant data items is classified into uniform request that is when the occurrence rate of $d_1, d_2 \dots d_{\max}$ is equally distributed and non-uniform request, where the occurrence rate is otherwise. The non-uniform request is indicated by the corresponding frequency of occurrence f , for each d . Also, $\sum f = f_1 + f_2 + \dots + f_{d_{\max}} = 1$. Subsequently, the range size of request for each d is defined, then calculate the average size $((r_{(d)min} + r_{(d)max})/2)$. $r_{(d)min}$ represents the minimum size of a request, whilst $r_{(d)max}$ relates to the maximum size of a request for the same d .

Similarly, for server service rate, the rate for each d is determined, and find the average service rate $\bar{\mu}$. The formulas in table 3.1 are applied to find the \bar{d} , \bar{r} , and $\bar{\mu}$ for uniform and non-uniform request.

Table 3.1. Uniform and Non-Uniform formulas

<i>Uniform Request</i>	<i>Non Uniform Request</i>
$\bar{d} = \frac{\sum_{i=1}^{d \max} i}{d \max}$	$\bar{d} = \frac{\sum_{i=1}^{d \max} i \times f_i}{d \max}$
$\bar{r} = \frac{\sum_{i=1}^{d \max} \frac{r_{i \min} + r_{i \max}}{2}}{d \max}$	$\bar{r} = \frac{\sum_{i=1}^{d \max} \left(\frac{r_{i \min} + r_{i \max}}{2} \right) \times f_i}{d \max}$
$\bar{\mu} = \frac{\sum_{i=1}^{d \max} \mu_i}{d \max}$	$\bar{\mu} = \frac{\sum_{i=1}^{d \max} \mu_i \times f_i}{d \max}$

The formula in Table 3.1 of the analytical model of both the broadcast channel and on-demand channel can then be modified as follows:

$$T_B \approx \frac{[(x_t - \bar{d})^2 + (2 \times \bar{d} \times x_t) + (2 \times \bar{d} \times (x_t - \bar{d})) + (x_t - \bar{d})] \times s}{2 \times x_t \times b_s} \quad (3.9)$$

Pre-determined number of requests:

$$T_D \approx \frac{\bar{r}}{b_r} + \frac{\sum_{i=1}^n i \times \frac{1}{\bar{\mu}}}{n} + \frac{\bar{d} \times s}{b_s} \quad (3.10)$$

If the server is at light load $\rho = \left(\frac{\lambda}{\bar{\mu}} \leq 1 \right)$, then:

$$T_D \approx \left(\frac{\bar{r}}{b_r} + \frac{1}{\bar{\mu}} + \frac{\bar{d} \times s}{b_s} \right) \quad (3.11)$$

or else if the server is at heavy load $\rho = \left(\frac{\lambda}{\bar{\mu}} > 1 \right)$, then:

$$T_D \approx \left(\frac{\bar{r}}{b_r} \right) + \frac{\sum_{i=1}^{q_{\max}} \left(i \times \frac{1}{\bar{\mu}} \right) - \left[(i-1) \times \frac{1}{\bar{\lambda}} \right]}{q_{\max}} + \left(\frac{\bar{d} \times s}{b_s} \right) \quad (3.12)$$

3.4 Formulating Data Broadcast Ordering and Scheduling Schemes

The optimum number of broadcast channels determines the number of channels required to broadcast a set of database items. It is optimum in a sense that the average query access time of accessing a broadcast channel will always outperform the average query access time over an on-demand channel in any situation. In this section, the query access time is further minimized considering the access patterns of mobile client accessing the broadcast data items. A knowledge of the access patterns enables the broadcast server to construct a desirable data broadcast stream, which is concerned with broadcast ordering and scheduling scheme of data items.

The scheme is further classified into two categories, namely a broadcast program *without replication* and *with replication*. A broadcast program is without replication when all data items are broadcast with equal frequencies or uniform frequency, which is also called a flat broadcast program (see Figure 3.6 (a)). Whilst, a broadcast program with replication corresponds to the case that the data items appear with different frequencies (see Figure 3.6 (b)). In this example, data item #1 is broadcast more frequently than others.

In this thesis, the ordering schemes consider data allocation over multiple broadcast channels. Moreover, it also includes single and multiple data items requests. The

scheme is designed to ensure that most requested data items stay close to each other in the channel. Consequently, the average query access time of the mobile client to retrieve the desired database items is expected to be reduced.

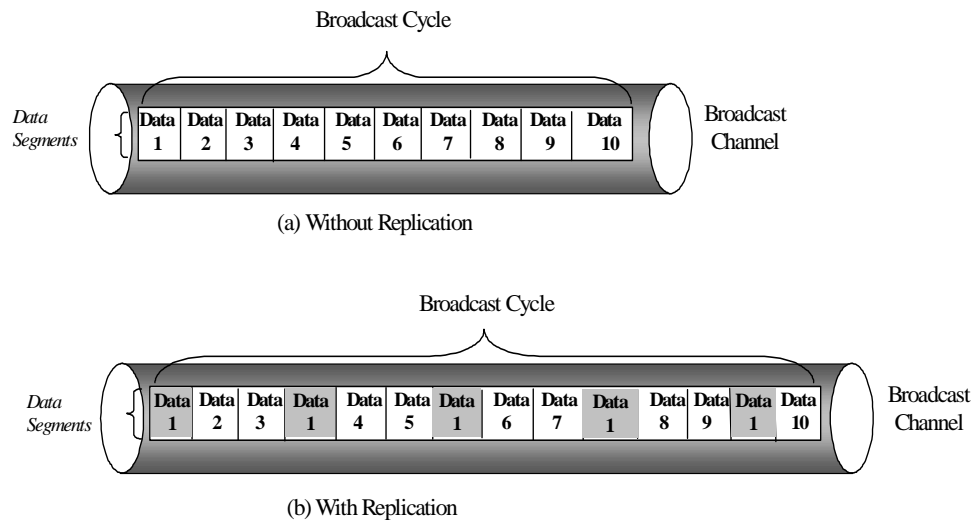


Figure 3.6. Broadcast program – with and without replication

A number of applications in this category include a situation when the mobile client wants to obtain information about more than one stock price at the same time (i.e. to list the stock prices of all car companies under General Motors corp.) or when the client wants to retrieve the weather forecast of a number of cities concurrently (i.e. to list the weather forecast of all cities in a certain region). To accommodate this type of request, the relationship between one data item and the others will be considered. Although the retrieval of multiple data items is the primary objective, the proposed scheme also considers single data item retrieval. Thus, the application of this scheme is not limited to either case.

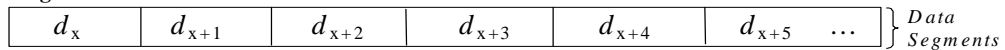
In order to be able to organise the placement of broadcast data items, it is essential to obtain information about the behaviour of mobile users in accessing the broadcast

information. To find the query access patterns of mobile clients, either one of the following mechanism may be incorporated. One is to collect the access information from each mobile client at regular interval; the second is to determine the access information from the behaviour of offline or desktop users with the assumption that the same data items are accessible through the internet. Thus, it can be assumed that offline users and wireless users have similar access patterns. Once the query patterns information is received by the broadcast server, the statistics will be compiled and the broadcast organisation scheme will be implemented. Some analysis on query patterns and access information has been reported by the Microsoft research group in (Adya et al ,2001). It was also found that the query patterns of mobile and stationary users accessing information via the internet do not closely follow the zipf distribution.

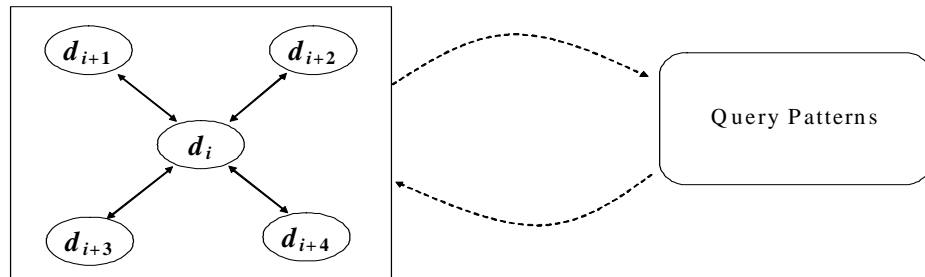
3.4.1 Without Replication

Let denotes $D = \{d_1, d_2 \dots d_m\}$, be a set of data items to be broadcast by the server, and Q as a set of queries $\{Q_1, Q_2, \dots, Q_n\}$. In this case, it is assumed the data item has an equal size and the order of the retrieval can be arbitrary, which means if any of the required data by Q_i arrives in the channel, it will be retrieved first. Each query, Q_i , accesses a number of data items j , where $j \subset D$. The broadcast channel is indicated by C , and the length of the broadcast cycle in a channel is given by BC . Let denotes the broadcast schedule as $S = \{d_x, d_y, \dots, d_z\}$. Similarly, the broadcast program for each channel is defined by SC .

Stage 1:



Stage 2:



Stage 3:

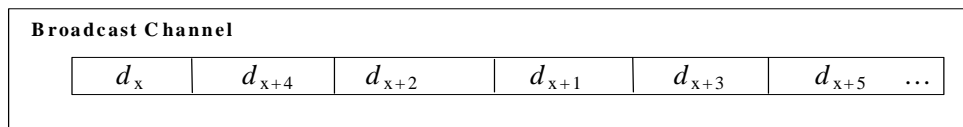
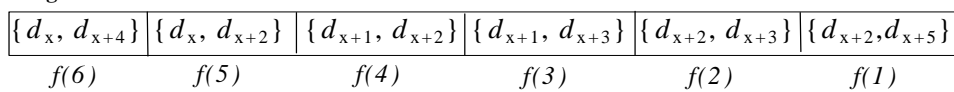


Figure 3.7. Broadcast ordering scheme –without replication

This scheme is depicted in Figure 3.7. The first stage is to list the data items in a sequence order. The second stage is to analyse the relationship between data items and calculate the access frequency of each pair of data item according to the given query patterns. The final stage is to order the pairs of data items based on the value of the access frequency in a descending order. Subsequently, they are placed into one or more broadcast channels. The number of broadcast channels will follow the requirement of the optimum number of channels required to broadcast the data items.

A detailed mechanism of this scheme is described as follows:

- *Firstly*, the data items will be ordered based on the popularity of the item in each query. The popularity of data items is decided by the access frequency of each data item given by each query in Q . Subsequently, the data items will be sorted in

decreasing order of value of access frequency. The result will be temporarily stored in STI , and incorporated into the final broadcast order S after the second stage is finalized. The algorithm of this stage is given in Figure 3.8.

Algorithm 1. Broadcast Ordering Algorithm – Stage 1

Input: An array of data items (D), queries (Q_i) with their related data items (j_i), and the total number of data items to be broadcast (m).

Output: An array of broadcast program (STI)

Procedure:

1. Initialise $i = 1, T = 0$
 2. **Do while** $i \leq m$
 3. **For** $n = 1$ to Q_{\max}
 4. Count the frequency of d_i in Q_n
 5. Store and cumulate the result in temporary variable T
 6. **If** Q_{\max} then
 7. $STI_{(i)} = T$
 8. Clear T
 9. **End if**
 10. **Next** n
 11. increment i
 12. **Loop**
 13. Check the largest value in STI
 14. Ordered the value STI such that $STI(1) \geq STI(2) \geq \dots \geq STI(m)$
 15. **End Procedure**
-

Figure 3.8. Stage 1 algorithm – without replication

- *Secondly*, the relationship between one data item and the other will be analysed.

The access frequency of each data item in relation to other data items will be assessed from each query in Q . This stage is required in order to group the related data items close to each other. Thus, the number of calculations required for this purpose will be $\frac{m \times (m-1)}{2}$, m is the total number of data items in D .

Figure 3.9 illustrates this stage. It shows that each data item to be broadcast will be analysed for its relationship with other data items based on Q . The calculation is done in a sequential basis. i in Figure 3.9 reads the data item number. This process iterates until each data item has been counted. A single process includes a two-way relationship of the data. Figure 3.10 depicts an example where there are

5 data items to be broadcast. Each data item will be counted for its relationship with the next sequence of the data item from given Q . The double point arrows indicate the two-way relationship of the two data. In this case, $D = \{d_1, d_2, d_3, d_4, d_5\}$, and count the frequency of the two data items, which appear at the same time in a query. Start from d_1 , which is counted towards the relation with the next sequences of the data item that is d_2, d_3, d_4, d_5 .

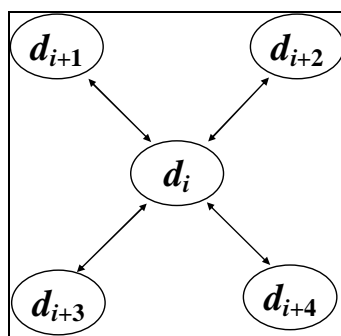


Figure 3.9. The architecture of the proposed method

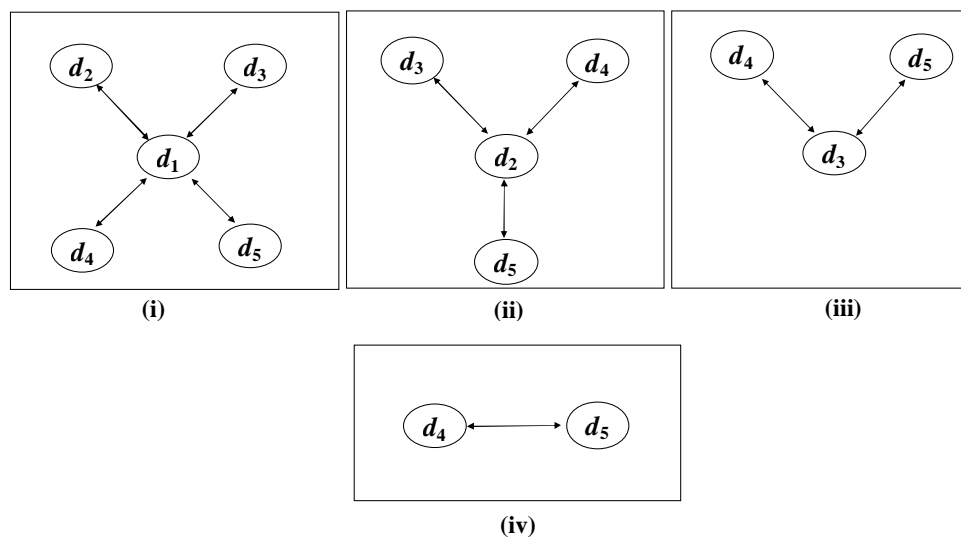


Figure 3.10. An example of the proposed method process

This process continues sequentially until $m-1$ data item. After all data items have been analysed, they are sorted in non-increasing order based on the access

frequency with other data items, and stored in $ST2$. Each allocation will involve two data items. If there is a duplication of data items within $ST2$, then data items with the highest $ST2$ value are kept and the rest are removed. Figure 3.11 shows the algorithm of the second stage.

Algorithm 2. Broadcast Ordering Algorithm – Stage 2

Input: An array of data items (D), queries (Q_i) with their related data items (j_i), and the total number of data items (m).

Output: Array of broadcast program ($ST2$).

Procedure:

1. Initialise $i = 1, T = 0$
 2. **Do while** $i < m$
 3. **For** $k = (i + 1)$ to m
 4. **For** $n = 1$ to Q_{\max}
 5. Count the frequency of d_i retrieved at the same time with d_k in Q_n
 6. Store and cumulate the result in temporary variable T
 7. **If** Q_{\max} then
 8. $TDarray(d_i, d_k) = T$
 9. Clear T
 10. **End if**
 11. **Next** n
 12. **Next** k
 13. increment i
 14. **Loop**
 15. Check the largest value in $TDarray$
 16. Ordered the value in $TDarray$ from the largest to the smallest value such that $TDarray(1) \geq TDarray(2) \geq \dots \geq TDarray(\max)$
 17. The coordinate value for each $TDarray$ determine the data item
 18. Place the coordinate value for each $TDarray$ to $ST2$ with the non-increasing order. Check for any duplicate data item within $TDarray$. Data item can only appear one for each cycle. Thus, in case there is any duplication, data item with the highest $TDarray$ value are kept and the rests are removed.
 19. **End Procedure**
-

Figure 3.11. Stage 2 algorithm– without replication

- The *final stage* is to combine the result of the first and second stages or $ST1$ and $ST2$ correspondingly to form a single broadcast order S . The allocation is started by moving data items in $ST2$ into S also in non-increasing order. This process continues until just one access frequency is left to be processed. Subsequently, the result from the $ST1$ is obtained, analyze it against S , the data item in $ST1$ that has existed in S will be discarded. The left over data item is put into S in non-increasing order as well. The broadcast order is then allocated over

multiple channels. Assuming the required number of data items for each channel is known, which is indicated by BC , once the BC is reached, the subsequent data items will be broadcast in another channel. The algorithm of this mechanism is given in Figure 3.12.

Algorithm 3. Broadcast Ordering Algorithm – Stage 3

Input: Two arrays of Broadcast schedule ($ST1$ and $ST2$), optimum length of broadcast cycle in a channel (BC)
Output: An array of Final Broadcast program (S), two dimensional array of broadcast program for each channel (SC)

Procedure:

1. **For** $i = 1$ to $ST2_{(max)}$
2. Move $ST2_{(i)}$ to S
3. **Next** i
4. **For** $i = 1$ to $ST1_{(max)}$
5. Check if $ST1_{(i)}$ exists in S
6. **If** $ST1_{(i)} \subset S$ then
7. skip
8. **Else**
9. Move $ST1_{(i)}$ to S_{max+1}
10. **End if**
11. **Next** i
12. Let channel number = x
13. Initialise $x = 1, i = 1, j = 1, k = 1$
14. **Do While** $i \leq S_{(max)}$
15. **If** $k \leq BC$
16. $SC(x, j) = S(i)$
17. increment j
18. **Else**
19. **if** $k > BC$ then
20. $k = 1 ; j = 1$
21. $x = x + 1$
22. **end if**
23. **End if**
24. increment i
25. increment k
26. **Loop**
27. **End Procedure**

Figure 3.12. Final stage algorithm– without replication

A knowledge of the broadcast order enables us to determine the sequence of data items that minimise the average access time. The next issue to consider is to specify the broadcast program. The broadcast program indicates the schedule of the first item in S at the broadcast cycle, and the sequential items follow the order that has been generated. To achieve the most effective broadcast program, one must note the

statistical patterns of users when they start listening or probing into the channel. For example, assume the pattern follows the behaviour of normal distribution. The broadcast order *SC* and *SIC* can be allocated to the program in such a way that the average access time can be minimised. For instance, this is done by allocating the first item in each *SC* and *SIC* at point $3/4$ in the broadcast program. This point can be chosen to minimise the chance of clients missing the data of interest and having to wait for the subsequent cycle. In this way, the majority of requests are served within a single broadcast cycle. However, this means about 50% of requests have to wait for a short time period but the overall performance will be better since only a fraction of requests needs to wait for the next cycle. The broadcast program can then be generated at regular intervals.

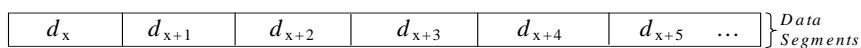
3.4.2 With Replication

This scheme is similar to the previous broadcast ordering and scheduling scheme without replication except that in this case a certain degree of replication is applied so that data items with high access frequency will be broadcast more often than the others.

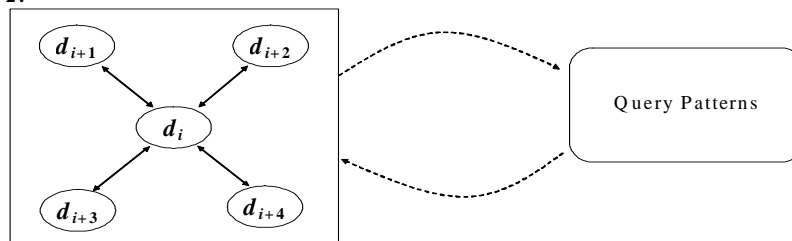
This scheme is partially replicated in a sense that only data items with access frequency greater than one will be replicated in a certain order. The data items with one or less access frequency will be located in a separate broadcast channel. An example is depicted in Figure 3.13. In Figure 3.13, the first stage contains the list of data items with their frequency of occurrence. The next stage analyses the relationship between data items, and order the result based on the access frequency of the pair of data item in each query based on the query patterns. In the third stage, the

pairs of data items with access frequency greater than 1 are allocated to a separate channel. Subsequently, they are organised in a certain way so that pairs with greater access frequency arrive more frequently in the channel. The pairs of data items with access frequency one or less will be organised in descending order based on the access frequency with no particular arrangement of the data items and no replication involved. The final result of the broadcast program in two channels can be split following the optimum number of channels required to broadcast the data items.

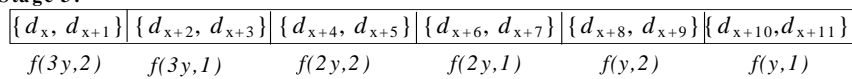
Stage 1:



Stage 2:



Stage 3:



Stage 4:

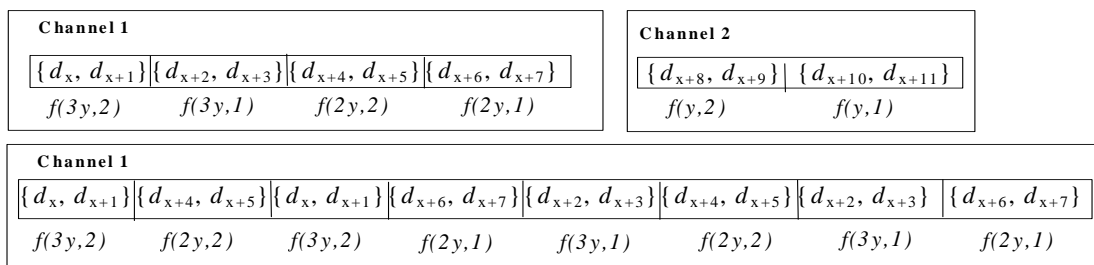


Figure 3.13. Broadcast ordering scheme – with replication

A detailed mechanism of the steps is described as follows:

- *Firstly*, the access frequency of each data item given by Q is calculated. Subsequently, the data items will be sorted in decreasing order of value of access

frequency. The result will be temporarily stored in ST_1 . The algorithm of this stage is given in Figure 3.14.

Algorithm 1. Broadcast Ordering Algorithm – Stage 1

Input: An array of data items (D), queries (Q_i) with their related data items (j_i), and the total number of data items to be broadcast (m).

Output: An array of broadcast program (STI)

Procedure:

1. Initialise $T = 0$
 2. **Do while** $counter1 \leq m$
 3. **For** $counter2 = 1$ to Q_{max}
 4. Count the frequency of $d_{counter1}$ in $Q_{counter2}$
 5. Store and cumulate the result in temporary variable T
 6. **If** Q_{max} then
 7. $STI_{(counter1)} = T$
 8. Clear T
 9. **End if**
 10. **Next**
 11. increment $counter1$
 12. **Loop**
 13. Check the largest value in STI
 14. Ordered the value STI such that $STI(1) \geq STI(2) \geq \dots \geq STI(m)$
 15. **End Procedure**
-

Figure 3.14. Stage 1 algorithm– with replication

- *Secondly*, the relationship between one data item and the other will be analysed.

The access frequency of each data item in relation to other data items will be assessed from given Q . This stage is required to allocate the related data items so that they are close to each other. Thus, the number of calculation required for this purpose will be $\frac{m \times (m-1)}{2}$, m is the total number of data items in D . Each data item will be counted for its relationship with the next sequence of the data item from given Q .

This process continues sequentially until $m-1$ data item. After all data items have been analysed, they are sorted in non-increasing order based on the access frequency with other data item, and stored in $TDarray$. Each allocation will involve two data items. Figure 3.15 shows this stage algorithm.

Algorithm 2. Broadcast Ordering Algorithm – Stage 2

Input: An array of data items (D), queries (Q_i) with their related data items (j_i), and the total number of data items (m).

Output: Array of broadcast program ($TDarray$).

Procedure:

1. Initialise $T = 0$
 2. **Do while** $counter1 < m$
 3. **For** $counter2 = (counter1 + 1)$ to m
 4. **For** $counter3 = 1$ to Q_{max}
 5. Count the frequency of $d_{counter1}$ retrieved at the same time with $d_{counter2}$ in $Q_{counter3}$
 6. Store and cumulate the result in temporary variable T
 7. **If** Q_{max} then
 8. $TDarray(d_{counter1}, d_{counter2}) = T$
 9. Clear T
 10. **End if**
 11. **Next**
 12. **Next**
 13. increment $counter1$
 14. **Loop**
 15. Check the largest value in $TDarray$
 16. Ordered the value in $TDarray$ from the largest to the smallest value such that $TDarray(1) \geq TDarray(2) \geq \dots \geq TDarray(max)$
 17. The coordinate value for each $TDarray$ determine the data item
 18. **End Procedure**
-

Figure 3.15. Stage 2 algorithm – with replication

- *Thirdly*, the broadcast data is split into two channels. For pairs of data items in the $TDarray$, the data items are separated based on the access frequency. Pairs of data items with access frequency greater than one are allocated to one channel, whilst the rest are in another channel. An example of this stage can be seen in Figure 3.13. As shown in the Figure, there are six pairs of data items, each of which is unique. The access frequency (f) indicates how many times the pairs are accessed based on the query pattern. $f(3y,1)$ means the relevant data items have three times appeared consecutively in the list of query patterns ($3y$), and one reads as pairs that have the related access frequency. Having allocated the data items with greater than one access frequency to a separate channel, the order of the items is then organised in a certain manner. The data items with high access frequency will be broadcast more frequently than the others. Figure 3.13 stage 3 depicts how to organize the data items. In case there is duplication of data items

within two pairs, then one of them is kept and the other is removed. Figure 3.16 shows the algorithm of this mechanism.

Algorithm 3. Broadcast Ordering Algorithm – Stage 3

Input: An array of data items (D), the total number of data items (m), $TDarray$

Output: Array of broadcast program ($ST2$).

Procedure:

1. Initialise $counter1 = 0$, $counter5 = 0$
2. Find the highest access frequency in $TDarray$, and allocate it to $counter1$
3. **For** $counter2 = counter1$ To 2 Step -1
4. **For** $counter3 = i$ to $(m-1)$
5. **For** $counter4 = (i+1)$ to m
6. **If** $TDarray(counter3, counter4) = counter2$ Then
7. $counter5 = counter5 + 1$
8. Store the result in temporary two dimensional array ($array1$)
9. $array1(counter2, counter5) = counter2$
10. **End If**
11. **Next**
12. **Next**
13. $counter5 = 0$
14. **Next**
15. Initialise $counter2 = 1$
16. **Do**
17. **Do While** $array1(counter1, counter2) \neq \text{Empty}$
18. **For** $counter3 = i$ to $(m-1)$
19. **For** $counter4 = (i+1)$ to m
20. **If** $TDarray(counter3, counter4) = array1(counter1, counter2)$ Then
21. Store and cumulate the result ($counter3, counter4$) in temporary array ($array2$)
22. Initialise $counter5 = 1$
23. **For** $counter6 = i$ to $(m-1)$
24. **For** $counter7 = (i+1)$ to m
25. **If** $array1(counter1-1, counter5) \neq \text{Empty}$ Then
26. **If** $TDarray(counter6, counter7) = array1(counter1-1, counter5)$ Then
27. Store and cumulate the result ($counter6, counter7$) in temporary array ($array2$)
28. $counter5 = counter5 + 1$
29. **End If**
30. **Else** Exit For
31. **End If**
32. **Next**
33. **If** $array1(counter1-1, counter5) = \text{Empty}$ Then
34. Exit For
35. **End If**
36. **Next**
37. $counter2 = counter2 + 1$
38. **End If**
39. **If** $array1(counter1, counter2) = \text{Empty}$ Then
40. Exit For
41. **End If**
42. **Next**
43. **If** $array1(counter1, counter2) = \text{Empty}$ Then
44. Exit For
45. **End If**
46. **Next**
47. **Loop**
48. $counter1 = counter1 - 1$
49. $counter2 = 1$
50. **Loop Until** $counter1 < 2$
51. **For** $counter1 = \text{LBound}(array2)$ To $\text{UBound}(array2)$
52. Check for replication for every two pairs of data items
53. Eliminate the duplicate data
54. Store the result in $ST2$.
55. **Next**
56. **End Procedure**

Figure 3.16. Stage 3 algorithm – with replication

- The *final stage* is concerned with organizing the data items in the other channel. The *ST1* and *TDArray*, which contains one or less access frequency is combined. The process begins by allocating the data item in *TDArray* into *ST3* also in non-increasing order. Subsequently, the result from the *ST1* is taken, analyse it against *ST3*, the data item in *ST1* that has existed in *ST3* will be discarded. The left over data item is put into *ST3* in non-increasing order as well. The algorithm of this mechanism is given in Figure 3.17.

Having applied all the four stages, the final broadcast program is then available from *ST2* and *ST3*. This program can be generated at regular intervals.

Algorithm 4. Broadcast Ordering Algorithm – Final Stage

Input: Two arrays of Broadcast schedule (*ST1* and *TDArray*)

Output: An array of Broadcast program (*ST3*)

Procedure:

1. **For** *counter1* = *i* to (*m-1*)
 2. **For** *counter2* = (*i+1*) to *m*
 3. **If** *TDarray(counter1, counter2)* = 1 **Then**
 4. Store and Cumulate (*counter1, counter2*) to temporary array (*array1*)
 5. **End if**
 6. **Next**
 7. **Next**
 8. **For** *counter1* = LBound(*array1*) To UBound(*array1*)
 9. Check for replication for every two pairs of data items
 10. Eliminate the duplicate data
 11. Store the result in *ST3*.
 12. **Next**
 13. **For** *counter1* = 1 to *ST1* (*max*)
 14. Check if *ST1 (counter1)* exists in *ST3*
 15. **If** *ST₁ (counter1) ⊂ ST3* then
 16. skip
 17. **Else**
 18. Move *ST1 (counter1)* to *ST3_{max+1}*
 19. **End if**
 20. **Next**
 21. **End Procedure**
-

Figure 3.17. Final stage algorithm– with replication

3.5 Performance Evaluation

This section evaluates the performance of the proposed data broadcast management schemes. The evaluation is divided into two parts according to the proposed strategies: (a) determining optimum number of broadcast channels, and (b) formulating broadcast ordering and scheduling schemes. The simulation is carried out using a simulation package *Planimate*, animated planning platforms (Seeley, 1997). *Planimate*[®] is a software platform designed for prototyping, developing and operating highly visual and interactive business applications. It is a discrete event simulation and end user application development platform. The *Planimate*[®] visual platform incorporates several fundamental, built-in capabilities such as:

- animation,
- handling of concurrency,
- visual workflow modelling,
- dynamic time-based modelling (simulations),
- tools to create interactive GUIs.

These fundamental capabilities - animation, concurrency, workflow modelling, and time-based modelling - are not present in other visual platforms such as Visual Basic[™], or Spreadsheets, etc.

The platform is designed for inter-operability and for TCP/IP networking. It was written in C++, and included its own sophisticated animation and interactive GUI (*iFlow*[™]). It is a truly object-orientated in design. *PLANimate*[™] runs under Windows. Appendix A provides further details on this simulation platform.

3.5.1 Determining Optimum Number of Broadcast Channels

This evaluation section relates to Chapter 3.3 of this chapter. Three cases are introduced, which represent the three categories of on-demand model, namely: pre-determined request, request arrival rate (lightly load) and request arrival rate (highly load). In these cases, the analytical models are applied to determine the optimum number of broadcast data items. Simulated evaluations have been developed to verify the results of the analytical models.

- *Simulation Setup*

In the simulation environment both arrival rate and server service time is exponentially distributed with a given average value. In the simulation model, three facilities are built to behave as multiple mobile clients, the simulation process is run for five iteration times, and derive the average result accordingly. Each evaluation includes requests that return one or more number of items. Subsequently, the request is classified into two scenarios: one is uniform request and the other is non-uniform request. Table 3.2 and 3.3 contains set of parameter of concerns.

Table 3.2. Parameters of concern - determining optimum number of channels

Parameter	Description	Initial Value
<i>Broadcast channel</i>		
x_t	Total number of Broadcast Items	450
<i>On demand channel</i>		
n	Number of request (pre-determined number)	50
b_r	Uplink Bandwidth	9600 bytes
<i>Broadcast and On demand channel</i>		
b_s	Downlink Bandwidth	19200 bytes
s	Size of each data item	1000 bytes

Table 3.3. Set of parameters for Uniform and Non-Uniform request

Number of Returned Data items (d)	1	2	3	4
<i>Non-Uniform Request</i>				
Frequency of Occurrence (f)	0.4	0.3	0.2	0.1
<i>Uniform Request</i>				
Frequency of Occurrence (f)	0.25	0.25	0.25	0.25
<i>Non-Uniform and Uniform Request</i>				
Size of Request (r)	50-100 bytes	101-150 bytes	151-200 bytes	201-250 bytes
Service Rate (μ)	4 request per sec	3 request per sec	2 request per sec	1 request per sec

- *Performance Results*

Case 1: To find the optimum number of broadcast items with a pre-determined number of requests (uniform and non-uniform requests).

This case is categorized into a pre-determined number of requests with uniform and non-uniform request. As shown in Figure 3.18, it tries to find the cross line of on-demand channel and broadcast channel. The average access time of on-demand channel is the threshold point, which appears in a straight line since the number of broadcast items does not affect its value. The intersection point indicates the optimum number of broadcast items in a channel.

From Table 3.4, it can be seen that (see performance analysis of this section for accurate results) the optimum number of broadcast items with uniform request, that is between 390-400 data items (analytical result), and 400-410 data items

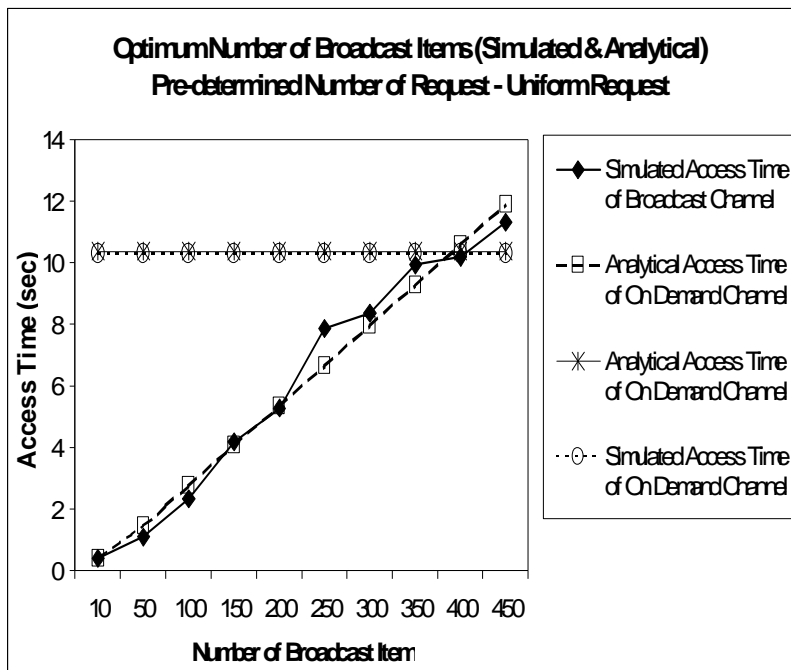
(simulation result). Similarly, Table 3.5, shows the optimum broadcast items for non-uniform request.

Case 2: Include this arrival rate of request (λ) as a parameter in this simulation. It is specified $\lambda = 1$ per two seconds, to obtain server utilization $\rho < 1$.

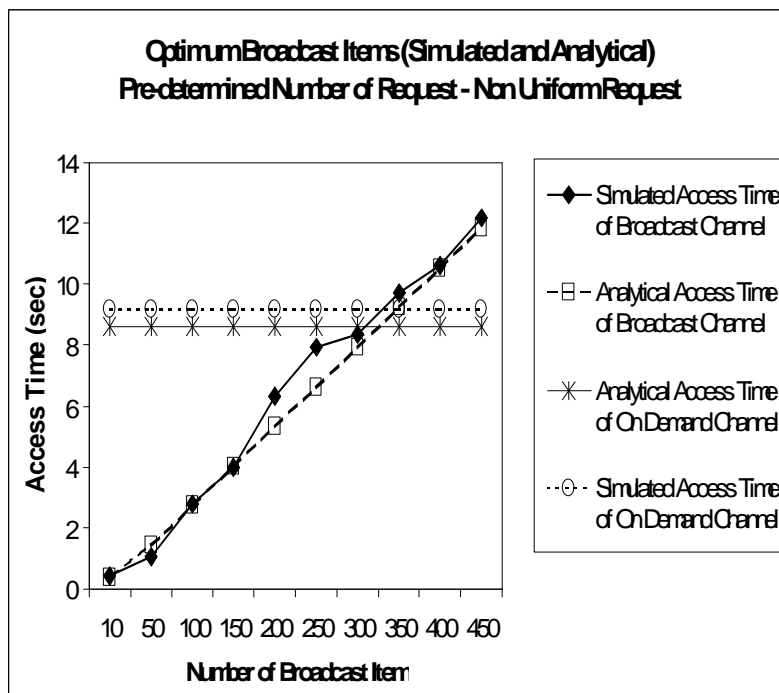
As shown in Figure 3.19, the on-demand channel performs well when the traffic request is low. The broadcast channel seems to have to allocate its data items in every 10-15 data items to a new channel to keep up with the performance of on demand channel. However, this is not a good case since the number of channels may be excessive and the bandwidth may be excessively wasted.

Case 3: In this case, request arrival rate is specified, as $\lambda = 4$ per second to obtain server utilization $\rho > 1$.

Figure 3.20 denotes the optimum number of items when the server is at heavy load. Figure 3.20 (a) shows the optimum number of broadcast items with a uniform request, which is approximately 300 data items from both the analytical and simulation results. However, there is a gap between the analytical and simulated result in Figure 3.20 (b) when involving a non-uniform request. The explanation for this might be due to the exponential distribution of the arrival rate affecting the result quite substantially for the on-demand channel, especially when the server is overloaded with the processing of non-uniform requests.



(a) Pre-determined Number of Request with Uniform Request



(b) Pre-determined Number of Request with Non-Uniform Request

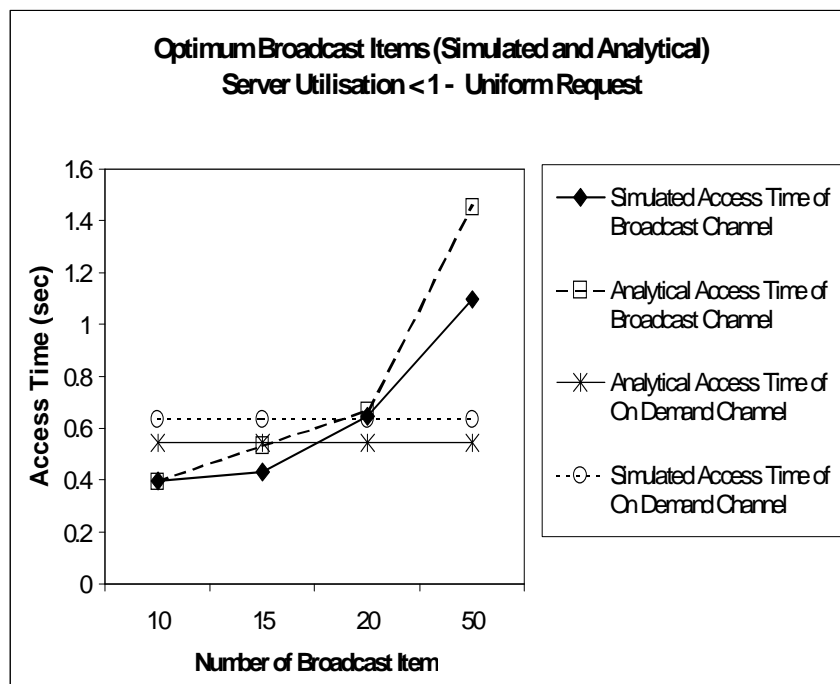
Figure 3.18. Optimum number of broadcast items with pre-determined number of requests (uniform and non-uniform requests)

Table 3.4. Average access time of on-demand and broadcast channel (pre-determined number of requests with uniform requests)

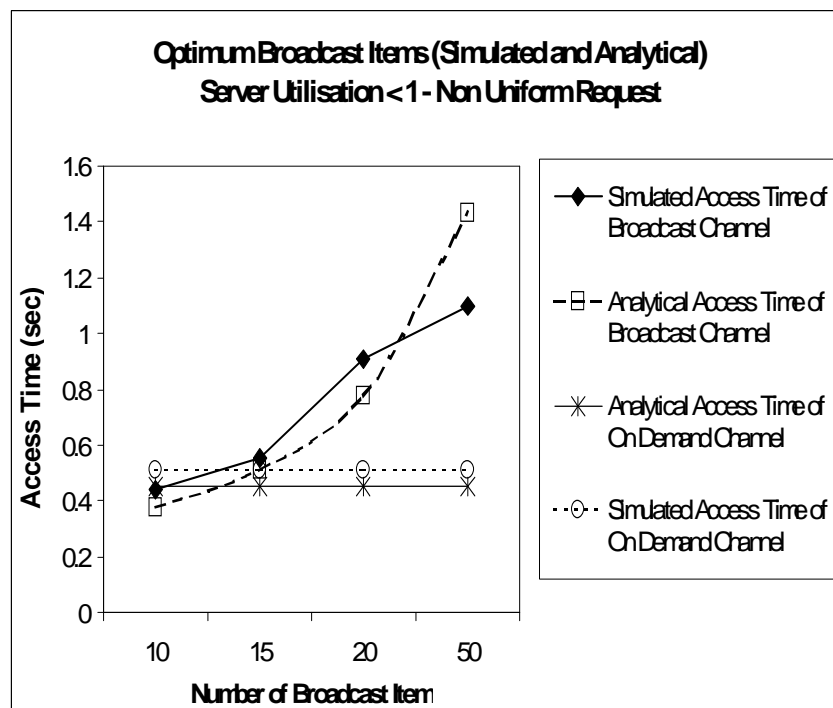
On-Demand Channel		Broadcast Channel		
Analytical Average Access Time (sec)	Simulated Average Access Time (sec)	Number of Broadcast Items	Analytical Average Access Time (sec)	Simulated Average Access Time (sec)
10.3458	10.3013	10	0.394	0.396
		50	1.454	1.0991
		100	2.758	2.326
		150	4.061	4.183
		200	5.363	5.279
		250	6.666	7.8675
		300	7.968	8.373
		350	9.270	9.943
		400	10.572	10.173
		450	11.874	11.295

Table 3.5. Average access time of on demand and broadcast channel (pre-determined number of requests with non-uniform requests)

On-Demand Channel		Broadcast Channel		
Analytical Average Access Time (sec)	Simulated Average Access Time (sec)	Number of Broadcast Items	Analytical Average Access Time (sec)	Simulated Average Access Time (sec)
8.617	9.196	10	0.375	0.396
		50	1.429	1.0466
		100	2.733	2.794
		150	4.035	4.0075
		200	5.338	6.339
		250	6.640	7.925
		300	7.942	8.3725
		350	9.244	9.725
		400	10.546	10.63
		450	11.849	12.189

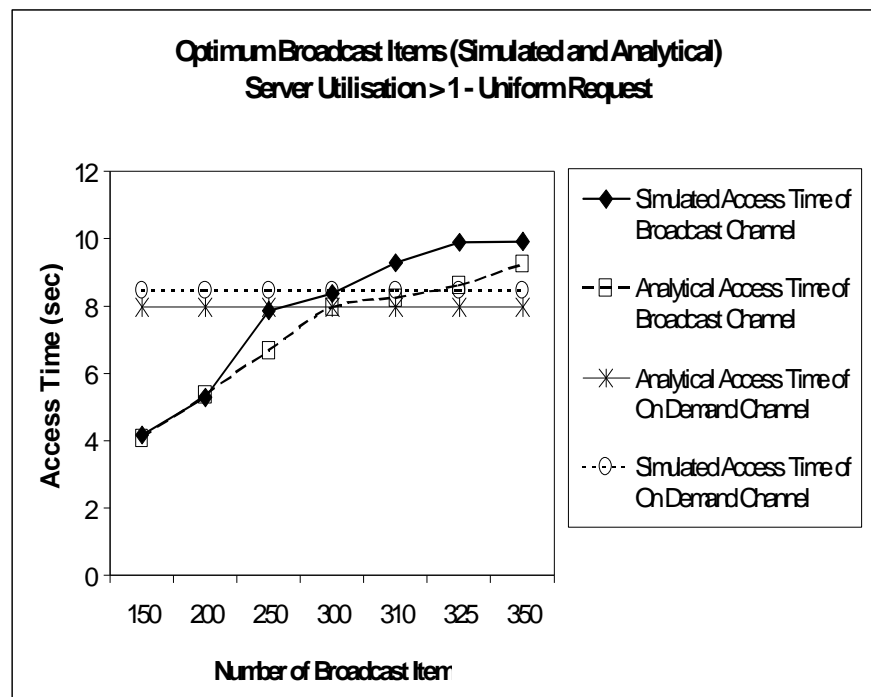


(a) Server Utilisation < 1 with Uniform Request

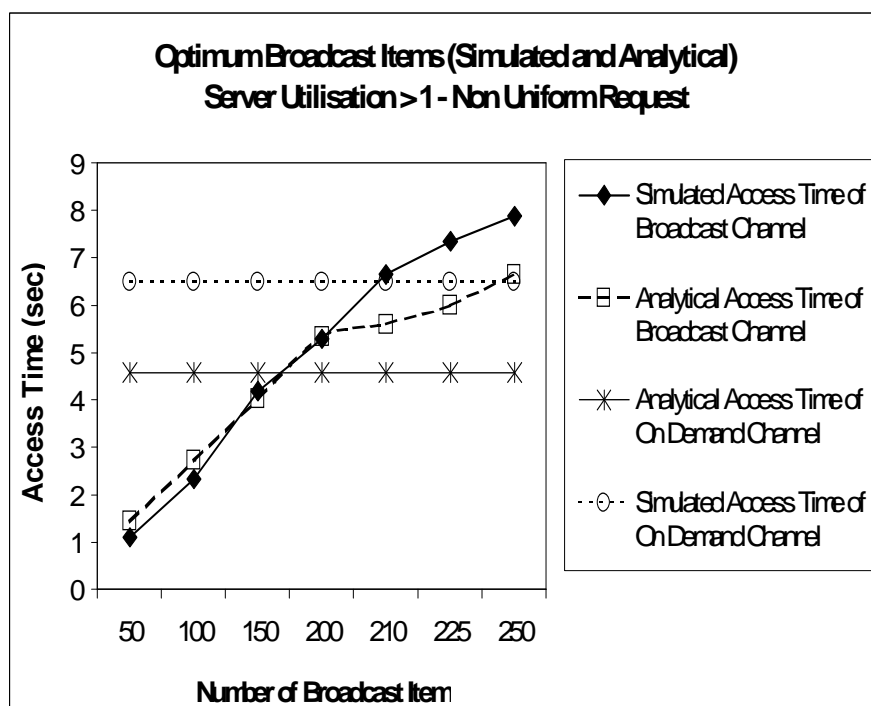


(b) Server Utilisation < 1 with Non Uniform Request

Figure 3.19. Optimum number of broadcast items with server utilisation $\rho < 1$ (uniform and non-uniform requests)



(a) Server Utilisation > 1 with Uniform Request



(b) Server Utilisation > 1 with Non Uniform Request

Figure 3.20. Optimum number of broadcast items with server utilisation $\rho > 1$ (uniform and non-uniform request)

- *Performance Analysis*

Table 3.6 shows the optimum number of broadcast items derived from the proposed analytical and simulation results for the three cases. The analytical calculation differs about 5 percent on an average from the simulation values. It mostly presents a lesser value than the simulation. Thus, the analytical models can be considered very close to the actual values. Subsequently, having known the optimum number of items in a broadcast channel, it can be decided how many channels are required to broadcast a certain amount of data items. Consequently, the average access time of broadcast channel can be optimised.

Table 3.6. Simulated and analytical performance in determining optimum number of broadcast items

Optimum Number of Broadcast Items						
		Uniform/ Non Uniform	Analytical	Simulated	Difference	Error Rate
Case 1	Pre-determined Number of Request	Uniform	391	400	9	2.3%
		Non Uniform	325	311	14	4.5%
Case 2	Server Utilization <1	Uniform	15	16	1	6.2%
		Non Uniform	12	12	0	0%
Case 3	Server Utilization > 1	Uniform	300	305	5	1.6%
		Non Uniform	170	205	35	17.1 %
Average Error Rate:						5.28%

3.5.2 Formulating Broadcast Ordering and Scheduling Schemes

The previous performance evaluation section shows the accuracy and effectiveness of the proposed analytical models to determine the optimum number of broadcast

channels. In this evaluation section, the performance of the proposed broadcast ordering and scheduling schemes (without and with replication) are studied. This evaluation section relates to chapter 3.4 of this chapter

- *Without Replication*

This section studies the performance of the proposed broadcast ordering and scheduling method without replication. The proposed scheme is analysed and compared with the conventional method. The conventional method relates to an arbitrary method when the data items are broadcast in an arbitrary sequence. To determine the optimum number of items in a channel, the proposed scheme as given in Section 3.3 of this chapter is applied.

The simulation environment is set to apply exponential distribution for data items' inter-arrival rate given an average value. The simulation is run for thirty iterations, and derives the average result accordingly. In the query profile, request that return one and more number of items are considered. A parameter, which indicates the number of dependent items in the query needs to be determined. Subsequently, both situations where the access frequencies of each query are uniform and varied, will be analysed.

In this simulation, the statistical patterns or access patterns of users probing into the channel are set to follow the behaviour of normal distribution. The parameters of concern for this study are given in Table 3.7. The results of this study are shown in Figure 3.21 and Figure 3.22.

Table 3.7. Parameters of concern – broadcast ordering and scheduling scheme without replication

Parameters	Value
Size of each data Item	2KB
Number of Data Items	30 and 45
Bandwidth	64Kbps
Query Patterns/Profiles	10
Number of Dependent Items in Query	1-4
Number of Broadcast Channel	3
Number of Request	30

It can be seen from Figure 3.21 that the proposed ordering and scheduling scheme outperforms the conventional method with an average access time that is about twice less. The access frequencies have also been varied for each query pattern, and it results in a slightly lower access time as compared to the uniform pattern frequencies.

In Figure 3.22, the number of data items to be broadcast is modified. It can be seen that the increase of broadcast items severely affects the conventional method. The data items have been increased twice as much as the number of data items in Figure 3.21, and the result for the conventional one indicates that the access time improves twice as much. In contrast, the performance results of the proposed scheme only slightly increase or can be considered as minor increases. Consequently, the gap between the proposed method and the conventional method is larger. The reason for these two cases is that since the most requested items are allocated close to each other, the increase of data items would not affect much of the overall query performance. Furthermore, with the same items requested many times, the better the performance will be.

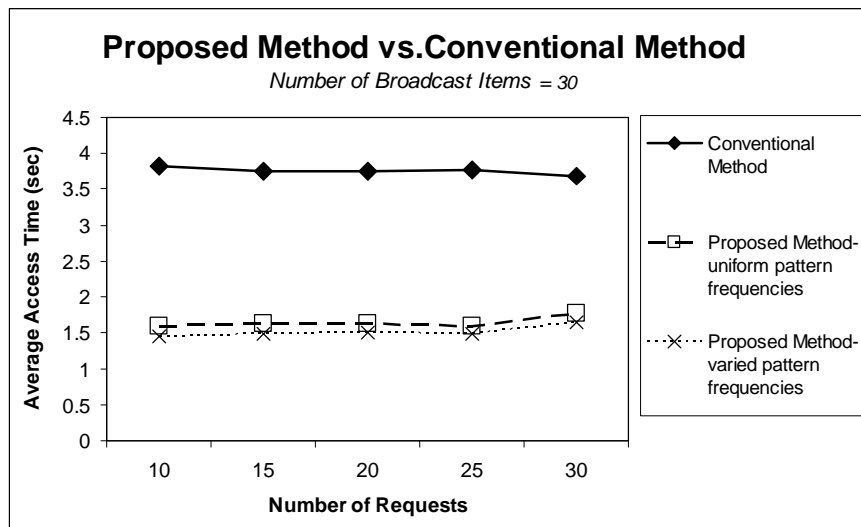


Figure 3.21. Conventional vs proposed method (30 broadcast data items)

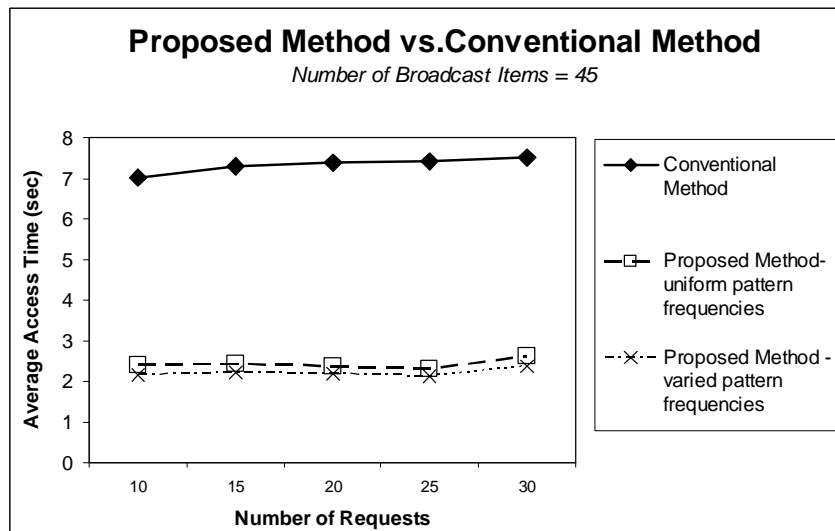


Figure 3.22. Conventional vs proposed method (45 broadcast data items)

- *With Replication*

In this section, the performance of the proposed broadcast ordering and scheduling method with replication is analysed and compared against the conventional method which is when the data items are broadcast without any specific order. The client's probe time of accessing the broadcast channel is varied. Since clients can

independently tune into the channel within a broadcast cycle, it is interesting to see how the access time performance will be affected. There are four different access distributions that need to be considered, namely: uniform, skewed to the left, skewed in the middle or normal distribution, and skewed to the right. Skewed to the left, in the middle and to the right is when majority of the clients start listening to the channel in the early broadcast cycle, middle of the broadcast cycle or at the end of the broadcast cycle, respectively. Uniform access distribution is when the number of clients accessing the channel at early, middle or end of the broadcast cycle is equally distributed.

The simulation environment is set to apply exponential distribution for data items inter-arrival rate given an average value. The simulation is run for thirty iterations, and derive the average result accordingly. The parameters of concern of this study are given in Table 3.8. The results are shown in Figures 3.23 to 3.26.

Table 3.8. Parameters of concern – broadcast ordering and scheduling scheme with replication

Parameters	Value
Size of each data Item	2KB
Number of Data Items	20 and 40
Bandwidth	64Kbps
Number of Query Patterns/Profiles	10
Number of Dependent Items in Query	1-4
Number of Requests	5-30

In Figure 3.23, the proposed scheme outperforms the conventional method with about half the average access time. Figure 3.24 depicts the access time performance while the number of broadcast items is 40. It is shown that the result follows the trend in Figure 3.23 with a slight increase in the access time in both schemes.

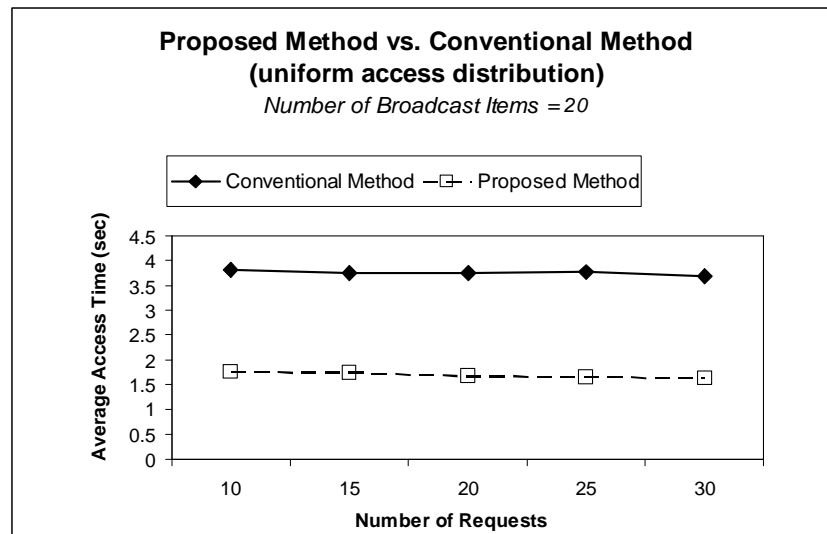


Figure 3.23. Conventional vs proposed method (20 broadcast data items)

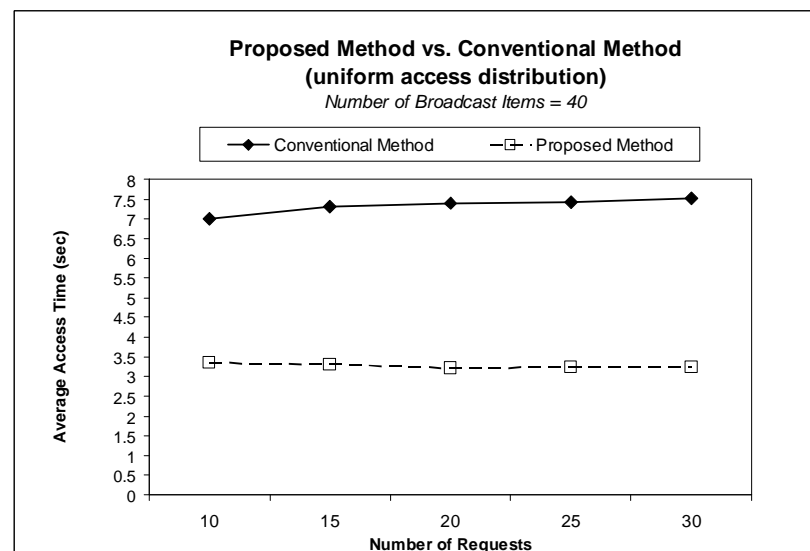


Figure 3.24. Conventional vs proposed method (40 broadcast data items)

Figures 3.25 and 3.26 show the access time performance of the proposed method with uniform and skewed access distribution. The results suggest that when the majority of clients start probing into the channel at the early cycle, the average access time is the lowest of all. On the other hand, when the majority of clients start listening into the channel in the middle of the broadcast cycle, the average access time increased, and is the highest among the other distributions.

The reason can be explained as follows: the proposed method organises the data items in non-ascending order based on the access frequency. The data items with the highest access frequency are broadcast more frequently than the others and are located in early broadcast cycle. Therefore, when the access distribution is skewed to the left, the average access time is the lowest one. However, when the access distribution is skewed in the middle, the majority of requests may need to wait for approximately a half of the cycle before the next cycle arrives, which has the most requested data items located at the front. The same reason applies to the case where the access distribution is skewed to the right. This causes the majority requests to wait for about a quarter of the cycle to obtain their desired data.

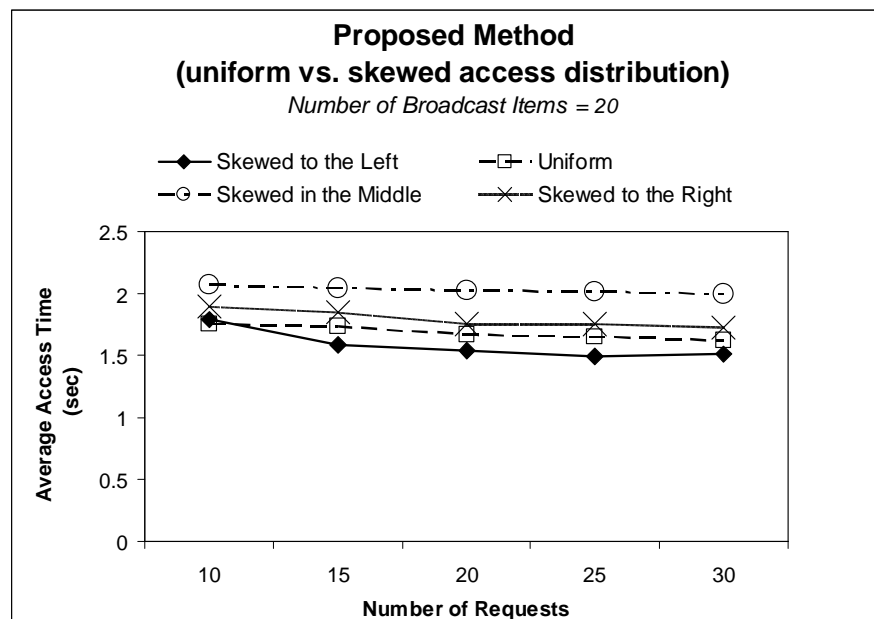


Figure 3.25. Proposed method (uniform vs skewed access distribution): 20 broadcast data items

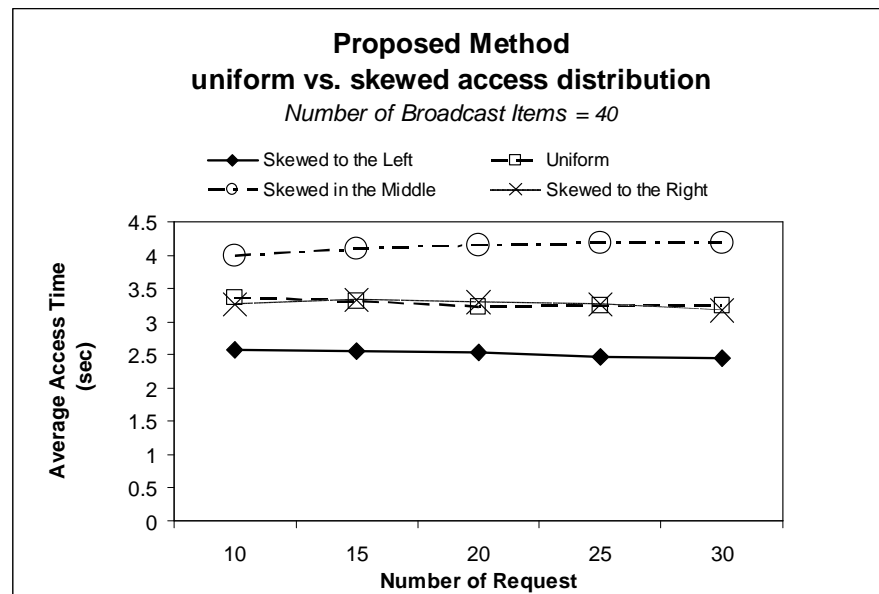


Figure 3.26. Proposed method (uniform vs skewed access distribution): 40 broadcast data items

3.6 Discussion

This chapter presented data broadcast management schemes to optimise and minimise the query access time of mobile clients when retrieving broadcast database items. The proposed method is divided into two stages: (i) to develop analytical models and utilise the analytical models to determine the optimum number of data items in a channel, (ii) to investigate broadcast scheduling and allocation schemes that concern with access patterns of mobile client accessing the broadcast data items. These schemes are further classified into two categories, namely a broadcast program *with replication* and *without replication*.

Extensive simulation experiments have been carried out to observe the performance of the analytical model in determining the optimum broadcast items. It is found that the analytical results are considerably very close to the simulation ones. Following the verification of the analytical model, the performance of the proposed broadcast

scheduling and allocation schemes as compared to the conventional method, is investigated. Conventional strategy is widely used as a broadcasting scheme when the data items are broadcast in an arbitrary sequence. It is learnt that the proposed scheme provides a substantially lower access time. In the case of the broadcast program with replication, the access distributions of clients accessing the broadcast program is evaluated. There are four different access distributions that are considered, namely: uniform, skewed to the left, skewed in the middle or normal distribution, and skewed to the right. The result suggests that when the majority of clients start probing into the channel at the early cycle, the average access time is the lowest of all. On the other hand, when the majority of clients started listening into the channel in the middle of broadcast cycle, the average access time increased, and is the highest among the other distributions.

3.7 Conclusion

Data broadcasting is an effective way to keep up with the number of clients in a cell and their frequency of requests. Mobile clients are able to retrieve the required data without sending any request to the server, which result in power conservatoin. However, with the increased number of items, the performance of the broadcast scheme will deteriorate since clients have to wait for a substantial amount of time before receiving their desired data items.

To maintain the performance of the broadcast strategy over a large set of broadcast items, an optimal broadcast channel to determine the optimum number of database items to be broadcast in a channel is introduced. The optimum number is used as an

indication of when the broadcast cycle needs to be split into different channel, forming multiple channels. Subsequently, broadcast ordering and scheduling schemes are proposed in order to further reduce the query access time of clients obtaining desired data items over the broadcast channel. These schemes are classified into two categories, namely: a broadcast program with replication and one without replication.

The main contributions of this chapter are summarized as follows.

- Analytical Model of On-demand Query is presented. It takes into account the server service rate, request arrival rate, number of requests, size of request, number of data item to be retrieved by each request, uplink and downlink bandwidth.
- Analytical Model of Broadcast-based Query is presented. The analytical model of the broadcast channel is calculated based on a variety of factors like the number of data items to be broadcast, the number of data items to be retrieved by each request, size of data item, and bandwidth.
- Optimum Number of Broadcast Channel is presented. The analytical models for both the broadcast and on-demand channels are applied to locate the optimum value of broadcast items. The access time of the on-demand channel is used as a threshold point. Several factors are taken into account, which include: request arrival rate, service rate, number of request, size of data item, size of request, number of data item to retrieve, and bandwidth. The optimum number of broadcast items is changed dynamically depending on the value of these factors. A simulation model has been developed to observe the performance of the analytical models in determining the optimum broadcast

items. It is found that the proposed analytical model is considerably close to accurate as compared with the simulation results.

- Data Broadcast Ordering and Scheduling Scheme - without Replication is presented. This scheme is concerned with the flat broadcast program. It orders and schedules the data items in the broadcast channel in such a way that most clients will have a minimum query access time. Simulation experiments have been carried out to analyse the performance of this proposed scheme. It is found that the proposed scheme provides a substantially lower average query access time as compared with the conventional method.
- Data Broadcast Ordering and Scheduling Scheme - with Replication is given. This scheme is concerned with the non-flat broadcast program. This scheme considers a certain degree of data replication to broadcast the highest access frequency more often than the others. Some performance evaluations have been carried out to analyse the performance of the proposed scheme. Based on the simulation results, the proposed scheme is able to provide a substantially better average query access time as compared with the conventional method. The result also indicates that clients who begin listening into the channel at the early broadcast cycle will obtain the lowest access time.

However, it should be noted that this chapter is concerned with query access time only, and since there is no broadcast directory which lets clients know when each of

the data items is being broadcast, clients will have to listen or tune in to the channel from the time they probe into the broadcast channel until the desired data items are received. Although, this chapter attempts to minimise the query access time, clients will still have to consume a substantial amount of power when listening to unnecessary data items.

The next chapter (Chapter 4) introduces the broadcast indexing scheme which will enable clients to efficiently and effectively obtain data items of interest over the broadcast channel.

Chapter 4

Indexing Schemes for Multi-channel Data Broadcasting

4.1 Introduction

Broadcast indexing is a desirable technique to apply in a mobile broadcast environment due to its ability to provide accurate information so that a client can tune in at the appropriate time for the required data (Lee, Leong and Si, 2002). In this scheme, some form of directory is broadcast along with the data, the clients obtaining the index directory from the broadcast and using it in subsequent reads. The information also contains the exact time of the data to be broadcast. As a result, mobile clients are able to conserve energy by switching into power saving mode or doze mode and back to active mode when the data is about to be broadcast.

However, the presence of the index in the broadcast cycle will greatly affect the query access time. It was claimed that the best access time is when there is no index broadcast along with data items (Imielinski et al, 1994). However, clients will then

waste their battery power listening to unnecessary data until the desired data arrives. The following two methods are optimal in the one-dimensional space of access time and tuning time.

- **Optimal access time:** The best access time occurs when no index is involved in the data items. The broadcast cycle is minimal in this way. However, the tuning time in this case is worse.
- **Optimal tuning time:** The server broadcasts the index segment at the beginning of each broadcast cycle. Mobile clients that require data items with primary key N tune into the channel and wait until the beginning of the next cycle. Clients can switch to doze mode until the next broadcast cycle arrives to obtain the index. It then follows the index pointer to the data items with the primary key N . This scheme provides the worse access time because the client has to wait until the beginning of the next cycle. However, the tuning time is minimal.

It is definitely of interest to obtain an efficient indexing structure for broadcast databases that reduce the client's tuning time while maintaining the query access time as low as possible.

The next sections of this chapter are organized as follows. Section 4.2 provides the outline of this chapter in detail. It is then followed by the proposed indexing schemes for traditional queries in Section 4.3. Section 4.4 explains the proposed Global indexing scheme for location-dependent queries. Section 4.5 presents the performance evaluations of the two categories of index broadcasting schemes. Section 4.6 describes the architecture of the proposed broadcast system and the index

maintenances. Section 4.7 provides discussion on this chapter. Finally, section 4.8 concludes the chapter.

4.2 Preliminaries

This chapter *presents* data broadcast indexing schemes for minimising client's tuning time and power consumption and maintaining a very low overall query access time when retrieving broadcast database items in multiple-channel environments. The proposed schemes are classified into two categories based on the type of queries involved, namely: Traditional Queries, and Location-Dependent Queries.

Traditional queries are related to queries which obtain non-location dependent data or the query results do not depend on the client's current location. This is different from location-dependent queries where the query results are based on the client's location and thus the data involved is categorised as location-dependent data.

In this thesis, both types of queries are studied. Depending on the service given by the wireless information provider, this thesis provides efficient broadcast indexing schemes, which are tailored to serve location dependent data as well as non-location dependent data. Figure 4.1 shows the framework of this chapter within the broadcast-based information system.

In general there are two indexing methods, namely: the index tree method and the signature method (see Chapter 2). This thesis is concerned with the index tree structure in a multiple-channels broadcast environment. The advantages of the index-

tree structure over the signature method include less power consumption, shorter-tune in time, and random data access (Hurson and Jiao, 2005).

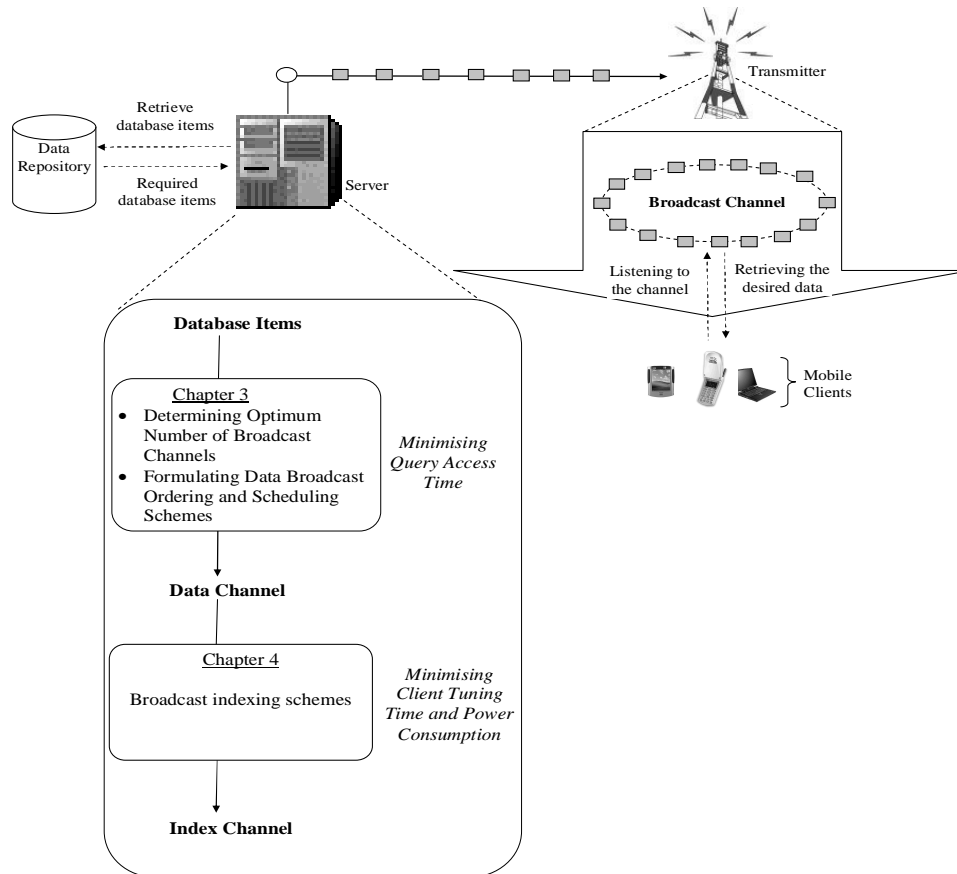


Figure 4.1. Chapter framework

- Traditional Queries

This thesis introduces three broadcast indexing schemes over multi-channel data broadcasting for traditional queries, namely: (i) non-replicated indexing scheme (NRI), (ii) partially-replicated indexing scheme (PRI), and (iii) fully-replicated indexing scheme (FRI). These three indexing schemes are designed based on the B^+ tree structure.

The B^+ tree structure has the advantage of having all data pointers stored on the leaf nodes, which is considered better than storing data pointers in the non-leaf nodes like the original B trees (Elmasri and Navathe, 2003). Consequently, the structure of the leaf nodes differs from the structure of non-leaf nodes, which makes it easier for a mobile client to interpret the nodes. Moreover, when all of the leaf nodes are at the same level, it guarantees that no nodes will be at a high level which may cause unbalanced distribution and thus a waste of bandwidth since the proposed schemes are concerned with index replication and partitioning in multiple-channel environments. When being broadcast, each physical pointer to the neighbouring leaf node, as well as an actual data item, is replaced by a time value, which indicates when the leaf node or data item will be broadcast.

The NRI scheme is designed where the entire index structure is partitioned into a number of disjoint and smaller indices. Each of these small indices is placed in a dedicated index channel. This is slightly different from PRI or what is called the Global Indexing scheme in such a way that it has some degree of replication while NRI has none. Each index channel has a different part of the entire index structure, and the overall structure of the entire index is still preserved. The advantage of PRI over NRI is that clients can tune in to any of the index channels and still be able to find the right pointer to the desired data item even though this pointer is located in a different index channel. However, when the channel is vulnerable to noise and signal distortion, the FRI scheme might be the one to consider as this is where the entire index structure is fully replicated to all available index channels. When the

interruption occurs, the client can move to the other channel and retrieve the desired data items.

- Location-Dependent Queries

In a mobile environment, the kind of information requested is generally location-dependent; that is mobile client's location is relevant to the information requested, or the information requested is based on a particular location (Lee et al, 2002). With a variety of promising applications, such as *location dependent information services* or so-called LDIS (e.g. traffic reports, news, navigation maps, and so on) and nearest-neighbour queries (e.g. finding the nearest petrol station), location-dependent information services will soon become an integral part of our daily lives.

Considering the potential of LDIS in the mobile telecommunication industry, the indexing scheme for traditional queries is modified in order to serve location-dependent queries. With regard to this issue, a novel index architecture called the Global indexing scheme for location-dependent query is introduced. The architecture of this scheme is based on the partially-replicated indexing (PRI) scheme for traditional queries. Similarly, this scheme is applied in a multi-channel wireless environment and it is designed to accommodate location-dependent queries in a most effective and efficient way.

- Traditional and Location-dependent Queries

In this thesis, the proposed broadcast indexing schemes for traditional and location-dependent queries consider the data and index segment located in different broadcast channels. This is due to the consequences of having both index and data in the same

channel, which causes a trade-off between minimizing the time listening to the channel (or also called tuning time), and the query access time. The consequence of minimizing one of them is the increase of the other (Imielinski T., Viswanathan S. and Badrinath, 1997; Chung, 2005). Thus, separate channel placement of index and data segments is designed to reduce the conflicts and to obtain an optimal method in the two-dimensional space of access time and tuning time. Figure 4.2 illustrates the situation when index segment and data segment are in separate channels. The Figure shows the access and tuning time to retrieve data item # 3, which starts from the time the client probes into the beginning of the index channel until the desired data item has been obtained. With regard to the broadcast indexing performances, the parameters of concern are *index access time*, *client's tuning time* and *power consumption*.

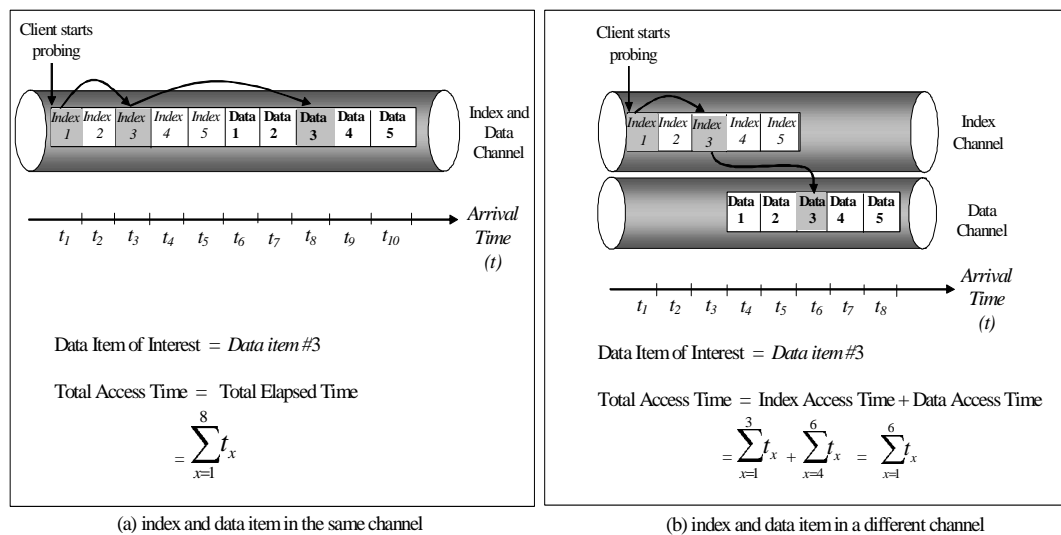


Figure 4.2. Integrated and separated index channel

4.3 Index Broadcasting for Traditional Queries

The broadcast indexing scheme is an effective way to retrieve database items over a broadcast channel efficiently. This technique is able to satisfy the requirement of data retrieval in a mobile environment, particularly in minimizing the amount of power consumption (Chehadeh, Hurson and Miller, 2000). The indexing scheme is able to reduce the tuning time by providing accurate information for a client to tune in for the required data items.

In this scheme, some form of directory is broadcast along with the data, the clients obtaining the index directory from the broadcast and using it in subsequent reads. The information generally also contains the exact time of the data to be broadcast. Figure 4.3 illustrates a tree index used to accommodate 54 data records. The tree index starts from the root followed by three level indexes. The first level of the index consists of three data buckets (a_1, a_2, a_3) and each bucket has three pointers to the data buckets in the next level. Subsequently, there are nine data buckets ($b_1, b_2, b_3, b_4, b_5, b_6, b_7, b_8, b_9$) in the second level and each of them has three pointers to the last level. The box in the bottom level represents the final data bucket and each of the boxes holds 2 data records. The leaf node level also contains a pointer which indicates the time when the data arrives in the channel which enables mobile clients to switch to a power saving mode and back to the active mode when the data is about to arrive. This mechanism helps clients to conserve power on their mobile devices.

Figure 4.4 depicts the application of the tree-indexing scheme in multiple-channel environments when the index and data segment are located in a separate channel. The

Figure illustrates the query processing mechanism from the time a client starts probing into the index channel until the desired data item #6 is obtained from the data channel. It can be seen from the Figure that clients will be able to conserve power consumption as they wait for the relevant data item to arrive in the channel. It is desirable to obtain efficient index broadcasting scheme which minimise the index access time, client's tuning time and power consumption which is the focus of this chapter.

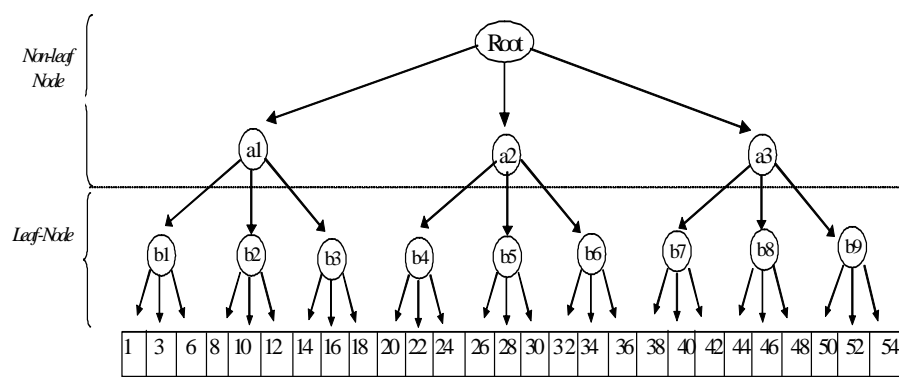
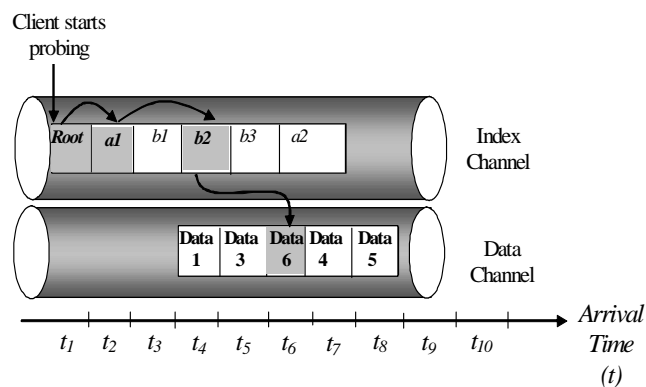


Figure 4.3. A sample tree index



Data Item of Interest = Data item #6

$$\text{Index Access Time} = \sum_{x=1}^4 t_x$$

Tuning Time = Root + a1 + b2 + Data bucket 6

Figure 4.4. Tree index in a broadcast channel environment

Considering the advantages of the tree-indexing mechanism, three indexing schemes are proposed, namely:

- i. *Non-Replicated Index (NRI)*.
- ii. *Partially-Replicated Index (PRI)*.
- iii. *Fully-Replicated Index (FRI)*.

For each scheme, the number of channels required to broadcast a certain amount of data items is known. The entire index structure is designed based on the B^+ tree structure. It consists of non-leaf nodes, and leaf nodes. A leaf node is the bottom-most index that consists of up to k keys, where each key points to actual data items, and each node has one node pointing to a right-side neighbouring leaf node. Unlike a leaf node, a non-leaf node may consist of up to k keys and $k+1$ pointers to the nodes on the next level on the tree hierarchy (i.e. child nodes). All child nodes, which are on the left-hand side of the parent node, have key values less than, or equal to, the key of their parent node. On the other hand, keys of child nodes on the right-hand side of the parent node are greater than the key of their parent node. When being broadcast, each physical pointer to the neighbouring leaf node, as well as the actual data item, is replaced by a time value, which indicates when the leaf node or data item will be broadcast.

The B^+ tree structure is more desirable than the original B trees since it has the advantage of having all data pointers stored on the leaf nodes (Elmasri and Navathe, 2003). This is due to the structure of the leaf nodes differing from the structure of non-leaf nodes, which makes it easier for a mobile client to interpret the nodes. Moreover, when all leaf nodes are at the same level, it guarantees that no nodes will

be at a high level which may cause an unbalanced distribution and thus a waste of bandwidth since the three indexing schemes are concerned with index replication and partitioning in multiple-channel environments.

A simple scenario is to broadcast major stock indices for countries around the world. There are 30 stock indices altogether to be broadcasted, and it is assumed the optimum number of indices in a data channel is 10 (Refer to Chapter 3 of this thesis for a discussion on determining the optimum number of data items in a channel). Subsequently, there are 3 data channels, each channel containing 10 stock indices. In this case, a table consisting of records of IDs, country, stock name, and stock value is constructed. The stock values are taken from (<http://finance.yahoo.com/>). The index is constructed based on the order of the data item in the data channels. The data items in the data channels are placed based on the proposed broadcast data ordering and scheduling scheme (see Chapter 3.4), and once it reaches the optimum number of indices, a new data channel is created. Figure 4.5 depicts the stages to follow for the index construction and restructuring process. The data channel and index structure are illustrated in Figure 4.6. For the sake of simplicity, in this example the data ordering of the table is designed to be the same as that of the data channels.

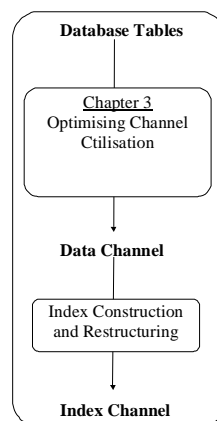
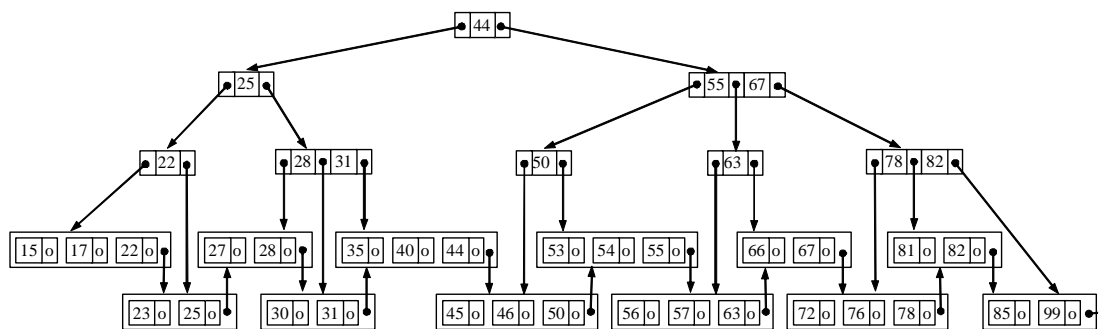


Figure 4.5. Index construction and restructuring Stages

Table (ID, Country, Stock Name, Stock Value):

30	Argentina	MerVal	903.680	25	Indonesia	Jakarta Composite	722.709	46	Taiwan	Taiwan Weighted	5,777.32
72	Brazil	Bovespa	18,796.91	28	Japan	Nikkei 225	10,849.63	50	Austria	ATX	1,798.41
44	Canada	S&P TSX Composite	8,191.61	17	Malaysia	KLSE Composite	793.97	54	Belgium	BEL-20	2,391.09
67	Chile	IPSA	1,412.79	81	New Zealand	NZSE 50	2,564.730	57	Czech Republic	PX50	750.80
53	Mexico	IPC	9,821.000	85	Pakistan	Karachi 100	5,454.69	76	Denmark	KFX	252.53
99	United States	S&P 500	1,098.70	22	Philippines	PSE Composite	1,521.61	82	France	CAC 40	3,603.26
55	Australia	All Ordinaries	3,366.800	23	Singapore	Straits Times	1,754.96	15	Germany	DAX	3,803.10
78	China	Shanghai Composite	1,562.688	27	South Korea	Seoul Composite	768.46	56	Italy	MIBTel	20,513.000
63	Hong Kong	Hang Seng	11,276.86	31	Sri Lanka	All Share	1,321.19	40	Netherlands	AEX General	327.39
66	India	BSE 30	5,069.87	35	Thailand	SET	609.72	45	Norway	OSE All Share	327.39

Index (B+ Tree):



Data Channel 1				Data Channel 2				Data Channel 3			
30	Argentina	MerVal	903.680	25	Indonesia	Jakarta Composite	722.709	46	Taiwan	Taiwan Weighted	5,777.32
72	Brazil	Bovespa	18,796.91	28	Japan	Nikkei 225	10,849.63	50	Austria	ATX	1,798.41
44	Canada	S&P TSX Composite	8,191.61	17	Malaysia	KLSE Composite	793.97	54	Belgium	BEL-20	2,391.09
67	Chile	IPSA	1,412.79	81	New Zealand	NZSE 50	2,564.730	57	Czech Republic	PX50	750.80
53	Mexico	IPC	9,821.000	85	Pakistan	Karachi 100	5,454.69	76	Denmark	KFX	252.53
99	United States	S&P 500	1,098.70	22	Philippines	PSE Composite	1,521.61	82	France	CAC 40	3,603.26
55	Australia	All Ordinaries	3,366.800	23	Singapore	Straits Times	1,754.96	15	Germany	DAX	3,803.10
78	China	Shanghai Composite	1,562.688	27	South Korea	Seoul Composite	768.46	56	Italy	MIBTel	20,513.000
63	Hong Kong	Hang Seng	11,276.86	31	Sri Lanka	All Share	1,321.19	40	Netherlands	AEX General	327.39
66	India	BSE 30	5,069.87	35	Thailand	SET	609.72	45	Norway	OSE All Share	327.39

Figure 4.6. Data channel structure and index using B+ tree

Assume that in the index tree, the maximum number of node pointers from any non-leaf node is 4, and the maximum number of data pointers from any leaf node is 3. The number of index channels used for index-tree partitioning is the same number as the data channels. In this example, there are 3 data channels, a similar number to the index channels. The index-tree structure is partitioned into 3 index channels based on the ID attribute. The ID is used as the index-partitioning attribute. With 3 index

channels, the index-partitioning attribute will be the highest ID over 3 index channels. The result is index channel 1 holds data IDs between 1 to 37, index channel 2 holds data IDs between 38 to 67, and the rest go to index channel 3. It must be stressed that the location of each leaf node is the not same as where the actual data is broadcasted. Having known the index-partitioning attribute, it is then necessary to apply a traversal algorithm to assign the index nodes into the index channels.

4.3.1 Non-replicated indexing (NRI) scheme

A *Non-Replicated Indexing* (NRI) scheme is where the entire index structure is partitioned into a number of disjoint and smaller indices. Each of these small indices is placed in a separate index channel. A Post Order Traversal algorithm needs to be applied to allocate the index nodes into designated index channel. The Postorder Traversal for the NRI scheme is depicted in Figure 4.7, whilst, the algorithm is shown in Figure 4.8.

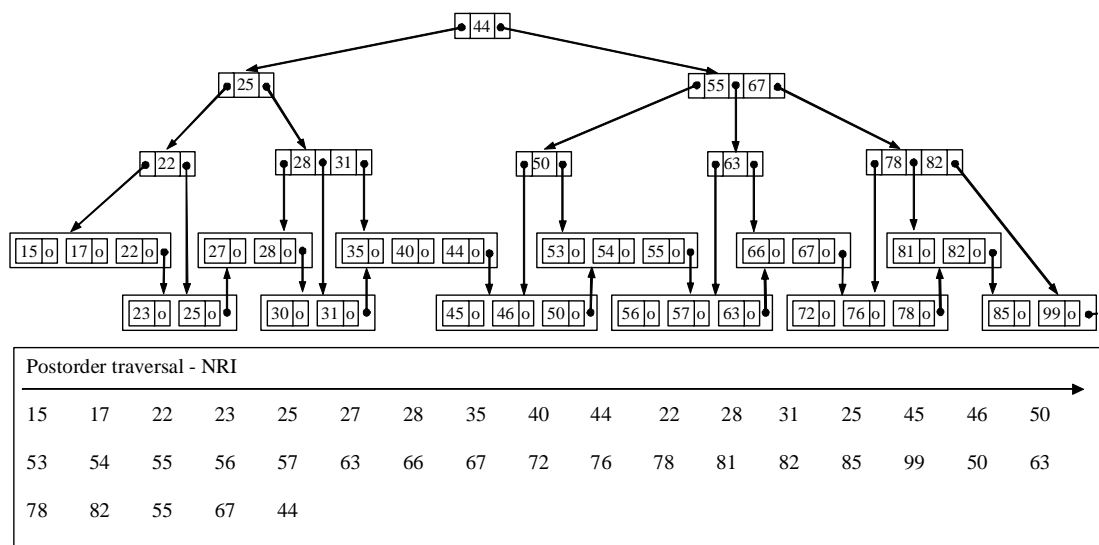


Figure 4.7. Post Order Traversal - NRI


```

Procedure TraverseNRI(T:tree) Is
  Begin
    If T = null then
      Return;
    Else
      TraverseNRI(T.left);
      TraverseNRI(T.right);
      VisitTree(T);
    End if;
  End Traverse NRI;

Procedure VisitTree(T:tree) is
  Begin
    Check the index node with index-partitioning attribute;
    Assign index node into designated index channel;
  End VisitTree;

```

Figure 4.8. Post Order Traversal algorithm with Index Allocation Scheme (NRI)

Figure 4.9 shows the composition of each index channel with its local index starting from the initial index tree. By utilizing this indexing scheme, the mobile client needs to know in advance the index channel that contains the desired index key.

The data retrieval mechanism in this scheme can be described as follows:

- The mobile client tunes in to one of the index channels (which must be the index channel that has the desired index key). Assuming that each mobile client is only able to listen to a single channel at any time, it can either broadcast some sort of index channel directory prior to each index cycle, or the mobile client can send a request to the server to supply the index channel directory.
- The mobile client tunes in to the index channel that contains the right key to the desired data and follows the index pointer. While waiting for the index to arrive, the mobile client can switch to power saving mode.

- The mobile client tunes back in at the index channel that has the right index key, which points to the data channel that contains the desired data item. It indicates a time value of the data to arrive in the data channel.
- The mobile client tunes in to the relevant data channel, and switches back to power saving mode while waiting for the data item to arrive.
- The mobile client switches back to active mode just before the desired data item arrives, and retrieves the information.

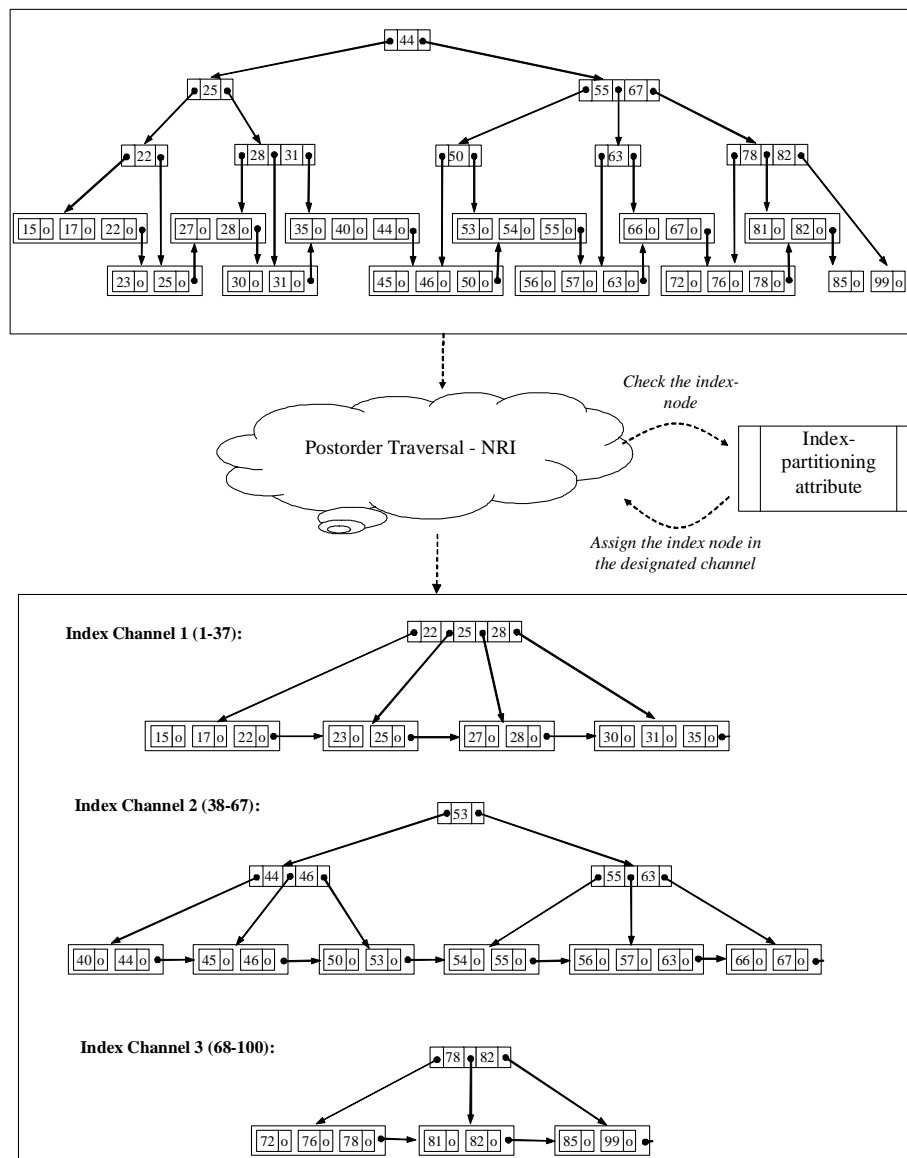


Figure 4.9. Non-replicated indexing scheme

4.3.2 Partially-replicated indexing (PRI) scheme

The NRI scheme requires mobile client to know in advance the index channel that contains the desired index key. Since the client can listen to only one channel at a time, this requirement can be realised by either broadcasting some sort of index channel directory prior to each index cycle, or the mobile client send a request to the server to supply the index channel directory. This drawback can be overcome with the *Partially-replicated indexing* (PRI) or what is called the *Global Indexing scheme*. This scheme is different from the NRI in that it has some degree of replication, while NRI has none. Each index channel occupies a different part of the entire index structure, and the overall structure of the entire index is still preserved. The advantage of PRI over NRI is that clients can tune in to any of the index channels and still be able to find the right pointer to the desired data item even though this pointer is located in a different index channel.

In the PRI scheme, the ownership rule of each index node is that the index channel owning a leaf node also owns all nodes from the root to that leaf node. Consequently, the root node is replicated in all index channels, and non-leaf nodes may be replicated in some channels. Additionally, if a leaf node has several keys belonging to different index channels, this leaf node is also replicated in the channels owning the keys.

Similar to NRI scheme, a Post Order traversal algorithm needs to be applied in order to allocate the index nodes to the designated index channel. The Postorder traversal for PRI scheme is depicted in Figure 4.10 and the corresponding algorithm is shown in Figure 4.11. The main difference between the NRI and PRI Post Order traversal

algorithm is that the PRI algorithm traverses the index bucket rather than the index nodes. The algorithm analyses each index node within each bucket prior to allocating the index bucket to relevant index channel based on the index-partitioning attributes. Figure 4.12 shows the PRI construction and restructuring process starting from the initial index tree. Notice from Figure 4.13 that the fifth leaf node (35, 40, 44) of PRI scheme is replicated in channel 1 and 2 because key 35 belongs to index channel 1, while keys 40 and 44 belong to index channel 2.

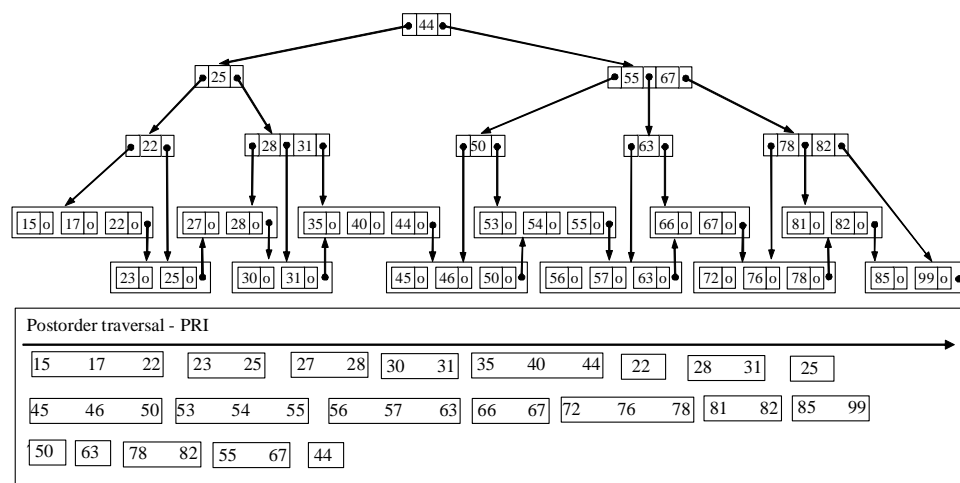


Figure 4.10. Post Order Traversal - PRI

Procedure TraversePRI(T:tree) Is

Begin

If T = null then

Return;

Else

TraversePRI(T.left);

TraversePRI(T.right);

VisitTree(T);

End if;

End TraversePRI;

Procedure VisitTree(T:tree) Is

Begin

If the index bucket is the root of index tree-structure then

Assign the index bucket across entire index channels;

Else

Check the index bucket with index-partitioning attribute;

If any of the index nodes in the bucket satisfy the index-partitioning attribute then

Assign the index bucket into relevant index channel;

End if;

End if;

End VisitTree;

Figure 4.11. Post Order Traversal algorithm with index allocation scheme (PRI)

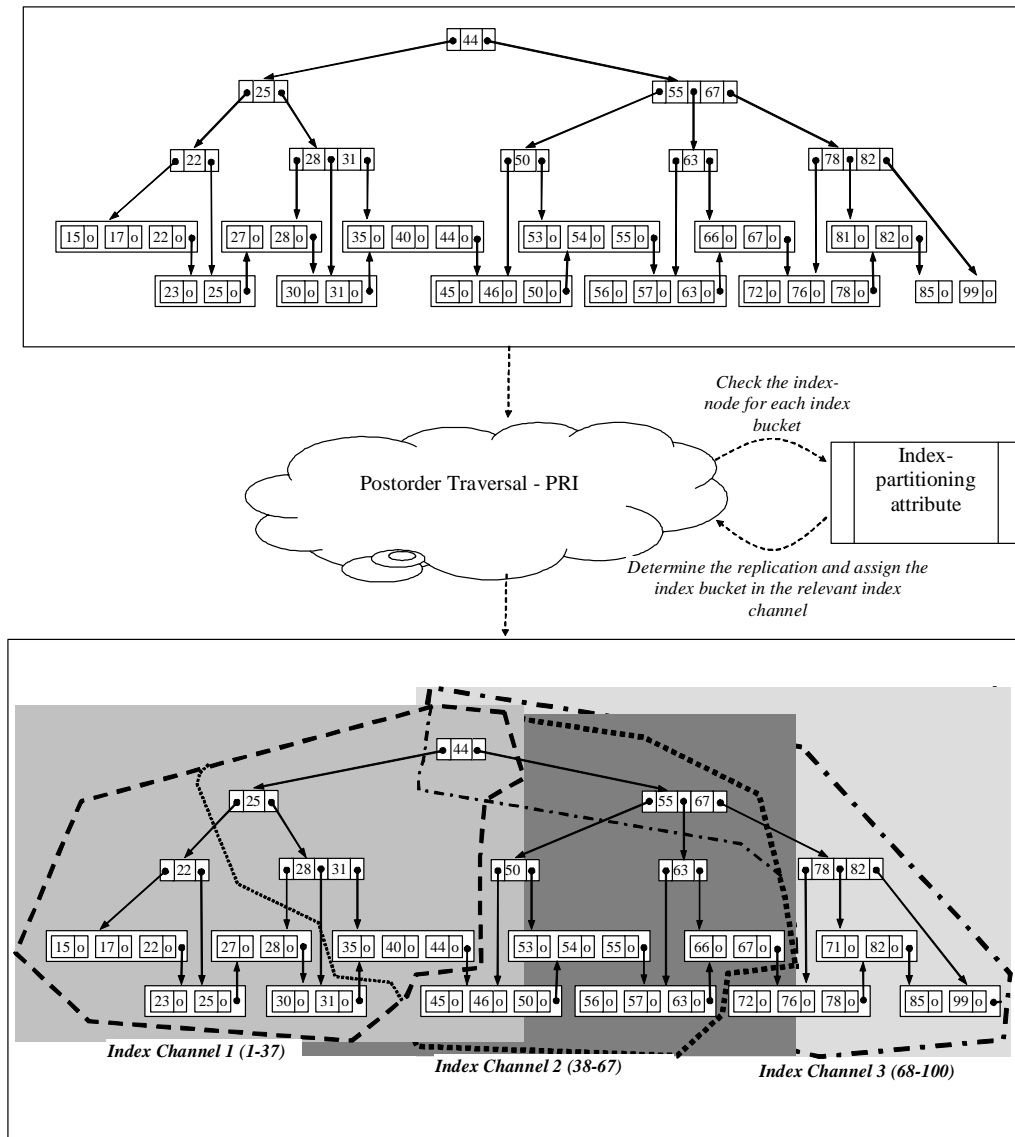


Figure 4.12. Construction and restructuring of Partially-Replicated Indexing scheme

Also notice that some non-leaf nodes are replicated, whereas others are not. For example, the non-leaf node 22 is not replicated and located only in index channel 1, whereas non-leaf node 25 is replicated to index channel 1 and 2. It is also clear that the root node is fully replicated. As mentioned earlier, the index is broadcast separately with the data, and each index key points to the relevant data channel.

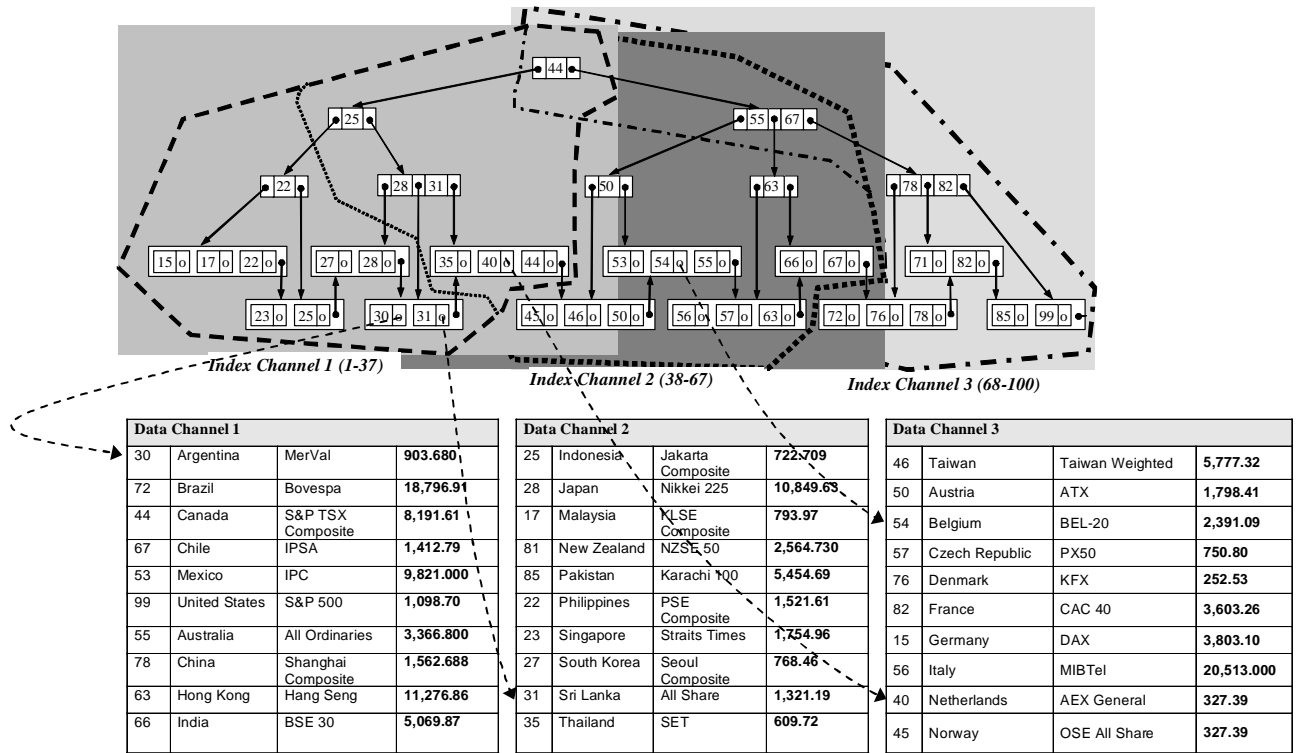


Figure 4.13. Partially-replicated indexing scheme

In this scheme, once the right index is found in a specific index channel, the mobile client switches to the right data channel and waits for the data of interest to arrive. The data structure for the PRI index can be described as follows: *If a child node exists locally*, the node pointer points to this local node only, even when this child node is also replicated in other index channels. For example, from node 44 at index channel 1, there is only one node pointer to the local node 25. The child node 25 at index channel 2 will not receive an incoming node pointer from the root node 44 at index channel 1; instead it will receive one node pointer from the local root node 44 only.

If a child node does not exist locally, the node pointer will choose one node pointer pointing to the nearest child node (in case if multiple child nodes exist somewhere

else). For example, from the root node 44 at index channel 1, there is only one outgoing right node pointer to child node (55,67) at index channel 2. In this case, it is assumed that index channel 2 is the nearest neighbour of index channel 3. The child node (55,67), which also exists at index channel 3, will not receive a node pointer from root node 44 at index channel 1.

Using this single node pointer model, it is always possible to trace a node from any parent node. For example, it is possible to trace to node (78,82) from the root node 44 at index channel 1, although there is no direct link from root node 44 at index channel 1 to its direct child node (55,67) at index channel 3. Tracing to node (78,82) can still be done through node (55,67) at index channel 2.

A more formal definition for the single node pointer model is as follows. First, given a parent node is replicated when its child nodes are scattered at multiple locations, there is always a direct link from whichever copy of this parent node to any of its child nodes. Second, using the same methodology as the first statement above, given a replicated grandparent node, there is always a direct link from whichever copy of this grandparent node to any of the parent nodes. Considering the first and the second statements above, it can be concluded that there is always a direct link from whichever copy of the grandparent node to any of its child nodes.

Data retrieval mechanism in this scheme can be described as follows:

- The mobile client tunes in to one of the index channels (i.e. can be of any index channel).

- The mobile client follows the index pointer to the right index key. The pointer may lead to another index channel that contains the relevant index. While waiting for the index to arrive, mobile clients can switch to power saving mode.
- The mobile client tunes back in to the index channel that has the right index key, which points to the data channel that contains the desired data item. It indicates a time value when the data will arrive in the data channel.
- The mobile client tunes in to the relevant data channel, and switches back to power saving mode while waiting for the data item to arrive.
- The mobile client switches back to active mode just before the desired data item arrives, and retrieves the information.

4.3.3 Fully-replicated indexing (FRI) scheme

In a wireless environment, the channel is generally susceptible to noise and signal distortion. Consequently, the Fully-replication indexing (FRI) scheme is the one to be considered as this is where the entire index structure is fully replicated to all available index channels. This scheme provides the solution to problems like data distortion, noise, and signal distortion as the mobile client can switch to another index channel to avoid any of these problems while still being able to retrieve the desired data items.

The index nodes in each channel may be broadcast in different interval. The data retrieval mechanism in this scheme is similar to PRI. The difference is that this scheme does not have a pointer to the other index channels.

4.4 Index Broadcasting for Location-dependent Queries

As mobility is one of the unique characteristic of a wireless environment, the kind of information requested is generally location-dependent; that is the mobile client's location is relevant to the information requested, or the information requested is based on a particular location (Lee et al, 2002). This request is generally known as a location-dependent query, and such services can be called *Location-dependent Broadcast Services*. The PRI indexing scheme for answering traditional queries is modified to serve location-dependent queries in a mobile broadcast environment.

4.4.1 Preliminaries

Although location-dependent query services exist in traditional computing environments (e.g. Guides@Yahoo), their greatest potential is in a mobile-pervasive computing environment, where users enjoy unrestricted mobility and ubiquitous information access (Lee et al, 2002). In location-dependent query, the location of mobile object becomes a parameter of the query. The value of the location parameter can be explicit or implicit. A query example of an explicit location parameter is "List all five-star hotels in Melbourne". In this query, the location is a component of the query. Thus, the movement of the user does not affect the query result. Another case is when the location parameter is implicit such as "List the nearest hotels or restaurants". The proposed broadcast indexing scheme focuses on the issues when the location parameter is implicitly determined.

The most well-known and commercially used satellite positioning system nowadays is the global positioning system (GPS). The GPS provides location identifier in the form

of coordinate tuples (ie. longitude and latitude). Satellite technologies have been widely used to locate the position of moving objects. It should be noted that the proposed scheme for location-dependent queries assumes the utilization of the satellite-based positioning system.

Location-dependent queries require location dependent data. Location dependent data can be categorized into static data or dynamic data. Static data relates to the stationary object, where the object location is not continuously updated in the database (e.g. restaurant, hotels). On the other hand, dynamic location data correspond to a moving object that requires constant update of the object's location in the database (e.g. taxis, ambulance, or even the mobile users themselves). The proposed indexing scheme is concerned with the static location-dependent data and there is only a single context involved in the data.

4.4.2 Global Index for Location-Dependent Queries

This section presents a novel broadcast indexing structure for location-dependent queries. The fundamental architecture of the proposed Global Index for Location-Dependent Queries is modified and derived from the *Partially-replicated indexing (PRI)* or the *Global Indexing scheme* for Traditional Queries which have been described in Section 4.2.2 of this chapter.

The proposed Global Index for Location-Dependent Queries applies the concept of valid scope (Xu et al, 2003). Valid scope defines the boundary of an area or region within which the query result is considered valid. In this thesis, a square shape is selected as the valid scope. A square is considered to be easier to construct since it

has same length and width. Furthermore, it requires only four points to represent the scope. This is desirable since the less processing required, the less power is required in order to analyse the index nodes. For simplicity, a geometric location is represented as a two dimensional coordinate (Jayaputera and Taniar, 2005).

Consequently, the information sent to represent the valid scope boundary will be small; thus mobile clients will have a more efficient query processing. The valid scope will be broadcast together with the data index. When the client initiates a query, the validity of the broadcast data will be checked by comparing the valid scope of the data instances with the client's current location. An example of the square valid scope is shown in Figure 4.14(a). In the Figure, there are four regions: region 1 (P1,L1), region 2 (P1,L2), region 3 (P2,L1) and region 4 (P2,L2). Each of the regions has a valid data instance that is D1, D2, D3, and D4, which is attached to regions 1, 2, 3, and 4 respectively.

Having a set of data instances and their relevant valid scopes, the issue of querying location dependent-data is how to obtain the right data instance efficiently. The proposed index is constructed based on the divisions of data regions. The space containing a set of data regions is partitioned until each sub-space covers *one* region only. The partition of two sub-spaces is represented by two straight lines forming a square (x-coordinate dimensional and y-coordinate dimensional). Figure 4.14(a) illustrates the partition.

Figure 4.14(b) depicts the index construction based on the space partition in Figure 4.13(a). It is tree-indexing based. The index node and its description are given in Figure 4.13(c) and Figure 4.13(d), respectively.

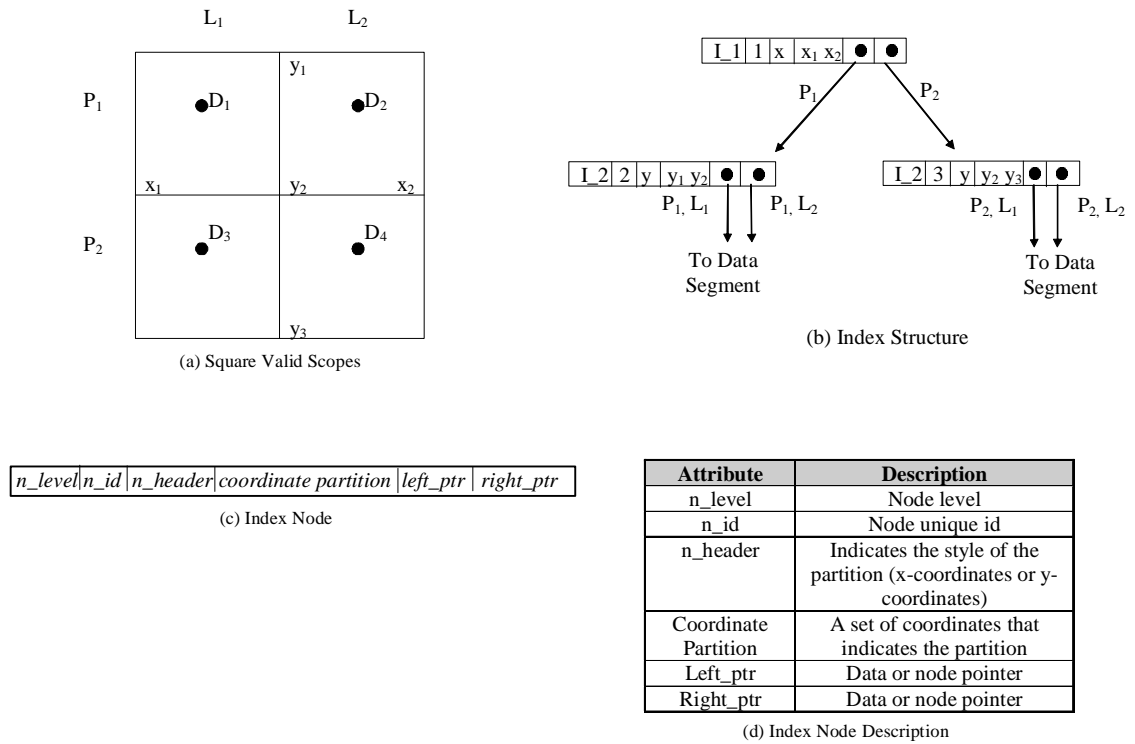


Figure 4.14. Index construction

The index node contains a header which indicates the partition style, whether it is *x*-dimensional partition or *y*-dimensional partition. The partition always proceeds from the top regions to the bottom regions and begins with an *x*-dimensional partition. Following the header is a set of coordinates for the partition itself and two pointers (left and right pointer). The left and right pointers in the *y*-dimensional partition indicate the upper and lower spaces respectively. Whilst, the left and right pointers in the *x*-dimensional partition divide the left and right space correspondingly. The pointers of the non-leaf node point to the relevant child nodes, while the leaf-node pointer goes to the data segment that contains the valid data instance in the particular region. These pointers contain the time when the node or the data instance will arrive in the channel. In this indexing scheme, each node has exactly two subsequent children.

Depending on the regions involved, when the final partition of the space forms a rectangular shape as shown in Figure 4.15, a different index construction can be obtained. In this example, the rectangle has more rows than columns; it comprises 6 squares and the index structure is then built accordingly. Figure 4.15 also presents how the index-tree structure will appear.

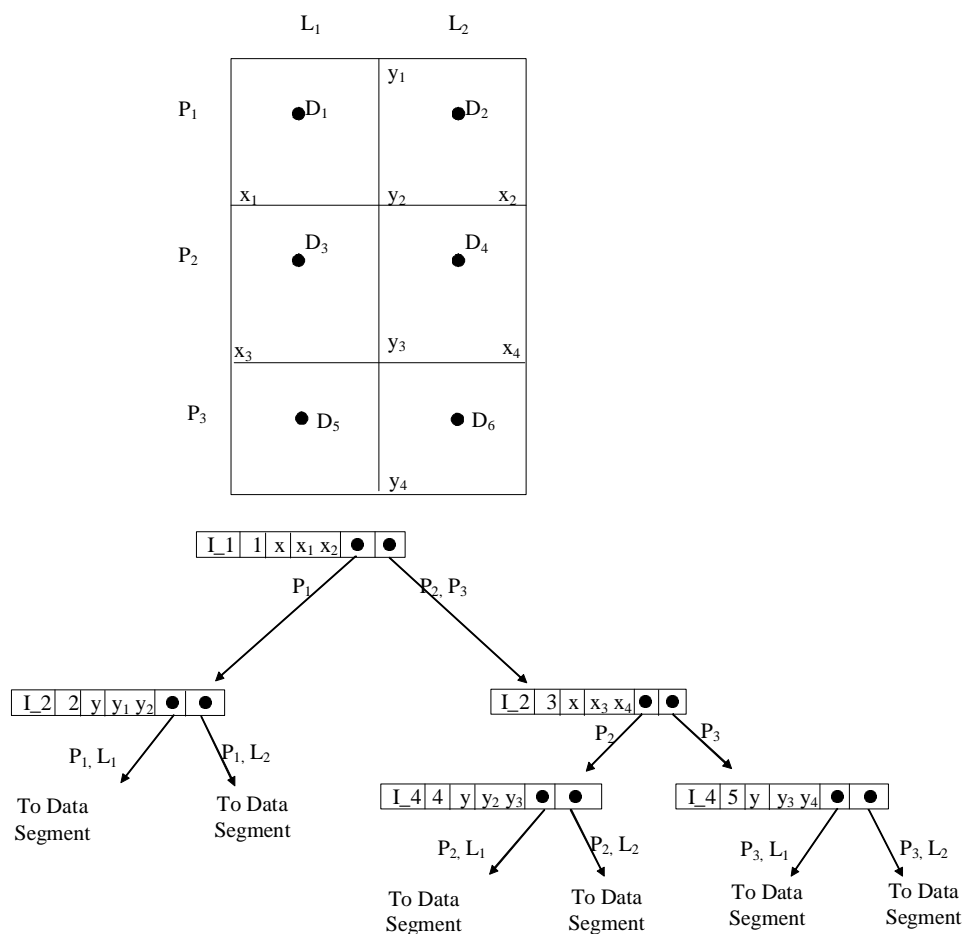


Figure 4.15. Unbalanced Index Model – Row

On the other hand when the rectangle has more columns than rows as given in Figure 4.16, the index will have an entirely different structure from the previous one. In this case, the index-tree height will be unbalanced as the x -dimensional partition includes an odd number of columns.

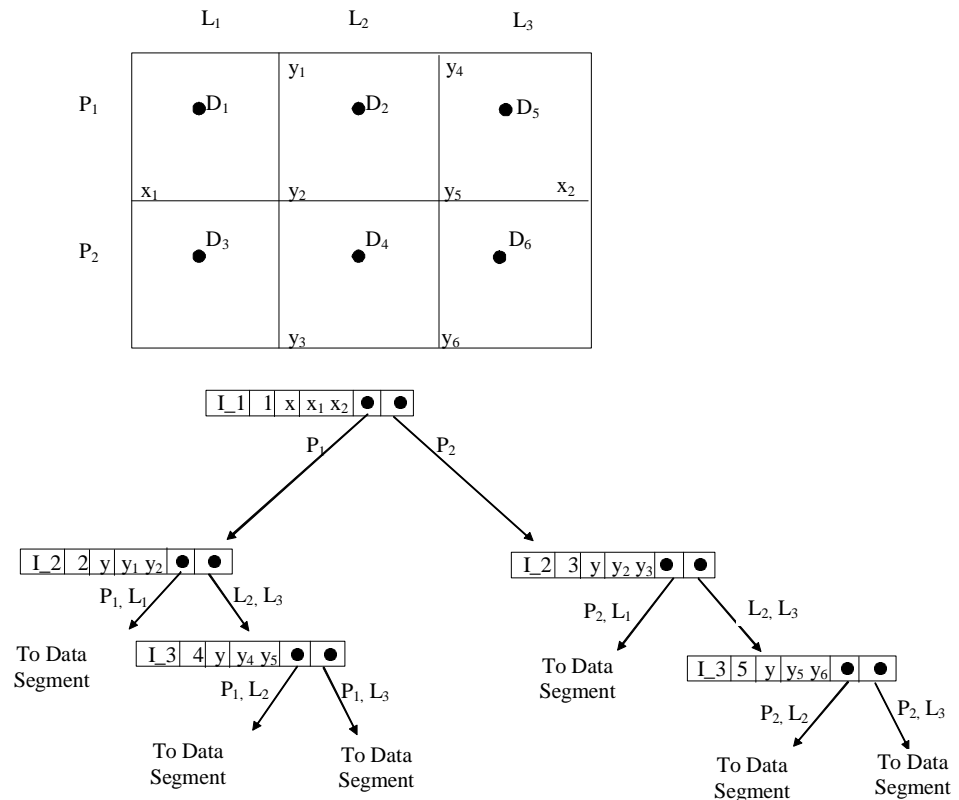


Figure 4.16. Unbalanced Index Model - Column

The process for querying location-dependent broadcast data can be described as follows:

- The client tunes in to the index channel
- The client searches the index by following a sequence of pointers. The right index is determined by locating the data region that contains the query point. The index indicates a time value for the data instance to arrive in the data channel.
- The client tunes in to the relevant data channel, and switches back to doze mode while waiting for the data item to arrive.
- The client switches back to active mode just before the desired data item arrives, and retrieves the information.

The proposed *global indexing* scheme is where the entire index structure is partitioned into a number of disjoint and smaller indices. Each of these small indices is placed in a separate index channel. The global indexing scheme has some degree of replication. Each index channel occupies a different part of the entire index structure, and the overall structure of the entire index is still preserved.

However, in this case the index structure is partitioned and broadcast to multiple index channels. Each index channel is dedicated to broadcast nodes of a specific region only. In this example, index channel 1 contains the index nodes for region P_1 , index channel 2 includes the index nodes for region P_2 , and index channel 3 comprises of index nodes of region P_3 . Each of the regions has a valid data instance. When there are multiple sets of data instances, these data can be grouped in the unit of packet (or frame) and the leaf-node pointer goes to the data packet that contains the valid data instance in the particular region. For example, in the GPRS network a packet can contain data up to 1600 bytes (Xu et al, 2003). Consequently, the data is accessed by clients in the packet unit.

Similar to the indexing schemes for traditional queries, a traversal algorithm to allocate the index nodes into the index channels needs to be applied. However, in this case, the index is partitioned based on the data region. The Postorder traversal algorithm for this Global Index model for location-dependent queries is depicted in Figure 4.17.

```

Procedure TraverseGlobalIndexLDQ(T:tree) Is
  Begin
    If T = null then
      Return;
    Else
      TraverseGlobalIndexLDQ(T.left);
      TraverseGlobalIndexLDQ(T.right);
      VisitTree(T);
    End if;
  End Traverse GlobalIndexLDQ;

Procedure VisitTree(T:tree) is
  Begin
    If the index node is in the top level of index tree-structure then
      Assign the index node across entire index channels;
    Else
      Check the index nodes region with the index-partitioning attribute (regions based);
      Assign the index node into designated index channel;
    End if;
  End VisitTree;

```

Figure 4.17. Post Order Traversal with Index Allocation Scheme (Regions based)

The ownership rule of each index node is that the index channel owning a leaf node also owns all nodes from the root to that leaf. Consequently, the root node is replicated in all index channels, and non-leaf nodes may be replicated in some channels. Figure 4.18 shows the PRI construction and restructuring process starting from the initial index tree.

A simple scenario is to broadcast weather conditions for two regions within three states in Australia. There are 6 regions altogether to be broadcast. A tree structure is used to describe the regions as illustrated in Figure 4.19 (a). Figure 4.19(b) depicts the broadcast structure according to the regions involved and Figure 4.19(c) presents a table consisting of records of IDs, city, weather condition, and temperature. All the data instances are broadcast in the data channel separate from the index channel.

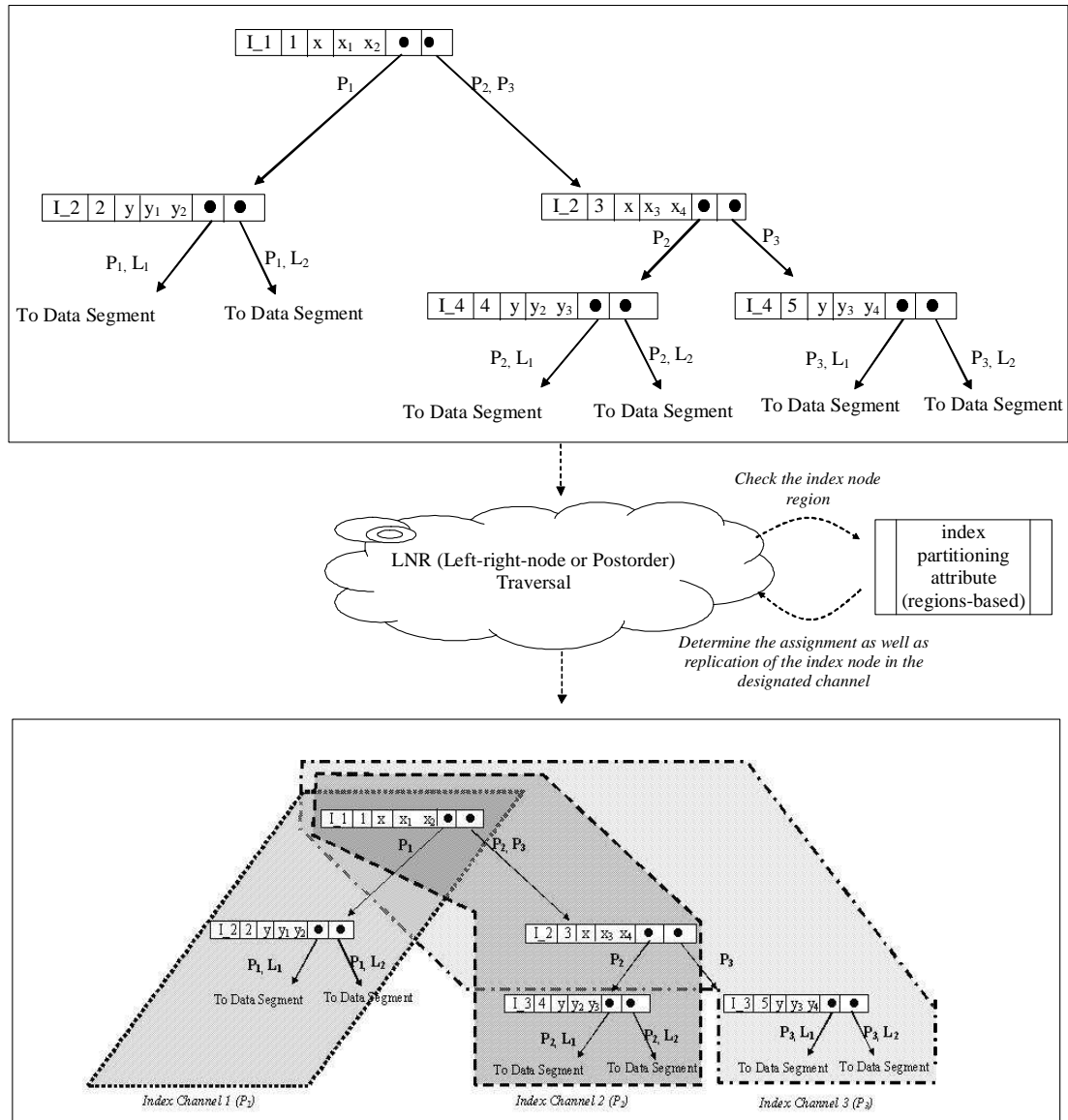
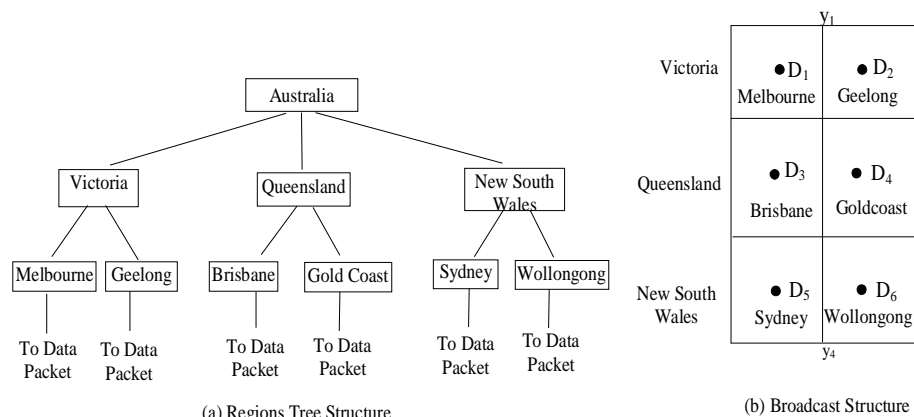


Figure 4.18. Global index model for location-dependent query with structuring process

Using the same example (shown in Figure 4.18), the Global index is depicted in Figure 4.20. In this example, the state attribute as the index-partitioning attribute is applied. It is assumed that the range partitioning rules used are that index channel 1 holds the regions within Victoria, index channel 2 holds the regions inside Queensland, and index channel 3 contains the regions within New South Wales. Notice from Figure 4.18 that the third non-leaf node is replicated in channels 1 and 2. The index is broadcast separately with the data, and each index key points to the

relevant data channel. Thus, once the right index is found in a specific index channel, the mobile client switches to the right data channel and waits for the data of interest to arrive.



(c) Sample Table

ID	City	Weather Condition	Temperature
1	Melbourne	Mostly Sunny	24°C
2	Geelong	Local Shower	15°C
3	Brisbane	Rain at times	12°C
4	Goldcoast	Mostly Fine	19°C
5	Sydney	Shower or Two	14°C
6	Wollongong	Partly Cloudy	10°C

Figure 4.19. Index construction – a scenario

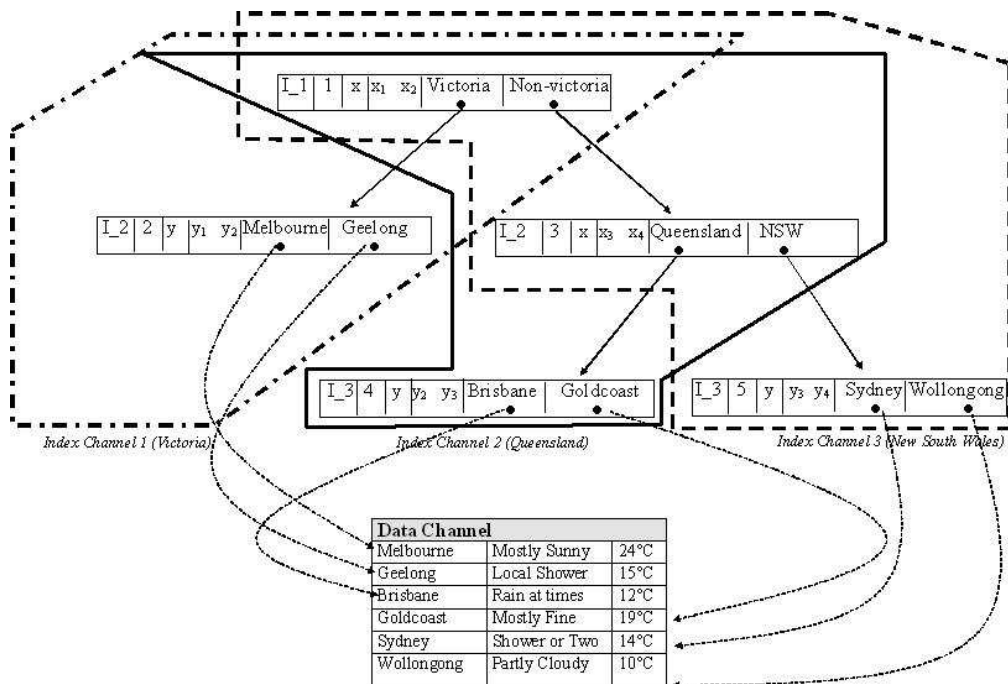


Figure 4.20. Global index model with a scenario

4.5 Performance Evaluation

This section studies the performance of the proposed broadcast indexing methods. The analysis is divided into sections based on the query types applicable for the proposed indexing schemes namely Broadcast Indexing over Multi Broadcast Channel for: (i) Traditional Queries, and (ii) Location-Dependent Queries. The performance indicators include index access time, client's tuning time and power consumption. The performance of the proposed schemes is also compared against the conventional one. Similar to the simulation platform which has been employed and described in Chapter 3 of this thesis, the simulation experiments in this chapter are also carried out using the *Planimate* simulation package, animated planning platforms (Seeley, 1997).

4.5.1 Traditional Queries

In this section, three cases are studied to in order to analyse the performance of the three broadcast indexing schemes, namely PRI, NRI, and FRI. In this simulation, the stock indices scenario as illustrated in Figure 4.6 with the same set of data items is applied. The simulation is run for a hundred iterations, and the average access time for up to 50 requests is calculated. The index inter-arrival rate is exponentially distributed. The parameters of concern in this study are given in Table 4.1.

Case 1 (a): To compare the performance of NRI, PRI and FRI in the context of single data item retrieval.

As shown in Figure 4.21, it can be seen that NRI outperforms both PRI and FRI. NRI provides access time that is approximately three times lower than the FRI average access time and about one and a half times better than PRI.

Table 4.1. Parameters of concern – PRI, NRI, FRI

Parameters	Value
<i>Partially-replicated indexing (PRI)</i>	
Index page in Channel 1	17
Index page in Channel 2	22
Index page in Channel 3	12
<i>Fully-replicated indexing (FRI)</i>	
Index page in Channel 1	41
Index page in Channel 2	41
Index page in Channel 3	41
<i>Non-replicated indexing (NRI)</i>	
Index page in Channel 1	13
Index page in Channel 2	18
Index page in Channel 3	9
<i>PRI - NRI - FRI</i>	
Node Pointer Size	5 bytes
Data Pointer Size	5 bytes
Indexed Attribute Size	4 bytes
Bandwidth	19200 bytes
Index Arrival Rate	4 index pages per sec

Another thing that can be seen from Figure 4.21 is that, in certain cases, the average access times decreases as the number of requests increases. The explanation for this may be the different indexes of interest for each client as well as non-uniform probe time or time when the client starts tuning in to the relevant index channel.

This situation normally happens when only a very few requests are involved in the analysis. The average access time can be large since the gap between the longest and the shortest access time that occurred from the request may also be large. As the number of requests is only limited, then the difference can be clearly seen. The result

can be stabilized when more requests are involved. Subsequently, the actual pattern of the access time becomes obvious.

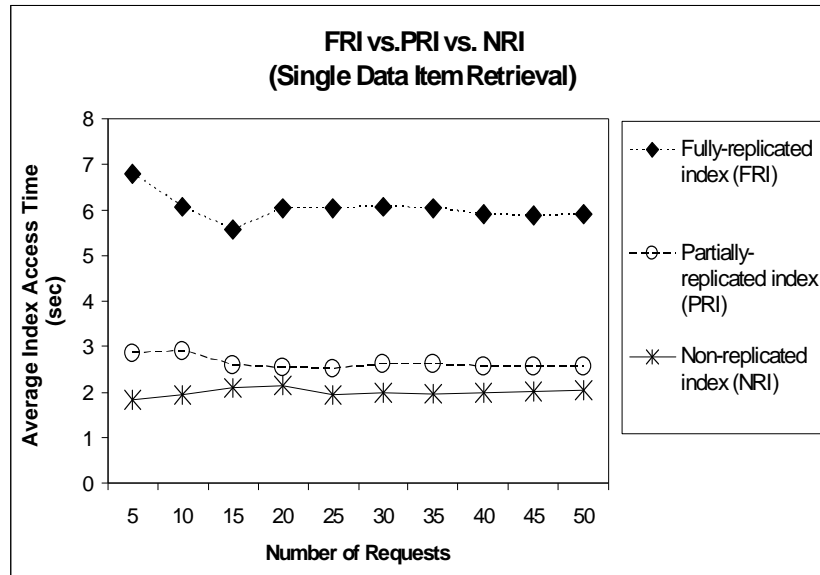


Figure 4.21. NRI vs PRI vs FRI (Single Data Item Retrieval)

Case 1 (b): To compare the performance of NRI, PRI, and FRI in the context of retrieval of two data items. In this case, the relevant indexes are retrieved in an unordered fashion, which means that the first desired index that arrives in the channel is obtained first.

Figure 4.22 shows that NRI still provides a better access time than the other two. This pattern follows the same trend as for case 1(a), but it has an overall longer access time. It can also be seen that the performance of PRI is getting close to NRI with less than one and a half lower index access time.

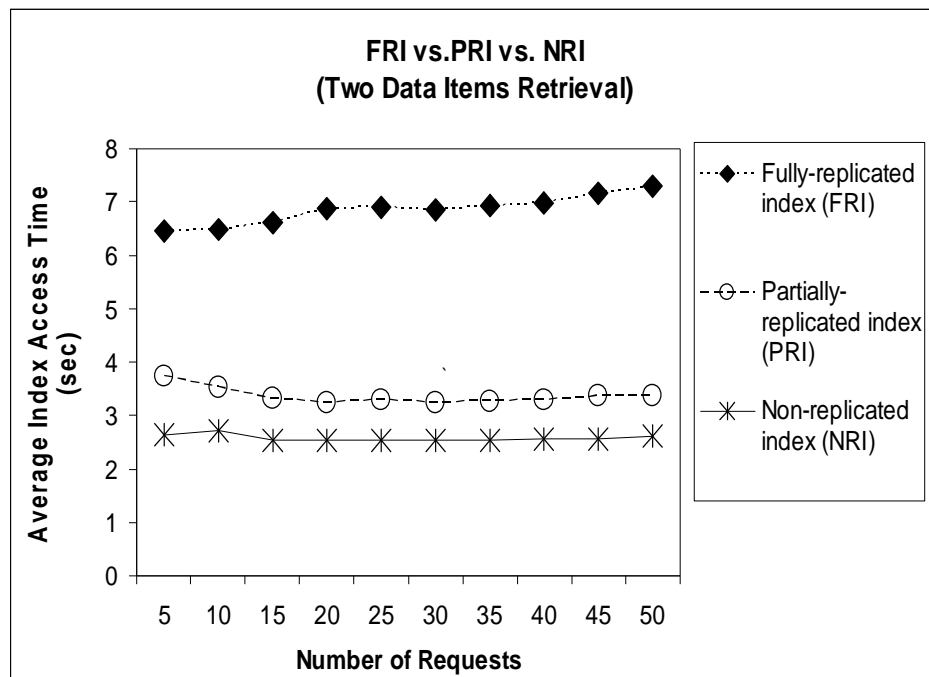


Figure 4.22. FRI vs PRI vs NRI (Two Data Items Retrieval)

Case 2: To compare the performance of PRI with a conventional method as suggested by Leong and Si (Leong and Si, 1995). The comparison includes one and two data items retrieval.

The indexing scheme in the conventional method is also designed to employ $B+$ -tree structure and is similar to FRI except that only a single broadcast channel is utilized. It can be seen from Figures 4.23 (a) and 4.23 (b) that PRI performs at nearly three times a lower average access time than the conventional one in both cases.

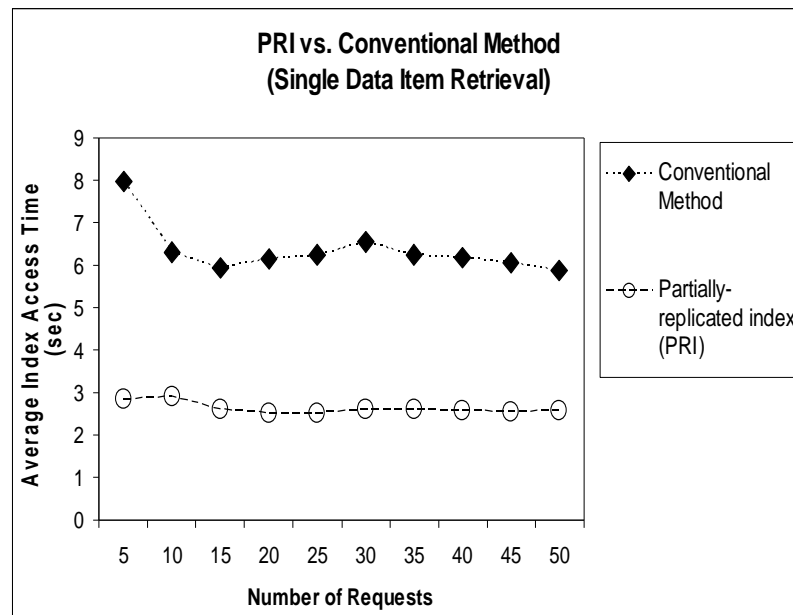


Figure 4.23(a). PRI vs Conventional Scheme (Single Data Item Retrieval)

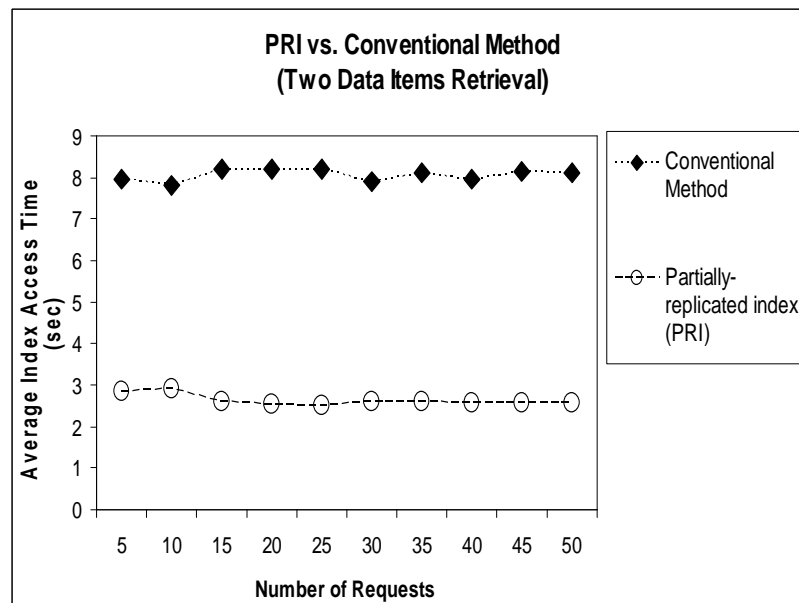


Figure 4.23(b). PRI vs conventional scheme (two data items retrieval)

Case 3: It should be noted that the PRI scheme replicates approximately 30% of the index pages compared with the NRI scheme. This contains the least possible duplication/redundancy while still having the advantage of the PRI scheme.

The replication enables a mobile client to tune in to any of the channels and follow the pointer to obtain the right index key. In this case, it is interesting to see the performance of the PRI scheme if 50% of the index pages are replicated as compared with 30%. The PRI with 50% replication is called PRI-50.

Figures 4.24 (a) and 4.24 (b) shows the performance of PRI and PRI-50 in the context of one and two data items retrieval respectively. The result suggests that the more replication involved, the longer will be the access time. This is due to the larger number of index nodes in each channel, which increase the index cycle correspondingly. In both cases of one and two data items retrieval, normal PRI outperforms the PRI-50. The average access time difference between these two schemes corresponds to the added percentage of the replication, which is about 20%.

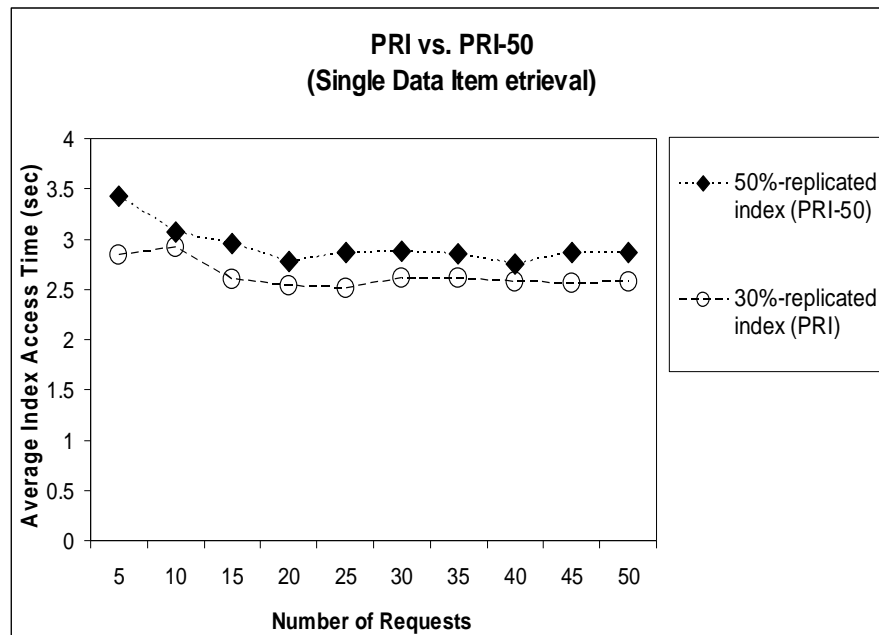


Figure 4.24(a). PRI vs PRI-50 (single data item retrieval)

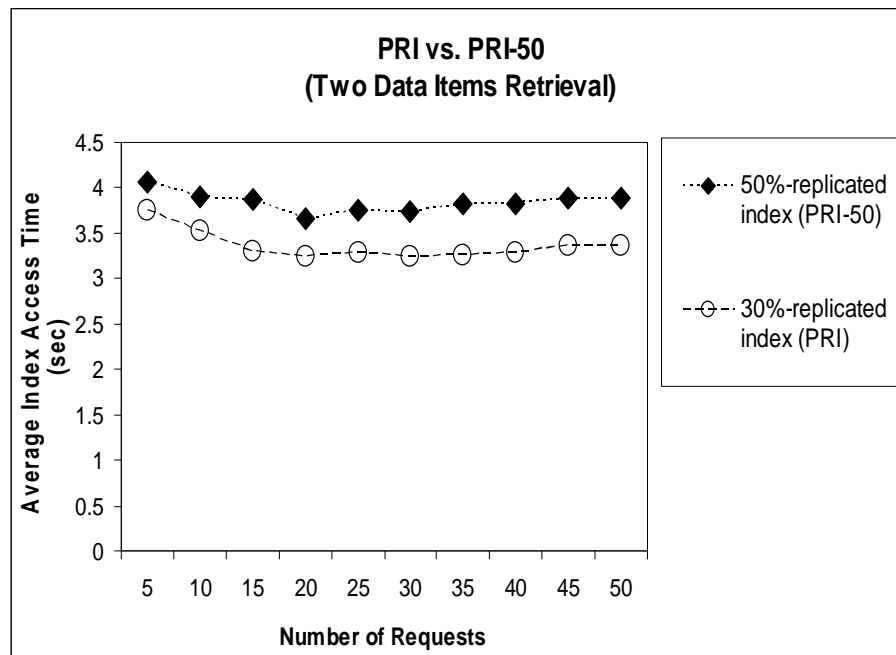


Figure 4.24(b). PRI vs PRI-50 (Two Data Items Retrieval)

- **Extended Simulation: Partially-replicated indexing (PRI) vs Fully-replicated indexing Scheme (FRI):**

In this section, the performance of the PRI is further analysed against FRI in different cases. It should be noted that in this extended simulation the term *Global Index* relates to PRI scheme, whilst *Non-Global Index* corresponds to the FRI scheme. The cases for comparison are concerned with varying the index-inter arrival rate and the skew request distribution in terms of one and two indexes retrieval. Similar to the initial performance evaluation, the simulation model here also applies the stock indices scenario as illustrated in Figure 4.6 with the same set of data items.

The cases are classified based on the relevant performance indicators including index access time, tuning time and power consumption. The simulation model is run for 50 iterations, and calculates the average access time for given number of request, which ranges from 5 to 40 numbers of requests with 5 request increment. The simulation

environment is set to apply exponential distribution for index arrival rate with given an average value. The parameters of concern are given in Table 4.2.

Table 4.2. Parameters of concern – Global Index and Non-Global Index (Traditional Queries)

Parameters	Value
Number of Index nodes in Global Index (Channel 1)	17
Number of Index nodes in Global Index (Channel 2)	22
Number of Index nodes in Global Index (Channel 3)	12
Number of Index nodes in Non-Global Index	41
Node Pointer Size	5 bytes
Data Pointer Size	5 bytes
Indexed Attribute Size	2 bytes
Bandwidth	64 Kbps
Index Arrival Rate	2 and 4 index pages per

- **Access Time**

Two cases are introduced: (i) to retrieve a single data item, and (ii) to retrieve two data items. For each case, the index arrival rate is varied from 2 and 4 index pages per *sec*. The performance of Global Index when there is a skew request distribution is also analysed. For example, Global Index with skew at channel 1 relates to majority (60%) of request concerns with data index at channel 1 and soon. The desired index pages to be retrieved are randomized.

Case 1: To compare the performance of Global Index with Non-Global Index in the context of single data item retrieval. This case applies an arrival rate of 4 index pages per *sec*.

It can be seen from Figure 4.25 that (a) the Global Index outperforms the Non-Global Index with about two to three times lower average access time. In the situation when skew request distribution occurs, among the three channels within Global Index, it is found from Figure 4.25 that (b) that index channel 3 provides a

slightly better average access time as compared with other channels. It indicates that the more requests with index key located in channel 3, the better the average access time. This is due to the short index broadcast cycle that exists in index channel 3, so that mobile clients do not wait too long to find the right index key.

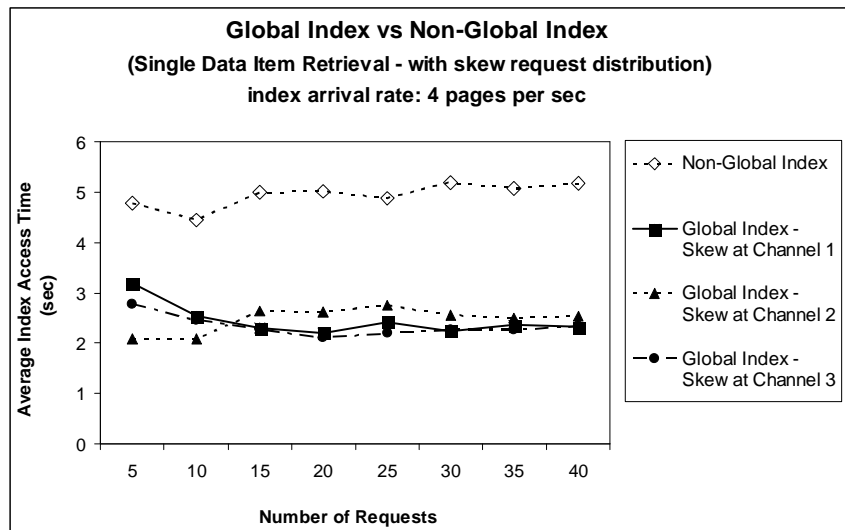


Figure 4.25(a). Global Index vs Non-Global Index (single data item retrieval): 4 index pages arrival rate per sec

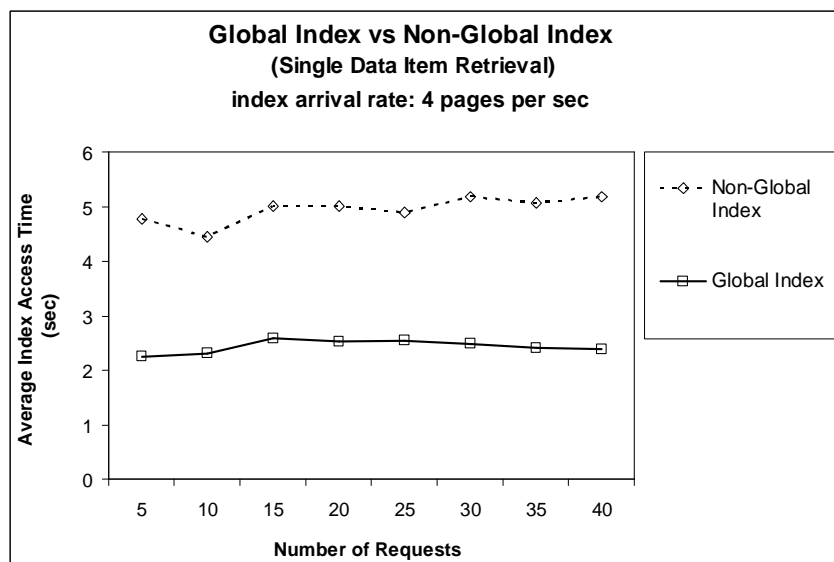


Figure 4.25(b). Global Index vs Non-Global Index (single data item retrieval) with Skew Request Distribution: 4 index pages arrival rate per sec

Case 2: Similar to Case 1. However, this time the index page arrival rate is modified to 2 index pages per *sec*.

As shown in Figure 4.26, the overall access time in this rate is approximately twice that of Case 1. It shows that the Global Index scheme outperforms the Non-Global Index with about half the access time. The access time follows the patterns in Case 1 very closely.

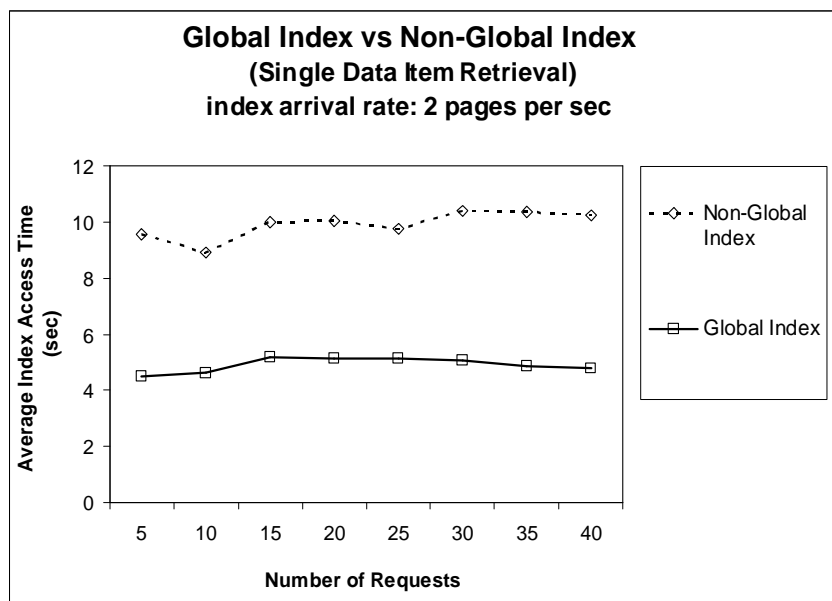


Figure 4.26(a). Global Index vs Non-Global Index (single data item retrieval): 2 index pages per sec.

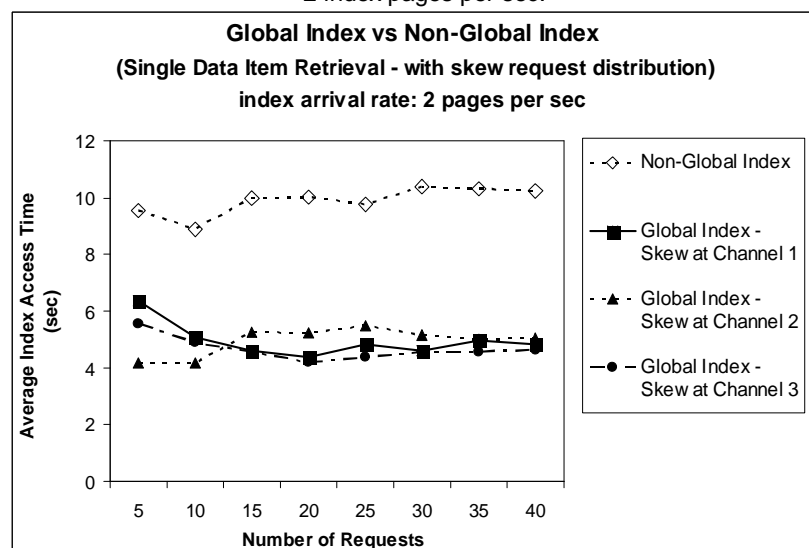


Figure 4.26(b). Global Index vs Non-Global Index (Single Data Item Retrieval) with Skew Request Distribution: 2 index pages arrival rate per sec

Case 3: To compare the performance of the Global Index and the Non-Global Index in the context of two data items retrieval. This case relates to an arrival rate of 4 index pages per *sec*.

Figure 4.27 (a) shows that the Global Index still outperforms the Non-Global Index with about one and a half to two times better index access time in any condition. . Similar to the previous cases, in the situation when skew request distribution occurs, among the three channels within Global Index, index channel 3 provides on average a slightly better access time performance as compared to other channels as what shown in Figure 4.27 (b)

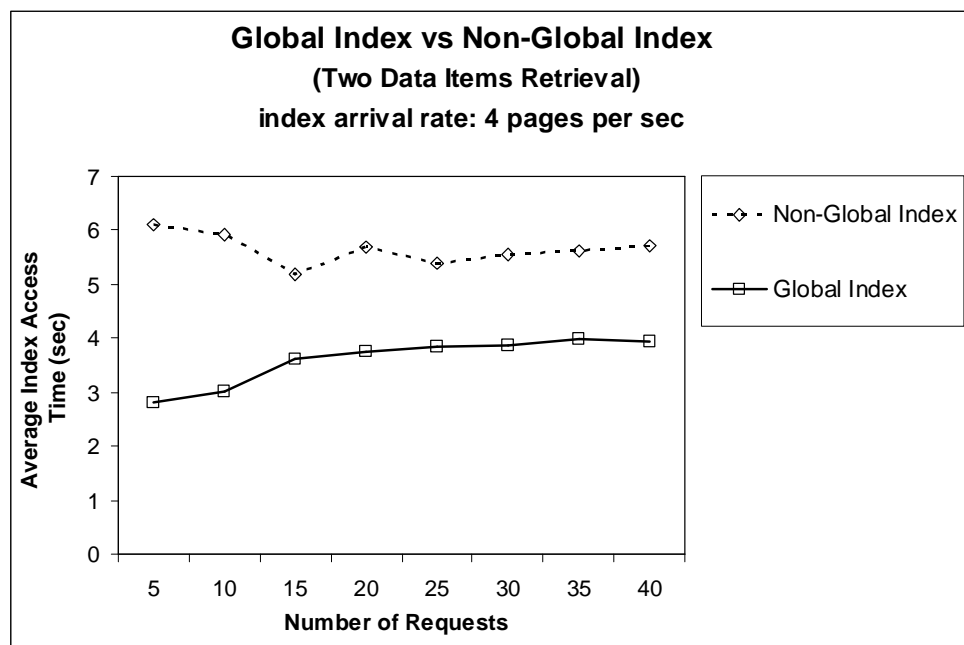


Figure 4.27(a). Global Index vs Non-Global Index (two data items retrieval):
4 index pages per sec

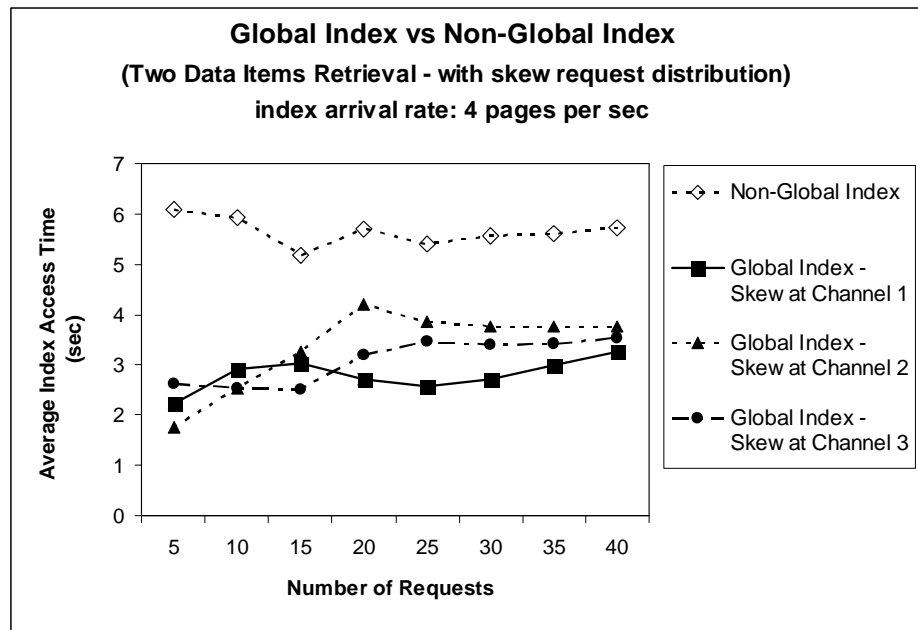


Figure 4.27(b). Global Index vs Non-Global Index (two data items retrieval) with skew request distribution: 4 index pages per sec

Case 4: Similar to Case 3, but this time we evaluated the arrival rate of 2 index pages per *sec*. Figure 4.28 depicts that the Global Index still provides a considerably better access time as compared with the Non-Global Index with approximately a one and a half to two times lower average access time in both circumstances.

Another thing that can be seen from Figures 4.25 to 4.28 is that in some cases, the average access time decreases as the number of requests increase. The explanation for this might be due to the probe time or time when the client starts tuning in to the relevant index channel not being uniform. Thus, a few requests may probe into the index channel at the same time that provides short access time, while others need to wait for some time before the index arrives. Furthermore, since the length of each index channel is generally short, and there is a greater number of requests, then the average access time may be decreased depending on the waiting time of the majority

requests in a particular index channel. With greater uniformity of probe time, the access time will be stabilised.

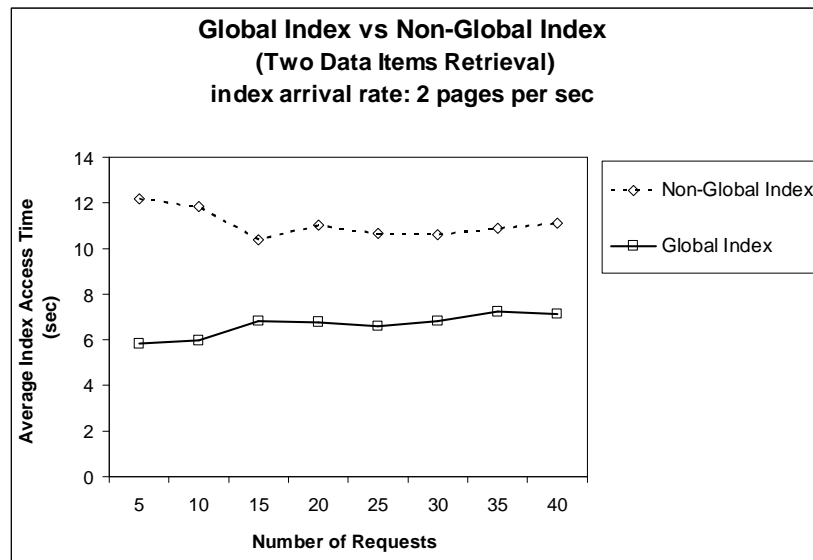


Figure 4.28(a). Global Index vs Non-Global Index (two data items retrieval): 2 index pages per sec

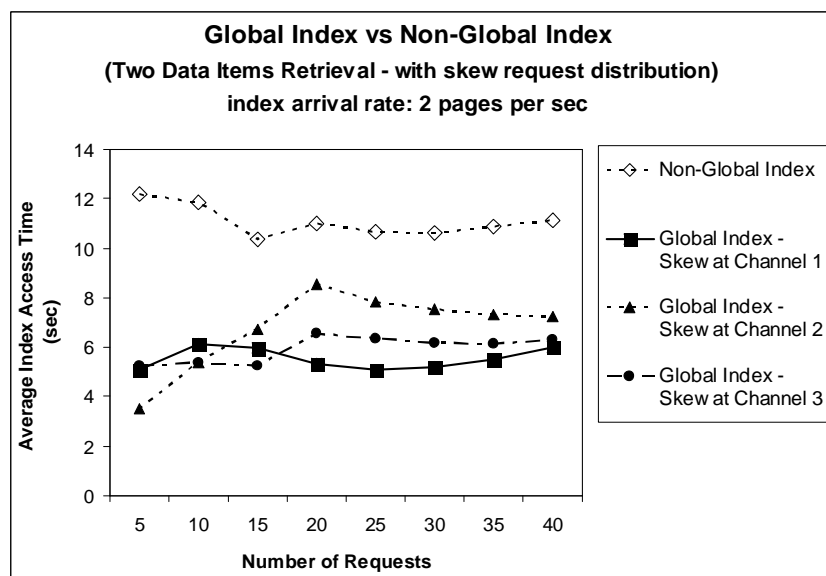


Figure 4.28(b). Global Index vs Non-Global Index (two data items retrieval) with skew request Distribution: 2 index pages per sec

- **Tuning Time**

Case 5: This case compares the tuning time of the Global index with non-global index scheme. Tuning time defines the amount of time clients must listen to the

channel in order to retrieve the desired data instances. The result can be used as an indication of the client power consumption. For the purpose of comparison, the tuning time is measured from the time client probes into the index channel until the final index key that points to the desired data item is found. The tuning time needed to obtain the desired data item is not included since both indexes retrieve the same data items, which can be easily incorporated whenever necessary.

As exhibited in Figure 4.29, the tuning time of mobile clients accessing the Global index is far less than the non-global index scheme. More importantly, the result also indicates that clients use much less power during query operation with the Global indexing scheme.

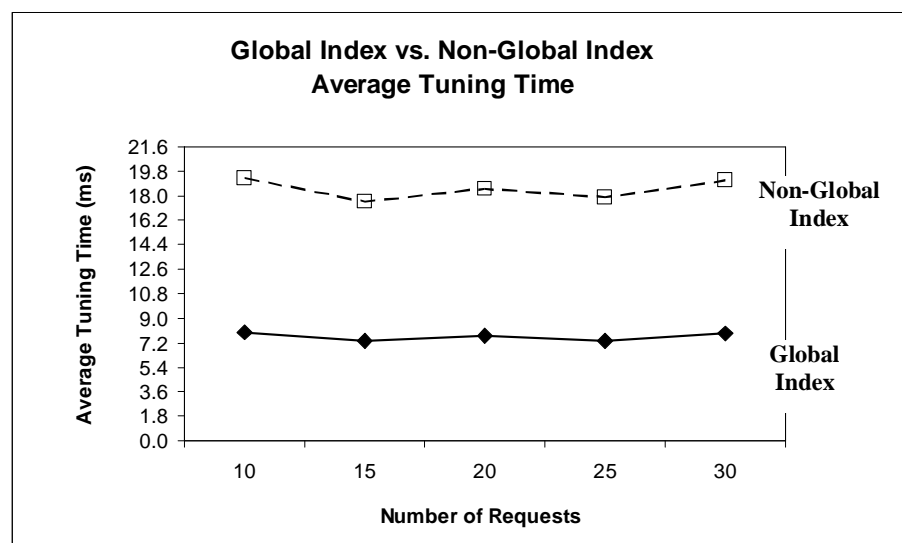


Figure 4.29. Tuning time: Global Index vs Non-Global Index

- **Power Consumption**

Case 6: In this case, the power consumption of mobile clients in accessing Global index will be analysed. According to Imielinski et al (1997), a device with a Hobbit chip (AT&T) requires about 250mW power consumption during active mode, and

50 μ W during power saving mode. Thus, it can be seen that the length of time of clients' listening and not listening to the channel has affected the power consumption. The power consumption of mobile clients is measured based on the duration of active mode and power saving mode that has been obtained from the simulations. For simplicity, other activities and components that require power consumption are disregarded, and assumed that 250mW relates to total power consumption. The calculation is determined from the following formula:

Power Consumption = (250 x *Time during active mode*) + (0.05 x *Time during power saving mode*). The analysis involves 30 broadcast items.

As depicted in Figure 4.30, the power consumption of clients accessing the Global index is substantially less than for the non-global index. With less power consumption, client can reserve battery power more efficiently, which is most desirable considering the limited power storage in mobile devices.

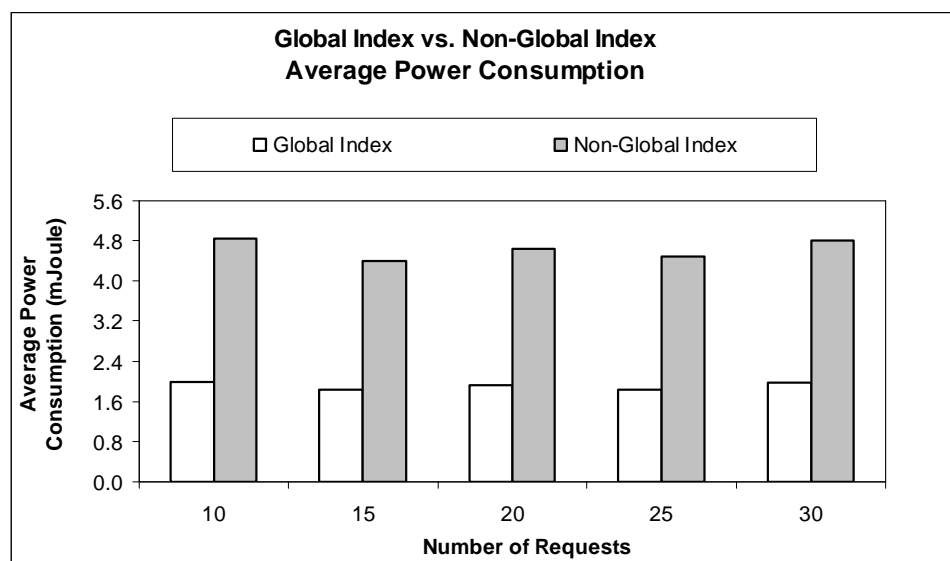


Figure 4.30. Power Consumption: Global Index vs Non-Global Index

4.5.2 Location-dependent Queries

In this section, the performance of the global index model for location-dependent queries is evaluated. The simulation model follows the weather scenario as illustrated in Figure 4.17. Table 4.3 shows the parameters of concern.

Table 4.3. Parameters of concern – Location Dependent Queries

Parameters	Value
level size	3 bytes
id size	3 bytes
header size	3 bytes
Node Pointer Size (each)	5 bytes
Data Pointer Size (each)	5 bytes
Index Coordinate Size	4 bytes
Bandwidth	19200 bytes
Index Arrival Rate	4 index pages per sec

The proposed global index incorporates three index channels, and compares the access time, tuning time, and power consumption performance with non-global index. Non global index refers to the conventional broadcast indexing scheme that includes the entire index structure in a single channel. Three cases are introduced to analyse the performance of the proposed indexing scheme. The index inter-arrival rate is set to be exponentially distributed.

Case 1: To compare the performance of square valid scope with polygon valid scope. Polygon is a general type of valid scopes (Lee et al, 2002). Figure 4.31 shows an example of the valid scope that uses the shape of a polygon.

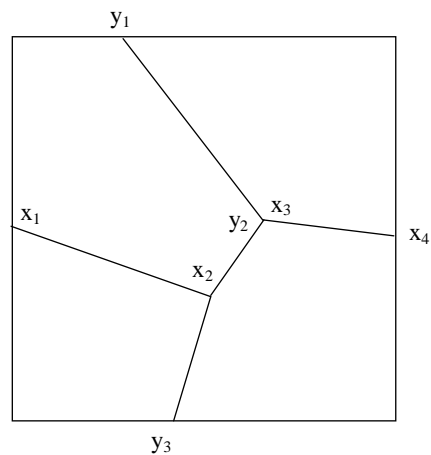


Figure 4.31. Polygon valid scope

The performance is concerned with the processing time of mobile clients when receiving these two different valid scopes. The processing time relates to the amount of time a client needs to obtain the attributes value in the index node. The time required to process such information will reflect to the amount of power consumption. In this simulation, it is assumed the client's processing time for each index attributes is 0.01 sec with exponential service distribution. The simulation is set to forty iterations and derives the average result accordingly.

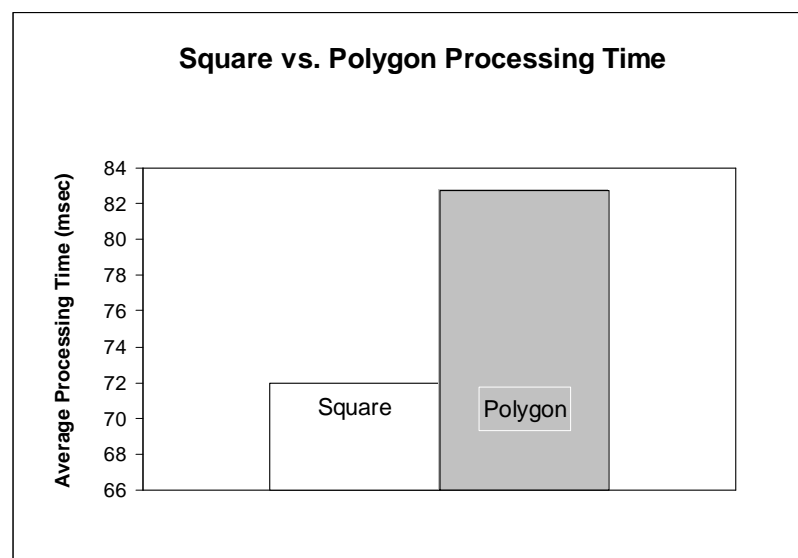


Figure 4.32. Square vs Polygon valid scope

As can be seen from Figure 4.32, square valid scope requires less client's processing time than polygon. With less processing time, the power consumption for mobile clients can be minimised, which is of great benefit to mobile devices.

Case 2: To compare the performance of the proposed global index with non-global index. In this case, there are 25 number regions involved, and the index tree is made balanced with 5 numbers of rows and columns. The simulation is set to fifty iterations and derives the average result accordingly.

The comparison includes the access time, tuning time and power consumption. Similar to the earlier case for Traditional Queries, the calculation to determine power utilisation is obtained from the following formula:

$$\text{Power Consumption} = (250 \times \text{Time during active mode}) + (0.05 \times \text{Time during power saving mode}).$$

Figure 4.33 (a) demonstrates that Global index outperforms non-global index with an average of one-third access time. The global index access time is more stabilised as it involves a shorter index cycle for each index channel. The tuning time performance as indicated in Figure 4.33 (b) shows that the proposed scheme provides a much better tuning time. This tuning time performance reflects the amount of power consumption required by mobile devices as is displayed in Figure 4.33 (c). It should be noted that the tuning time needed to obtain the desired data item is not included in evaluation since both indexes retrieve the same data items, which can be easily incorporated whenever necessary

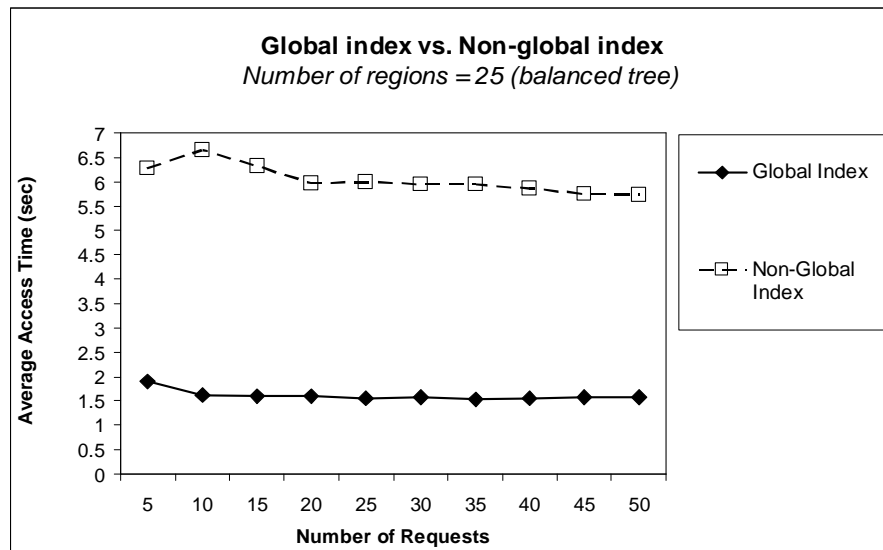


Figure 4.33(a). Global index vs Non-global index (balanced tree): access time

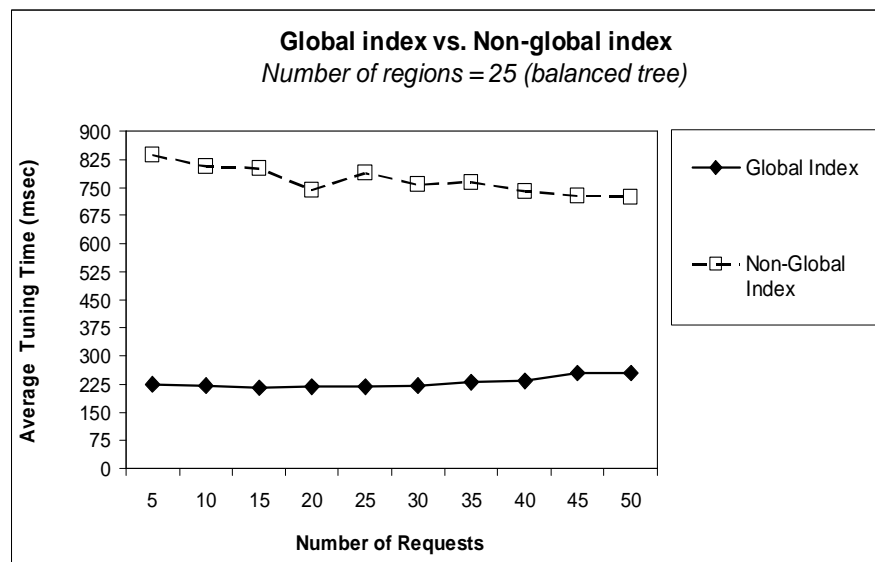


Figure 4.33(b). Global index vs Non-global index (balanced tree): tuning time

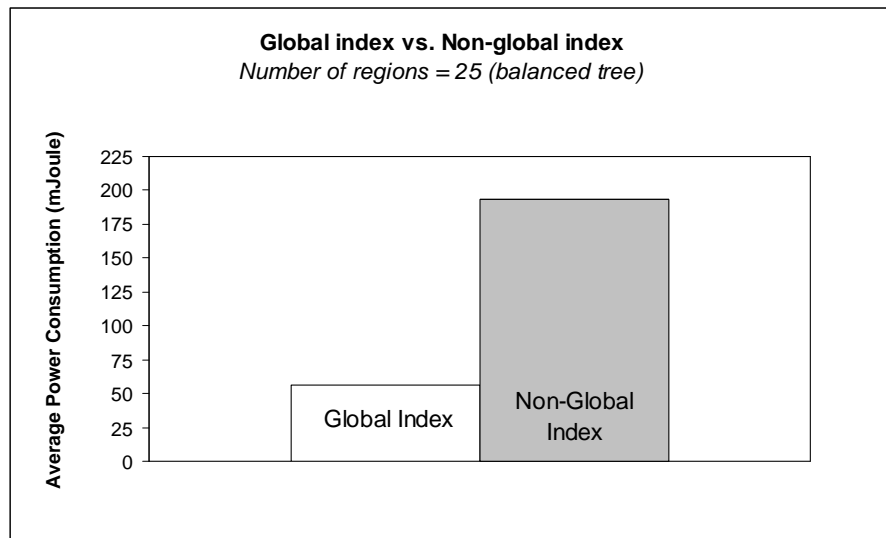


Figure 4.33(c). Global index vs Non-global index (balanced tree): power consumption

Case 3: To compare the performance of the proposed global index with non-global index. As opposed to Case 2, here there are 30 regions involved, and the index tree is therefore unbalanced with 2 rows and 15 columns.

As depicted in Figure 4.34 (a), it can be seen that the global index outperforms the non-global index with an approximately two times lower average access time. Similarly, in Figure 4.34 (b) the tuning time obtained by global index is much less than non-global index scheme. Another thing that we can see from Figure 4.34 (b) is that at certain points, the average tuning time decreases as the number of requests increases. The explanation for this may be due to the different indexes of interest for each client as well as non-uniform probe time or time when the client starts tuning in to the relevant index channel.

This situation normally occurs when only a very few requests are involved in the analysis but the index cycle is rather long. The average tuning time can be large since

the gap between the longest and the shortest tuning time that occurred from the requests involved may also be large. At that point, the number of requests is only small, so that the difference can be clearly seen. The result can be stabilized with a larger number of requests involved. Subsequently, the actual pattern can be seen. As shown in Figure 4.34 (c), the amount of power consumption using global indexing scheme is about half that of the non-global ones. This is most desirable considering the limited battery power of mobile devices.

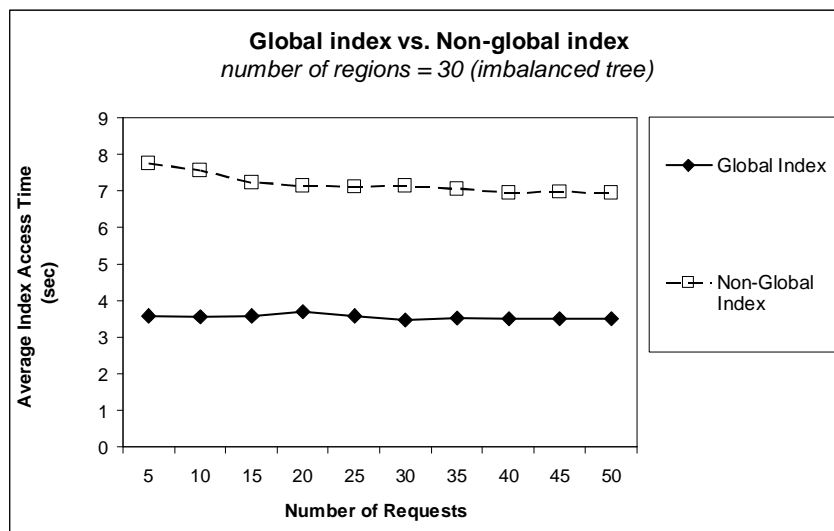


Figure 4.34(a). Global index vs Non-global index (unbalanced tree): access time

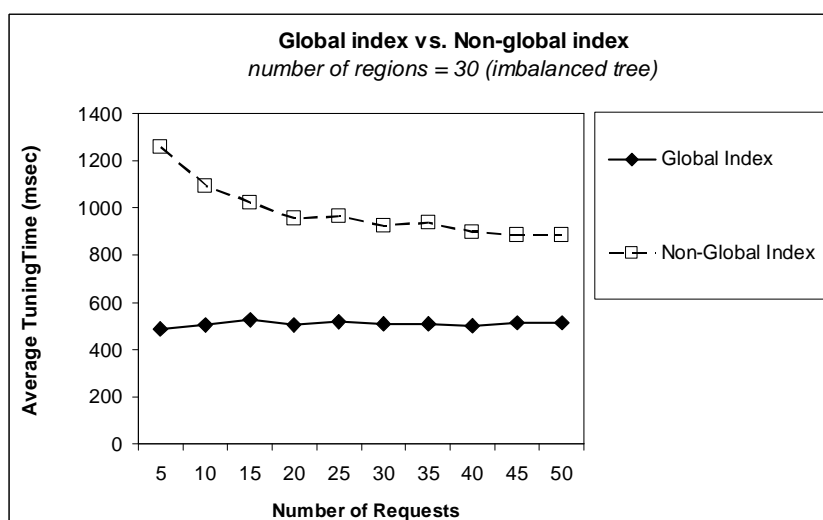


Figure 4.34(b). Global index vs Non-global index (unbalanced tree): tuning time

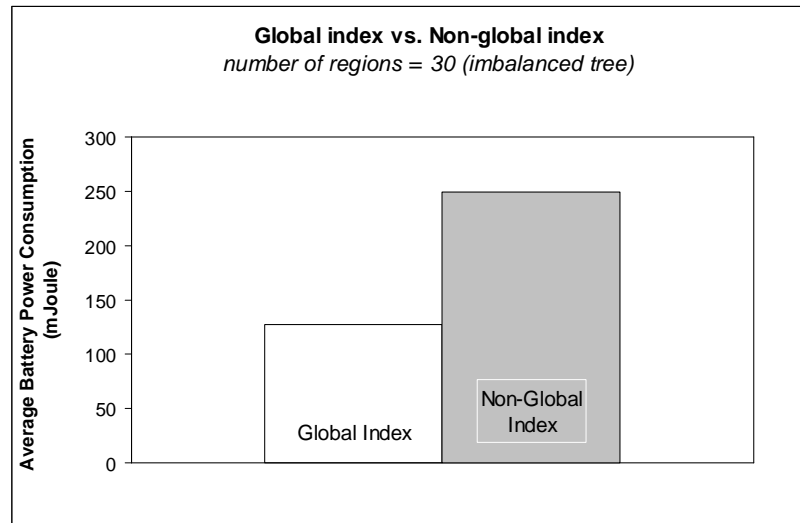


Figure 4.34(c).Global index vs Non-global index (unbalanced tree): power consumption

4.6 Index Maintenance

This section describes the process for maintaining the NRI and PRI indexing scheme. The maintenance involves the index restructuring process which includes an insertion and deletion of index node. This maintenance is not regularly required as all data items are broadcast and made available for mobile clients to obtain whatever data items they are interested in over wireless channels without having to send any queries to the server (see Chapter 3). It is necessary to carry out this process only when there are new data inserted in the databases. Whilst the index structure is being reconstructed, the server can keep broadcasting the earlier version of the broadcast structure from the cache until the new structure is ready to be applied. Figure 4.33 shows the architecture of the broadcast system.

As shown in Figure 4.35, the server periodically broadcasts data items according to a predetermined broadcast program. When a user submits a query to his or her mobile

device, the mobile device will retrieve the required data item from the broadcast channel by referring to the data index information. The server contains at least five entities namely: (i) broadcast scheduler, (ii) server cache, (iii) broadcast program generator, (iv) data indexing generator, and (v) query patterns. The mechanism of this broadcast system can be described as follows:

- The server retrieves the entire database items from data repository, the data repository transmits the results back to the server and they are processed by the broadcast program generator.
- The broadcast program generator concerns the ordering and allocation of data items in the broadcast channels. To obtain optimum placement of broadcast data items, the broadcast program generator needs to be informed about the behaviour of mobile users in accessing the broadcast information or query access patterns of mobile clients (this case relates to Chapter 3 of this thesis).
- Having determined the broadcast program and the related data channels, the broadcast program generator sends the information to the data indexing generator.
- The data indexing generator constructs the index based on the information from the broadcast program generator. The indexing generator applies the index structuring scheme as discussed in this chapter.
- The final broadcast program and the index structure are then transmitted to the server cache. The broadcast program generator will be able to update the program at any given time according to the query access patterns.
- The broadcast scheduler will have the final broadcast program and the relevant index structure from the server cache. It is then responsible for the scheduling

issues to broadcast the data items and their indexes before they are transmitted to the wireless channels.

- When data item values in the data repository are updated, the new values and the relevant identification will be sent to the server cache immediately. As such, the new data values will be directly made available for mobile clients.
- However, when there is insertion or deletion of data items in the data repository, the data repository will pass the information about the new or deleted data items to the broadcast program generator. The broadcast program generator will subsequently process the data item to determine the new broadcast program and this process continues to that described in step 2 of this mechanism.

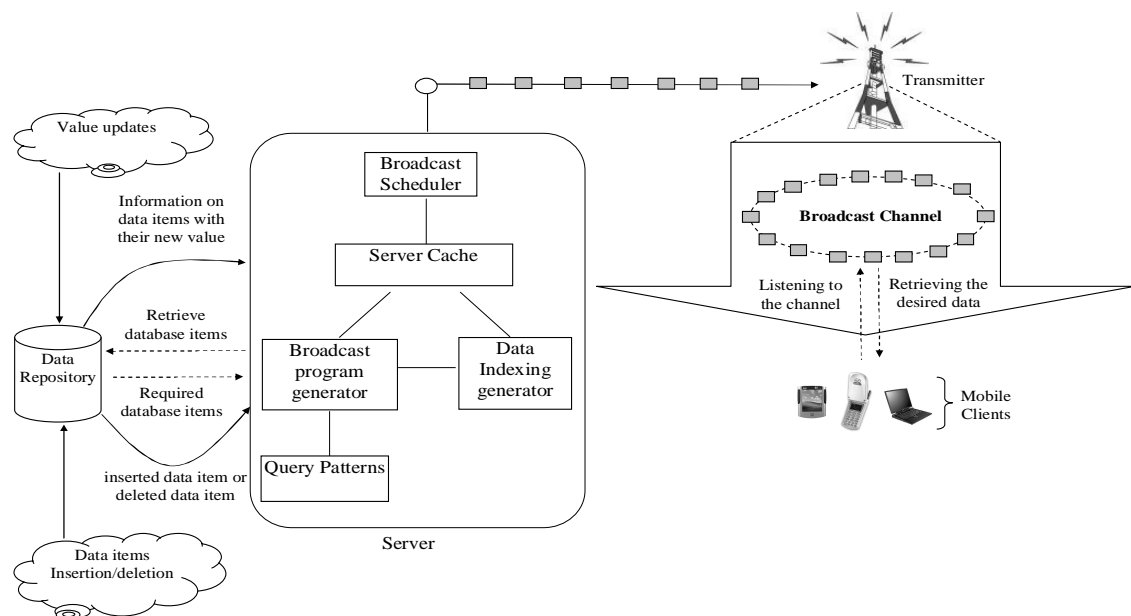


Figure 4.35. Architecture of a broadcast system

This section considers the case of NRI and PRI indexing scheme re-structuring when there is any insertion or deletion of the data item in the data repository.

a. NRI-Indexing Scheme

The mechanism of index maintenance is carried out as per normal index maintenance (Elmasri and Navathe, 2003). The insertion and deletion procedure is summarised as follows.

The index key insertion steps are as follows. First, search for an appropriate leaf node for the new key on the index tree according to the predefined index-partitioning attribute. Then, insert the new key entry to this leaf node, if there is still space in this node. However, if the node is already full, this leaf node must be split into two leaf nodes. The first half entries are kept in the original leaf node, and the remaining entries are moved to a new leaf node. The last entry of the first of the two leaf nodes is copied to the non-leaf parent node. Furthermore, if the non-leaf parent node is also full, it has to be split again into two non-leaf nodes, as has happened with the leaf nodes. The only difference is that the last entry of the first node is not copied to the parent node, but moved. Finally, a data pointer is established from the new key on the leaf node to the record.

The deletion process is similar to that of the insertion. Firstly, delete the record, and then, delete the desired key from the leaf node in the index tree (the data pointer is to be deleted as well). When deleting the key from a leaf node, it is possible that the node becomes underflow after the deletion. In this case, try to find a sibling leaf node (a leaf node directly to the left or to the right of the node with underflow) and redistribute the entries among the node and its sibling so that both are at least half full; otherwise, the node is merged with its siblings and the number of leaf nodes is reduced.

The index maintenance for FRI is similar to NRI indexing scheme except that FRI scheme should update the index structure in the entire channels rather than specific index channel as what NRI does.

b. PRI-Indexing Scheme

- *Node Insertion*

When a new key (record) is inserted into the PRI indexing scheme, the PRI scheme needs restructuring. The following example illustrates the index restructuring process. It is an insertion of entry 28 into the existing index. This example shows three stages of index insertion process. The stages are (i) the initial index tree and the desired insertion of the new entry to the existing index tree, (ii) the splitting node mechanism, and (iii) the restructuring of the index tree.

The initial index tree position is shown in Figure 4.36(a). When a new entry of 28 is inserted, the first leaf node becomes overflow. A split of the overflow leaf node is then carried out. The split action also causes the non-leaf parent node to overflow, and subsequently, a further split must be performed to the parent node (see Figure 4.36(b)).

Notice that when splitting the leaf node, the two split leaf nodes are replicated in index channels 1 and 2, although the first leaf node after the split contains entries of the first channel only (25 and 28 – the range of index channel 1 is 1-37). This is because the original leaf node (25, 30, 44) is already replicated in both channels 1 and 2. The two new leaf nodes have a node pointer linking them together.

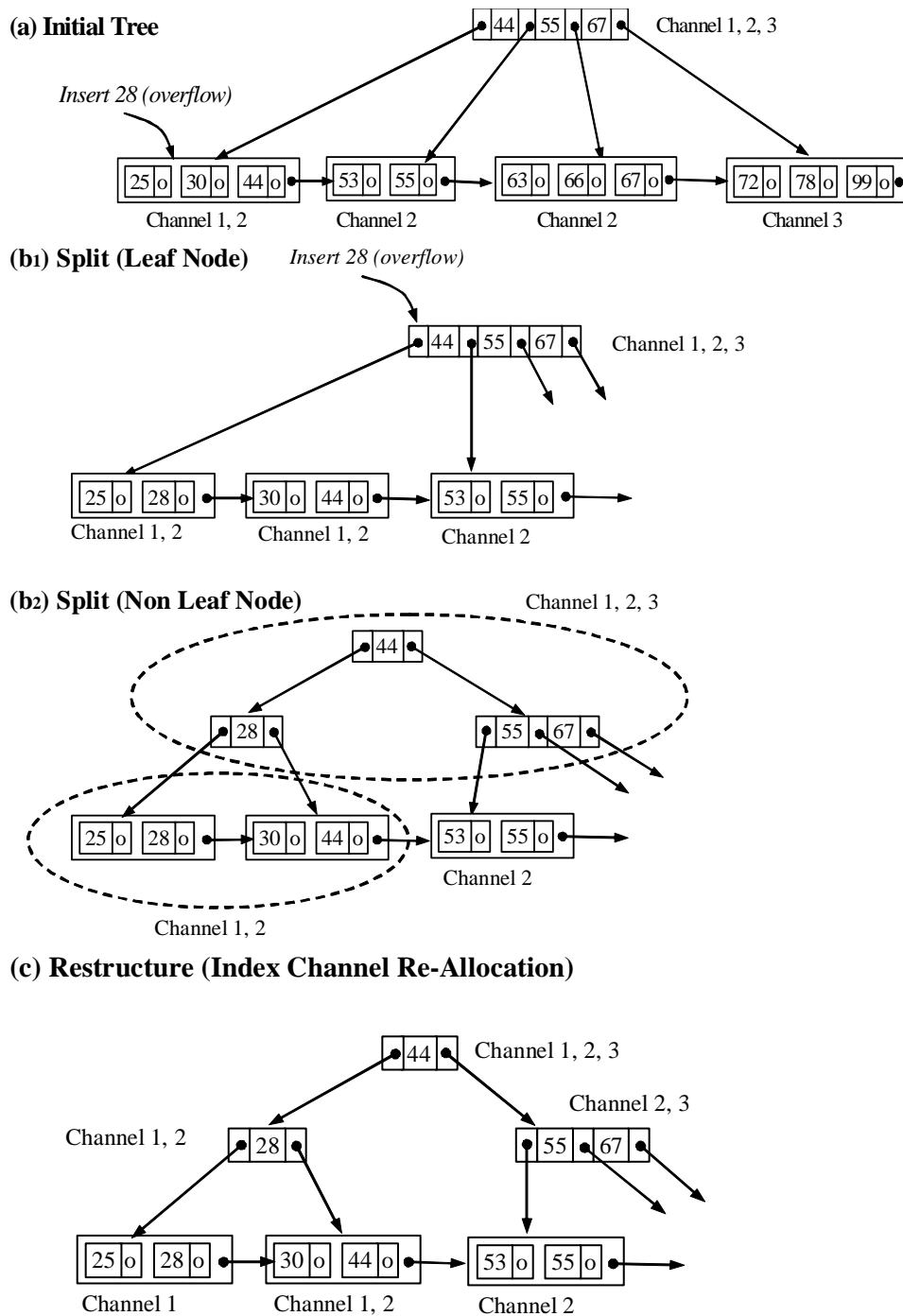


Figure 4.36. Index entry insertion

When splitting the non-leaf node (44, 55, 67) into two non-leaf nodes (28; and 55, 67), index channel 3 is involved because the root node is replicated in channel 3 too. The final step is the restructuring step. This step is necessary to ensure that each node has been allocated to the correct index channels. Figure 4.36(c) shows

a restructuring process. In this restructuring, index allocation is updated. This is done by performing a sequential traversal of the tree, finding the range of the node (min, max), determining the correct index channel(s), and reallocating to the designated channel(s). When reallocating the nodes to channel(s), each channel will also update the node pointers, pointing to its local or neighbouring child nodes. Notice that in the example, as a result of the restructuring, the leaf node (25, 28) is now located in channel 1 only (instead of channels 1 and 2).

- *Node Deletion*

To illustrate how node deletion works, consider the following example. We would like to delete entry 28, expecting to get the original tree structure shown previously before entry 28 is inserted. Figure 4.37 shows the current tree structure, the merge and collapse processes.

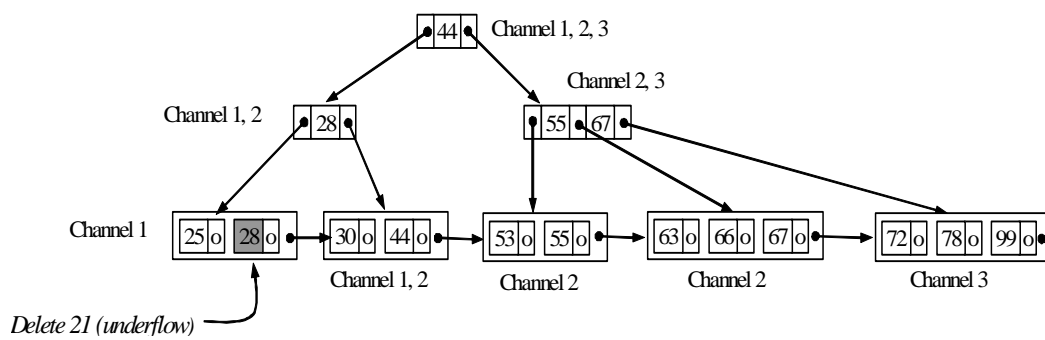
As shown in Figure 4.37(a), after the deletion of entry 28, the leaf node (25) becomes underflow. A merging with its sibling leaf node must be carried out. When merging two nodes, the index channel(s) which own the new node are the union of all channels owning the two old nodes. In this case, since node (25) is located in channel 1 and node (30, 44) is in channels 1 and 2, the new merged node (25, 30, 44) should be located in channels 1 and 2. Also, as a consequent to the merging, the immediate non-leaf parent node entry has to be modified to identify the maximum value of the leaf node, which is now 44, not 28.

As shown in Figure 4.37(b), the right node pointer of the non-leaf parent node (44) becomes void. Because the non-leaf node (44) has the same entry with its

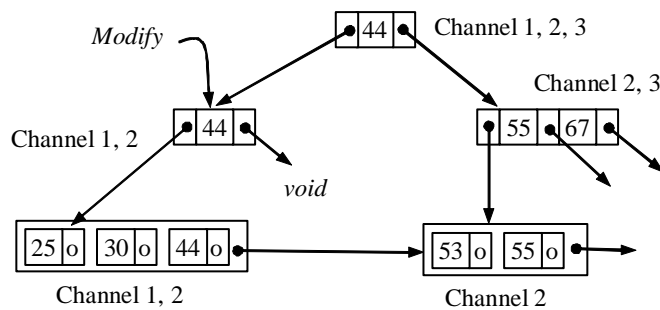
parent node (root node (44)), they have to be collapsed together, and consequently a new non-leaf node (44, 55, 67) is formed (see Figure 4.37(c)).

The restructuring process is the same as that for the insertion process. In this example, however, index channel allocation has been done correctly and hence a restructuring is not needed.

(a) Initial Tree



(b) Merge



(c) Collapse

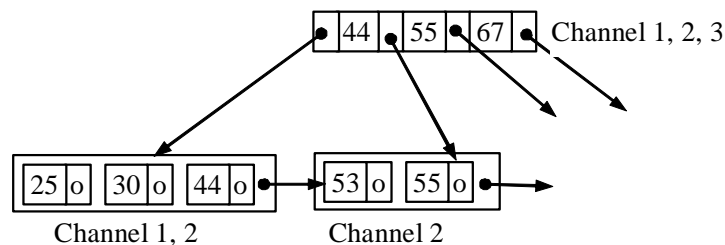


Figure 4.37. Index entry deletion

4.7 Discussion

This chapter has presented efficient data broadcast indexing schemes which are concerned with minimising the client tuning time and power consumption while at the same time optimising the overall query access time for obtaining desired data item in multiple channel environments. The proposed schemes are classified based on the type of queries involved namely: Traditional Queries, and Location-Dependent Queries.

- Traditional Queries

To accommodate this type of query, this thesis introduces three broadcast indexing schemes namely: (i) Non-replicated indexing scheme (NRI), (ii) Partially-replicated indexing scheme (PRI), and (iii) Fully-replicated indexing scheme (FRI). These three indexing schemes are designed based on $B+$ tree structure. The performance indicators include index access time, client's tuning time and power consumption. The simulation experiments have been extensively carried out. There are three cases for consideration whose aim is: (i) to compare the performance of NRI, PRI and FRI in the context of one and two data items retrieval, (ii) to study the performance of PRI against conventional method as what suggested by Leong and Si (Leong and Si, 1995), (iii) to analyse the performance of PRI when there are more replications involved in the channel.

Based on the three cases, it can be learnt that the Non-replicated indexing (NRI) scheme offers a slightly lower index access time as compared with the other two schemes. However, this scheme (NRI) requires the mobile client to determine the

right index channel that contains the index of interest. This can be easily solved if the client is able to listen in to more than one channel at any given time, but it also consumes a substantial amount of energy. In the simulation, it is assumed that where NRI is utilized, each mobile client is aware of the index channel directory. Thus, the drawbacks of this scheme are not apparent in the simulation, but they certainly need to be considered.

Otherwise, this constraint can be overcome by either broadcasting the index channel directory prior to the index cycle, or by sending the directory request to the server. The later scenario raised a few issues such as: (a) When a client leaves its cell and enters a new cell, the directory or index cached may need to be updated; (b) new clients in the cell need to obtain an index directory from the broadcast channel. This includes the mobile unit, which was turned off and turned on again. These issues may generate excessive traffic between the clients and the server. Other issues include larger index sizes to incorporate the index channel directory, and high power consumption for clients to send a request to the server. The PRI scheme does not have these drawbacks and most importantly it has a very small fraction of access time difference from NRI. Having noted that PRI has the same behaviour as a single channel system while providing a better access time, it can definitely be considered it as the better option.

A fully-replicated indexing scheme (FRI) provides the largest access time as compared to the other two schemes. However, this scheme might be considered when the channel is vulnerable to noise and signal distortion, as the client can move to the other channel and retrieve the desired data items without substantial disruption.

An extended experiment has been conducted to further analyse the performance of the PRI against FRI in six cases. The six cases for comparison are concerned with varying the index-inter arrival rate and the skew request distribution in terms of one and two indexes retrieval. The results suggest that the PRI (Global Index) scheme outperforms the FRI (Non-Global Index) scheme substantially in every aspect of evaluation.

- Location-dependent Queries

As mobility is one of the unique characteristics of the wireless environment, the kind of information requested is generally location-dependent; that is, the mobile client's location is relevant to the information requested or the information requested is based on a particular location (Lee et al, 2002). This request is generally known as location-dependent queries, and such services can be called *Location Dependent Information Services*. Since the focus is on data broadcasting scheme, such services can be called *Location Dependent Broadcast Services*.

This thesis also presents index architecture, called Global indexing scheme to accommodate location-dependent queries. This architecture is designed to minimize the average access time, tuning time, and power consumption for accessing relevant index items. Considering the advantages of PRI for Traditional Queries, the fundamental architecture of the Global Index for Location-Dependent Queries is adapted and derived from PRI scheme.

In the experiments, the performance of a square valid scope as compared with polygon one is studied. It is found that a square valid scope which incorporates the

Global indexing scheme reduces processing time for clients. The Global index scheme is analysed against the non-global index in which there two cases for consideration namely: balanced and unbalanced tree. For both cases, the Global index provides a substantially better average access time and tuning time performance as well as less power consumption than the non-global index model. As a result, it will be able to conserve a considerable amount of power.

4.8 Conclusion

The indexing scheme is a desirable mechanism to apply in a mobile broadcast environment due to its ability to provide accurate information that enables a client to tune in at the appropriate time for the required data. However, the presence of the index in the broadcast cycle will greatly affect the access time.

The main contributions of this chapter are summarized as follows.

- An efficient Indexing Scheme for Traditional Queries in Multi Broadcast Channel Environment is presented. Three indexing schemes are introduced. These three schemes are: (i) Non-replicated indexing (NRI), (ii) Partially-replicated indexing, and (iii) Fully-replicated indexing scheme (FRI). These indexing schemes incorporate a multi-index channel, and the data items are broadcast separately in data channels.

Extensive simulation models have been developed to analyse the performance of the proposed indexing schemes. The analysis includes a comparison of index access time, client tuning time and power consumption. It is considered that the Partially-replicated indexing scheme (PRI) is the better scheme for

index dissemination over multi-broadcast channels. Although PRI is utilizing multi-index channelling, it behaves like a single channel system. PRI has the ability to maintain a low overall query access time while still having the advantage of utilizing an index to inform mobile clients of when the desired data is to arrive.

- An efficient Indexing Scheme for Location-Dependent Queries in Multi Broadcast Channel Environment is introduced. An indexing architecture for location-dependent queries in multi broadcast channel environment, called global index has been presented. In the simulation experiments, the performance of this scheme has been extensively analysed including the processing time of mobile clients for processing the valid scope, query access time, client's tuning time and power consumption. The performance is compared with non-global indexing scheme in which there are two cases: balanced and unbalanced tree. For both cases, the Global Index provides a substantially better average access time than the Non-Global Index model. Moreover, the results suggest that the Global Index is capable of providing less client tuning time than the Non-Global Indexing scheme. As a result, it will be able to conserve a considerable amount of power, which is of great benefit in the mobile database environment.

Chapter 5

Hybrid Model

5.1 Introduction

In wireless computing, applications such as accessing airline schedules, stock activities, traffic conditions, and weather information using PDAs on the road are expected to become increasingly popular. It is noted, however, that several wireless computers including laptops and palmtops use batteries of limited lifetime for their operations and are not directly connected to any power source. As a result, query response time and energy saving are very important issues to resolve before the potential of wireless computing can be fully realised. In data broadcasting, query response time and energy saving issues are represented by three performance metrics namely: query access time, client tuning and power consumption.

In Chapter 3 of this thesis, strategies to optimise channel utilisation are studied. These strategies are concerned with minimising the query access time for obtaining

on-air broadcast database items. Subsequently, it is necessary to ensure power conservation for the clients through the reduction of client tuning time which also indicates its energy consumption. In light of this issue, Chapter 4 of this thesis presents index broadcasting schemes, which have the ability to provide accurate information for a client to tune in at the appropriate time for the required data. These indexing schemes are designed for minimising client tuning time and power consumption and maintaining the overall query access time at a considerably low level when retrieving broadcast database items in multiple channel environments.

In this chapter, a hybrid model combining the models for optimising query access time, tuning time and power consumption is presented. This hybrid model combines strategies presented in the previous two chapters (Chapters 3 and 4) of this thesis, which is built following the sequence of these two chapters forming a single broadcast system.

The performance of this hybrid model is evaluated using a simulation program as well as real-world implementation (prototype system). The prototype system has been built in a real wireless environment and it is used to represent an experimental test bed of the overall proposed data broadcasting schemes and to measure and study the effectiveness and efficiency of the schemes. The prototype system is also used to validate the results and to ensure the correctness of simulations when larger parameters are applied.

The rest of this chapter is organized as follows. Section 5.2 gives an overview of the hybrid model. Section 5.3 describes the performance evaluations of the hybrid model.

The prototype system of the hybrid model is outlined in Section 5.4. Section 5.5 presents the experimental results which are obtained from the prototype system and simulation program. It is followed by discussions of findings in Section 5.6, and Section 5.7 concludes the chapter.

5.2 Hybrid Model

A hybrid model consists of the strategies that have been presented in Chapters 3 and 4 of this thesis forming a single data broadcast system. This hybrid model is created following a sequence given in Figure 5.1, which also corresponds to the order of chapters in this thesis.

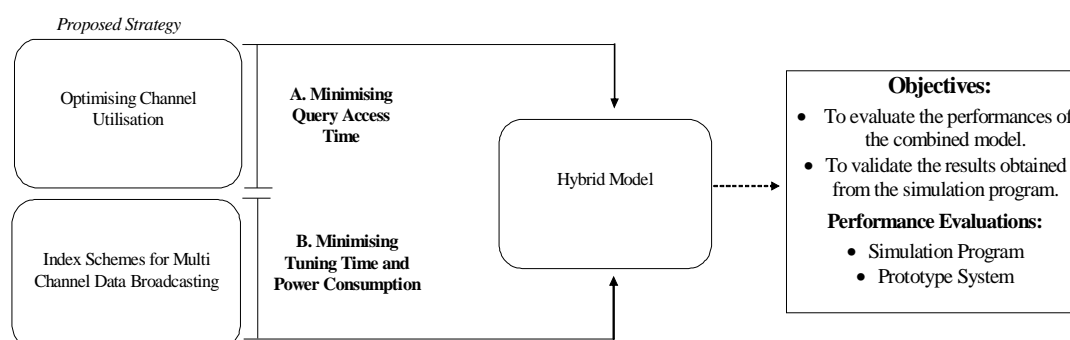


Figure 5.1. Chapter framework

As can be seen from the Figure, it proceeds from Chapter 3.3, where a model for finding the optimum number of broadcast channel is studied. After knowing the number of broadcast channels to broadcast a set of database items, data broadcast ordering and scheduling strategies then need to be applied (see Chapter 3.4). The final broadcast program is subsequently placed in the data channels. These strategies are concerned with minimising query access time when obtaining on air broadcast database items.

Although it attempts to minimise the query access time, clients will still have to consume a substantial amount of power while listening to a number of unnecessary data items. Therefore, it is necessary to ensure power conservation for the clients through the reduction of client tuning time which also indicates its energy consumption. This can be achieved using index broadcasting schemes which let clients know when each of the data items is being broadcast. These schemes enable mobile clients to conserve energy by switching into power saving mode or doze mode and back to active mode when the data is about to be broadcast.

Chapter 4 of this thesis presented data broadcast indexing schemes for minimising client tuning time and power consumption. The index is constructed based on the order of the data item in the data channels, which have been determined based on the strategies given in Chapter 3.

In this chapter, a hybrid model combining the models for optimising query access time, tuning time and power consumption, which have been presented in Chapter 3 and 4 of this thesis, is investigated. This combined model corresponds to the broadcast-based information system that can be utilised by wireless telecommunication providers to offer scalable, effective and efficient data broadcast delivery services to mobile clients. Figure 5.2 depicts the proposed data broadcasting system and its elements.

The hybrid model is illustrated in Figure 5.3, whilst Figure 5.4 depicts the access time and tuning time calculation of the model. Figure 5.4 shows the total access time and tuning time to retrieve data item # 3 as an example, which starts from the time the

client probes into the beginning of the index channel until the desired data item has been obtained. It can be seen from Figure 5.4 that the client is able to conserve power consumption using index information by tuning in only when the desired index or data item is about to arrive in the channel. However, the consequence of having the index is that there is a trade-off between minimizing the tuning time and the query access time. The consequence of minimizing one of them is the increase of the other (Imielinski T., Viswanathan S. and Badrinath, 1997; Chung, 2005).

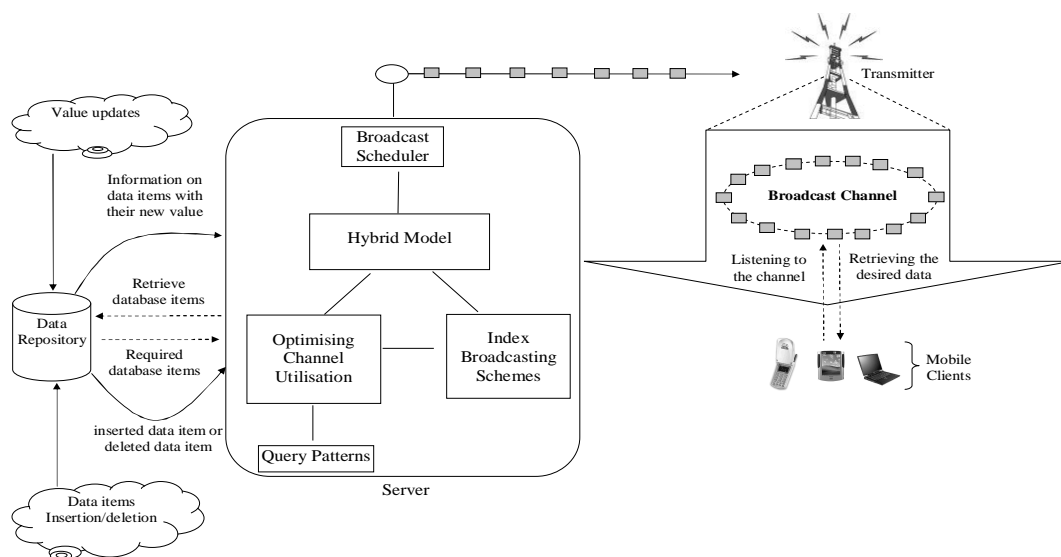


Figure 5.2. Architecture of a broadcast system

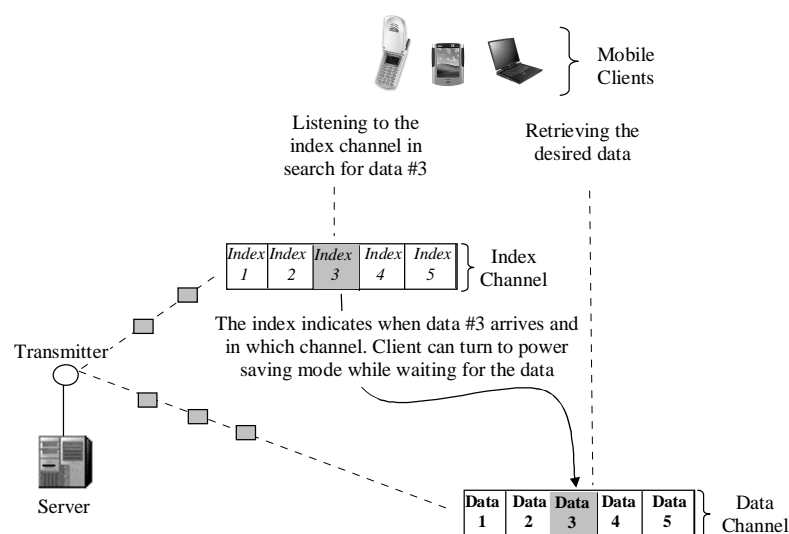
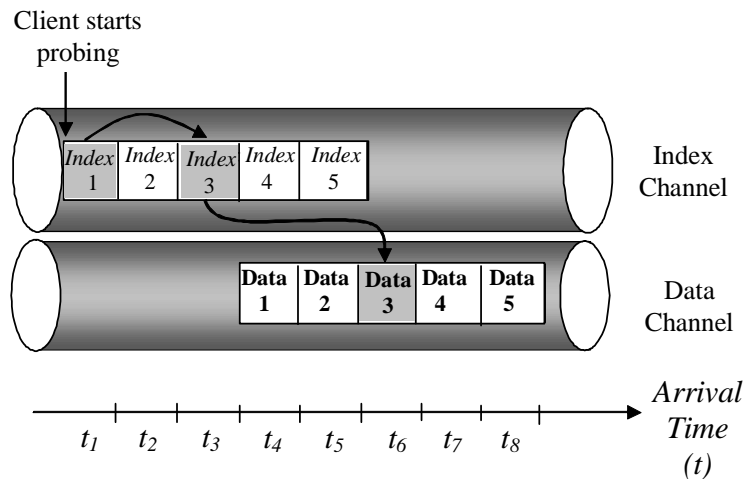


Figure 5.3. Data and index broadcasting in the hybrid model



$$\text{Access time to obtain data \# 3} = \sum_{x=1}^6 t_x$$

$$\text{Tuning time to obtain data \# 3} = t_1 + t_3 + t_6$$

Figure 5.4. Access time and tuning time calculation of the hybrid model

The client processing mechanism in this model can be described as follows:

- Mobile client tunes in to an index channel.
- Mobile client follows the index pointer to the right index key. The pointer may lead to another index channel that contains the relevant index. While waiting for the index to arrive, mobile clients can switch to power saving mode.
- Mobile client tunes back and retrieves the right index key, which points to the data channel that contains the desired data item. It indicates a time value for the data to arrive in the data channel.
- Mobile client tunes into the relevant data channel, and switches back to power saving mode while waiting for the data item to arrive.
- Mobile client switches back to active mode just before the desired data item arrives, and retrieves the information.

5.3 Performance Evaluations

This chapter is divided into two categories of performance evaluations: (i) the prototype system, and (ii) the simulation program.

- Prototype System

The prototype system provides a real-world implementation. It has been built in a real wireless environment and it is used to represent performance measurements of the hybrid model. In the prototype system, the proposed PRI or Global Indexing scheme for traditional queries (see Chapter 4.3) is combined with the broadcast ordering and scheduling scheme (see Chapter 3.4) forming a single broadcast system. The analysis is concerned with the proposed broadcast ordering and scheduling scheme without replication (refer to Chapter 3.4.1), and apply the Global Indexing scheme in order to minimise client's tuning time and power consumption.

As for performance comparisons, an existing data broadcast ordering and scheduling approach as discussed in (Prabhakara et al, 2000) has also been implemented in the prototype system. The existing ordering scheme relates to the basic ordering scheme that allocates the data items in the channel according to the access frequency in a non-decreasing order. The indexing scheme of the existing method follows the B^+ tree index structure, which is broadcast in an index channel. This concept was introduced in (Leong and Si, 1995).

- Simulation Program

The performance evaluation is also carried out using a simulation program. An extended simulation is presented to further analyse the performance of the combined

model of what have been implemented in the prototype system with larger parameters value. The simulation program is outlined in section 5.4a of this chapter and more detailed information about the simulation is available in appendix A of this thesis.

5.4. Prototype Model

This section describes the prototype model of a broadcast-based information system that has been developed for the experimental test-bed. In the model, the broadcast data is retrieved from a central database server and they relate to the share price indices context (see Appendix B).

The hardware technology used for both the client and the server devices is one desktop computer as a server and a notebook computer as a client. The server device is configured with Intel Pentium 4 CPU 2.4GHz , 1 GB RAM, 120 GB HDD. While, the notebook used is a Fujitsu P series Ultra-light Notebook with Transmeta Crusoe™ TM5800 (800MHz) and an integrated WLAN, 256MB SDRAM and 30GB HDD. The server device communicates to the client device over a wireless LAN. The standard wireless Ethernet networking technology 802.11b is utilized. The software technology on both computers uses Microsoft® Windows XP Professional edition as the operating system.

Both applications on the client and server device are programmed in Microsoft® Visual Basic® 6.0 Enterprise Edition. Microsoft® Visual Basic® 6.0 is an Object-Orientated Event-Driven high-level programming language. The server application uses version 2.7 of the Microsoft® ActiveX Data Objects (ADO) in Visual Basic® 6.0 to enable connectivity with the data source, Microsoft® Access® database.

Within every ADO, a structured query language (SQL) statement is used in the server application to access and manipulate the data stored in the database (Dietel et al, 1999). The Microsoft® Winsock control provides a standard application programming interface (API) to enable wireless communication in a UDP/IP network (Jones, Ohlund, 2002).

The prototype model implements a connectionless communication via UDP/IP by the standard application programming interface (API), Microsoft® Winsock Control. The Winsock control uses the IPv4 transport. IPv4 is the network protocol that is used by the Internet. It is used to assign a 32-bit address to each computer and to support unicasting, multicasting and broadcasting of data (Jones, Ohlund, 2002).

The design of the model comprises of three specific components. The three components are: (1) a data source, (2) a server application, and (3) a client application. The data source component is a simple Microsoft Access® database. The model retrieves information from the database. All communications are routed through a server component. The data source and server components are located on the same server device. The server component is an application that functions as a mediator between the data source and client application. Figure 5.5 illustrates these components and their relationships. The client application filters out broadcast information from the server.

The client processing mechanisms for the prototype model are illustrated from Figure 5.6 to 5.9. Figure 5.6 shows the initialization phase for the clients to select desired share prices. The system detects the required share prices with the relevant

identification numbers and tunes into an index channel (see Appendix B.2). In the index channel, the client receives an index key and follows the pointer containing the required identification number. The pointer may lead to another index channel that contains the relevant index. The client completely disconnects for a certain time interval indicated in the index key and deliberately does not receive index keys which are not related to the required identification number (see Figure 5.7).

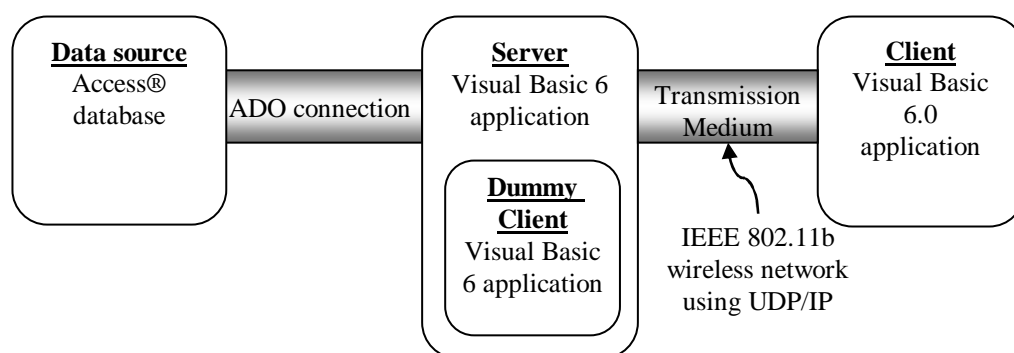


Figure 5.5. Prototype Components

The client reconnects after the time interval has lapsed. If an index key is still to be received, then back to the earlier step. Otherwise the client receives the required index that has a pointer to the desired share prices (see Appendix B.2). The client partially disconnects for a certain interval indicated in the index key. The client tunes in to the data channel as indicated in the index key and deliberately does not receive share prices which are not related to share prices with the required identification number (see Figure 5.8). The client reconnects after the timer interval has lapsed and receives the share price (see Figure 5.9).

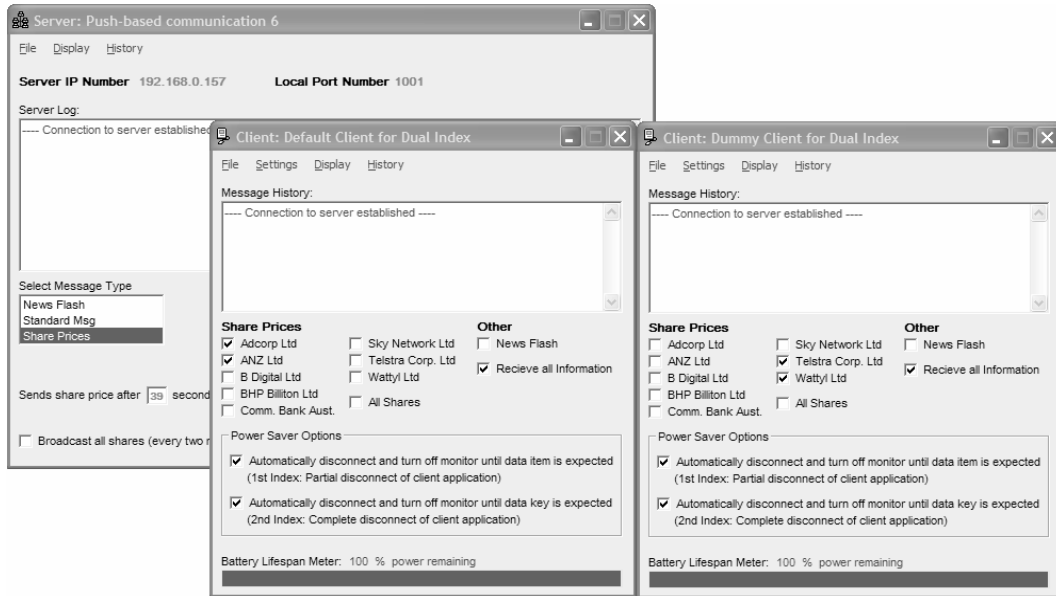


Figure 5.6. Initialisation Phase

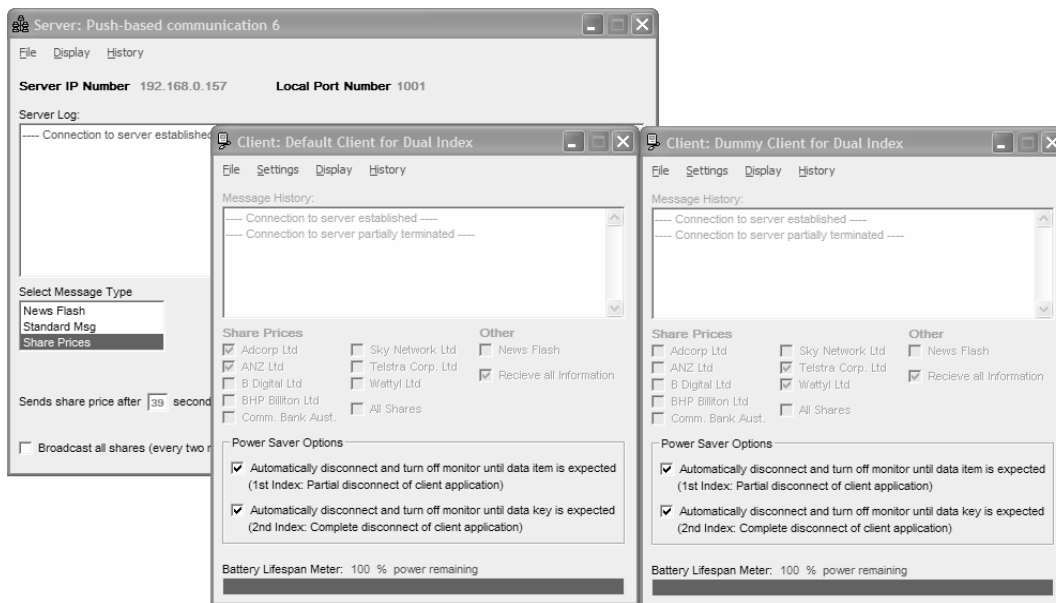


Figure 5.7. Client in power saving mode

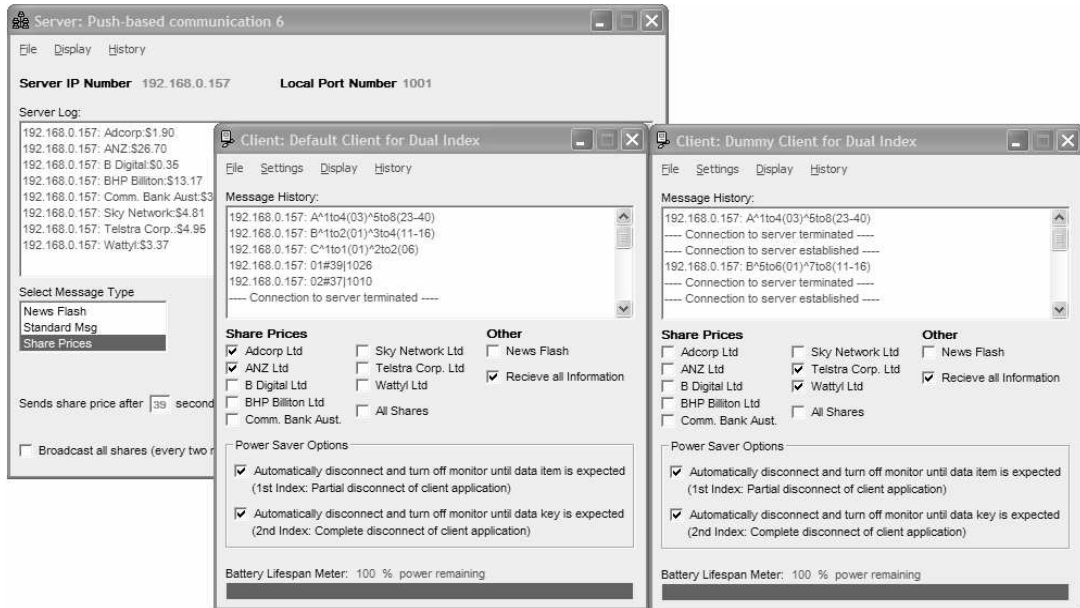


Figure 5.8. Client receiving index information and partial disconnection

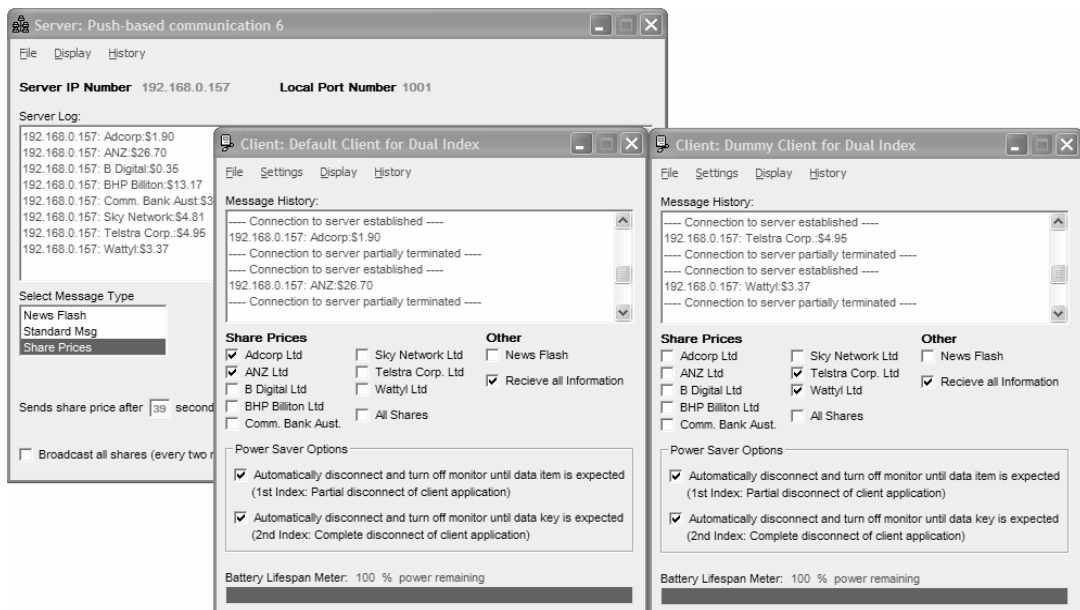


Figure 5.9. Client obtaining the selected share prices

5.5 Experimental Results

In this section, the experimental results are presented, which are obtained from simulation as well as real-implemented (prototype) experiments. The performance of

the proposed method is analysed and compared with the existing one. In this existing scheme, the data items are ordered and broadcast based on the access frequency without considering the relationship with other data items. This existing data ordering method was discussed in (Prabhakara et al, 2000). The indexing scheme of the existing method follows the B^+ tree index structure, which is broadcast in an Index channel.

Subsequently, the performance results derived from the simulation program against the prototype system are compared. In this evaluation, the statistical patterns of users who start listening or probing into the channel follows the behaviour of Gaussian distribution and each query pattern has equal access probability. The index structure is broadcast following a top-down approach and the index distribution model employs a non-replication model similar to that introduced in (Imielinski, Viswanathan and Badrinath, 1997). Table 5.1 shows the parameters of concern for these experiments.

The prototype system utilises the share price indices context for data broadcast services. The data content is retrieved from a data source, Microsoft® Access® database. The table stores records of the share prices details including company name, ASXcode, abbreviation, category, and price. The size of each record amounts to about 420 Kb. For simplicity, there are 8 records altogether and 5 query patterns in accessing the records. Each query pattern may request up to 4 share prices at the same time.

The bandwidth is determined from the standard bandwidth for IEEE802.11b. It is assumed that the optimum number of broadcast channels is 2 channels. This can be determined following the strategy in Chapter 3.3 of this thesis. The number of requests ranges from 10 to 30. The index node pointer follows the syntax given in Appendix B.2. The size of the pointer, as well as the index attribute, is determined from the table based on the database design as provided in Appendix B.1. The index node interval can be varied. In this case the interval is set to 0.2 sec per node.

Table 5.1. Parameters of concern – hybrid model

Parameters	Value
Size of data Items	420 Kb
Bandwidth	11Mbps
Query Patterns	5
Number of Dependent Items in Query	1-4
Number of Broadcast Channel	2
Number of Broadcast Data Items	8
Number of Requests	10-30
<i>Global Index</i>	
Node Pointer Size	24-32 bytes
Data Pointer Size	15 bytes
Indexed Attribute Size	8 bytes
Index Arrival Rate	1 index node per 5 sec interval
<i>Non-Global Index</i>	
Node Pointer Size	19-22 bytes
Data Pointer Size	10 bytes
Indexed Attribute Size	8 bytes
Index Arrival Rate	1 index node per 5 sec interval

a. Simulation Model

In this simulation, two software packages are deployed, namely Visual Basic 6.0 and *Planimate* or animated planning platforms (Seeley, 1997). The algorithm to determine the best broadcast order is coded in Visual Basic 6.0. The simulation is carried out using *Planimate* simulation tool. The simulation environment is set to

apply exponential distribution for index and data item inter-arrival rate given an average value. The simulation is run for thirty iterations, and derives the average result accordingly. In the query patterns, it considers requests that queried up to four numbers of broadcast items, which is reflected in the number of dependent items in the query.

b. Performance Analysis

The analysis involves three cases. The *first* case is to compare the query access time of the proposed method with the existing method. The comparison analysis is carried out using a simulation program and prototype system. The performance results of the two schemes are then provided. In the *second* case, the tuning time of Global Index in the proposed method is evaluated against Non-Global Index in the existing method. The results are based on the prototype system. The *third* case relates to determining the average power consumption of mobile clients for listening to the channels. This last case is related to the client's tuning time as indicated in the second case.

Case 1: As can be seen from Figure 5.10, the simulated-based query access time performance of the proposed method outperforms the existing method. The graph indicates that the proposed method provides a better access time compared with the existing approach by about one and a half times lower access time. The prototype based query access time performance as shown in Figure 5.11 confirms the superiority of the proposed method. The simulation result provides a slightly higher average access time for each measurement as compared with the prototype results.

The graph comparison of the query access time performance obtained from simulation and prototype experiments are given in Figure 5.12. It can be seen that the results from the two types of experiments are very close to one another. This proves the confidence of the accuracy of the simulation model. A more detailed comparison can be found in Table 5.2. It is indicated that the simulated results have an average of 5.572% error in comparison with the prototype ones.

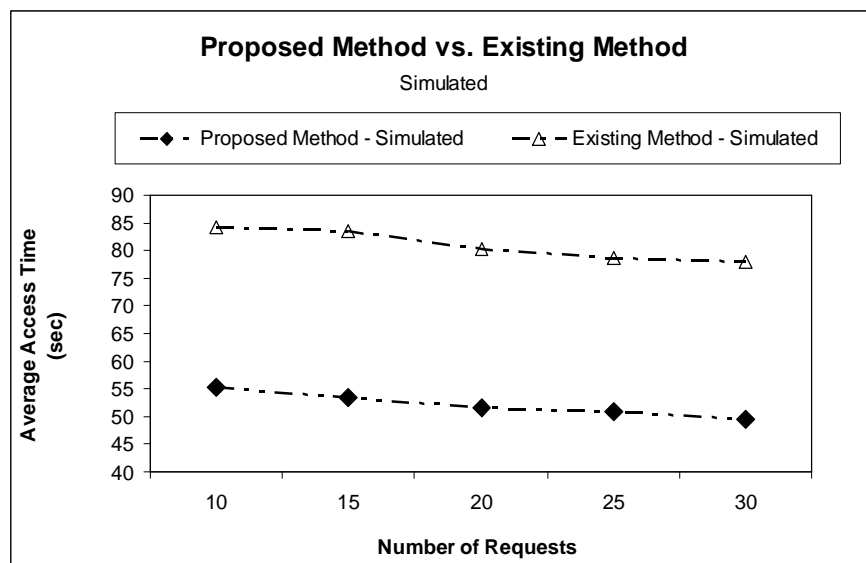


Figure 5.10. Query access time: proposed vs existing method (simulated)

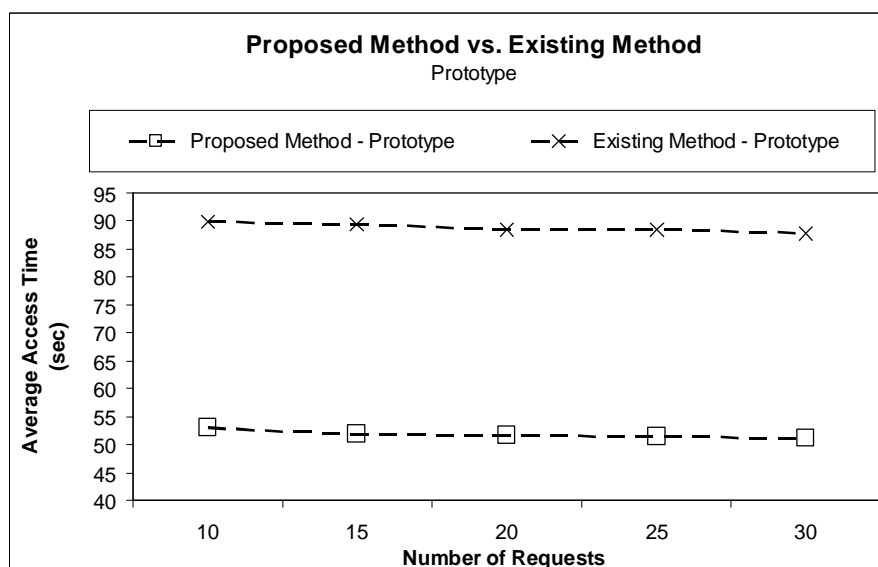


Figure 5.11. Query access time: proposed vs existing method (prototype)

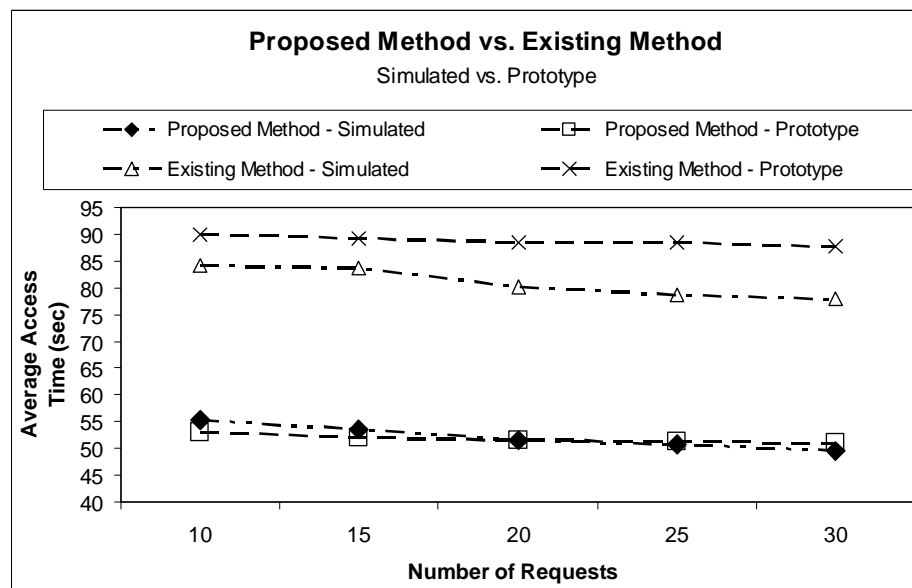


Figure 5.12. Simulated vs prototype: results comparison

Case 2: The amount of the client's tuning time is depicted in Figure 5.13. It shows that the tuning time of clients with Global Index in the proposed method is substantially lower than for the Non-Global Index in the existing method. Based on this result, one can calculate the power consumptions of mobile clients accessing the desired data items. To obtain the client's tuning time, it is necessary to determine the client's processing time per Index node. In simulation experiments, it is assumed the processing time per Index node is equal to the size of the Index over the bandwidth size per second. In this case, the processing time is derived from the prototype model. The results are averaged from 10 iterations and the client's processing time per Index node is found to be 0.0019 sec, which is fairly close from the formula used for the simulation experiments.

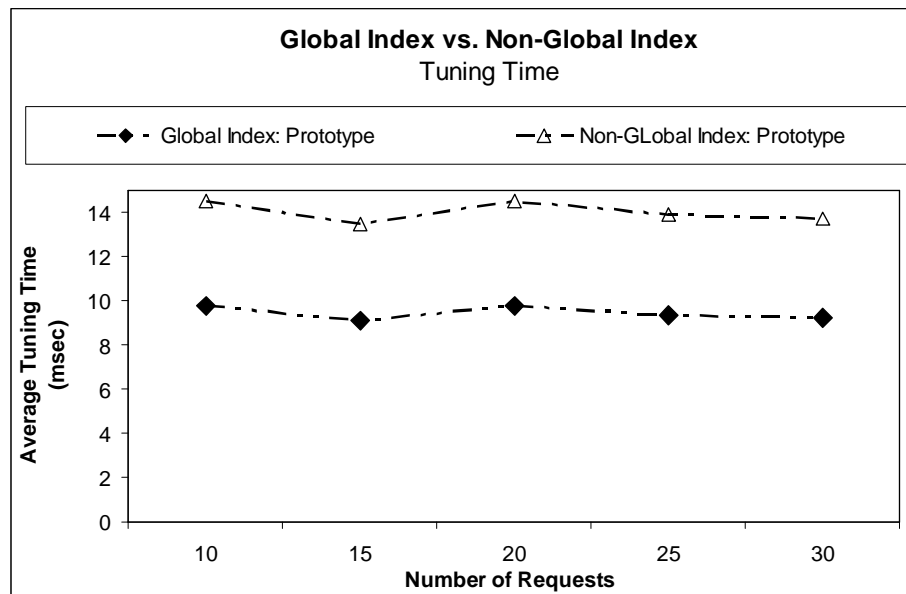


Figure 5.13. Tuning time: global index vs non-global index (prototype)

Case 3: Similar to the simulation experiments, the power consumption is measured based on the formula given in Imielinski et al (1997). $\text{Power Consumption} = (250 \times \text{Time during active mode}) + (0.05 \times \text{Time during power saving mode})$. Figure 5.14 shows the power consumption comparisons between the Global Index and Non-Global Index methods.

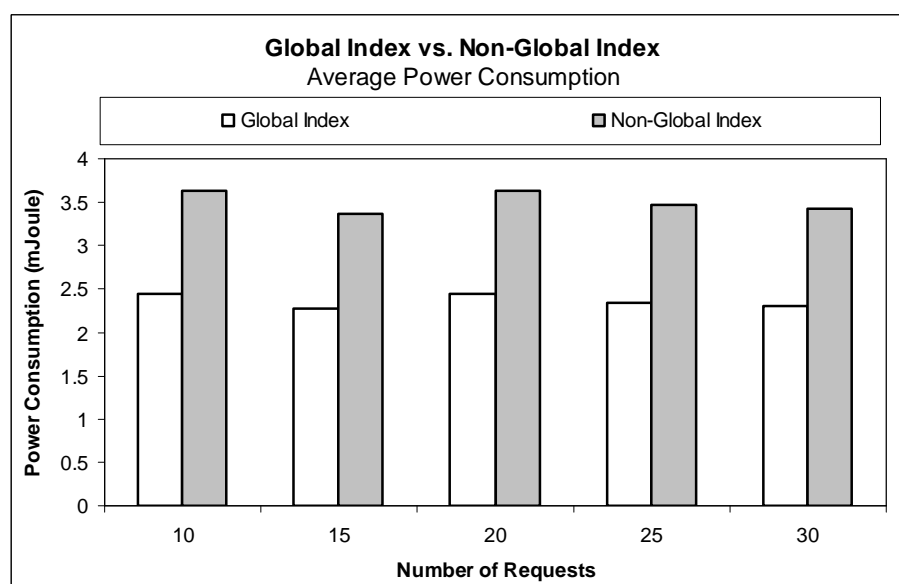


Figure 5.14. Power consumption: proposed vs existing method

Table 5.2. Simulated vs prototype experimental results

Average Access Time (s) Simulated vs Prototype System					
	Average Access Time (s)				
Number of Requests	10	15	20	25	30
Proposed Method (Simulated)	55.25	53.48	51.53	50.92	49.58
Proposed Method (Prototype)	52.98	51.95	51.67	51.32	51.13
Existing Method (Simulated)	84.14	83.58	80.21	78.67	77.92
Existing Method (Prototype)	89.88	89.31	88.53	88.42	87.73
Average Error Rate	5.572%				

5.5.1 Extended Simulation

To obtain a better knowledge of each method's performance behaviour, the relevant parameters are varied and extended into a larger scope. The broadcasting of the index structure in this case follows a breadth-first order. Similar to the previous experiments, the simulation is also carried out using two software packages, namely, Visual Basic 6.0 and *Planimate* or animated planning platforms (Seeley, 1997). The simulation environment is also set to apply exponential distribution for data and index inter-arrival rate given an average value. In the query profile, it considers requests that return one and up to four numbers of items. The parameters of concern are given in Table 5.3. The channel bandwidth is set to 64Kbps which follows the EDGE standard (Nokia, 1999).

The performance of the proposed method is studied in four cases. The first two cases are concerned with the *query access time* performances between the proposed

method and the existing as well as the conventional method; the third case and fourth case relate to *client tuning time* and *power consumption* respectively.

Table 5.3. Parameters of concern – extended simulation

Parameters	Value
Size of each data Item	2KB
Bandwidth	64Kbps
Query Patterns/Profiles	10
Number of Dependent Items in Query	1-4
Number of Broadcast Channel	3
Number of Broadcast Data Items	30 and 45
Number of Request	30
Client's processing time per index node	0.0019 sec
<i>Global Index</i>	
Node Pointer Size	24-32 bytes
Data Pointer Size	15 bytes
Indexed Attribute Size	8 bytes
Index Arrival Rate	4 index nodes per sec
<i>Non-Global Index</i>	
Node Pointer Size	19-22 bytes
Data Pointer Size	10 bytes
Indexed Attribute Size	8 bytes
Index Arrival Rate	4 index nodes per sec

Case 1: In this case the proposed method is compared with the conventional method. A conventional method is when the data items are broadcast without any specific order. The index structure in the conventional method follows B^+ tree index structure and is broadcast in a single index channel. Two analyses are considered, which involve 30 and 45 broadcast items.

In Figure 5.15(a), it can be seen that the proposed scheme outperforms the conventional method with more than two times lower than the average access time.

The number of broadcast items is also modified, and found from Figure 5.15 (b) that both access times improve accordingly. However, it should be noticed that the increase of average access time for the proposed method is far less than the conventional method. This is due to the ordering scheme that has been applied.

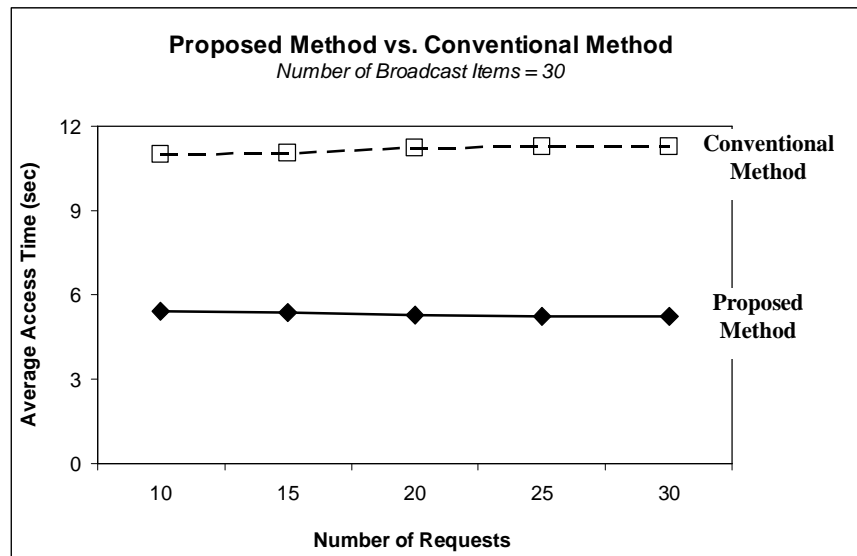


Figure 5.15(a). Proposed method vs conventional method: 30 number of broadcast data items

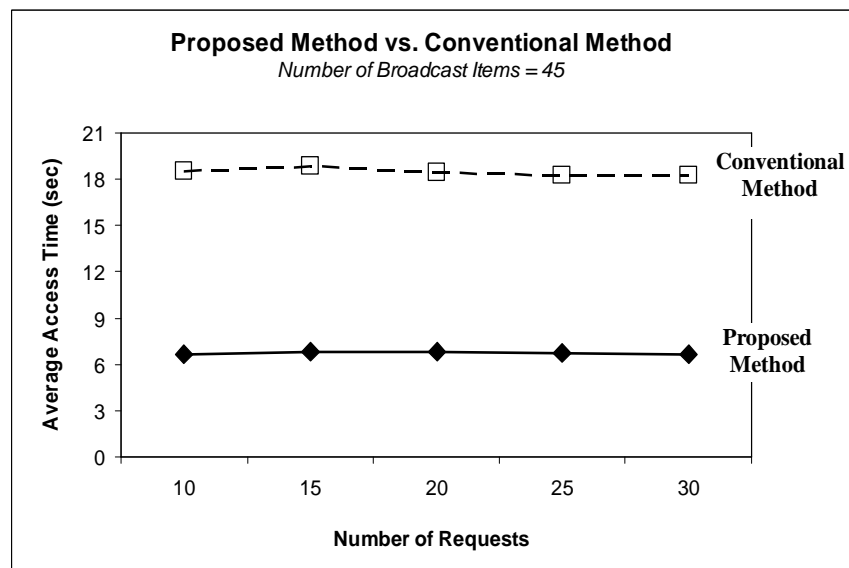


Figure 5.15(b). Proposed method vs conventional method: 45 number broadcast data items

Case 2: The existing ordering scheme relates to the basic ordering scheme that allocates the data items in the channel according to the access frequency in a non-decreasing order as what presented in (Prabhakara et al, 2000). The index structure is broadcast in an index channel. The analysis involves 30 and 45 broadcast items.

From Figure 5.16(a), it shows that the proposed method provides a better access than the existing method with about one and half times lower access time. A similar trend occurs when the number of broadcast items increased (see Figure 5.16(b)). It is shown that the increase of broadcast items severely affects the existing method, but not as much as the conventional one. On the contrary, the access time of the proposed scheme with the Global Index rises just a few seconds or can be considered as a minor increase. Consequently, the gap between the proposed method and existing method is increasing.

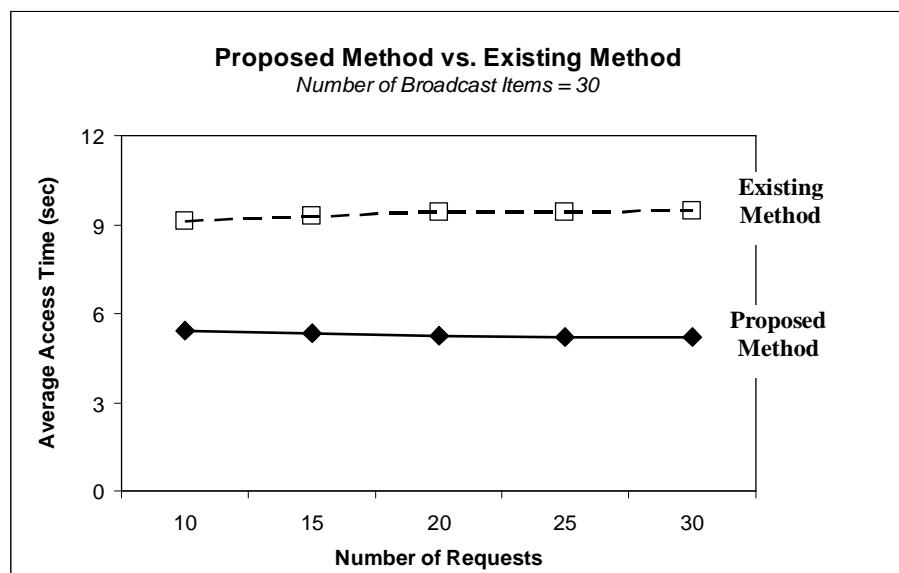


Figure 5.16(a). Proposed method vs existing method: 30 number broadcast data items

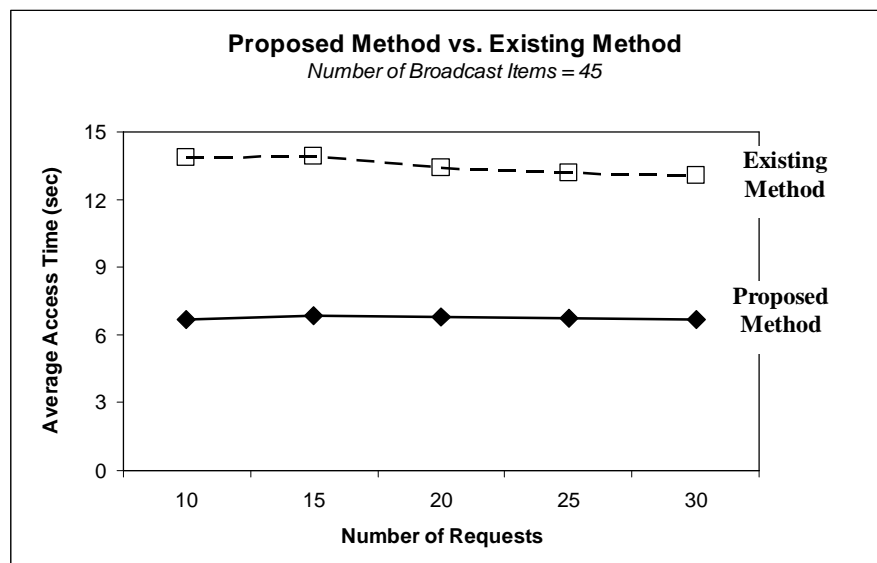


Figure 5.16(b). Proposed method vs existing method: 45 number of broadcast data items

Case 3: This case compares the tuning time of the Global Index with non-Global Index scheme. Non-Global Index refers to an existing tree-indexing based on $B+$ -tree structure, which utilizes a single broadcast channel to broadcast the index structure. The analysis involves 3 broadcast items. It is noted that the tuning time needed to obtain the desired data item is not included in the calculation since both indexes retrieve the same data items, which can be easily incorporated whenever necessary.

As demonstrated in Figure 5.17, the tuning time of mobile clients accessing the Global Index is far less than for the Non-Global Index scheme. More importantly, the result also indicates that clients consume much less power during query operation with the Global Indexing scheme.

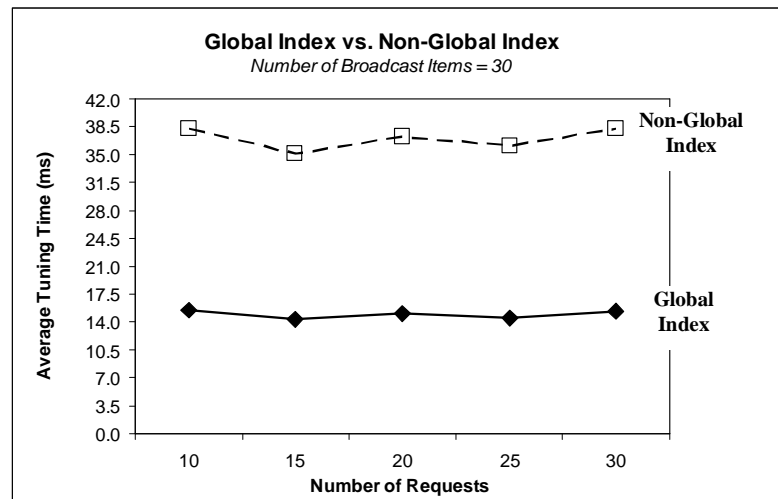


Figure 5.17. Tuning time: Global Index vs Non-Global Index

Case 4: In this case, the power consumption of mobile clients in accessing the Global Index and Non-Global Index are determined also from the formula given in Imielinski et al (1997). Power Consumption = $(250 \times \text{Time during active mode}) + (0.05 \times \text{Time during power saving mode})$. As depicted in Figure 5.18, the power consumption of clients accessing the Global Index is substantially less than for Non-Global Index. With little power consumption, client can reserve battery power more efficiently, which is very much desirable considering the lack of resources in mobile devices.

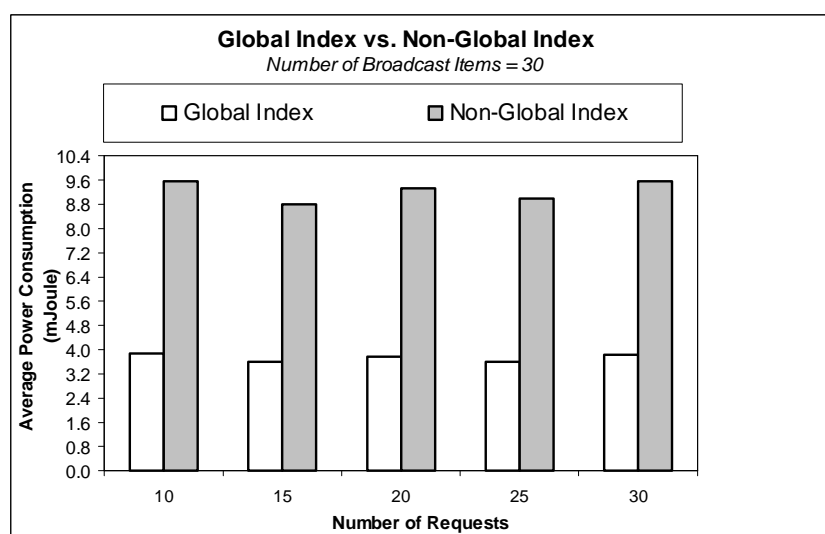


Figure 5.18. Power Consumption: Global Index vs Non-Global Index

5.5.2 Performance Analysis of each Proposed Strategy

In this section, performance results for each strategy in this thesis are provided. The analysis is conducted using a simulation program based on a given set of parameters. The performance metrics for comparison includes query access time, client tuning time and power consumption.

The first strategy is concerned with optimising channel utilisation (see Chapter 3). This strategy considers optimum number of broadcast channels and applies data broadcast ordering and scheduling scheme in order to minimise query access time. However, it does not include any broadcast directory which lets clients know when each of the data items is being broadcast; clients will have to listen or tune in to the channel from the time they probe into the broadcast channel until the desired data items are received. The second strategy is to apply an index broadcasting scheme. A PRI scheme (see Chapter 4.3.2) is selected for this comparison purposes since this scheme is considered as a better scheme as compared to others (see Chapter 4.7). However, this scheme does not involve ordering and scheduling any data items. The last strategy, a hybrid model, combines the PRI scheme and optimal channel utilisation method into a single broadcast system, which is described in this chapter. In this analysis, the simulation settings and the parameters of concern are the same as those applied in the extended simulation (see section 5.5.2) with 30 broadcast data items.

The average results of this analysis are given in Table 5.4. It can be seen from the table that the query access time is the lowest when there is no index directory broadcast with the data items. However, the tuning time of the client listening to the

channel is worse, which is related to the highest power utilisation. When an index broadcasting scheme is applied (PRI), the query access time increases accordingly since clients need to wait for the right index to arrive before obtaining desired data items. However, this scheme does not include the data broadcast ordering and scheduling scheme. Hence, this scheme provides the worse access time although PRI is considered the better scheme. However, the tuning time is minimal. The hybrid model, which integrates optimal channel utilisation method with the PRI scheme, provides the balance between query access time, client tuning time and power consumption performances. The tuning time and power consumption of the hybrid model and the second strategy are the same since both indexes retrieve the same data items.

Table 5.4. Parameters metrics – performance comparison

Performance Metrics	Optimising Channel Utilisation (with no Index)	Index Broadcasting Scheme (PRI)	Hybrid Model
<i>Number of Broadcast Data Items = 30</i>			
Average Query Access Time (sec)	1.9019	7.1465	5.2899
Average Client Tuning Time (msec)	1901.91	680.53	680.53
Average Power Consumption (Joule)	0.475	0.170	0.170

Another thing that can be seen is that the access time ratio of the hybrid model with the first strategy is less than the power consumption ratio between these two. The ratio difference is about 0.01 or 1 percent. Thus, it can be derived that the hybrid model is capable of providing better performances with regard to query access time and power consumption trade-off ratio as compared with the first strategy. However,

the utilisation of each strategy may be selected according to the level of importance of each of the performance indicators for the relevant broadcast-based mobile services.

5.6 Discussion

This section discusses the performance of the hybrid strategy. Two categories performance evaluations are applied: (i) prototype system, and (ii) simulation program. According to the evaluations, the following findings can be derived.

- Based on the prototype system, the results show that the performance of the hybrid model is substantially better than the existing one in every aspect of the evaluations including query access time, tuning time and power consumption. Moreover, it is found that the simulated result is very close to the prototype one that represents real-world performance measurement. This proves the confidence of the accuracy of the simulation model.
- In the extended simulation, the parameters involved in the prototype system are modified to represent a larger scope. The results suggest that the proposed model performs significantly lower for access time, tuning time and power consumption compared with the conventional as well as existing models.
- The performance analysis for each strategy in this thesis suggests that the hybrid model is capable of providing optimal query access time, client tuning and power consumption. The results of this analysis can be utilised as a guideline to determine the most suitable strategy to apply in the relevant data broadcast services. Applications such as stock indices and foreign exchange are probably

more interested to the non-indexing strategy as it requires the shortest access time possible to make a quick decision based on it. Whilst, election result and news broadcasting may not be too much concerned with the response time, thus the second strategy with indexing can be applied. Other applications like weather information or weather forecast services, tourist services, airline schedules, and route guidance may well apply a hybrid model data broadcast system.

5.7 Conclusion

This chapter presented a hybrid model of the combined strategies that were proposed in Chapters 3 and 4 of this thesis. The performance evaluations are carried out using a prototype system and a simulation program. The evaluations include comparisons with existing approaches. The validation of the simulation model using the prototype system has been achieved. An extended simulation has also proven the effectiveness of the proposed data broadcast management schemes.

The main contributions of this chapter are summarized as follows.

- Provides a hybrid strategy forming a single broadcast system. The hybrid model represents the broadcast-based information system that can be utilised by wireless telecommunication providers to offer scalable, effective and efficient data broadcast delivery services to mobile clients.
- Performance evaluations of the hybrid model have been carried out using empirical (simulation program) and practical evaluations (prototype system). Simulation models have also been validated. An extended simulation experiments have been carried out to prove the performance of

the hybrid model in a larger scope of parameters. Further performance evaluation can rely upon the simulation models.

- This chapter completes the proposed data broadcasting strategies (see Figure 5.19). In Chapter 2 of this thesis, a data broadcast management framework has been provided. Several research questions have been raised and they are summarised into two tasks as given in the left boxes in Figure 5.19. The proposed strategies concerning the two tasks have been presented in Chapters 3 and 4 of this thesis, respectively. The performance for each strategy has also been studied.

A hybrid model that combines the strategies of these chapters has been presented and the performance has been analysed. It is found that hybrid model is able to optimise the query access time, tuning time and power consumption of mobile clients for obtaining on-air broadcast database items.

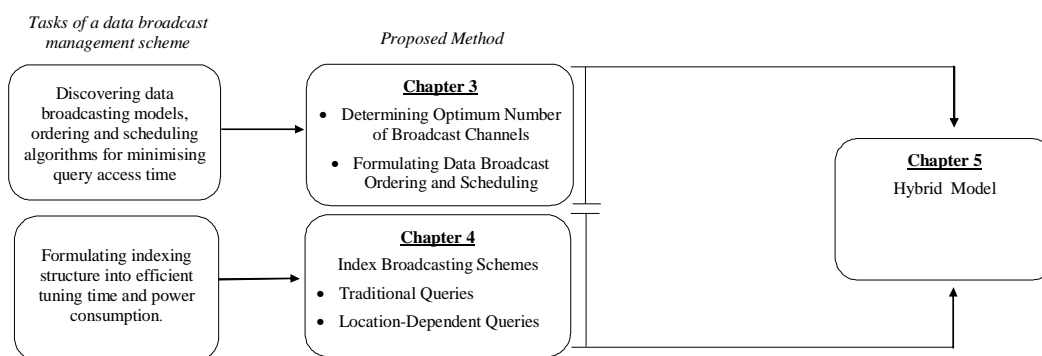


Figure 5.19. A framework of the proposed data broadcasting strategies

Chapter 6

Conclusion

6.1 Background

This thesis investigated data broadcast management schemes designed to achieve high performance data broadcast retrieval and maintain power efficiency. The main aim of this research was to study performance improvement of data broadcast query processing in a multi wireless channels environment. Attention is focused on three major areas of performance improvements: *query access time*, *client's tuning time* and *power/energy consumption*. The investigations include determining a model to obtain an optimum solution for broadcast channels, considering number of channels, number of data items requested, data ordering and replication, index structures, and data/index channel composition. The model is fundamental for optimizing query access and client's tuning times, which will make a significant impact on power consumption. In addition, the performance evaluation of the results has been carried out using a *simulation*, and *prototype* model.

6.2 Summary of the Research Results

The research presented in this thesis has addressed and resolved the outstanding problems of data broadcast query processing management highlighted at the end of Chapter 2. The achievements of this research are summarized as follows.

6.2.1 Minimising Query Access Time

- *Optimum Number of Broadcast Channel*

The analytical models for calculating query access time over broadcast and on-demand channel are presented. These analytical models are applied in order to locate the optimum value of broadcast items. The access time of the on-demand channel is used as a threshold point. This threshold point is utilised to determine the optimum number of channels required to broadcast a set of database items. Consequently, the query access time over the broadcast channel is maintained below the access time over an on-demand channel in any circumstances.

Several parameters are taken into account, which include request arrival rate, service rate, number of request, size of data item, size of request, number of data item to retrieve, and bandwidth. The optimum number of broadcast items is changed dynamically depending on the value of these parameters. The achievements in this section can be summarised as follows: (i) to present analytical models to determine the query access time over on-demand and broadcast channel and (ii) to provide analysis and comparison of the two models in order to determine the optimum number of broadcast channels.

The performance evaluation results show that the analytical models are considerably close to accurate.

- *Data Broadcast Ordering and Scheduling Scheme - without Replication*

The scheme is designed to ensure that the most requested data items stay close to each other in the channel. It orders and schedules the data items in the broadcast channel in such a way that most clients will have a minimum query access time. This scheme is concerned with a flat broadcast program in a multi broadcast channel environment. A flat broadcast program corresponds to the case where the data items appear with the same frequencies. The performance evaluations result shows that the proposed scheme provides a substantially better average query access time as compared with conventional method.

- *Data Broadcast Ordering and Scheduling Scheme - with Replication*

As opposed to the flat broadcast program, this scheme is concerned with the non-flat broadcast program. A non-flat broadcast program corresponds to the case where the data items appear with different frequencies. Generally, the most popular data item will be broadcast more often than others. This scheme considers a certain degree of data replication to broadcast the highest access frequency more often than the others in a multi broadcast channels environment. Based on the performance evaluation results, the proposed scheme is able to provide a substantially better average query access time as compared to conventional method. The result also indicates that clients who begin listening to the channel during the early broadcast cycle will obtain the shortest access time.

6.2.2 Minimising Client Tuning Time and Power Consumption

- *Efficient Indexing Scheme for Traditional Queries in Multi Broadcast Channel Environment*

Three indexing schemes for traditional queries in a multi broadcast channel environment are presented. These three schemes are: (i) Non-replicated indexing (NRI), (ii) Partially-replicated indexing, and (iii) Fully-replicated indexing scheme (FRI). These indexing schemes incorporate a multi index channel, and the data items are broadcast separately in data channels.

The analysis of the three indexing schemes includes index access time comparison, client's tuning time and power consumption. It is considered that the Partially-replicated indexing scheme (PRI) is the better scheme for index dissemination over multi broadcast channels. Although PRI is utilizing multi index channelling, it behaves like a single channel system. PRI has the ability to maintain a short overall query access time, while still having the advantages of utilizing an index to inform mobile clients of when the desired data will arrive.

- *Efficient Indexing Scheme for Location-Dependent Queries in Multi Broadcast Channel Environment*

A novel indexing architecture for location-dependent queries in multi-broadcast channel environment, called the Global indexing scheme is presented. This scheme is designed based on the concept of the PRI-model similar to that which was proposed for the traditional queries environment. The Global indexing scheme for answering traditional queries is modified to serve location-dependent

queries in a mobile broadcast environment. In the performance evaluation, it is found that the square valid scope, which is incorporated into the Global indexing scheme, provides less processing time for clients. Subsequent analysis includes a comparison of the proposed global index against non-global index in which there are two cases: balanced and unbalanced tree. For both cases, the global index provides a substantially better average access time than the non-global index model. As a result, it will be able to conserve a considerable amount of power, which is of great benefit in the mobile database environment.

6.2.3 Optimising Query Access Time, Client Tuning Time and Power Consumption

A hybrid model combining the models for optimising query access time, client tuning time and power consumption has been presented. A prototype system has been built in a real wireless environment and it is used to represent an experimental test bed of the hybrid model and to measure and study the effectiveness and efficiency of the schemes. The evaluation of this model includes a comparison with conventional as well as existing schemes. The performance results suggest the superiority of this model in every aspect of the evaluation. The prototype system is also used to validate the simulation results and to ensure the correctness of simulations when larger parameters are applied. The outcome of this investigation can be employed by wireless telecommunication providers to offer scalable, effective and efficient data broadcast delivery services to mobile clients.

6.2.4 Performance Evaluation

Simulation performance evaluation was carried out to demonstrate the efficiency of the proposed procedures. The cost models for calculating the performance of data broadcasting as well as data on-demand access are corroborated by simulation. A prototype system for the experimental approach of the hybrid model is able to strengthen the simulation results. Through these evaluations, the quantitative models are demonstrated to be highly valuable in representing the behaviour of broadcast-based data management schemes in a multi-channels environment.

6.3 Future Research

This section discusses some possible investigations that can be done in order to enhance the broadcast data management schemes that have been presented in this thesis.

- **Broadcast Management for Transitive Queries**

This thesis focuses on broadcasting a single entity type for the selective retrieval of database items. However, since most real-world application databases involve multiple entity types, transitive queries that access related data items belonging to different tables, are very common. For example, in a relational database (RDB), join operations are required to relate data items from different relations. The term entity type relates to a generic collection of data items with the same logical structure. It is desirable to extend the work in this thesis to consider transitive queries or queries that access related data items belonging to different entity types. The difficulty of the

server is to decide the broadcast order in advance, even without much knowledge of any query in the future. The complexity includes the modification of the relevant indexing scheme to include an integrated pointer to joint attributes from two or more entity types.

- **Multi Attributes Index Broadcasting**

This thesis is concerned with single attribute index broadcasting. It is definitely of interest to modify the index structure in order to accommodate multi-index attributes. Thus, it will be able to support mobile clients to find the data item efficiently using an alternative search key or combination thereof. For example, as in traditional queries, mobile users are enabled to initiate queries so as to retrieve all stock indices where the stock price is below \$50. Similarly, for location-dependent queries, it is desirable to accommodate constrained location-dependent queries; such queries include “find the nearest motels at cost less than \$50 per night”. This will require a more complex and vigorous mechanism.

- **Adaptive Broadcast Channels**

An adaptive broadcast channel is utilized in conjunction with a multiple broadcast channel mechanism. The distribution of data items over multiple channels with different distribution patterns enables a mobile client to switch among channels and retrieve the desired data (Leong and Si, 1995). This technique not only helps a mobile client to retrieve data item during signal distortion or channel failure, but also enhances the query response time.

With an adaptive broadcast channel, the number of broadcast channels is set to be dynamic. This technique is effective in handling an ever-changing environment. For example, when there are a very large number of mobile users in a cell, the number of data items of interest (hot items) increase, which subsequently increases the length of the broadcast cycle. This situation may cause some mobile clients to wait for a substantial amount of time before receiving the desired data item even with the help of different patterns of data distribution over multiple channels.

An adaptive broadcast channel is expected to learn the environment and broadcast the data items over a dynamic number of channels. Consequently, when there are an enormous number of broadcast items, the number of channels can be adjusted accordingly. On the other hand, when the number of broadcast items is small, a dynamic broadcast channel will shrink the number of channels, so the resource utilization is preserved.

An algorithm to find the optimal number of broadcast channels in a particular point in time depending of the number of users in a cell and query access patterns, needs to be investigated. The algorithm should be able to determine the best organization of data. Subsequently, the data is replicated over a dynamic number of broadcast channels with possibly a different data organization or different distribution schedule. This technique is expected to improve the response time for the client to retrieve the required data items.

- **Simultaneous Data Retrieval over Multi Channels**

One of the advantages of a data broadcasting scheme, which separates the index and the data items over multiple channels is that it will overcome the expected delay that

occurs from a broadcasting index directory, since the frequency of broadcasting the index affects the query performance. However, in a normal situation, when more than one channel is involved, the client has to move between channels to obtain the consecutive data items. To prevent a situation where a client needs to move from one channel to another especially when it involves several channels, a further investigation of a means to receive data simultaneously over multiple channels is necessary.

With a simultaneous data retrieval mechanism, a client can just tune in to more than one channel and filter the required data items in a single operation. This situation reduces the delay of missing the data items of interest, and reduces the overhead of switching channels, which results in an improved response time. However, some evaluation needs to be conducted since the operation may be at the cost of high power consumption.

- **Caching Management for Mobile Queries**

Data caching in mobile databases enables clients to obtain a high computing speed as server, by involving a much smaller volume of data items. Furthermore, as a wireless communication channel in general suffers from narrow bandwidth while it is also inherently asymmetric communication, *caching* of frequently accessed data items in a client's local storage becomes important for enhancing the performance and data availability of data access queries. Another advantage of caching is the ability to handle fault tolerance. This is related to the characteristics of mobile computing in which each mobile client connects or disconnects from the network frequently. In some situation, the MBS may not be accessible due to problems such as signal

distortion but a query can still be partially processed from caches and at least some of the query results can be returned to the user (Godfrey and Gryz, 1997).

A caching management strategy relates to how the client manipulates and maintains the data in its *cache* efficiently and effectively. In order to achieve this, a number of issues have to be considered and these issues require investigation. Some of these issues include *caching granularity*, *caching coherence strategy*, and *caching replacement policy*.

Caching granularity is used to find a suitable physical form of data to be cached. Data caching granularities include tuple caching, page caching, attribute caching, object caching, and hybrid caching. Data granularity broadcast by the server corresponds to data caching granularity of a mobile client. As for page caching, the locality that is favourable for one client may not be the same for another. Thus, the task of the server is to determine the data organization so the locality favors most of the clients.

Caching coherence strategy is defined to ensure valid cached items. Any updates of a cached item, should correspond to relevant mobile clients. It is important for mobile clients to have an accurate value for the cached item at any time. The update strategy can be initiated by the mobile client itself or by waiting for broadcast notification from the server to update the cache. The problem with notification is the synchronization between the client and the server. Since mobile clients are often disconnected, they have to re-synchronize every time they switch back on. A periodic broadcast report requires mobile clients to keep tuning into the channel.

Consequently, it consumes a substantial amount of energy. This is not the case with the update strategy initiated by mobile clients, as they are responsible with their own cache. The main concern of this strategy is to determine a refresh time for the cached items. It is difficult to find an accurate refresh time. If the refresh time is too short, too much energy is wasted in communicating with the server. On the other hand, if the refresh time is too long, the cached items may no longer be valid.

Due to cache storage limitation, mobile clients need what is called a *caching replacement policy*. The replacement policy is used to discard old cached items that are no longer relevant to current queries initiated by mobile clients. In other words, the caching replacement policy manages to retain only the most frequently accessed data items. The ultimate concern of this policy is to predict or determine the data items in the memory cache that will no longer be accessed in the near future. Therefore, it creates a space for new frequently accessed items to be accommodated within the cache memory. An inadequate replacement policy results in a waste of energy, as mobile clients need to contact the server in order to obtain the desired data.

- **Hybrid method of broadcasting, caching and pull based request**

The combination of caching, broadcasting and on-demand methods can constitute an effective technique to enhance the query performance. The caching technique provides speedy data retrieval since the data is available in local storage. However, a mobile unit has a limited capacity for storing data items. Thus, only a fraction of the desired data items can be cached. With the complement of a broadcasting technique, the caching problem is removed since clients can tune in to the channel to pick up

data of interest when the data is not available in the cache. Nevertheless, the quantity of data broadcast is changing based on the query access patterns. Therefore, the response time for clients' data retrieval may be severely affected.

As an alternative, mobile clients may send a direct request to the server when it is made aware of when the desired data items will appear in the broadcast channel. For example, a threshold as proposed in (Lee, Hu and Lee, 1997) can be employed to make a selection between broadcast and on-demand service. The indexing technique is used to indicate the time of the next desired data items to arrive in the broadcast channel; if the data item appears within the next threshold number of data frames, the client can choose to stay tuned in the channel; otherwise an on-demand service is required.

The possible drawback with this technique is power utilization. This technique enables the speeding up of data retrieval but at some point it may not be energy efficient since a number of checks may be needed before it can decide how the data is to be obtained. Thus, a thorough investigation is required.

References

- Abowd D. G., Atkeson G. C., Hong J., Long S., Kooper R. and Pinkerton M., "Cyberguide : A Mobile Context-Aware Tour Guide", *Wireless Networks*, **3**(5):421-433, 1997.
- Adya A., Bahl P., Qiu L., "Analyzing the Browse Patterns of Mobile Clients" In *Proceedings of the 1st ACM SIGCOMM Workshop on Internet Measurement*, pp.189-194, 2001.
- Acharya S., Alonso R., Franklin M. and Zdonik S., "Broadcast Disks: Data Management for Asymmetric Communication Environments", In *Proceedings of ACM Sigmod*, pp.199-210, 1995.
- Acharya S., Alonso R., Franklin M. and Zdonik S., "Prefetching from a broadcast disk", In *Proceedings of the International Conference on Data Engineering (ICDE)*, pp.276-285, 1996.
- Acharya S. and Muthukrishnan S., "Scheduling On-Demand Broadcast New Metrics and Algorithms", In *Proceedings of ACM/IEEE International Conference on Mobile Computing*, pp.43-54, 1998.
- Acharya S., Franklin M. and Zdonik S., "Balancing Push and Pull for Data Broadcast", In *Proceedings of ACM Sigmod Conference*, pp.183-194, 1997.
- Aksoy D. and Franklin M., "Scheduling for Large-Scale On-Demand Data Broadcasting", In *Proceedings of IEEE InfoCom Conference*, pp.651-659, 1998.
- Aksoy D., Altinel M., Bose R., Cetintemel U., Franklin M., Wang J., and Zdonik S., "Research in Data Broadcast and Dissemination", In *Proceedings of 1st International Conference on Advanced Multimedia Content Processing*, LNCS, **1554**:194-207, 1999.

- Ammar M., and J. Wong, "The Design of Teletext Broadcast Cycles", *Perform. Eval.*, **5**(4): 235-242, 1985.
- Argade, P.V., Aymeloglu, S., Berenbaum, A.D., dePaolis, M.V. Jr., Franzo, R.T., Freeman, R.D., Inglis, D.A., Komoriya, G., Lee, H., Little, T.R., MacDonald, G.A., McLellan, H.R., Morgan, E.C., Pham, H.Q., Ronkin, G.D., Scavuzzo, R.J., Woch, T.J. "Hobbit: a high-performance, low-power microprocessor", In *Proceedings of COMPCON*, pp.88-95, 1993.
- Asthana A., Cravatts M. and Krzyzanowski P., "An Indoor Wireless Systems for Personalised Shopping Assistance", In *Proceedings of IEEE Workshop on Mobile Computing Systems and Applications*, pp.69-74, 1994.
- Barbara D., "Mobile Computing and Databases-A Survey", *IEEE Transactions on Knowledge and Data Engineering*, **11**(1): 108-117, 1999.
- Barbara D. and Imielinski T., "Sleepers and Workaholics: Caching Strategies in Mobile Environments (Extended Version)", In *Proceedings of ACM Sigmod International Conference on Management of Data*, pp.1-34, 1994.
- Bar-Noy A., Naor J., and Schieber B., "Pushing Dependent Data in Clients-Providers-Servers Systems", In *Proceedings of the 6th ACM/IEEE on Mobile Computing and Networking*, pp. 222-230, 2000.
- Beckmann N., Kriegel H., Schneider R. and Seeger B., "An Efficient and Robust Access Method for Points and Rectangles", In *Proceedings of ACM Sigmod*, pp. 322 - 331 , 1990.
- Berg M., Kreveld M., Overmars M. and Schwarzkopf O., *Computational Geometry*, Springer-Verlag, New York, 1997.
- Bowen T. F., Gopal G., Herman G., Hickey T., Lee K. C., Mansfield W. H., Raitz J. and Weinrib A., "The Datacycle Architecture", *Communications of the ACM*, **35**(12): 71-81, 1992.
- Bria A., Gessler F., Queseth O., Stridh R., Unbehaun M., Wu J., and Zander J., "4th-Generation Wireless infrastructure: Scenarios and Research Challenges", *IEEE Personal Communications*, **8**(6):25-31, 2001.
- Cai Y. and Hua K. A., "An Adaptive Query Management Technique for Real-Time Monitoring of Spatial Regions in Mobile Database Systems", In *Proceedings of 21st IEEE International Conference on Performance, Computing and Communications*, pp.259-266, 2002.

- Chehadeh Y.C., Hurson A. R., and Miller L.L., "Energy-efficient indexing on a broadcast channel in a mobile database access system", In *Proceedings of the IEEE Conference on Information Technology* pp. 368-374, 2000.
- Chen G. and Kotz D., "A Survey of Context-Aware Mobile Computing Research", *Technical Report TR2000-381*, Department of Computer Science, Dartmouth College, November, 2000.
- Chung Y.D., "Indexing and Clustering of Wireless Broadcast Data", *Wireless Information Highways*, Katsaros D., Nanopoulos A., and Manolopoulos Y. (editors), Chapter 3, IRM Press Publisher, London, 2005.
- Chung Y.D. and Kim M.H., "Effective Data Placement for Wireless Broadcast", *Distributed and Parallel Databases*, 9(2): 133–150, 2001.
- Deitel H.M., Deitel P.J., and Nieto T.R., *Visual Basic 6: How to Program*, Prentice-Hall Inc, U.S.A., 1999.
- Drews P., Sommer D., Chandler R., and Smith T., "Managed runtime environments for next-generation mobile devices", *Intel Technology Journal*, 7(1), 2003.
- Ebling R.M., Hunt H.D.G. and Lei H., "Issues for Context Services for Pervasive Computing", *Proceedings of Middleware'01 Advanced Workshop on Middleware for Mobile Computing*, Heidelberg, November, 2001.
- Elmasri R. and Navathe S. B., *Fundamentals of Database Systems*, Fourth Edition, Addison Wesley, U.S.A., 2003.
- Feldman M.B., *Data Structures with Ada*, Reston Publishing Company Inc, 1985.
- Franklin M. J., "Challenges in Ubiquitous Data Management", *Informatics*: 24-33, 2001.
- Ganguly S. and Alonso R., "Query Optimization in Mobile Environments", In *Proceedings of Fifth Workshop on Foundation of Models and Languages for Data and Objects*, pp.1-17, 1993.
- Ganguly S., "Parallel Evaluation of Deductive Database Queries", *PhD Thesis*, The University of Texas, 1992.
- Ganguly S., Hasan W. and Krishnamurthy R., "Query Optimization for Parallel Executions", In *Proceedings of the 1992 ACM SIGMOD*, pp.9-18, 1992.
- Godfrey P. and Gryz J., "Semantic Query Caching for Heterogeneous Databases", In *Proceedings of the 4th Knowledge Representation meets Databases*

- Workshop (KRDB)*, pp.6.1-6.6, 1997.
- Gutting R. H., Bohlen M.,H., Erwig M., Jensen C. S., Lorentzos N. A., Schneider M., and Vazirgiannis M., “A Foundation for Representing and Querying Moving Objects”, *ACM Transactions on Database Systems Journal*, **25**(1): 1-42, 2000.
- Hameed S. and Vaidya N. H., “Log-time Algorithms for Scheduling Single and Multiple Channel Data Broadcast”, In *Proceedings of the 3rd ACM MOBICOM*, pp.90-99, 1997.
- Hameed S. and Vaidya N. H., “Efficient Algorithms for Scheduling Data Broadcast.”, *ACM/Baltzer journal of Wireless Network*, **5**(3): 183-193, 1999.
- Huang Y., Sistla P. and Wolfson O., “Data Replication for Mobile Computers”, In *Proceedings of the ACM SIGMOD*, pp.13-24, 1994.
- Huang J.L. and Chen M.-S., “Dependent Data Broadcasting for Unordered Queries in a Multiple Channel Mobile Environment”, In *Proceedings of the IEEE GLOBECOM*, pp.972-976, 2002.
- Huang J.L. and Chen M.-S., “Broadcast Program Generation for Unordered Queries with Data Replication”, In *Proceedings of the 8th ACM Symposium on Applied Computing (SAC-03)*, pp. 866-870, 2003.
- Huang J.-L. and Chen, M.-S., ”Dependent Data Broadcasting for Unordered Queries in a Multiple Channel Mobile Environment,” *IEEE Transactions on Knowledge and Data Engineering*, Vol. 16, No. 9, pp. 1143-1156, September 2004.
- Hu Q., Lee W.C. and Lee D. L., “Indexing Techniques for Wireless Data Broadcast under Data Clustering and Scheduling”, In *Proceedings of the 8th ACM International Conference on Information and Knowledge Management*, pp.351-358 1999.
- Hu Q., Lee D. L. and Lee W.C., “Optimal Channel Allocation for Data Dissemination in Mobile Computing Environments”, In *Proceedings of 18th International Conference on Distributed Computing Systems*, pp.480-487, 1998.
- Hull R., Neaves P. and Bedford-Roberts J., “Towards Situated Computing”, In *Proceedings of the First International Symposium on Wearable Computers*, pp.146-153, October 1997.

- Hurson A.R., and Jiao Y., "Data Broadcasting in Mobile Environment", *Wireless Information Highways*, Katsaros D., Nanopoulos A., and Manolopoulos Y. (editors), Chapter 4, IRM Press Publisher, London, 2005.
- Hurson A.R., Chehadeh Y.C., and Hannan J., "Object Organization on Parallel Broadcast Channels in a Global Information Sharing Environment", In *Proceedings of the 19th International Performance, Computing and Communications*, pp.347-353, 2000.
- Hurson A., and Jiao Y., *Data Broadcasting in a Mobile Environment*, Wireless Information Highways: Chapter 3, IRM Press, pp. 96-154, 2005.
- Imielinski T., Viswanathan S. and Badrinath B. R., "Energy Efficient Indexing on Air", *Proceedings of the ACM Sigmod Conference*, pp.25-36, 1994.
- Imielinski T., Viswanathan S. and Badrinath B. R., "Data on Air: Organisation and Access", *IEEE Transactions on Knowledge and Data Engineering*, **9**(3): 353-371, 1997.
- Imielinski T. and Badrinath B., "Mobile Wireless Computing: Challenges in Data Management", *Communications of the ACM*, Vol.37, No. 10, pp.18-28, October, 1994.
- Imielinski T. and Viswanathan S., "Adaptive Wireless Information Systems", *Proceedings of SIGDBS (Special Interest Group in Database Systems) Conference*, October, 1994.
- Imielinski T., Viswanathan S. and Badrinath B. R., "Data on Air: Organisation and Access", *IEEE Transactions on Knowledge and Data Engineering*, **9**(3): 353-371, 1997.
- Jayaputera, J., and Taniar, D., "Data Retrieval for Location-Dependent Query in a Multi-cell Wireless Environment", *Mobile Information Systems*, IOS Press, **1**(2), 2005.
- Jagadish H.V., "Linear Clustering of Objects with Multiple Attributes", In *Proceedings of the ACM SIGMOD*, pp. 332-342, 1990.
- Jones, A. and Ohlund, J., *Network Programming for Microsoft Windows*, Microsoft Press, Redmond, Washington, U.S.A, 2002.
- Jung H-d., You Y-h, Lee J-j and Kim K., "Broadcasting and Caching Policies for Location-Dependent Queries in Urban Areas", In *Proceedings of the of the 2nd international workshop on Mobile commerce*, pp. 54-59, 2002.

- Kirkpatrick D.G., "Optimal Search in Planar Subdivisions", *SIAM J. on Computing*, **15**(2):28-35 1983.
- Kollios G., Gunopulos D., and Tsotras V.J., "On Indexing Mobile Objects", In *Proceedings of ACM PODS*, pp.261-272, 1999.
- Kottkamp H.-E. and Zukunft O., "Location-Aware Query Processing in Mobile Database Systems", In *Proceedings of ACM Symposium on Applied Computing*, pp.416-423, 1998.
- Kubach U. and Rothermel K., "An Adaptive Location-Aware Hoarding Mechanism", In *Proceedings of the 5th IEEE Symposium on Computers and Communications*, pp.615-620, 2000.
- Kubach U. and Rothermel K., "A Map-Based Hoarding Mechanism for Location-Dependent Information", In *Proceedings of Second International Conference on Mobile Data Management*, pp.145-157, 2001.
- Lee, G., and Lo S-C, "Broadcast Data Allocation for Efficient Access on Multiple Data Items in Mobile Environments", *Mobile Networks and Applications*, **8**:365-375, 2003.
- Lee, D.K., Xu, J., Zheng, B. and Lee, W-C, "Data Management in Location-Dependent Information Services", *IEEE Pervasive Computing*, **2**(3):65-72, July-Sept, 2002.
- Lee K. C. K., Leong H. V. and Si A., "Semantic Data Access in an Asymmetric Mobile Environment", In *Proceedings of the 3rd Mobile Data Management*, pp.94-101, 2002.
- Leong H. V. and Si A., "Database Caching Over the Air-Storage", *The Computer Journal*, **40**(7):401-415, 1997.
- Leong H. V. and Si A., "Data Broadcasting Strategies over Multiple Unreliable Wireless Channels", In *Proceedings of the 4th Information and Knowledge Management*, pp.96-104, 1995.
- Liberatore V., "Multicast Scheduling for List Requests". In *Proceedings of IEEE INFOCOM Conference*, pp.1129-1137, 2002.
- Lee W.C., Hu Q. and Lee D. L., "Channel Allocation Methods for Data Dissemination in Mobile Computing Environments", In *Proceedings of the 6th IEEE High Performance Distributed Computing* pp.274-281, 1997.
- Lee K. C. K., Leong H. V. and Si A., "Semantic Data Access in an Asymmetric

- Mobile Environment”, In *Proceedings of the 3rd Mobile Data Management*, pp.94-101, 2002.
- Lee W.C. and Lee D. L., “Using Signature techniques for Information Filtering in Wireless and Mobile Environments”, *Journal on Distributed and Parallel Databases*, **4**(3): 205-227, 1996.
- Lee D. L., Hu Q. and Lee W.C., “Indexing Techniques for Data Broadcast on Wireless Channels”, In *Proceedings of the 5th Foundations of Data Organization*, pp. 175-182, 1998.
- Lee, D.K., Xu, J., Zheng, B. and Lee, W-C, “Data Management in Location-Dependent Information Services”, *IEEE Pervasive Computing*, **2**(3):65-72, July-Sept, 2002.
- Madria S.K., Bhargava B., Pitoura E. and Kumar V., “Data Organisation for Location-Dependent Queries in Mobile Computing”, In *Proceedings of ADBIS-DASFAA*, pp. 142 – 156, 2000.
- Malladi R. and Davis K.C., “Applying Multiple Query Optimization in Mobile Databases”, In *Proceedings of the 36th Hawaii International Conference on System Sciences*, pp. 294 – 303, 2002.
- Myers B.A. and Beigl M., “Handheld Computing”, *IEEE Computer Magazine*, **36**(9):27-29, 2003.
- Nokia, “Enhanced Data Rates for GSM Evolution EDGE”, *White Paper*, Nokia Telecommunications, Finland, 1999.
- Paulson L.D., “Will Fuel Cells Replace Batteries in Mobile Devices?” *IEEE Computer Magazine*, **36**(11): 10-12, 2003.
- Pitoura E., and Samaras G., *Data Management for Mobile Computing*, Kluwer Academic Publishers, London, 1998.
- Pfoser D., "Indexing the Trajectories of Moving Objects", *IEEE Data Engineering Bulletin*, **25**(2): 3-9, 2002.
- Pfoser D. and Theodoridis Y., "Generating Semantics-Based Trajectories of Moving Objects", *International Journal of Computers, Environment, and Urban Systems*, **27**(3): 243-263, 2003.
- Prabhakara K., Hua K.A, and Oh J., “Multi-Level Multi-Channel Air Cache Designs for Broadcasting in a Mobile Environment”, In *Proceedings of the IEEE*

- International Conference on Data Engineering (ICDE'00)*, pp. 167-176, 2000.
- Roos T., Myllymaki P., and Tirri H., "A Statistical Modelling Approach to Location Estimation", *IEEE Transactions of Mobile Computing*, **1**(1): 59-69, 2002.
- Saltenis S., and Jensen C.S., "Indexing of Moving Objects for Location-Based Services", In *Proceedings of ICDE*, pp. 463-472, 2002.
- Saltenis S., Jensen C.S., Leutenegger S.T., and Lopez M.A., "Indexing the Positions of Continuously Moving Objects", In *Proceedings of ACM SIGMOD*, pp.331-342, 2000.
- Schilit B., Adams N. and Want R., "Context-Aware Computing Applications", *Proceedings of IEEE Workshop on Mobile Computing Systems and Applications*, pp. 85-90, December, 1994.
- Seydim A.Y., Dunham M. H. and Kumar V., "Location-dependent Query Processing", In *Proceedings of the Second International Workshop on Data Engineering on Mobile and Wireless Access (MobiDE'01)*, pp.47-53, 2001.
- Seeley D. et al, *Planimate[™]-Animated Planning Platforms*, InterDynamics Pty Ltd, 1997.
- Sharma C., *Wireless Internet Enterprise Applications*, John Wiley & Sons Inc., U.S.A., 2001.
- Si A. and Leong H. V., "Query Optimization for Broadcast Database", *Data and Knowledge Engineering*, **29**(3): 351-380, 1999.
- Sistla A. P., Wolfson O., Chamberlain S. and Dao S., "Modeling and Querying Moving Objects", In *Proceedings of the 13th International Conference on Data Engineering*, pp.422-432, 1997.
- Stan M. C., and Skadron K., "Power-Aware Computing", *IEEE Computer Magazine*, **36**(12):35-38, 2003.
- Taniar D., Rahayu J.W., "ATaxonomy of Indexing Schemes for Parallel Database Systems", *Distributed and Parallel Databases*, **12**(1): 73-106, 2002.
- Tao Y., Papadias D. and Sun J., "The TPR*-Tree: An Optimized Spatio-Temporal Access Method for Predictive Queries", In *Proceedings of VLDB Conference*, pp. 790-801, 2003.

- Tran D.A., Hua K.A, and Jiang N., "A Generalized Design for Broadcasting on Multiple Physical-Channel Air-Cache", In *Proceedings of the ACM SIGAPP Symposium on Applied Computing (SAC'01)*, pp. 387-392, 2001..
- Triantafillou P., Harpantidou R. and Paterakis M., "High Performance Data Broadcasting: A Comprehensive Systems 'Perspective", In *Proceedings of the 2nd International Conference on Mobile Data Management (MDM 2001)*, pp.79-90, 2001.
- Trivedi K. S., Dharmaraja S. and Ma X., "Analytic modelling of handoffs in wireless cellular networks", *Information Sciences*, **148**: 155-166, 2002.
- Tsalgatidou A., Veijalainen J., Markkula J., Katasonov A., and Hadjiefthymiades S., "Mobile E-Commerce and Location-Based Services: Technology and Requirements", In *Proceedings of the 9th Scandinavian Research Conference on Geographical Information Services*, pp. 1-14, 2003.
- Theodoridis Y., Silva R. and Nascimento M., "On the Generation of Spatiotemporal Datasets", In *Proceedings of the 6th International Symposium on Spatial Databases*, pp. 147-164, 1999.
- Vaughan-Nichols J. S., "Bull Market for IEEE 802.11 WLAN Chipsets", *Computer Magazine*, Vol.35, No.11, pp. 17-19, November, 2002.
- Want R., Schilit N.B., Adams I.N., Gold R., Petersen K., Goldberg D., Ellis R.J. and Weiser M., "The ParcTab Ubiquitous Computing Experiment", *Mobile Computing*, Imielinski T. and Korth H.F. (editors), Chapter 2, Kluwer Academic Publishers, Boston, 1996.
- Wong J.W., "Broadcast Delivery", *Proceedings of the IEEE*, **76**(12):1566-1577, 1988.
- Worthington B. L., Ganger G. R. and Patt Y., "Scheduling Algorithms for Modern Disk Drivers", In *Proceedings of the ACM SIGMETRICS*, pp.241-251, 1994.
- Wolfson, O., "Moving Objects Information Management: The Database Challenge", Vision Paper, In *Proceedings of the 5th Workshop on Next Generation Information Technologies and Systems (NGITS'2002)*, pp. 75-89, 2002.
- Wolfson O., Sistla P., Xu B., Zhou J., Chamberlain S., Yesha Y., and Rische N. "Tracking Moving Objects Using Database Technology in DOMINO", In *Proceedings of the 4th Workshop on Next Generation Information Technology and Systems*, pp.112-119, 1999.

- Xu J., Lee D-L, Hu Q., and Lee W-C., *Data Broadcast*, Handbook of Wireless Networks and Mobile Computing: Chapter 11, John Willey & Sons, pp. 243-365, 2002.
- Xu J., Zheng B., Lee W-C., and Lee D.L., "Energy Efficient Index for Querying Location-Dependent Data in Mobile Broadcast Environments", In *Proceedings of the 19th IEEE Int. Conf. on Data Engineering (ICDE '03)*, pp. 239-250, 2003.
- Xu J., Zheng B., Zhu M. and Lee D.L., "Research Challenges in Information Access and Dissemination in a Mobile Environment", In *Proceedings of the Pan-Yellow-Sea International Workshop on Information Technologies for Network Era*, pp. 1-8, 2002.
- Yajima E., Hara T., Tsukamoto M. and Nishio S. "Scheduling and Caching Strategies for Correlated Data in Push-based Information Systems", *ACM SIGAPP Applied Computing Review*, **9**(1): 22-28, 2001.
- Yee W. G., and Navathe S.B., "Efficient Data Access to Multi-Channel Broadcast Program", In *Proceedings of the ACM International Conference on Information and Knowledge Management (CIKM'03)*, pp. 153-160, 2003.
- Zaslavsky A., and Tari S., "Mobile Computing: Overview and Current Status", *Australian Computer Journal*, **30**(2): 42-52, 1998.
- Zheng B., Xu J, Lee D.L., "Cache Invalidation and Replacement Strategies for Location-Dependent Data in Mobile Environments", *IEEE Transactions on Computers*, **51**(10): 1141-1153, 2002.
- Zheng B., Lee W-C and Lee D.L., "Selecting the Best Valid Scopes for Wireless Dissemination of Location-dependent Data", In *Proceedings of the 18th ACM Symposium on Applied Computing (SAC'03)*, pp. 860-865, 2003.
- Zheng B., and Lee D.L., "Processing Location-Dependent Queries in a Multi-cell Wireless Environment, In *Proceedings of the 2nd ACM International Workshop on Data Engineering for Wireless and Mobile Access (MobiDE'01)*, pp.54-65, 2001.

Appendix A

Simulation Model

A.1 Simulation Platform Overview

- *Planimate*, animated planning platforms is a software platform designed for prototyping, developing and operating highly visual and interactive applications (Seeley, 1997). Figure A.1 depicts the initial interface of Planimate simulation platform.



Figure A.1. Planimate Simulation Platform

- *Planimate* has a number of simulation objects from the object palette which represent different activities for simulating the behaviour of the real-environment. The following diagram (Figure A.2) is an enlargement of the Object Palette with a table of object names.

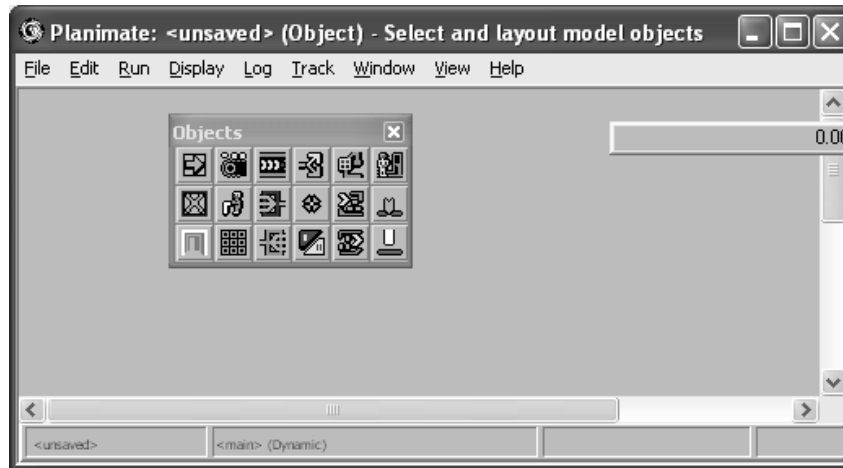


Figure A.2. Objects in Planimate

- Items, or temporary entities, will flow through the system and interact with the permanent objects (see Figure A.3). The paths by which they will move through the network of objects needs to be defined.

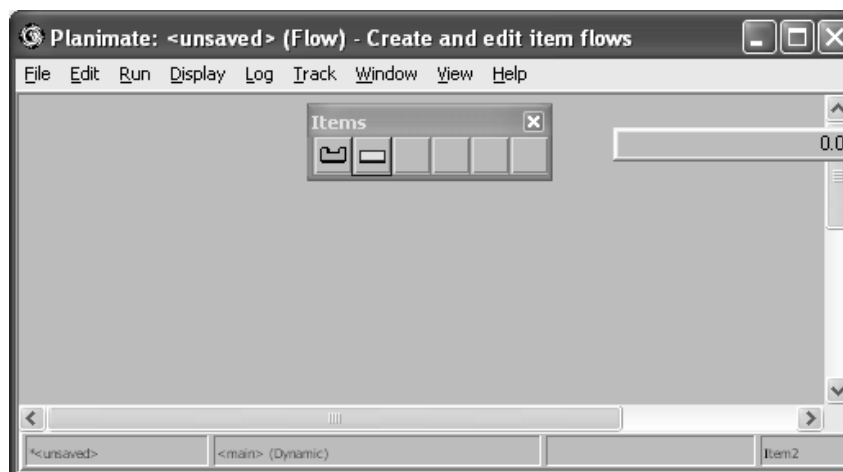


Figure A.3. Entities in Planimate

A.2 Data Broadcasting Model

- A data broadcasting model is shown in Figure A.4. In this example, there are four servers, four broadcast transmitters, four wireless channels, and four mobile clients. The receptors in this case represent mobile devices which listen to broadcast channel, and select the data items based on client requests. Each mobile client receives broadcast data items from individual server. The broadcast interval can be set in periodic manner.

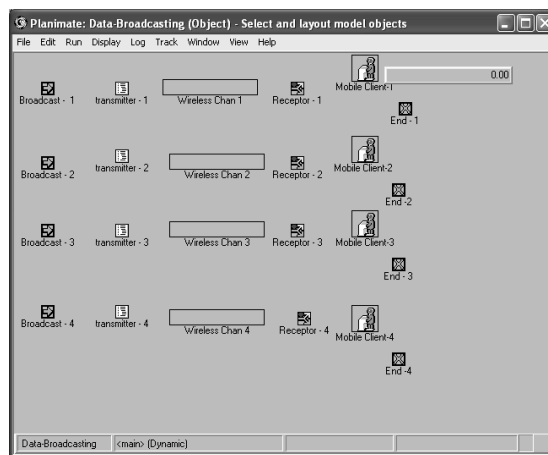


Figure A.4. Data Broadcasting Model

- Figure A.5 shows the flow of the broadcast data items from the server through wireless channel to mobile client.

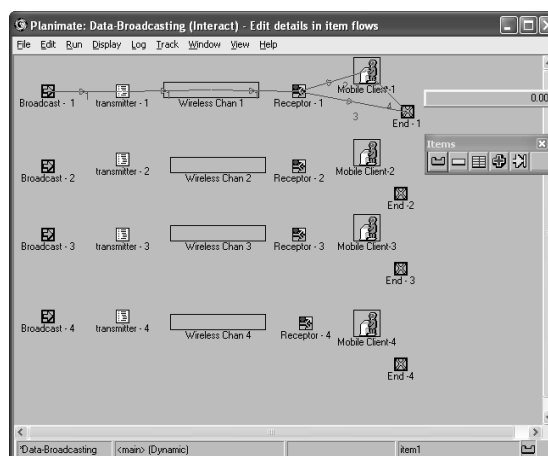


Figure A.5. Data Broadcasting Model with flow

- The image shown in Figure A.6 demonstrates the parameters setting for broadcast arrival rate including the distribution patterns.

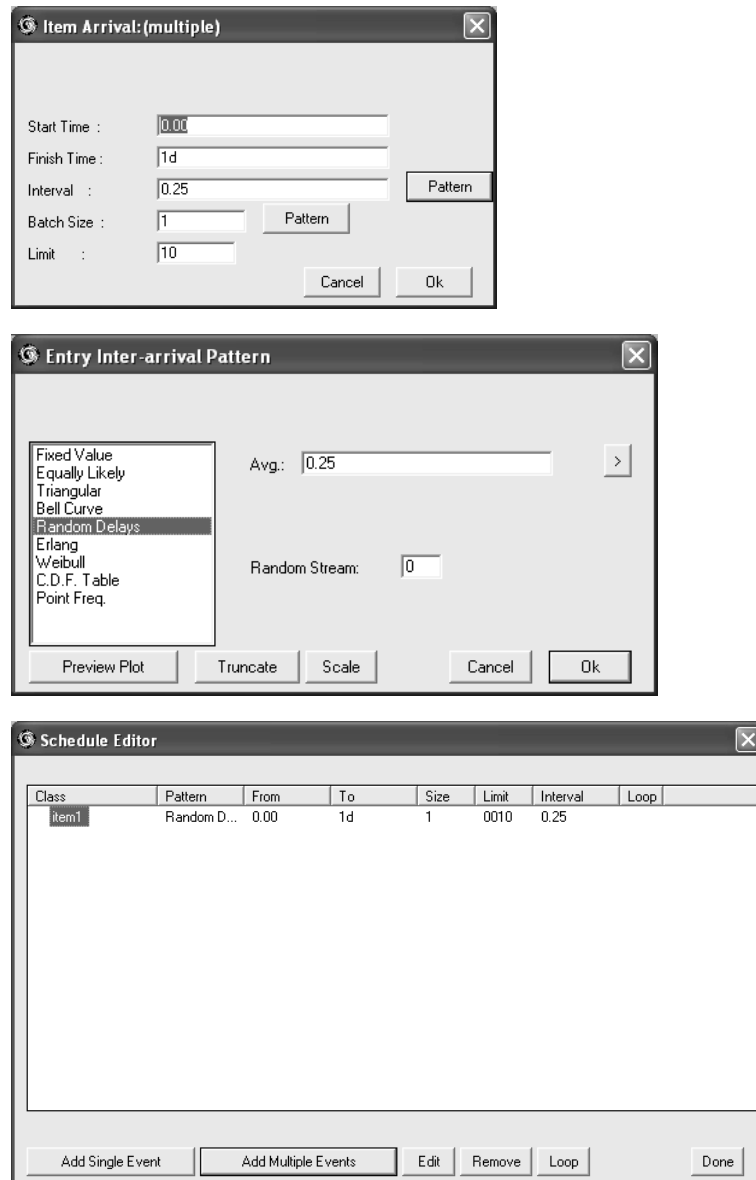


Figure A.6. Parameters Setting – Data Broadcast Arrival Rate

- The plot in Figure A.7 indicates that the chosen distribution is the exponential. The vertical dotted line in the plot indicates the mean value.

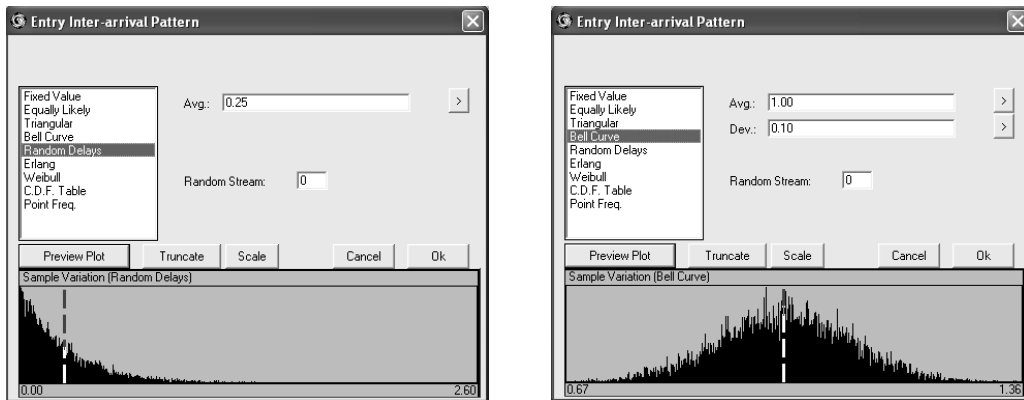


Figure A.7. Distribution Setting

- Figure A.8 shows image of labelling each broadcast items. This is applied for simulating the broadcast ordering and scheduling schemes. Clients will be able to select the desired data items based on their preferences.

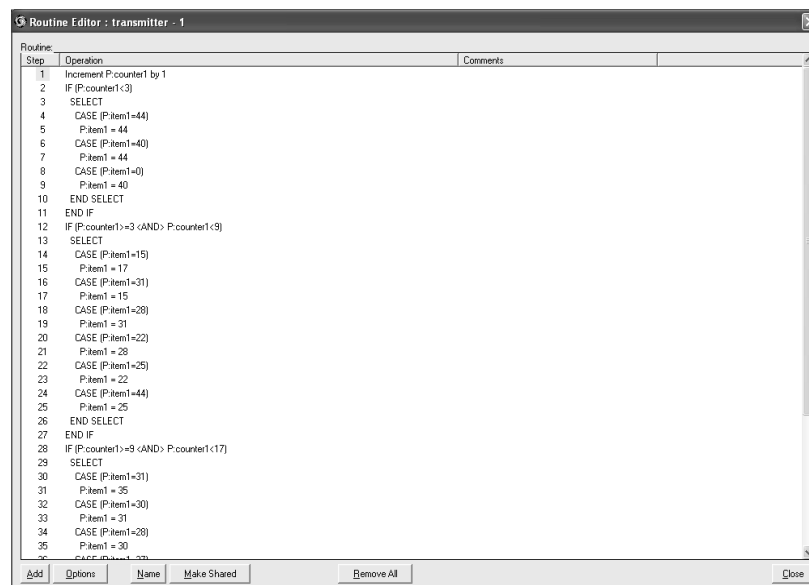


Figure A.8. Broadcast Items Ordering Setting

- The image in Figure A.9 shows the interface for specifying the data items of interest of mobile clients. Based on this setting, clients will be able to select the desired broadcast data items.

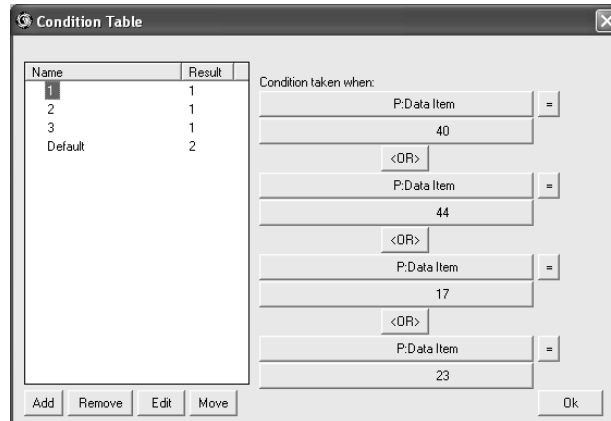


Figure A9. Condition Setting: Data Filtering (Mobile Clients)

- Figure A.10 depicts the parameter setting for the clients in downloading broadcast data items.

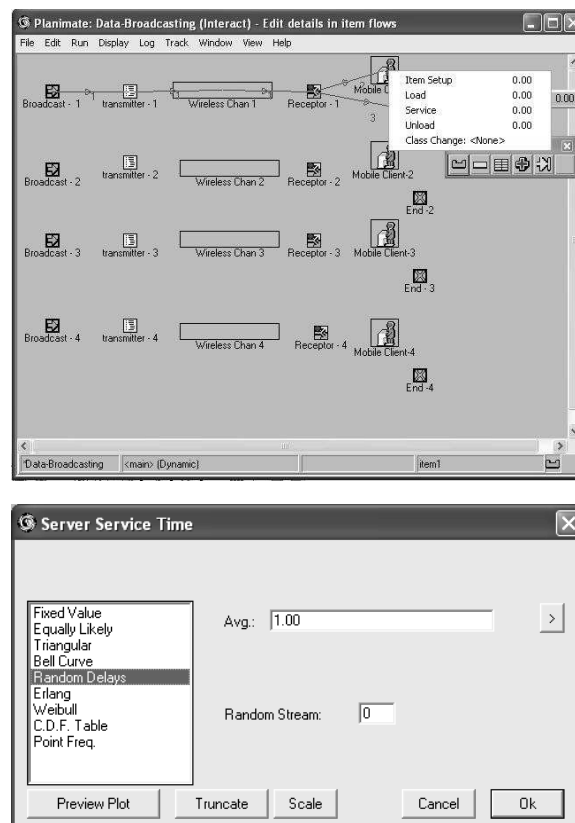


Figure A.10. Service Time parameter setting

- Figure A.11 shows the data broadcasting model with three broadcast channels. Clients can only receive data items from one channel at any given time.

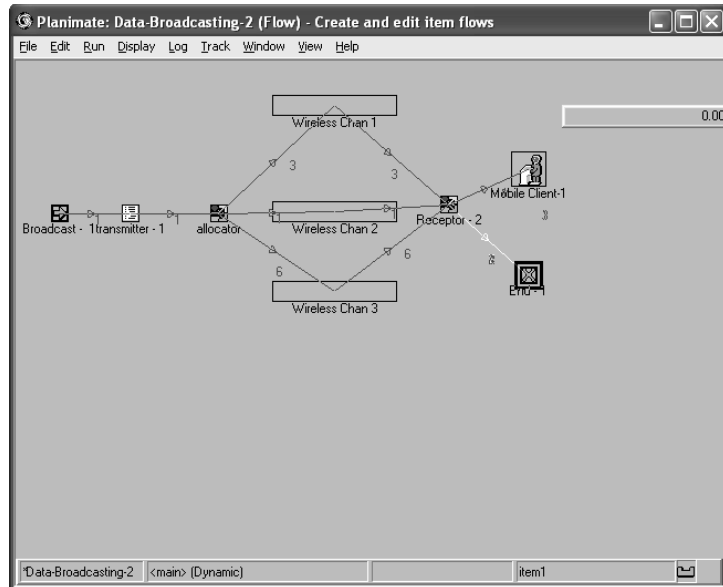


Figure A.11. Data Broadcast Model – 3 broadcast channels

- Figure A.12 illustrates the combined model of the data broadcasting system, which consists of data and index channels. The condition is set on the receptor items which select the index and broadcast items according to the client's requests.

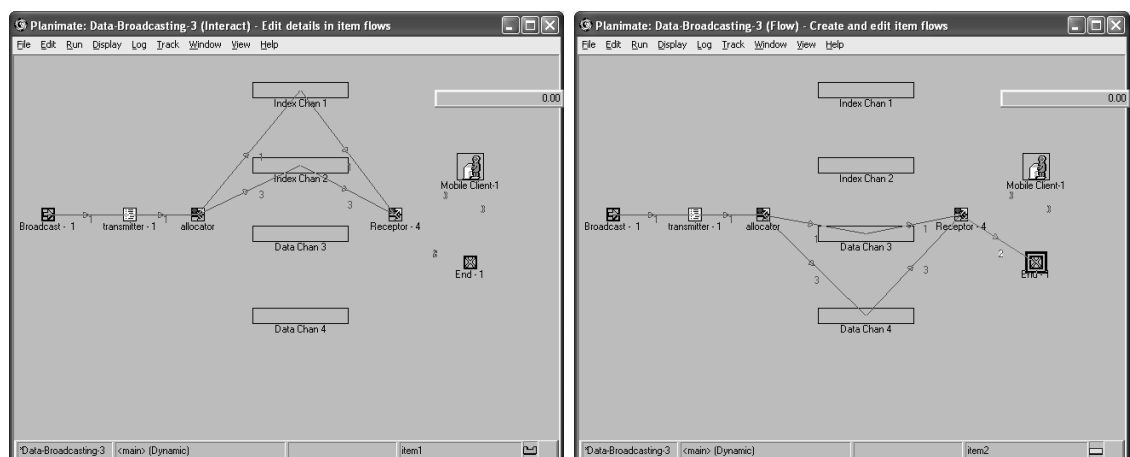


Figure A.12. Data Broadcast Model – Index and Data Channel (combined model)

- Figure A.13 demonstrates the simulation mode setting. The simulation can be set in single run or multiple run based on the given parameters settings.

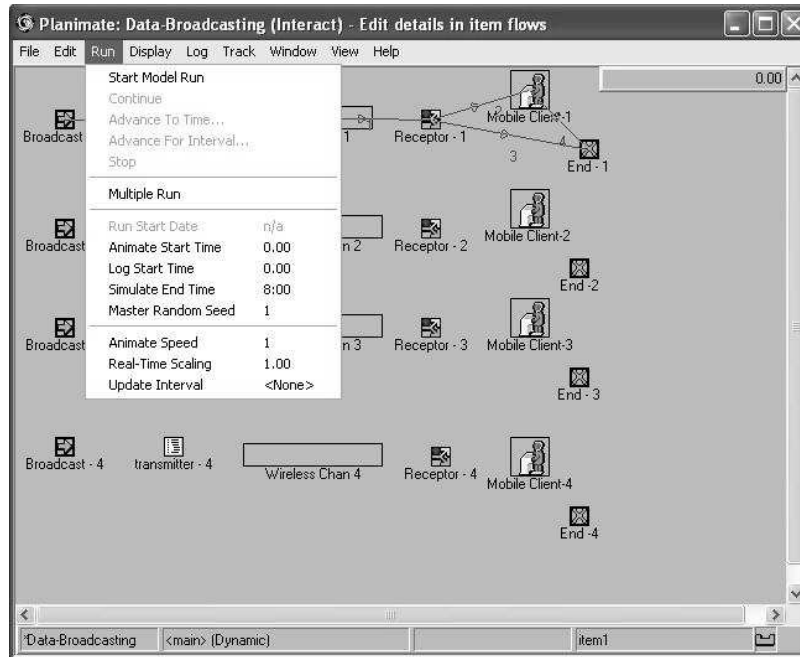


Figure A.13. Simulation mode setting

- Figure A.14 shows the graphical interface when the simulation is running on single mode.

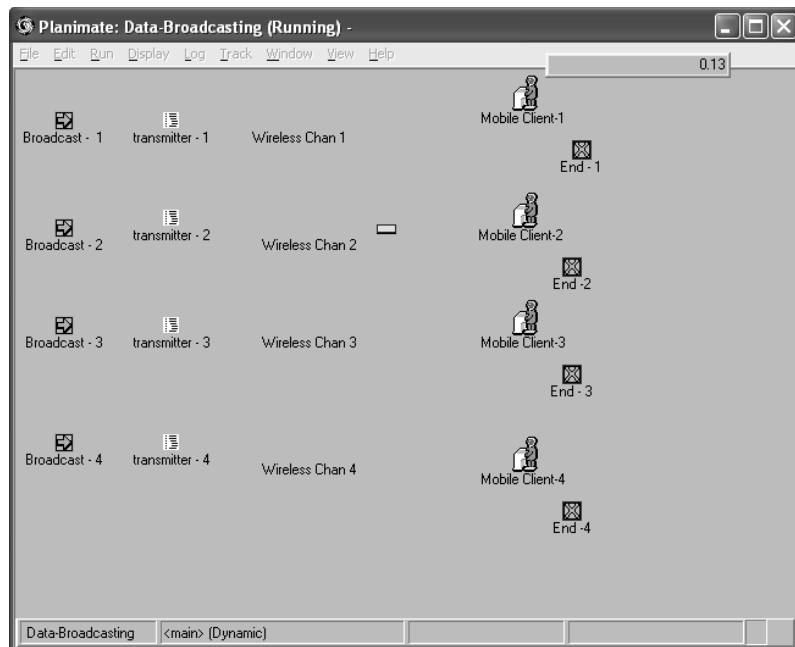


Figure A.14. Simulation Running mode

- The image in Figure A.15 illustrates the parameter setting for multiple run modes.



Figure A.15. Multiple run mode

A.3 Simulation Results

- Figure A.16 shows the legend of the simulation results, which are derived from multiple run mode.

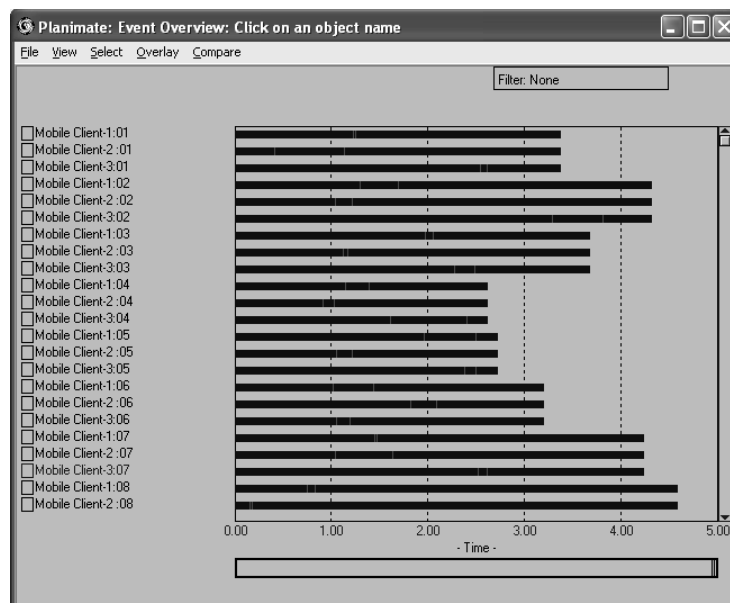
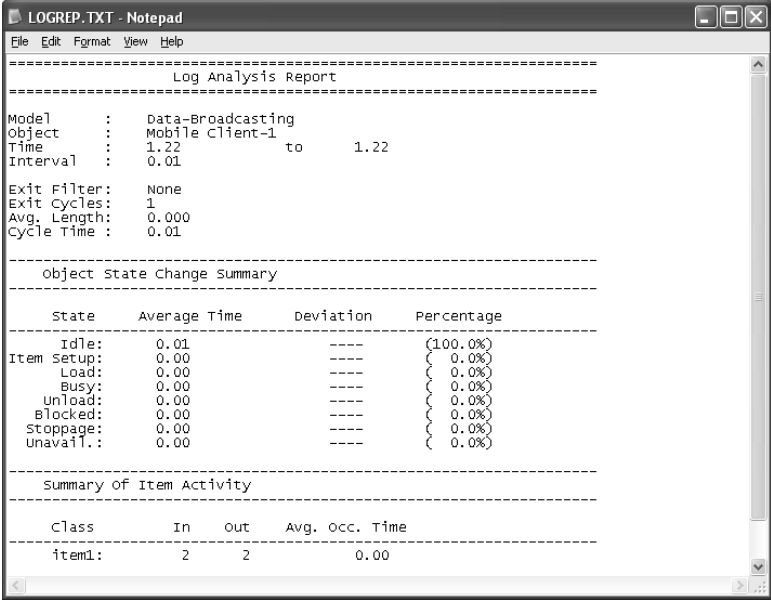


Figure A.16. simulation results - collective

- Figure A.17 depicts the simulation results from multiple run mode as per iteration. It can be seen the time the data items are received, the time difference between one data item and another, number of data items received, etc.



```

LOGREP.TXT - Notepad
File Edit Format View Help
-----
Log Analysis Report
-----
Model   :   Data-Broadcasting
Object  :   Mobile Client-1
Time    :   1.22      to      1.22
Interval : 0.01

Exit Filter: None
Exit Cycles: 1
Avg. Length: 0.000
Cycle Time : 0.01

-----
object State Change Summary
-----
      State   Average Time   Deviation   Percentage
-----
Item Idle:    0.01           ----      (100.0%)
  Setup:    0.00           ----      (  0.0%)
   Load:    0.00           ----      (  0.0%)
   Busy:    0.00           ----      (  0.0%)
  Unload:    0.00           ----      (  0.0%)
 Blocked:    0.00           ----      (  0.0%)
 Stoppage:    0.00           ----      (  0.0%)
 Unavail.:    0.00           ----      (  0.0%)

-----
Summary of Item Activity
-----
Class      In   out   Avg. Occ. Time
-----
item1:      2   2     0.00

```

Figure A.17. Simulation result per iteration

Appendix B

Prototype of Data Broadcast System

B.1 Database Design

The data broadcasting are demonstrated on the server application by the *share price* message type. The Microsoft® Access® database is the data source used in the prototype model. It is comprised of three tables entitled *tblMessageType*, *tblIndexTree* and *tblSharePrices*. Figure B.1 illustrates the three tables in the database of the prototype model. The table, *tblMessageType* stores records of the different messages broadcast from the server application. The table, *tblSharePrices* stores records of the share prices details. Each share price is identified by an index that acts as a primary key in the Share Prices table. Finally, the table, *tblIndexTree* stores details about the *index tree architecture* for the share prices. The index tree identifies when the records in the table *tblSharePrices* are broadcast.

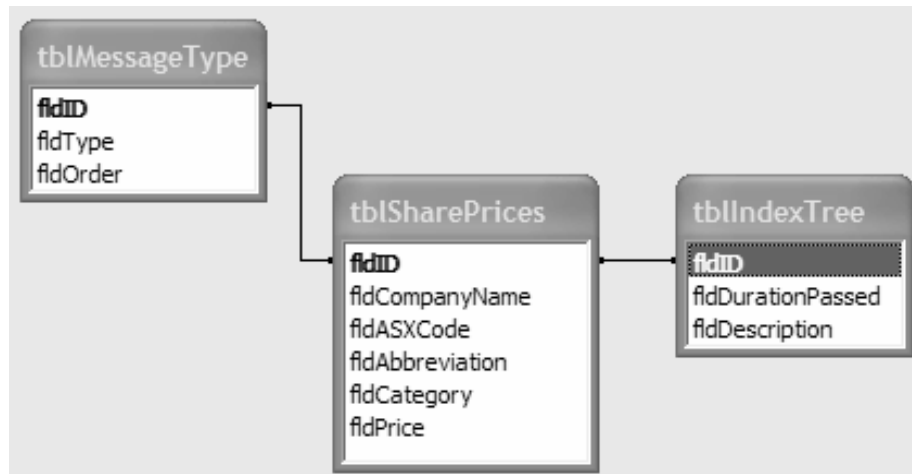


Figure B.1. Tables in the proposed broadcast model

B.2 Data Broadcast System: Server Implementation

- **Data broadcasting scheme**

In the prototype system, a datagrid is included for the user to select the share price to broadcast. Figure B.2 illustrates the server's interface required to broadcast database information.

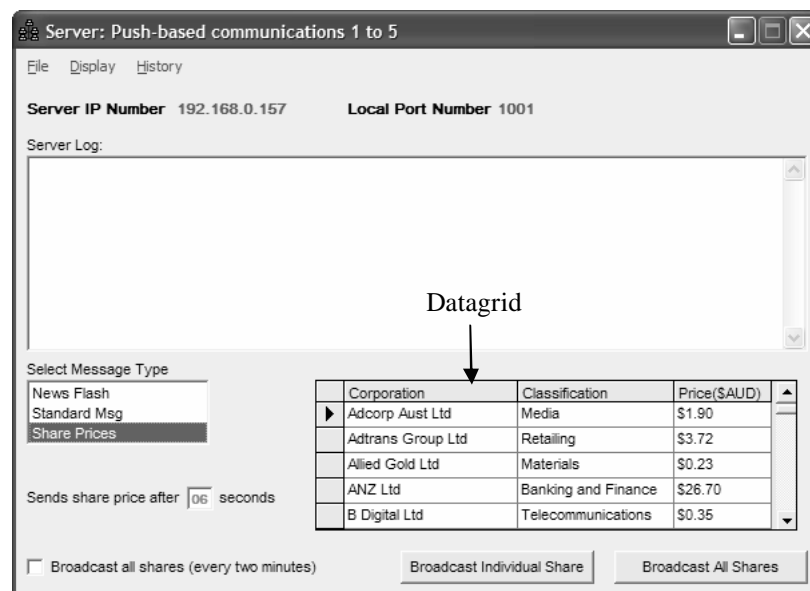


Figure B.2. Server's interface for data broadcasting system

- **Index broadcasting scheme**

The server application is designed to broadcast the *Share Price* message periodically with index broadcasting schemes. Figure B.3 illustrates the concept of a tree index used to accommodate 54 data records. The tree index starts from the root followed by three level indexes. The first level of the index consists of three data buckets (a1, a2, a3) and each bucket has three pointers to the data buckets in the next level. Subsequently, there are nine data buckets (b1, b2, b3, b4, b5, b6, b7, b8, b9) in the second level and each of them has three pointers to the last level. The box in the bottom level represents the final data bucket and each of the boxes holds 2 data records. In this context, the root and first level of the tree index are referred to as share structure keys level and the second level relates to the share data keys level.

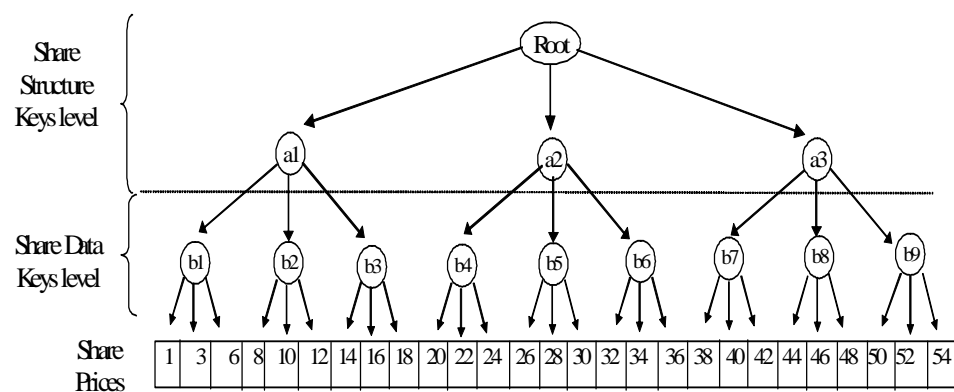


Figure B3. Index tree structure

In order to describe how the index key is interpreted, Figure B.4 provides a tree structure diagram of all the share structure keys broadcast to the client application. These share structure keys are broken up into three hierarchical levels that are represented by the letters A, B and C. The first letter on every data represents its designated hierarchical level.

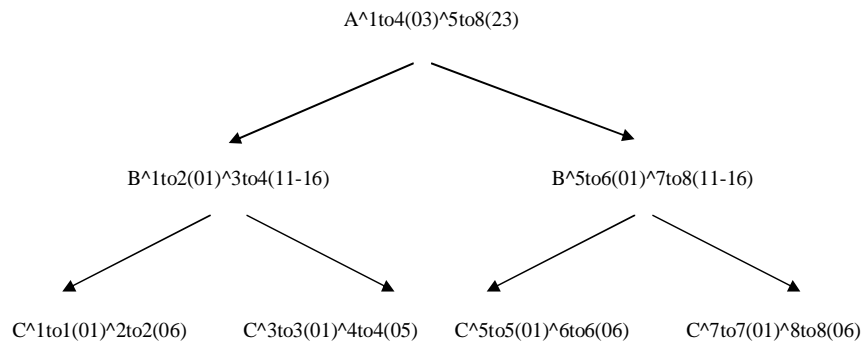


Figure B4. Tree diagram of the share structure keys level

For example, the share structure key $B^{5to6(01)^7to8(11-16)}$ is designated to the hierarchical level B. The higher the designated hierarchical level of the share structure key, the greater the range difference is for the next batch of keys. The current share structure key has been implemented so that a share structure key with the hierarchical level of A has a range of eight; the share structure key with a hierarchical level of B has a range of four; and, a share structure key with a hierarchical level of C has a range of two. The designated hierarchical level on the share structure key also identifies the type of key that is expected; share structure key or share data key. A share structure key with the designated hierarchical level of A or B identifies when the *share structure keys* are broadcast. While a share structure key with the hierarchical level of C identifies when the *share data keys* are broadcast.

The share structure key is comprised of two more elements: (1) the range of the next share structure keys or share data keys broadcast and (2) the time this batch of share structure keys or share data keys are broadcast. The range when the next share structure key(s) or share data key(s) are broadcast, are identified by the next four values after the ^ symbol. These four values indicate the range of the next batch of incoming share structure keys or share data keys. Each share structure key is

comprised of two ranges. Beside each range, the share structure key contains the time interval, when the next batch of keys is broadcast. The time is represented in seconds and is the two values in between the forward and backward brackets, (**). Refer to Figure B.5 for a visual representation of the share structure key(s).

The share structure key for the Global index is rather different as it includes the port numbers in which the key are to be broadcast into. This is applied as Global Index considers a certain degree of replications. Figure B.6 depicts the share structure key for Global Index Scheme.

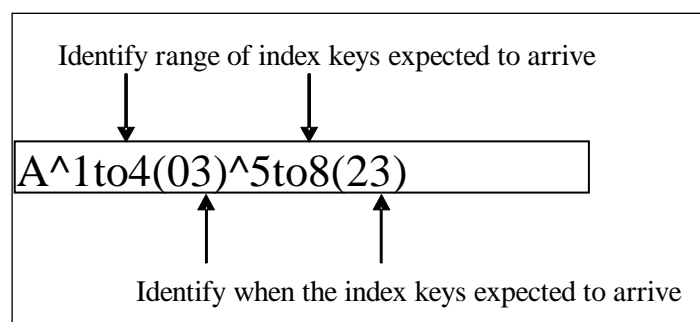


Figure B.5. Share structure key

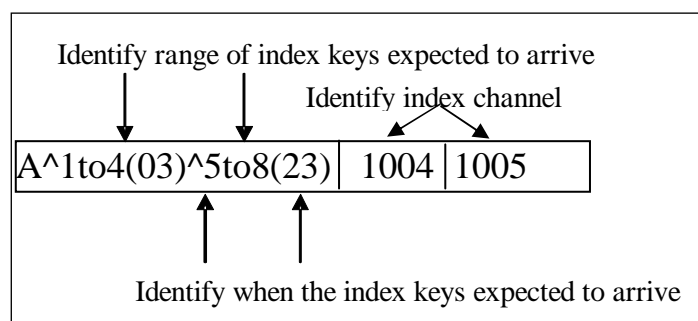


Figure B.6. Share structure key – Global Index

By understanding these features, a share structure key can be interpreted. For instance, the share structure key $A^{1to4}(03)^{5to8}(23)$ identifies that the share structure keys for share prices with the identification number from one to four are

broadcast in three seconds, and the share structure keys for share prices with the identification number of five to eight are broadcast in twenty-three seconds.

The share data key is similar except it also identifies the designated port number. The share data key has three characteristics: (1) Unique Identifier, (2) Interval and (3) Port Number. An example of a share data key is, 01#06|1014. In this example the '01', is the unique identifier, the '06' is the interval represented in seconds and '1014' is the designated port number the share price will be broadcast to. The server application interprets this share data key and broadcasts the share price with the unique identifier of '01' to port 1014 after six seconds has elapsed since the share data key has been broadcast. Figure B.7 provides a visual illustration of this share data key.

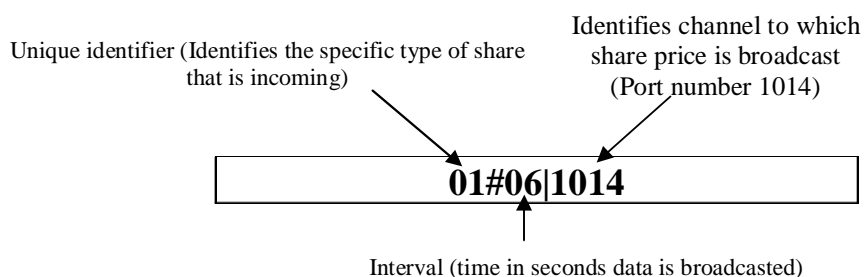


Figure B.7. Share data key

The share prices are broadcast periodically by a checkbox. Figure B8 illustrates the check box required to execute periodic data and index broadcasting-based communications in the server application. Figure B.9 and B.10 show the image when the index is broadcast and the share prices follow, respectively.

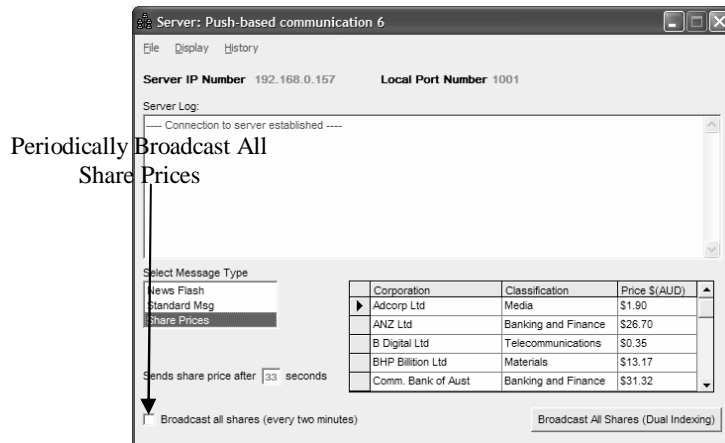


Figure B.8. Server application with indexing scheme

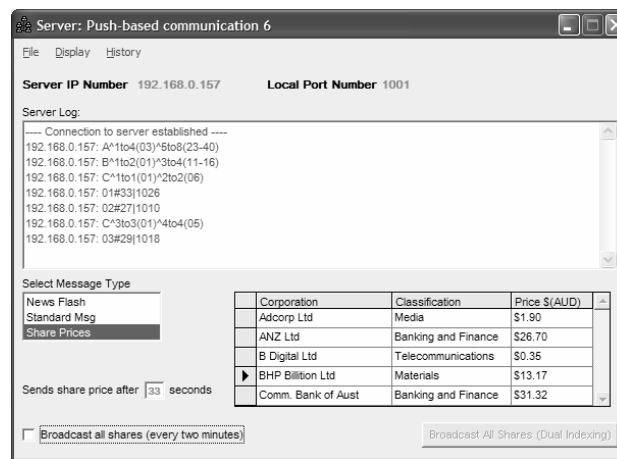


Figure B.9. Share structure and share data keys broadcast

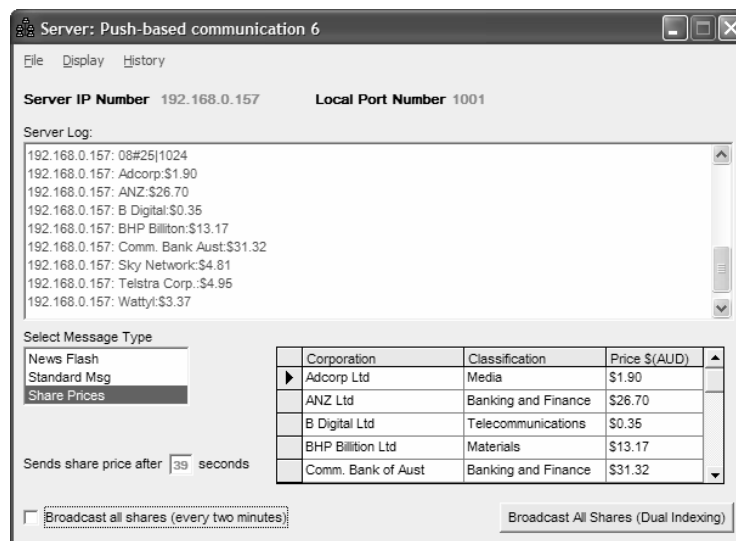


Figure B.10. Share prices broadcast

- **Multi-channeling Environment**

The server application is designed so that the client applications are able to utilize multi-channeling. Multi-channeling is exhibited by the server application's ability to broadcast data to multiple channels throughout the network. Specifically, the server application uses a range of port numbers to broadcast a range of different information. Figure B.11 illustrates the multi-channeling feature.

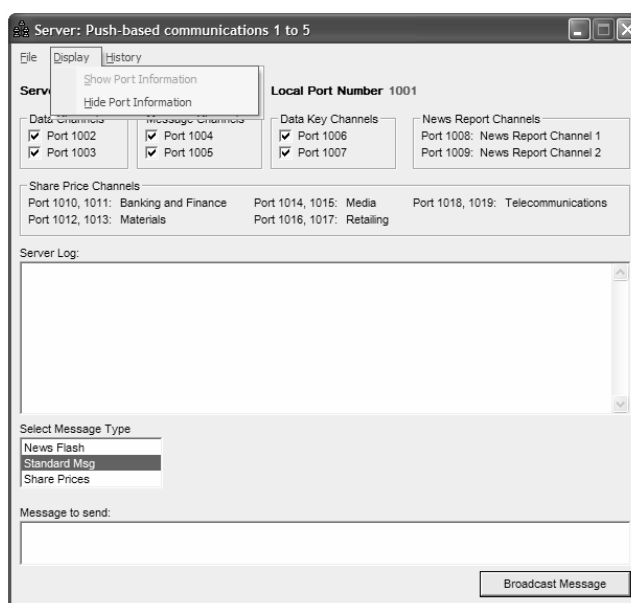


Figure B.11 Multi-channelling

The data items from the data source, Microsoft® Access Database are divided into several equal-sized channels and apply the *data partitioning broadcast mechanism* (Leong and Si, 1995). By adopting the data partitioning broadcast mechanism, the clients will be able to retrieve broadcast data without any conflict. For instance, if it is assumed that each share price data item can only be broadcast every ten seconds, it would require a broadcast cycle of 300 seconds for thirty share price data items. However, by implementing the data partitioning mechanism, the ten second interval

for each share price is maintained except the share prices are broadcast at different starting points among multiple channels.

B.3. Data Broadcast System: Client Implementation

This section describes queries processing from client application: (1) data filtering; and (2) data filtering through index.

- **Data Filtering**

The client application is able to execute the data broadcasting scheme by filtering the broadcast data, to select and display only the desired incoming data items. A series of checkboxes in the client application provides the user with the option to select the desired data. The incoming data is selected and displayed when the corresponding checkbox has a value of one, i.e. is selected. If the checkbox has a value of zero, i.e. not selected, the corresponding incoming data is not selected and displayed. Figure B.12 (a and b) displays the initial client application for setting up query, whilst Figure B.13 shows client application when the query results are obtained.

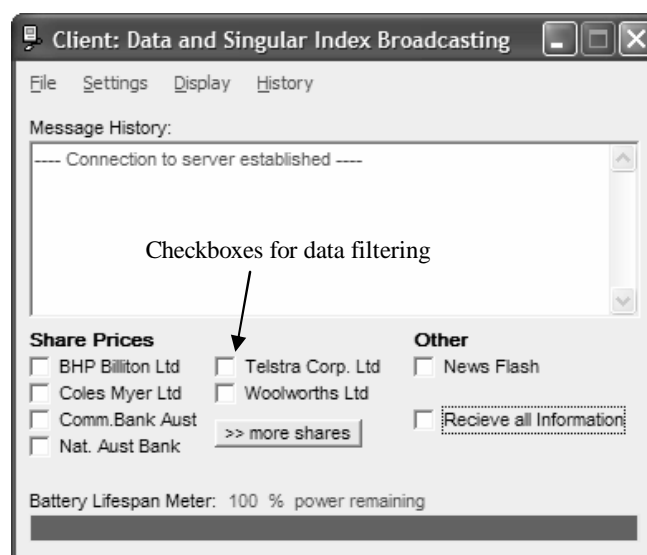


Figure B.12a. Client application – setting up query



Figure B.12b. Client application – setting up query

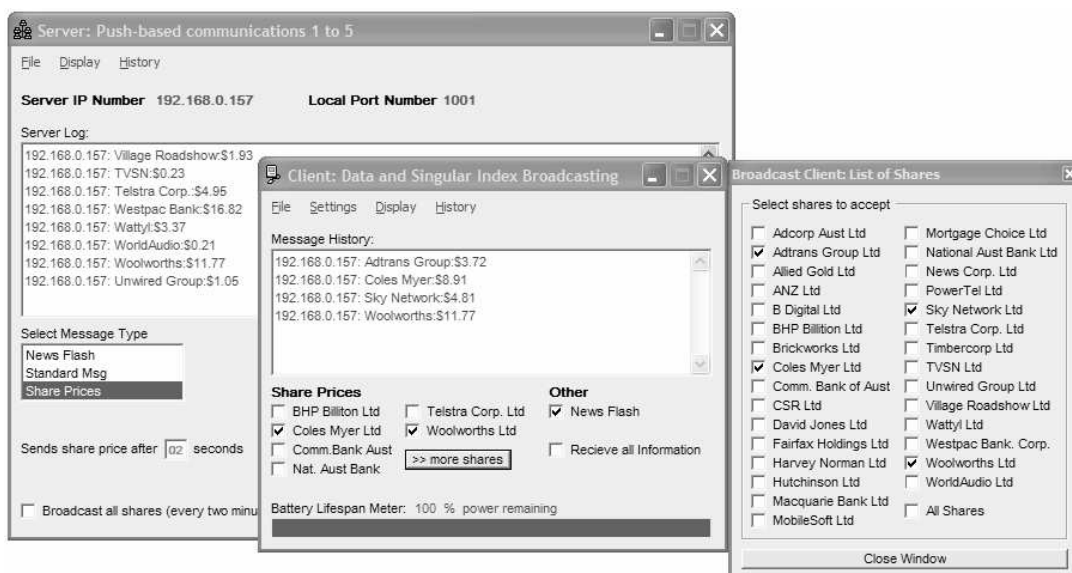


Figure B.13. Client application – displaying results

- **Data filtering through index**

Data filtering through index is when an index is broadcast with a data item. By interpreting the data key, the client application is able to switch to energy conserving mode when the desired data is not scheduled to arrive. This power saving incorporates the following features: automatically partially disconnects and turns off

monitor after idle, automatically turns on monitor and connects to server application when data arrives and automatically partially disconnects after data arrives and monitor stays on for at least a few seconds. The client application is able to determine when the incoming data is expected to arrive by interpreting the index key broadcast from the server. Despite, the client application being partially disconnected, it is still able to interpret index keys. The share index key has two important characteristics: unique identifier and interval. The unique identifier the specific type of data expected to arrive, while, the interval is the expected time in seconds when the data is broadcast to the client application. The key also has an additional characteristic that specifies the channel or port number, which was explained in the previous section.

The client application interprets each index key individually by a two-step process: (1) it checks if the data key's unique identifier corresponds to the data required by the user, and either 2a) implements a timer control to turn the monitor on and connect to server application when data arrives; or 2b) ensures that the monitor remains off and the client application remains partially disconnected. When the unique identifier of the data key corresponds to the data required by the user, step 2a) is executed; otherwise step 2b) is executed. In particular, the client application interprets a structure key to determine whether to receive or deliberately not receive the incoming data key.

The two index structures interpreted by the client application to execute the index broadcasting scheme are the share structure key and share data key. The share structure key identifies when the next individual share data key(s) or share structure

key(s) is scheduled to arrive. Specifically, the share structure key identifies two main elements: (1) those which identify which share structure key(s) or share data key(s) are expected to arrive and (2) the time in seconds, when the share structure key(s) will arrive. A share structure key is more informative than a share data key and, unlike the share data key, the share structure keys are dependent upon each other.

The index broadcasting scheme is enabled on the client application by the two checkboxes, *chkSingularIndex* and *chkDualIndex*. When the two checkboxes are selected, a four-step process is periodically executed:

1. Partially disconnects and turns off monitor when application is idle
2. Implements index broadcasting scheme to interpret structure keys
3. Implements index broadcasting scheme to interpret data keys
4. Implements data broadcasting scheme to filter incoming data items

The *chkSingularIndex* is applied automatically whenever *chkDualIndex* is selected. However, the client can select the *chkSingularIndex* independently to only enable steps 1, 3 and 4 above. Figure B.14 illustrates the location of the checkboxes, *chkSingularIndex* and *chkDualIndex* in the client application for dual index scheme.

The client application conserves power by being able to switch off the monitor and disconnect (completely) when the share structure key is not required. Power is conserved when the client application is disconnected (completely) because data is not to be received, therefore the source code usually required to filter out the incoming data is not executed. Figure B.15 (a, b and c) depicts the image when client application is executing indexing method based on the following scenario.

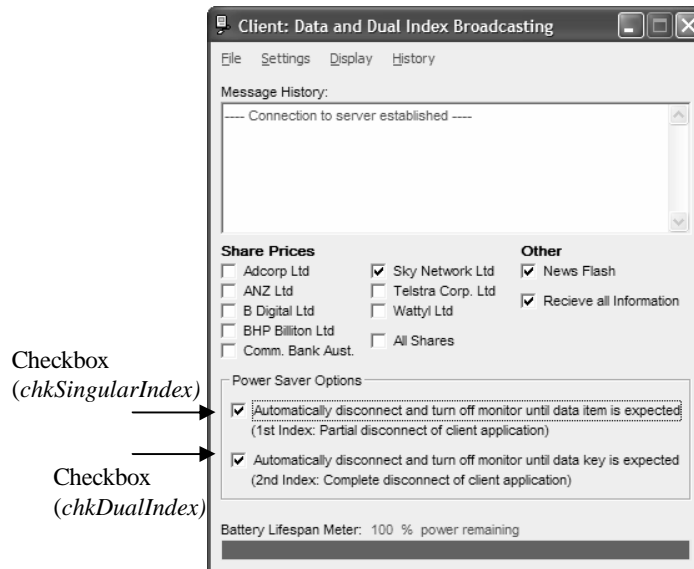


Figure B.14. *chkSingularIndex* and *chkDualIndex* checkboxes

Scenario: User only require to receive the share price of Sky Network Limited

Steps executed by the client application when indexing is enabled:

1. Automatically partially disconnects and turns off monitor after five seconds idle
2. Receives share structure key $A^{1\text{to}4}(03)^{5\text{to}8}(23)$
3. Examines whether the user has checked any of the share price checkboxes
4. Detects that share price with the identification number six is required
5. Completely disconnects for a time interval of twenty-three seconds
 - Deliberately does not receive share structure keys or share data keys related to share prices with the identification number between one to four
6. Reconnects after the time interval of twenty-three seconds has passed
7. Receives share structure key $B^{5\text{to}6}(01)^{7\text{to}8}(11-16)$
8. Remains connected as share price with the identification number of six is broadcast in one second

9. Receives share structure key C^5to5(01)^6to6(06)
10. Completely disconnects for one second
 - Deliberately does not receive share data key for the share price with the identification number five.
11. Reconnects after the timer interval of one second has passed
12. Receives share data key for the share price with the identification number six
13. Executes steps required for data indexing and data filtering

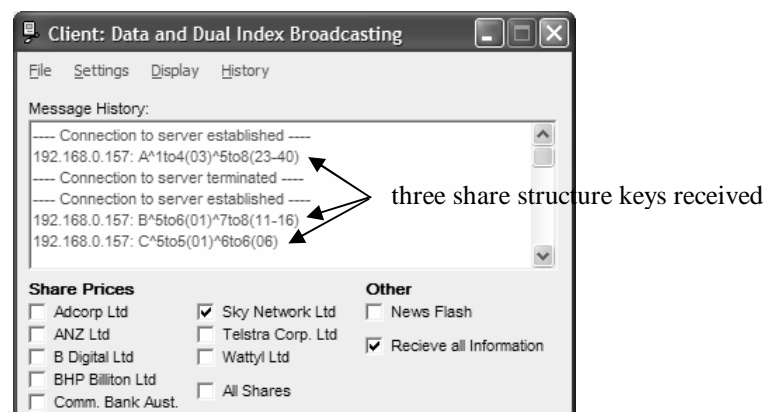


Figure B.15(a). Three share structure keys are received

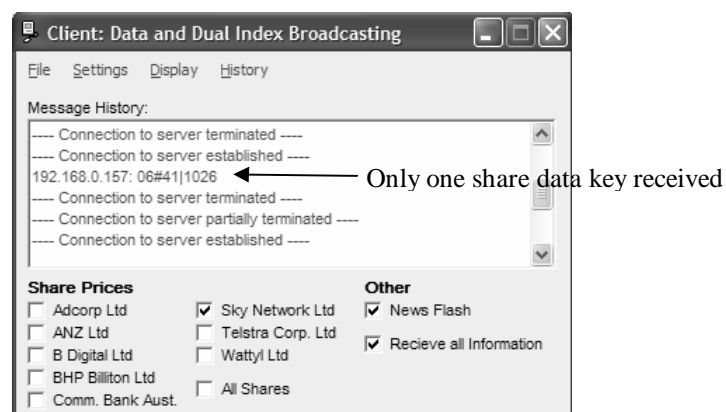


Figure B.15(b). One share data key is received

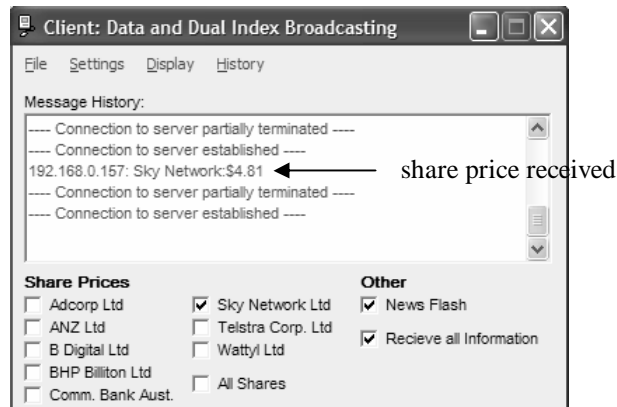


Figure B.15(c). Client receives the desired share price

- **Multi-channelling Environment**

Both the data and index broadcasting schemes are capable of being used in a multi-channel environment. Figure B.16 illustrates when the multi-channelling function is executed in the client application. The image demonstrates the client application a) tuning to the designated channel of the required data; then b) tuning back to the index channel when the required share price is received.



Figure B.16. Displays the current channel

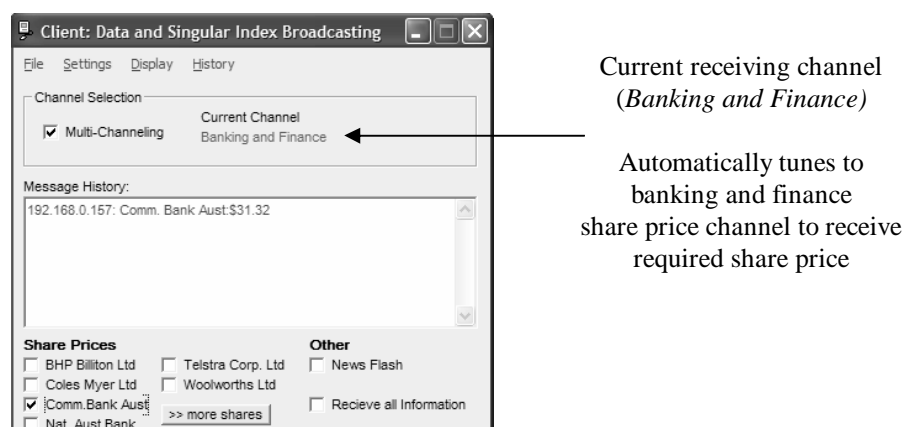


Figure B.16(a). Tunes to required channel, Share Prices: Banking and Finance

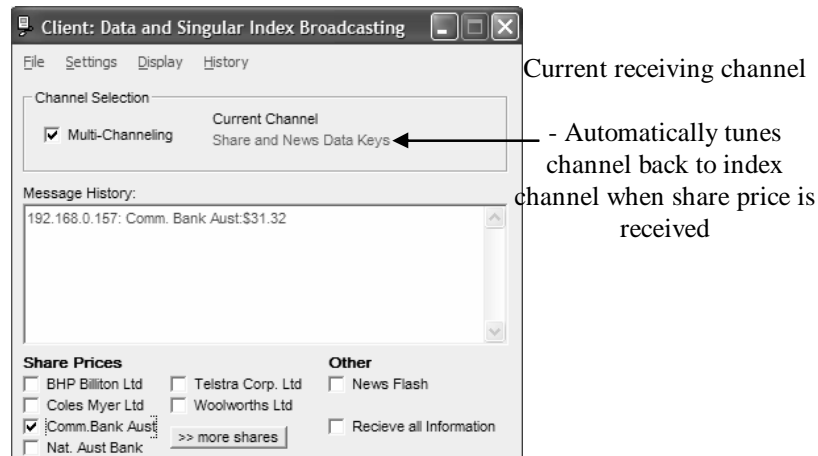


Figure B.16(b). Client application executing multi-channelling and data indexing

B.4 Sample Source codes for Server Implementation

- `chkPeriodicBroadcastShares_Click()`

```
Private Sub chkPeriodicBroadcastShares_Click()

    ' ** Notes:
    ' ** - Periodically broadcasts all shares
    ' ** Either
    ' ** - 1) Calls timer control, tmrPeriodicBroadcast
    ' ** - 2) Disables timer control, tmrPeriodicBroadcast

    'If checkbox is selected, enter condition statement
    If chkPeriodicBroadcastShares.Value = 1 Then

        'Set interval of timer control to 60000
        tmrPeriodicBroadcast.Interval = 60000
        'Enable timer control, tmrPeriodicBroadcast
        tmrPeriodicBroadcast.Enabled = True

    'If checkbox is not selected enter condition statement
    Else

        'Disable timer control, tmrPeriodicBroadcast
        tmrPeriodicBroadcast.Enabled = False

    End If

End Sub
```

- `cmdBroadcastAllShares_Click()`

```
Private Sub cmdBroadcastAllShares_Click()

    ' ** Notes:
    ' ** Broadcasts all shares prices in entire collection
    ' ** 1) Disable share price command buttons
    ' ** 2) Retrieves share prices from data source, MS Access
    ' ** 3) Call timer to enable share buttons
```

```

'Disable Share Price command buttons
'- Therefore unable to broadcast share prices
'- Ensure that all share prices are broadcast
cmdBroadcastShare.Enabled = False
cmdBroadcastAllShares.Enabled = False

'** Data source: Microsoft Access Database
'Broadcasts All Shares in the Categories L to Z
'Moves to first record in Active Data Object (ADO) recordset
adoShares.Recordset.MoveFirst

'Loop stops when the ADO recordset reaches the end
Do Until adoShares.Recordset.EOF = True

    'Calls the cmdBroadcastShare_Click method
    cmdBroadcastShare_Click
    'Moves to next record in ADO recordset
    adoShares.Recordset.MoveNext

Loop

'Enable all share price buttons when broadcasting finished
tmrEnableShareButtons.Interval = 60000
tmrEnableShareButtons.Enabled = True

End Sub

```

- **cmdBroadcastShare_Click()**

```

Private Sub cmdBroadcastShare_Click()
On Error GoTo ErrorHandler

    '** Notes:
    '** Method called when Individual Share Price broadcast
    '** - 1) Stores share price key ID of selected share
    '** - 2) Stores share price of selected share
    '** - 3) Calls sendDataKey method (Broadcast Share Key)
    '** - 3) Stores interval for selected share price
    '** - 4) Calls timer control array, tmrSharePrice(Index)
    '** - Timer broadcasts the share price

    'Declared string to store share price interval
    Dim strShareInterval As String

    35 share prices can be sent simultaneously
    'If there is greater than 35 share prices, earliest is overwritten
    If intShareCounter = 36 Then
        intShareCounter = 1
    End If

    'Retrieves the share price key value from the data source
    strSharePriceArray(intShareCounter, 1) = adoShares.Recordset!fldID
    'Retrieves the share price value from the data source
    'Stores the share price in a 2-dimensional array
    strSharePriceArray(intShareCounter, 2) =
adoShares.Recordset!fldAbbreviation + ":" +
adoShares.Recordset!fldPrice

```

```

**Calls the sendDataKey method
  'Method is used to send the Share Price Data Key
sendDataKey (strSharePriceArray(intShareCounter, 1))

  'Store the share interval as a string
  '- Break share price key so interval is separate
strShareInterval = Mid(adoShares.Recordset!fldID, 4, 4)

**Value of counter determines timer control that is manipulated
  'Interval of share key timer is made to equal interval of share key
tmrSharePrice(intShareCounter).Interval = Mid(strShareInterval, 1,
  2) * 1000
  'tmrSharePrice(Index), Timer Control Array is enabled
tmrSharePrice(intShareCounter).Enabled = True
  'The share price counter is incremented by 1
  '- Ensure timers are not overwritten, unless when greater then 35
  intShareCounter = intShareCounter + 1

ErrorHandler:

  If Err.Number <> 0 Then
    'Calls method to provide appropriate message boxes if errors
    'occur
    processErrors (Err.Number)
    'Continue resuming application despite error
    Resume Next
  End If

End Sub

```

- **sendDataKey(String)**

```

Private Sub sendDataKey(strData As String)

  ** Notes:
  ** - Broadcasts the Data Keys
  ** - Broadcasts Standard Messages
  ** 1) Data Channels are: 1001, 1002 and 1003
  ** 2) Data Key Channels are: 1006 and 1007

  'Set the address of the winsock control to broadcast address
wskUDP.RemoteHost = "255.255.255.255"

  **Data Broadcast Port
  'Change remote port to 1001
wskUDP.RemotePort = 1001
  'Broadcast the Share Price retrieved from share price array
wskUDP.SendData strData

  ** Data Broadcast Port
  'If Port 1002 is checked enter condition statement
  If chkPort1.Value = 1 Then

    'Change remote port to 1002
    wskUDP.RemotePort = 1002
    'Broadcast the incoming data, strData to network
    wskUDP.SendData strData
  End If

```

```

** Data Broadcast Port
'If Port 1003 is checked enter condition statement
If chkPort2.Value = 1 Then

    'Change remote port to 1003
    wskUDP.RemotePort = 1003
    'Broadcast the incoming data, strData to network
    wskUDP.SendData strData

End If

** Data Key Broadcast Port
'If Port 1006 is checked enter condition statement
If chkPort5.Value = 1 Then

    'Change remote port to 1006
    wskUDP.RemotePort = 1006
    'Broadcast the incoming data, strData to network
    wskUDP.SendData strData

End If

** Data Key Broadcast Port
'If Port 1007 is checked enter condition statement
If chkPort6.Value = 1 Then

    'Change remote port to 1007
    wskUDP.RemotePort = 1007
    'Broadcast the incoming data, strData to network
    wskUDP.SendData strData

End If

End Sub

```

- **sendShareBodyData(Integer)**

```

Private Sub sendShareBodyData(intIndex As Integer)

    ** Notes:
    ** - Broadcasts the Share Price: ONLY
    ** 1) Data Channels are: 1001, 1002, 1003
    ** 2) Share Channels range from: 1010 to 1019

    'Set the address of the winsock control to broadcast address
    wskUDP.RemoteHost = "255.255.255.255"

    **Data Broadcast Port
    'Change remote port to 1001
    wskUDP.RemotePort = 1001

    'Broadcast the Share Price retrieved from share price array
    wskUDP.SendData strSharePriceArray(intIndex, 2)

    ** Data Broadcast Ports
    'If Port 1002 is checked enter condition statement
    If chkPort1.Value = 1 Then

```

```

'Change remote port to 1002
wskUDP.RemotePort = 1002
'Broadcast the Share Price retrieved from share price array
wskUDP.SendData strSharePriceArray(intIndex, 2)

End If

'** Data Broadcast Port
'If Port 1003 is checked enter condition statement
If chkPort2.Value = 1 Then

    'Change remote port to 1003
    wskUDP.RemotePort = 1003
    'Broadcast the Share Price retrieved from share price array
    wskUDP.SendData strSharePriceArray(intIndex, 2)

End If

'** Share Price Broadcast Port
'Port number specified in share price key stored as an integer
Dim intPortNumber As Integer
intPortNumber = Mid(strSharePriceArray(intIndex, 1), 7, 10)

'Changes remote port to port specified in share price key
wskUDP.RemotePort = intPortNumber

'Broadcasts the Share Price retrieved from share price array
wskUDP.SendData strSharePriceArray(intIndex, 2)

'** Share Price Broadcast Port
'Broadcasts the data to the next port (increment of 1)
'Corresponds to the Share Key Port
'- (eg. key states port is 1001, therefore next port is 1002)
'Increment port number by 1
intPortNumber = intPortNumber + 1
'Change remote port number by an increment of 1
wskUDP.RemotePort = intPortNumber
'Broadcast the Share Price retrieved in share price array
wskUDP.SendData strSharePriceArray(intIndex, 2)

End Sub

```

B.5 Sample Source codes for Client Implementation

- processDataFiltering(String)

```

Private Sub processDataFiltering(strIncomingData As String)

'** Notes:
'** - Method called directly from wskUDP_DataArrival(Long) method
'** - Displays incoming data according to user's preference
'** - Incoming data displayed in textbox, txtHistory

'Declaration of two counters: Share and Character counter
Dim intShareCounter As Integer
Dim intCharacterCounter As Integer

```

```

'Declaration of string: Data displayed on textbox, txtHistory
Dim strDisplayData As String

'If checkbox, chkAll is checked then enter condition statement
If chkAll.Value <> 1 Then

    'Share counter is equal to one
    intShareCounter = 1

    'Assumption: Only 30 share prices exist
    'Loops until share counter is less than sixty-two
    Do While intShareCounter < 31

        'If any strListOfSharesArray(integer, 2) = 1 then enter
        'condition statement
        ' - Equivalent to if any share checkboxes are checked
        If strListOfSharesArray(intShareCounter, 2) = "1" Then

            'Character counter is equal to one
            intCharacterCounter = 1

            'Declaration of integer: Used to determine length of
            'share price
            Dim intShareLength As Integer

            'Loop until character counter is less then length of
            'incoming data
            Do While intCharacterCounter <= Len(strIncomingData)

                'intShareLength is equal to the length of share
                'price
                intShareLength =
                Len(strListOfSharesArray(intShareCounter, 1))

                'If incoming data is equal to share price, enter
                'condition statement
                If Mid(strIncomingData, intCharacterCounter,
                intShareLength) =
                strListOfSharesArray(intShareCounter, 1) Then

                    intShareLength = intShareLength + 7

                    'String, strDisplayData is equal to the share
                    'price plus 7 characters after
                    strDisplayData = Mid(strIncomingData,
                    intCharacterCounter, intShareLength)

                    '**Calls method, addIncomingText
                    'Displays string, strDisplayData on txtHistory
                    addIncomingText strDisplayData, txtHistory

                End If

                'Increment character counter by one
                intCharacterCounter = intCharacterCounter + 1

            Loop

```

```

    End If

    'Increment share counter by one
    intShareCounter = intShareCounter + 1

    Loop

    'Enters this condition statement if checkbox, chkAll is checked
    '- Applies no filtering
    Else

        '**Calls method, addIncomingText
        'Displays incoming data on textbox
        addIncomingText strIncomingData, txtHistory

    End If

End Sub

```

- processShareDataKey(String)

```

Private Sub processShareDataKey(strIncomingData As String)

    '** Notes:
    '** - Method called directly from wskUDP_DataArrival(Long) method
    '** - Interprets the share data key
    '** - If any share price selected to be recieved by user

    'Declaration of two counters
    Dim intShareCounter As Integer
    Dim intCounter As Integer

    'Stores Share Data Key
    Dim strShareKey As String
    'Stores Port number
    Dim strPort As String
    'Stores the interval of news data key
    Dim timeInterval As Double

    'Share counter is equal to one
    intShareCounter = 1

    '** One share price broadcast per message
    'Maximum of 61 share prices can recieved simultaneously
    Do While intShareCounter < 31

        'If checkbox is checked, i.e. value of one
        If strListOfSharesArray(intShareCounter, 2) = 1 Then

            'Counter is equal to one
            intCounter = 1

            'Loop until character counter is less then length of
            'incoming data
            Do While intCounter <= Len(strIncomingData)

                'If incoming data is equal to #, enter condition
                'statement
            End Do
        End If
    End Do
End Sub

```

```

If Mid(strIncomingData, intCounter, 1) = "#" Then

    'String, strShareKey is equal to the share data key
    strShareKey = Mid(strIncomingData, intCounter - 2, 5)
    'Stores the port number from the data key
    strListOfSharesArray(Mid(strShareKey, 1, 2), 3) =
    Mid(strIncomingData, intCounter + 4, 4)

End If

    'Increments counter by one
    intCounter = intCounter + 1
Loop

'Counter is equal to one
intCounter = 1

'Loops until counter reaches 31
Do While intCounter < 31

    'Interprets to check if user requires data
    '- Enters condition statement if user requires data
    If Mid(strShareKey, 1, 2) = intCounter And
    strListOfSharesArray(intCounter, 2) = 1 Then

        'Calculates and stores time interval for shares
        timeInterval = Mid(strShareKey, 4, 5)
        timeInterval = timeInterval - 0.45

        'Stores the port number into port number array
        intPortArray(intCounter) =
        strListOfSharesArray(intCounter, 3)

        'Sets interval and enable timer control
        tmrRecieveShareData(intCounter).Interval =
        timeInterval * 1000
        tmrRecieveShareData(intCounter).Enabled = True

    End If

    'Increments counter by one
    intCounter = intCounter + 1

Loop

'Assumed only one share price in message
'- Therefore share counter equal to 31
intShareCounter = 31

End If

'Increments share counter by one
intShareCounter = intShareCounter + 1

Loop
End Sub

```