

2004

A mobile database design approach in problem-based learning

Sunita Sharma
San Jose State University

Follow this and additional works at: https://scholarworks.sjsu.edu/etd_theses

Recommended Citation

Sharma, Sunita, "A mobile database design approach in problem-based learning" (2004). *Master's Theses*. 2622.
DOI: <https://doi.org/10.31979/etd.p4s9-k9d2>
https://scholarworks.sjsu.edu/etd_theses/2622

This Thesis is brought to you for free and open access by the Master's Theses and Graduate Research at SJSU ScholarWorks. It has been accepted for inclusion in Master's Theses by an authorized administrator of SJSU ScholarWorks. For more information, please contact scholarworks@sjsu.edu.

**A MOBILE DATABASE DESIGN APPROACH IN
PROBLEM-BASED LEARNING**

A Thesis

Presented To

The Faculty of the Department of Computer Engineering

San Jose State University

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

by

Sunita Sharma

August 2004

UMI Number: 1424493

Copyright 2004 by
Sharma, Sunita

All rights reserved.

INFORMATION TO USERS

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleed-through, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

UMI[®]

UMI Microform 1424493

Copyright 2005 by ProQuest Information and Learning Company.

All rights reserved. This microform edition is protected against
unauthorized copying under Title 17, United States Code.

ProQuest Information and Learning Company
300 North Zeeb Road
P.O. Box 1346
Ann Arbor, MI 48106-1346

© 2004

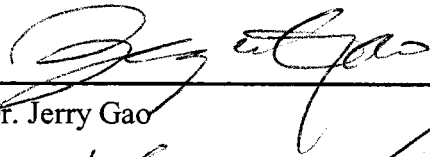
Sunita Sharma

ALL RIGHTS RESERVED

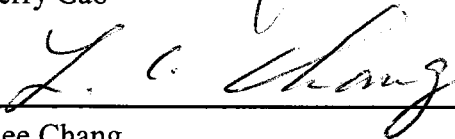
APPROVED FOR THE DEPARTMENT OF COMPUTER ENGINEERING



Dr. Weider Yu, Thesis Advisor



Dr. Jerry Gao



Dr. Lee Chang

APPROVED FOR THE UNIVERSITY



ABSTRACT

A MOBILE DATABASE DESIGN APPROACH IN PROBLEM-BASED LEARNING

by Sunita Sharma

Mobile devices have been in existence for a few years, however in the last couple of years they have started playing a significant role in our personal and professional life. A number of mobile applications and services are already on the market and more are expected to become available in the future. As more demand is placed on mobile devices, it will become critical to properly manage the limited storage, display, and processing power of mobile devices.

The purpose of this thesis is to research mobile computing and the role of mobile databases in mobile computing environments. It aims to identify methodologies for a good mobile database design and will apply these to design a problem-based learning application.

ACKNOWLEDGEMENTS

I am greatly indebted to my advisor, Dr. Weider Yu, for the constant guidance and support he provided in the course of this research. I feel very honored to have worked with him on this thesis.

I would also like to sincerely thank the members of my thesis committee, Dr. Jerry Gao and Dr. Lee Chang, for taking time out of their busy schedules and providing invaluable feedback on my research.

Finally, I would like to thank my husband, Tarun Bharti, and son, Mihir Bharti, for their understanding and support.

TABLE OF CONTENTS

List of Tables	viii
List of Figures	ix
1.0 Introduction.....	1
1.1 Mobile Computing.....	1
1.1.1 Architecture.....	2
1.1.2 Advantages.....	4
1.1.3 Disadvantages.....	5
1.1.4 Challenges.....	6
1.2 Research Overview.....	7
1.3 Organization of Chapters.....	8
2.0 Mobile Transactions.....	10
2.1 Transaction Models.....	15
2.1.1 Kangaroo.....	15
2.1.2 Pro-Motion.....	17
2.1.3 Cluster.....	18
2.1.4 Multi-Database.....	19
2.1.5 Transaction Model Comparison.....	22
2.2 Optimization Techniques.....	30
2.2.1 Compression.....	30
2.2.2 Summarization.....	31
2.2.3 Caching.....	34
3.0 Mobile Database.....	36
3.1 Architecture.....	37
3.2 Database Management Systems.....	41
3.2.1 IBM DB2 Everyplace.....	41
3.2.2 Oracle9i Lite.....	42
3.2.3 SQL Server for Windows CE.....	43
3.2.4 Sybase UltraLite.....	44
3.2.5 Mobile DBMS Comparison.....	46
3.3 Mobile Database Design Methodology.....	47
3.4 Mobile Database Design Factors and Potential Solutions.....	54
3.4.1 Responsiveness.....	55
3.4.2 Data Consistency and Concurrency.....	56
3.4.3 Synchronization and Conflict Resolution.....	57
3.4.4 Security.....	57
3.4.5 High Availability.....	58
4.0 Problem-Based Learning Environment.....	59
4.1 Process Diagram.....	61
4.2 Advantages.....	62
4.3 Disadvantages.....	63

4.4 Challenges	64
4.5 High Level Design	67
5.0 Analysis and Conclusions	81
References	87
Appendix A: Glossary.....	93
Appendix B: Abbreviations	95
Appendix C: User Interface	97
Appendix D: Database Schema	112
Appendix E: Use Cases.....	125

LIST OF TABLES

Table 1: ACID Attributes of Transaction Models	22
Table 2: Side-by-Side Comparison of Transaction Models.....	22
Table 3: Transaction Model Evaluation Scenario 1 – Kangaroo	24
Table 4: Transaction Model Evaluation Scenario 2 - Pro-Motion.....	26
Table 5: Transaction Model Evaluation Scenario 3 - Multi-Database.....	27
Table 6: Transaction Model Evaluation Scenario 4 - Cluster.....	29
Table 7: Mobile Database Management System Comparison	46
Table 8: PBL Application Database Entities – Setup	68
Table 9: PBL Application Database Entities – Transactional.....	69
Table 10: PBL Application Setup User Interfaces	73
Table 11: PBL Application Mobile User Interfaces.....	74
Table 12: PBL User Interface and Entity Mapping.....	76
Table 13: PBL Departments Setup.....	77
Table 14: PBL Users Setup.....	78
Table 15: PBL Lookups Setup	78

LIST OF FIGURES

Figure 1: Mobile Computing Model	3
Figure 2: A Typical Mobile Computer System.....	4
Figure 3: Mobile Computing - Advantages, Disadvantages, and Challenges.....	7
Figure 4: Online Mobile Transaction Scenario.....	13
Figure 5: Offline Mobile Transaction Scenario	14
Figure 6: Basic Structure of Kangaroo Transaction.....	16
Figure 7: Pro-Motion System Architecture.....	17
Figure 8: Multi-Database Transaction Processing Manager Architecture	20
Figure 9: Global Transaction Manager	21
Figure 10: Model Characteristics and Preference Matching - Kangaroo.....	25
Figure 11: Model Characteristics and Preference Matching - Pro-Motion.....	27
Figure 12: Model Characteristics and Preference Matching - Multi-Database.....	28
Figure 13: Model Characteristics and Preference Matching – Cluster	30
Figure 14: Mobile Database Architecture with Summary Databases	31
Figure 15: Mobile Database Environment.....	38
Figure 16: Client-Server Database Architecture.....	39
Figure 17: Distributed Database Architecture.....	39
Figure 18: Push-Pull Approach.....	41
Figure 19: IBM DB2 Everyplace Architecture	42
Figure 20: Oracle9iLite Architecture	43
Figure 21: SQL Server for Windows CE Architecture	44
Figure 22: Sybase UltraLite Architecture	45
Figure 23: Database Design Process.....	48
Figure 24: Detailed Design Database Step.....	50
Figure 25: Mobile Database Design Problems and Potential Solutions	55
Figure 26: Problem-Based Learning Process Diagram.....	61
Figure 27: PBL - Advantages, Disadvantages, and Challenges.....	66
Figure 28: Basic SQL Server CE Architecture	67
Figure 29: PBL Entity Relationship Diagram.....	72
Figure 30: Load Balancing.....	86

1.0 INTRODUCTION

Mobile applications and services have been in existence for more than five years, however, not necessarily in their current form. They have received greater emphasis and importance in recent years. A vast amount of information is already available from the World Wide Web and can be made available to the wireless users with the help of technologies such as cHTML, XML, VXML, and WML.

Wireless providers are realizing the market potential and are trying to provide applications and services to fulfill the needs of mobile device users. At the same time, they are also increasing customer loyalty by differentiating themselves in a highly competitive wireless market. In other words, providers and their partners are trying to offer more than just the phone service by promoting mobile lifestyles. The mMode service from AT&T Wireless is a perfect example of one such initiative (AT&T Wireless, 2004).

To summarize, usage of mobile devices is on the rise and wireless providers are coming up with new mobile applications and services. It is clearly an emerging market with very high potential for growth.

1.1 Mobile Computing

Mobile computing refers to the applications and services that individuals can use while on the move using their mobile devices, such as laptops, tablet PCs, cell phones, and PDAs. Forman (1994) suggests, “Mobile computing – the use of a portable computer capable of wireless networking – will very likely revolutionize the way we use computers” (p. 38). Increasingly, the global workforce is becoming mobile and

consumers are demanding more from their mobile devices. Providing mobility to these users is the primary factor for development of new applications and services in this area. The latest advances in technology have also allowed manufacturers to add more functionality and computing power to mobile devices, while making them affordable.

Mobile applications help businesses perform their tasks more efficiently and effectively. An example of a commercial mobile application is the handheld device that car rental companies use to print on-the-spot receipts for individuals returning their cars to the drop off locations. Consumers also enjoy the convenience offered by mobile applications. An example of a consumer mobile application is a cell phone that can be used to search for local restaurants, make reservations, and receive personalized driving directions. Other mobile applications and services available on the market include games on cell phones, custom weather forecasts, navigation, live traffic information, emergency assistance, security, theft control, location sensitive billing, local advertising, and travel services. More recently, the focus has been on using the location of a mobile user as the basis for providing specific services. Commercial activity based on the geographic location of the consumer increases the quality and amount of business. Clearly, technological advances in mobile computing are helping businesses connect to their most qualified customers.

1.1.1 Architecture

A mobile computing environment consists of mobile networks and a fixed network. One or more mobile devices and a mobile support station constitute a mobile network. The communication between a mobile device and fixed network takes place through a

mobile support station. Each mobile support station serves a fixed geographical area, also known as a cell. As a mobile device moves from one cell to another, the mobile support station in the first cell hands over the control to the mobile support station in the second cell. The following figure represents a mobile computing environment:

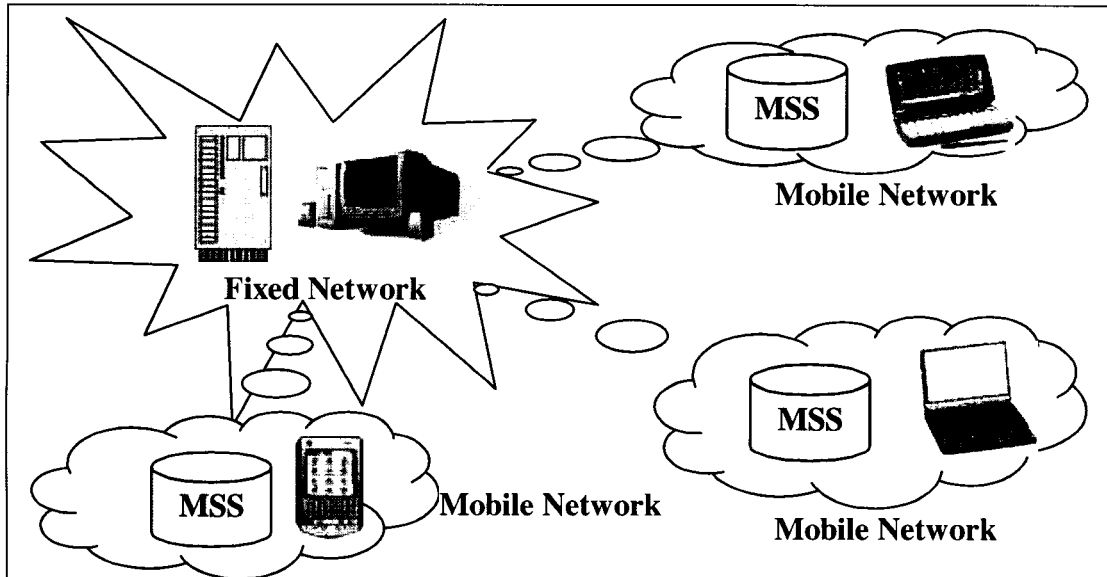


Figure 1: Mobile Computing Model

The distance between cells in a particular area depends upon the level of coverage needed in that area. For example, it is essential to provide good coverage in all locations within a metropolitan area. This can only be achieved if cells are close enough, so that the mobile device remains connected to the network all the time. On the other hand, the areas that need not have full coverage, the cells can be farther apart.

According to Racherla and Das (1996), in a typical mobile computing environment, cells may be operating in different frequencies, providing coverage for different types of mobile devices. The following figure demonstrates how two frequencies can provide enough bandwidth for four wireless users with two large

coinciding cells. Likewise, the same two frequencies can serve eight wireless users with four small non-interfering cells.

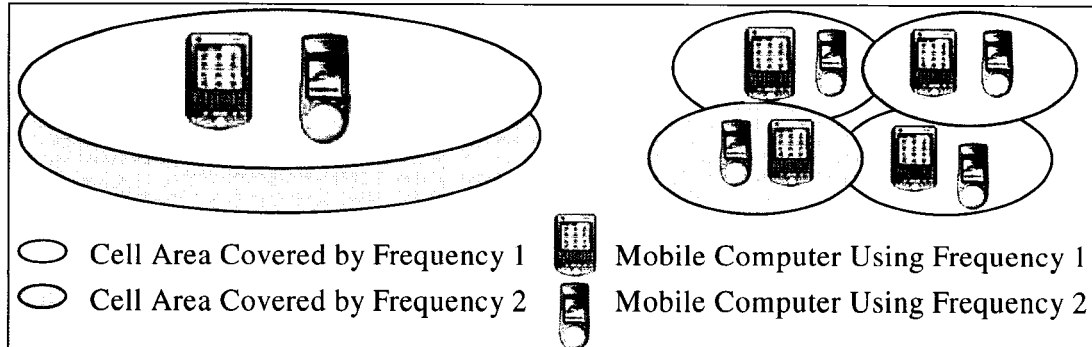


Figure 2: A Typical Mobile Computer System

Source: Racherla, G. & Das, A. (1996). Mobile Computing [Electronic Version]. *IEEE Potentials*, 13-15.

1.1.2 Advantages

Mobile applications and services offer many benefits, some of which are described below.

Convenience – They provide quick and easy access to the desired information, such as phone number lookup, movie information, and a TV Guide. They also provide, without the assistance of the local yellow pages directory, the ability to find a great restaurant in a neighborhood where the consumer happens to be driving.

Productivity – Productivity of the employees can be improved in many ways through the use of mobile services. For example, the customer care department of a company can keep track of its field technicians through GPS-enabled mobile phones. If a customer reports a problem, customer care personnel can dispatch a technician in close proximity of the customer (Thomas, 2004).

Safety – GPS-enabled mobile phones provide individuals with access to emergency services from any location and also help emergency personnel identify the precise location of the victim (FCC, 2001).

They also provide the consumers with the ability to get in-vehicle or roadside assistance. An example of such a service is the OnStar system, which comes preinstalled in luxury General Motors cars and provides services, such as roadside assistance and car phone (OnStar Corporate Information, 2003).

Savings – Consumers can receive customized advertising and marketing offers from manufacturers and stores, thereby saving them money on products and services. Moreover, consumers can save valuable time by using a mobile phone, PDA, or any other handheld device to quickly place orders.

Tracking – Mobile services allow for easy tracking of assets and individuals. Sales organizations can use them to trace and coordinate salespeople. Fleet managers can use them to better manage their fleet by tracking their drivers and vehicles (MobileIN, 2004).

1.1.3 Disadvantages

While mobile applications and services offer many benefits, they have some disadvantages as well.

Privacy – One of the major drawbacks of mobile services is that consumers are afraid of losing their privacy. Some personalized mobile services require consumers to provide search criteria, including their current location. Once the consumer provides this information, service providers may use it for other purposes (Appelbe, 2003).

Unsolicited Marketing – Based on the information available in public records, companies market their products via phone, email, direct mail, etc. Mobile devices capable of receiving text and graphic messages may be targeted to display advertisements for products and services that are of no interest to the consumer (Detecon, 2003).

1.1.4 Challenges

There is no doubt that mobile computing offers various benefits to its users. However, it poses serious technical challenges for the wireless infrastructure and service providers. According to Forman and Zahorjan (1994), the following are some of the challenges:

Complexity – Designing software for mobile computing is quite difficult as compared to designing software for stationary systems. One of the main issues is how to provide offline access when the mobile user may not be connected to the network.

Bandwidth – The bandwidth assigned to a mobile user can vary depending upon the number of users connected to the network. Moreover, some applications, such as turn-by-turn driving directions, may require a large amount of data transfer. Due to limited bandwidth, this can be a frustrating experience for the mobile user.

Infrastructure – A mobile device may need to work in different types of networks. Depending on the network in which it needs to work, transmission speeds and protocols may need to be changed.

Security – This is another major concern for mobile services. There is a probability of hackers intercepting radio signals and compromising the security of applications and data. In addition, these services are dependent upon a shared public

infrastructure where there is less control and awareness of the security that is being employed.

The following figure summarizes the advantages, disadvantages, and challenges of mobile computing:

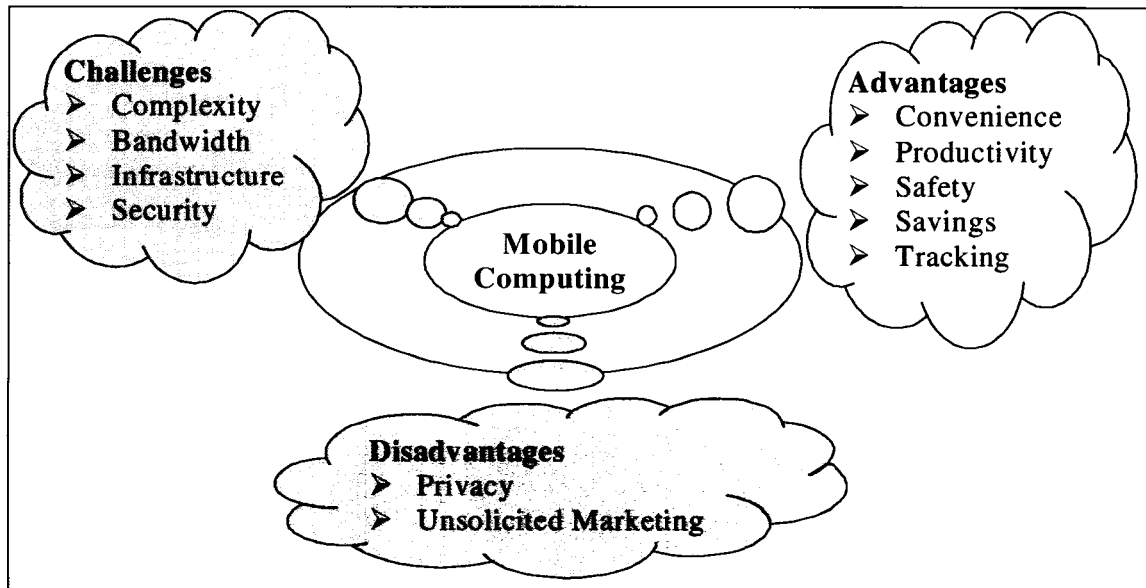


Figure 3: Mobile Computing - Advantages, Disadvantages, and Challenges

1.2 Research Overview

This research aims at developing an approach for a good mobile database design and applying it to design a mobile database to support a problem-based learning environment. Some of the key issues studied as part of this research are as follows:

- Mobile computing architecture, its advantages, disadvantages, and challenges
- Mobile database architecture and mobile database management systems currently available in the market
- Transaction models and optimization techniques used for mobile transactions
- Mobile database design process and methodologies

- Mobile database design problems and solutions
- Concepts of problem-based learning
- Advantages, disadvantages, and challenges of problem-based learning
- Design of problem-based learning application

1.3 Organization of Chapters

Chapter 1.0 provides an overview of the mobile computing architecture, along with its advantages, disadvantages, and challenges.

Chapter 2.0 discusses the mobile database transactions and their characteristics. It then provides details on various transaction models that can be employed in a mobile computing environment, including an evaluation model that can be used to determine the best transaction model given certain parameters. In addition, it details the optimization techniques employed in a mobile computing environment.

Chapter 3.0 describes the architecture of a mobile database. It gives a brief description of some of the mobile database management systems already available in the market followed by a side-by-side comparison of the same. It also summarizes the challenges faced when designing mobile databases and provides a methodology that can be followed while designing a mobile database. Then, it categorizes the problem areas and suggests ways to overcome them.

Chapter 4.0 defines problem-based learning and explains it through a process diagram. It further discusses the advantages, disadvantages, and challenges faced in problem-based learning environments. The database schema and entity relationship

diagram for the problem-based learning application have been developed to demonstrate the application of the mobile database design approach identified in earlier chapters.

Chapter 5.0 summarizes the results of the analysis carried out as part of this research and its applicability to the problem-based learning application. It also provides the conclusion of this research with some suggestions on further work that can be done in this area.

2.0 MOBILE TRANSACTIONS

A transaction can be defined as a set of operations that have some end result. In a database environment, a transaction can be a set of read and write operations completing with either a commit or a rollback. According to Clark and Demir (2003), the four attributes that define the characteristics of a transaction are atomicity, consistency, isolation, and durability. These are also known as ACID attributes of a transaction.

Atomicity – This means that all the operations that are part of a transaction are dependent on one another. Even if one of the operations fails, the whole transaction must be rolled back. A transaction is considered successful and can be committed only if all of its operations are successful.

Consistency – It defines the state of a database, which is free from data distortion. Every database transaction should preserve the data consistency. Data consistency can be achieved by defining the required constraints that dictate the relationship of data items.

Isolation – This means that every transaction should be processed independent of another transaction. In other words, a transaction should not interfere with the processing of any other transaction.

Durability – This means that once a transaction is committed, it becomes persistent in the database. Therefore, even in the event of a database failure, the changes performed by a transaction should not be lost.

Given above attributes, transaction management is a critical aspect of database computing. This is even more critical for mobile transactions due to the following issues:

Disconnection – Mobile devices may not always be connected to the central database server. While disconnected, a mobile application may be using stale data. It is also possible that the mobile device gets disconnected in the middle of a transaction. Appropriate mechanisms should be in place to take care of such unplanned disconnections. For example, forcing a complete roll back from every node in case of a disconnection will ensure atomicity of mobile transactions.

Limited Resources – Mobile devices are constrained by the limited resources they possess. They run on batteries, which may be discharged if some power-hungry process is running. They have limited storage, so we cannot store a large amount of data on the mobile device. They also have a small size, which restricts what can be displayed on the screen. In addition, they have limited computational power, so large calculations can take time.

If a large amount of data needs to be moved from a mobile device to the central database server and vice-versa, it may consume more power and bandwidth, resulting in poor battery life and slow performance. Proper care must be taken to avoid any long running transactions. Moreover, only a limited amount of data can be stored on the mobile device, so the available storage space should be used wisely. Keeping the small display size in mind, the data to be displayed to the user should be carefully selected. Furthermore, since mobile devices have limited computational power, large data processing should be performed at the central database server.

Consistency – Mobile transactions make it very difficult to maintain data consistency. The reason is that multiple mobile devices can update the same set of

cached data while disconnected from the central database server. Once the devices reconnect, all the local transactions need to be committed to the central database server, which can lead to conflicts. The data consistency can be ensured only if appropriate conflict resolution techniques are in place (Clark and Demir, 2003).

Validation Techniques – If mobile devices are engaged in updating data in the central database server, it becomes essential to have proper validation techniques. The old cached data on mobile devices makes this task even more complex. Database constraints and proper sanity checks can be added to validate the data (Clark and Demir, 2003).

Deadlock – A deadlock is a situation in which two transaction are waiting on resources held by one another. It is almost impossible to create a deadlock free transaction system, however, proper mechanisms could be devised to solve the deadlock. For instance, a timeout parameter could be used to limit the wait time to acquire a particular resource. Similarly, if a transaction fails in the middle, the resources held by the transaction should be freed so that they can be used by other transactions (Clark and Demir, 2003).

Security – Security is one of the major concerns for mobile transactions. Since the data transmission between the mobile device and the central database server takes place through a wireless network, unauthorized users may intercept and misuse the data. Therefore, the data should be properly secured through proper access control mechanisms and encryption techniques (Clark and Demir, 2003).

Fault Tolerance – It is vital to have a fault tolerant system, which is able to recover gracefully in case of a failure. The failure could be at any level – mobile device, mobile support station, or the central database server. Appropriate strategies should be in place to ensure recovery irrespective of where the failure occurs (Clark and Demir, 2003).

High Availability – Businesses are increasingly becoming dependent on databases. Therefore, it is critical to ensure around-the-clock availability of database applications. One of the techniques to increase availability is to replicate data across multiple stationary hosts, preferably at different sites. While this improves availability, it requires higher maintenance, as changes made to one host must be replicated to other hosts to ensure data consistency.

Mobile users may travel across cells, so a mobile transaction can be initiated from one cell and can be terminated on another. The following figure represents a typical online mobile transaction:

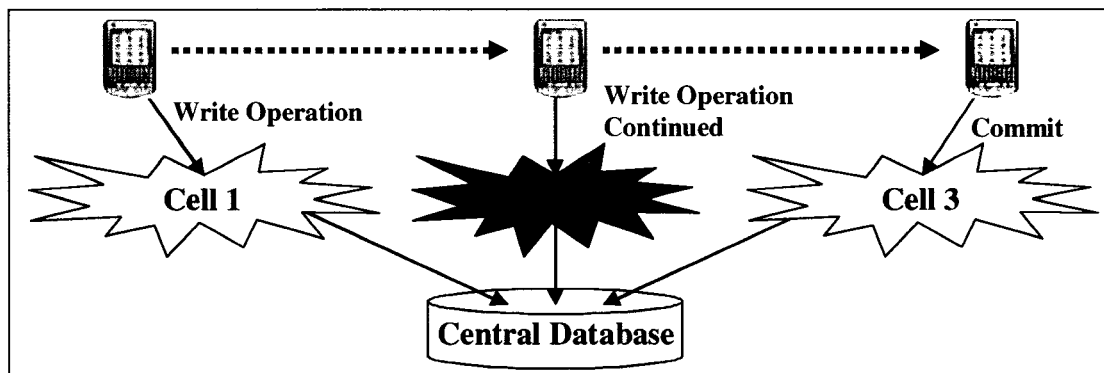


Figure 4: Online Mobile Transaction Scenario

From a mobile user's perspective, the following is taking place in the online mobile transaction scenario:

- The user connects from the first cell and begins a write operation.

- While the write operation is in progress, the user enters the second cell.
- The user completes the write operation, enters the third cell, and issues a commit.

The above transaction will complete successfully only if the operation initiated through each cell is saved properly. In other words, when the mobile user issues a commit or a rollback from the third cell, the operations from all of the three cells need to be applied.

In another scenario, a mobile user may be performing mobile transactions while disconnected from the network. In this case, a mobile transaction is initiated on the mobile device, however it is committed only when the device is connected to the network. The following figure represents a typical offline mobile transaction:

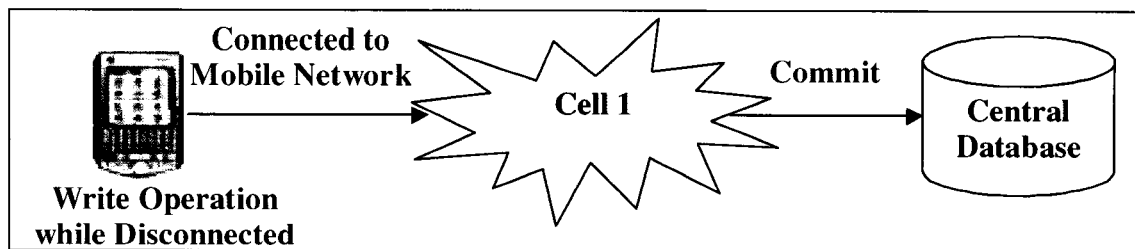


Figure 5: Offline Mobile Transaction Scenario

From a mobile user's perspective, the following is taking place in the offline mobile transaction scenario:

- The user performs a write operation on the mobile device. The data in the mobile database is updated to reflect that transaction.
- The user then connects to the mobile network and commits the transaction.

The transaction will complete successfully only if it does not conflict with any other ongoing or completed transaction. Once committed, the updated data will be available to other mobile users.

2.1 Transaction Models

Mobile devices may not be connected to the central database server all the time. Therefore, it is critical that ACID attributes of mobile transactions are maintained. Since traditional techniques do not function properly in mobile computing environments, new mechanisms are needed. According to Dunham, Helal, and Balakrishnan (1997), any mobile transaction model should satisfy the following conditions:

- It should use the built-in support provided by the multi-database systems as its foundation and then more features should be added.
- The transaction control should move along with the mobile devices.
- It should be flexible in supporting the atomicity of the transactions.
- It should support the long-lived transactions.

A number of transaction models have been developed to handle transactions in mobile computing environments. Some of the transaction models suitable for mobile computing environments are discussed below.

2.1.1 Kangaroo

This model intends to capture the moving nature of mobile devices. In this model, when a user initiates an operation, a Kangaroo Transaction is created, which serves as a unique identifier for that specific transaction. When the user moves from one cell to another, a sub-transaction called Joey Transaction is created at each cell. For example, if

a mobile transaction involves three cells, a total of three Joey Transactions will be created. One Joey Transaction encompasses one or more local and global transactions. A transaction is said to be Local if it is executed on the local database. A transaction is considered Global in a multi-database environment. The first Joey Transaction is usually a BEGIN transaction and the last one is a COMMIT or ABORT operation. Therefore, the last Joey Transaction is a deciding factor whether the Kangaroo Transaction should be committed or rolled back (Dunham, Helal, and Balakrishnan, 1997).

The following figure represents the basic structure of a Kangaroo Transaction:

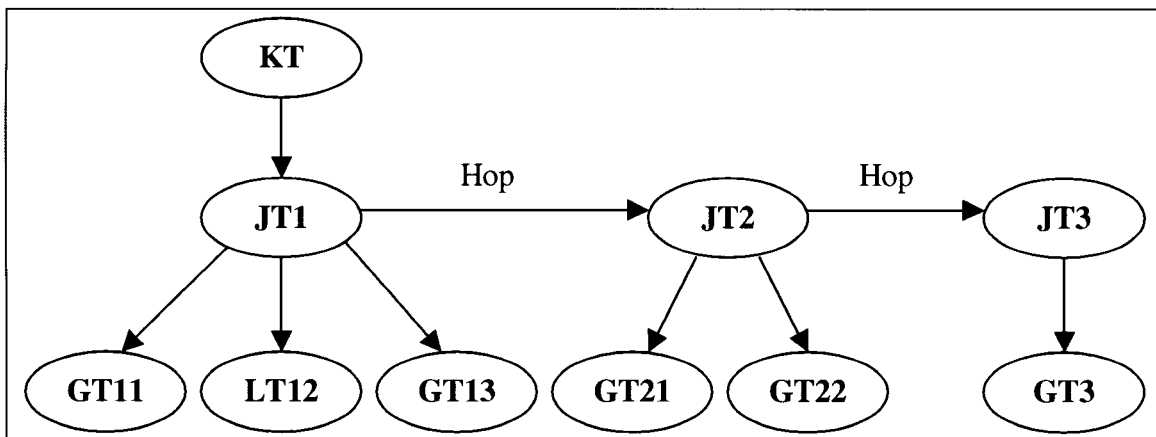


Figure 6: Basic Structure of Kangaroo Transaction

Source: Seydim, A. (1999). *An Overview of Transaction Models in Mobile Environments*. Retrieved December 29, 2003, from http://enr.smu.edu/~yasemin/mobile_trans.pdf

As demonstrated in the figure above, a Kangaroo Transaction is created at the originating base station. Also, a Joey Transaction is created to logically group all of the local and global transactions that occur in the first cell. Once the mobile device enters the second cell, a new Joey Transaction is created. Similarly, a new Joey Transaction is created as the mobile device enters the third cell. This Kangaroo Transaction is

considered to be successful if all three Joey Transactions are completed successfully. If any of these Joey Transactions fail, the whole Kangaroo Transaction should be aborted.

2.1.2 Pro-Motion

This model supports transactions in which mobile devices are disconnected from the central database server. Transactions are allowed through the data cached on mobile devices, but there are restrictions in terms of operations allowed and data expiration. The mobile devices are required to abide by the rules dictated by the central database server. The local transactions that occur on a mobile device are committed to the database server when it reconnects to the network. Once the local transactions for a mobile device are saved in the central database, other mobile devices can query the updated data. The following figure describes the Pro-Motion system architecture:

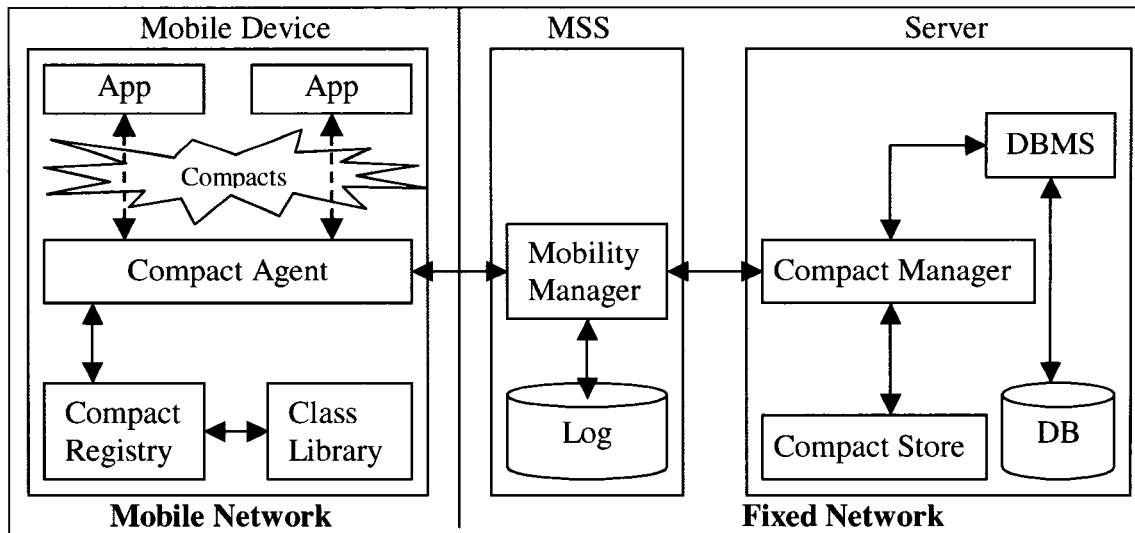


Figure 7: Pro-Motion System Architecture

Source: Seydim, A. (1999). *An Overview of Transaction Models in Mobile Environments*. Retrieved December 29, 2003, from http://engr.smu.edu/~yasemin/mobile_trans.pdf

As demonstrated in the figure above, the data is cached on mobile devices in the form of a “compact,” which represents the smallest unit of data available on a mobile device for local transaction management. The Compact Manager on the server creates and maintains compacts and interacts with the Compact Source to manage the assignments. The Compact Agent on the device helps manage the compacts locally. The communication between the Compact Agent and the Compact Manager is coordinated through the Mobility Manager, which resides on each Mobile Support Station. The server keeps track of the updates to data items contained in a compact. When a mobile device requires some data, it sends the request to server through the Mobility Manager. The server creates compacts based on this request and sends them to the mobile device through the mobility manager. In addition, the server makes an entry in the Compact Registry, indicating the mobile device, so that it can track any updates to these compacts by mobile devices. The Compact Agent manages the transactions that take place on a mobile device. Once the mobile device is disconnected from the server, it keeps a record of committed transactions. Once it reconnects, it sends that information to the server so that it can be committed in the central database server (Clark and Demir, 2003).

2.1.3 Cluster

Closely related data that is stored together is referred to as a “cluster.” The data may be clustered based upon its physical location and usage and may be stored on a single host or multiple hosts. According to Clark and Demir (2003), in a cluster model, mobile transactions can be categorized as follows:

Weak Transaction – This type of transaction accesses the data that resides in the same cluster. The transaction can either be committed in the cluster or globally. The global commit is allowed only when the relevant clusters merge. A mobile application that requires access to data within the cluster can use weak transactions. A mobile application that requires access to some global data may also use weak transactions as long as certain level of data inconsistency is permitted.

Strict Transaction – Only consistent data can be accessed using these transactions. Strict transactions are allowed to access data used by other strict transactions or by weak transactions that have been committed globally.

A cluster can span across fixed host and mobile devices. Due to the fact that mobile devices can be disconnected from fixed hosts most of the time, the clusters can be recreated or merged during reconnection. Also, upon traversal to a different cell, a mobile host can move to a different cluster. The data can also be clustered as needed by the users irrespective of its location. In other words, the data that is most often accessed by a particular user can form a cluster. The data can be replicated at different clusters. Although the data within a cluster needs to be consistent, different sets of rules can be applied to different clusters. The network bandwidth plays a major role in deciding the degree of consistency across clusters (Pitoura and Bhargava, 1995).

2.1.4 Multi-Database

This model aims at managing mobile transactions in a multi-database environment. It offers a global interface to multiple independent local databases, thus

providing the feel of a distributed database. The following figure represents the multi-database architecture:

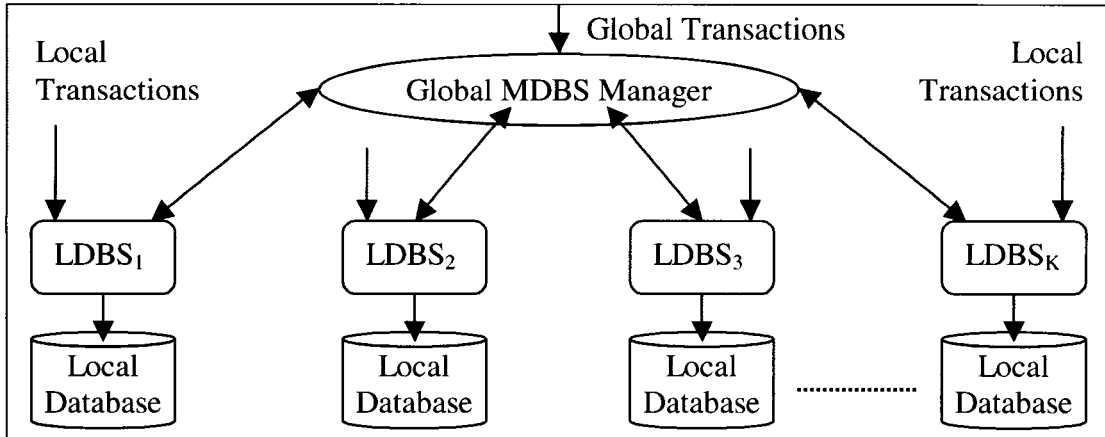


Figure 8: Multi-Database Transaction Processing Manager Architecture

Source: Segun, K., Hurson, A., Desai, V., Spink, A., & Miller, L. (2003). *Transaction Management in a Mobile Data Access System*. Retrieved April 23, 2004, from <http://www.comp.nus.edu.sg/~yuenck/35.pdf>

As demonstrated in the figure above, all mobile users can connect to a common global database management system without having to know the structure of the underlying local databases. This global interface takes care of routing requests from mobile devices to the corresponding local databases. Mobile devices connect to the mobile support stations and access the Mobile Multi-Database System using an interface known as Global Transaction Manager. The following figure provides the architecture of GTM:

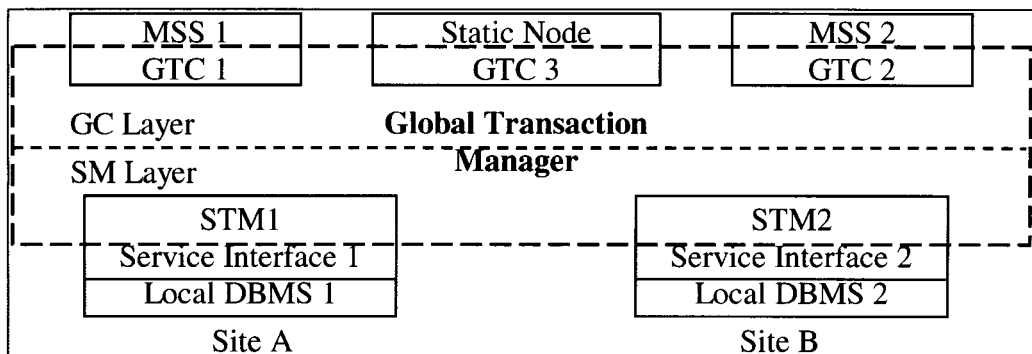


Figure 9: Global Transaction Manager

Source: Dirckze, R. & Gruenwald, L. (2000). A pre-serialization transaction management technique for mobile multidatabases [Electronic version]. *Mobile Networks and Applications*, 5, 311-321.

The GTM is responsible for coordinating the transactions executed by mobile devices on the MMDBS. The GTM consists of two components:

- Global Transaction Coordinator
- Site Transaction Manager

There is one instance of GTC running on each MSS and one instance of STM running locally on each database. STM manages the site-specific transactions at the local DBMS level. GTC takes care of forwarding global transactions to the corresponding STM. All of the transactions performed by mobile devices are communicated to the STM through the GTC. Similarly, the STM sends the results of a transaction commit or rollback to the GTC that, in turn, communicates them to the mobile devices. In addition, if a mobile device is disconnected, the GTC saves the communication from the STM and hands it to the mobile device once it is connected to the network. The GTC also takes care of aborting the failed transactions issued by a mobile device if it gets disconnected so

that they do not cause any problems for other mobile devices. In short, the GTC acts as a resource coordinator for mobile devices in its space (Clark and Demir, 2003).

2.1.5 Transaction Model Comparison

A side-by-side comparison of ACID attributes of above transaction models is provided in the table below:

Table 1: ACID Attributes of Transaction Models

Model	Atomicity	Consistency	Isolation	Durability	Execute In
Kangaroo	Maybe	No	No	No	Fixed Network
Pro-Motion	Yes	Yes	Yes	Yes	MU or Fixed Network
Cluster	No	No	No	No	MU or Fixed Network
Multi-Database	No	No	No	No	Mu or Fixed Network

Source: Seydim, A. (1999). *An Overview of Transaction Models in Mobile Environments*. Retrieved December 29, 2003, from http://enr.smu.edu/~yasemin/mobile_trans.pdf

According to Clark and Demir (2003), the transaction models discussed earlier possess different characteristics. A side-by-side comparison of the characteristics of these transaction models is provided in the table below:

Table 2: Side-by-Side Comparison of Transaction Models

Attributes	Transaction Models			
	Kangaroo	Pro-Motion	Multi-Database	Cluster
Online Transaction Management	Yes	No	No	No

Attributes	Transaction Models			
	Kangaroo	Pro-Motion	Multi-Databas e	Cluster
Offline Transaction Management	No	Yes	No	No
Works in Multi-tier Networks	Yes	Yes	Yes	No
Allows User Interaction	No	Yes	No	No
More Power Consumption	No	Yes	No	No
More Bandwidth Usage	No	Yes	No	No
Requires Fixed and Reliable Network	Yes	Yes	Yes	No
Local Transaction Management on Mobile Units	No	Yes	No	No
Automatic Transaction Timeout	No	Yes	Yes	No
High Data Transfer	No	Yes	No	No
Works in Distributed Database Environment	Yes	Yes	Yes	Yes
Support for Short Lived Transactions	Yes	Yes	Yes	Yes
Support for Long Lived Transactions	Yes	No	Yes	Yes

As part of this research, an evaluation model to find the best transaction model was developed. This evaluation model takes two user inputs: preference for a specific transaction model characteristic and its importance on a scale of one to ten, with one being the least desired and ten being the most desired. Based on the user inputs, the evaluation model derives a score for each transaction model. The transaction model with

the highest score is preferred. The following are some of the scenarios of this evaluation model:

Scenario 1 – In this scenario, the Kangaroo transaction model comes out to be the most suitable, given the user preferences for model characteristics and their importance.

Table 3: Transaction Model Evaluation Scenario 1 – Kangaroo

Model Characteristics	Transaction Models				User Inputs	
	Kangaroo	Pro-Motion	Multi-Database	Cluster	Preference	Importance (1-10)
Online Transaction Management	No	No	No	No	Yes	10
Offline Transaction Management	Yes	No	No	No	No	10
Works in Multi-tier Networks	Yes	Yes	No	No	Yes	10
Allows User Interaction	Yes	No	No	No	No	4
Consumes More Power	No	Yes	No	No	Yes	10
Uses More Bandwidth	Yes	Yes	No	No	No	6
Requires Fixed and Reliable Network	Yes	Yes	Yes	No	No	10
Local Transaction Management on Device	Yes	Yes	No	No	No	1
Automatic Transaction Timeout	Yes	Yes	No	No	No	10
High Data Transfer	No	Yes	No	No	Yes	9

Model Characteristics	Transaction Models				User Inputs	
	Kangaroo	Pro-Motion	Multi-Database	Cluster	Preference	Importance (1-10)
Works in Distributed Database Environment	Yes	Yes	Yes	Yes	Yes	2
Support for Short Lived Transactions	Yes	Yes	Yes	Yes	Yes	10
Score	63	41	43	53		
Preferred Transaction Model						

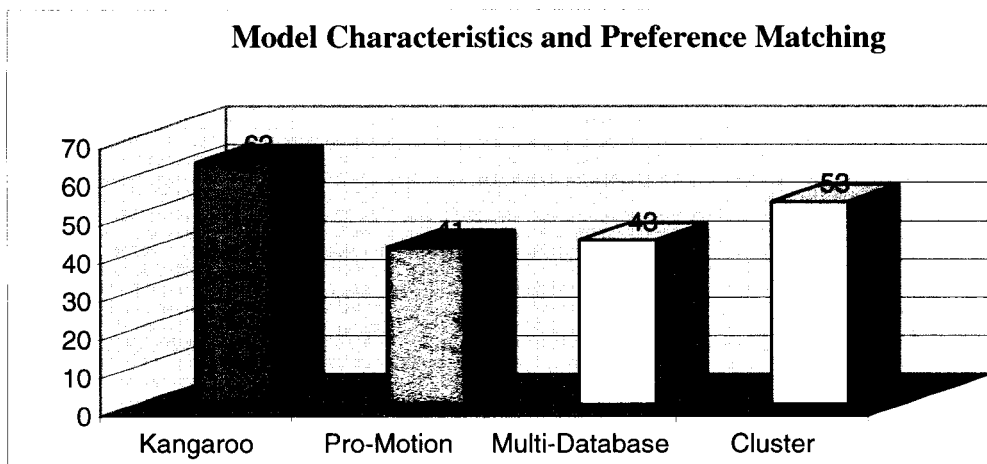


Figure 10: Model Characteristics and Preference Matching - Kangaroo

Scenario 2 – In this scenario, the Pro-Motion transaction model comes out to be the most suitable, given the user preferences for model characteristics and their importance.

Table 4: Transaction Model Evaluation Scenario 2 - Pro-Motion

Model Characteristics	Transaction Models				User Inputs	
	Kangaroo	Pro-Motion	Multi-Database	Cluster	Preference	Importance (1-10)
Online Transaction Management	Yes	No	No	No	No	10
Offline Transaction Management	No	Yes	No	No	Yes	10
Works in Multi-tier Networks	No	Yes	Yes	No	Yes	10
Allows User Interaction	No	Yes	No	No	No	4
Consumes More Power	No	Yes	No	No	Yes	10
Uses More Bandwidth	No	Yes	No	No	No	6
Requires Fixed and Reliable Network	Yes	Yes	Yes	No	No	10
Local Transaction Management on Device	No	Yes	No	No	No	1
Automatic Transaction Timeout	No	Yes	Yes	No	No	10
High Data Transfer	No	Yes	No	No	Yes	9
Works in Distributed Database Environment	No	Yes	Yes	Yes	Yes	2
Support for Short Lived Transactions	No	Yes	Yes	Yes	Yes	10
Score	43	61	43	53		
Preferred Transaction Model		Pro-Motion				

Model Characteristics and Preference Matching

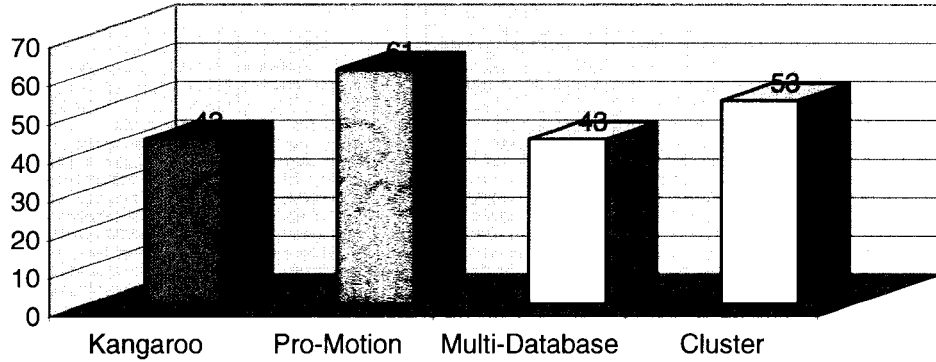


Figure 11: Model Characteristics and Preference Matching - Pro-Motion

Scenario 3 – In this scenario, the Multi-Database transaction model comes out to be the most suitable, given the user preferences for model characteristics and their importance.

Table 5: Transaction Model Evaluation Scenario 3 - Multi-Database

Model Characteristics	Transaction Models				User Inputs	
	Kangaroo	Pro-Motion	Multi-Database	Cluster	Preference	Importance (1-10)
Online Transaction Management	Yes	No	No	No	No	10
Offline Transaction Management	No	Yes	No	No	No	10
Works in Multi-tier Networks	No	Yes	Yes	No	Yes	10
Allows User Interaction	No	Yes	No	No	No	4
Consumes More Power	No	Yes	No	No	No	10
Uses More Bandwidth	No	Yes	No	No	No	6

Model Characteristics	Transaction Models				User Inputs	
	Kangaroo	Pro-Motion	Multi-Database	Cluster	Preference	Importance (1-10)
Requires Fixed and Reliable Network	No	Yes	Yes	No	Yes	10
Local Transaction Management on Device	No	Yes	No	No	No	1
Automatic Transaction Timeout	No	Yes	Yes	No	No	1
High Data Transfer	No	Yes	No	No	Yes	9
Works in Distributed Database Environment	No	Yes	Yes	Yes	Yes	2
Support for Short Lived Transactions	No	Yes	Yes	Yes	Yes	10
Score	64	51	73	54		
Preferred Transaction Model			Multi-database			

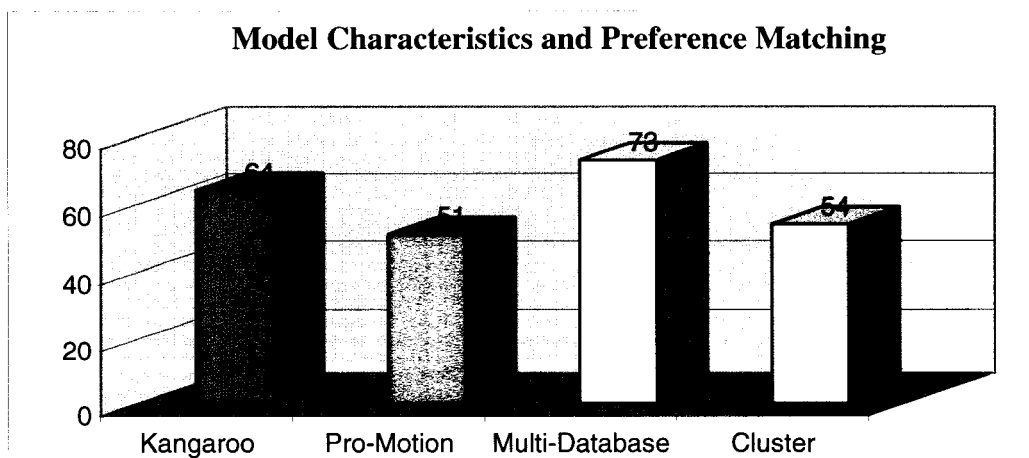


Figure 12: Model Characteristics and Preference Matching - Multi-Database

Scenario 4 – In this scenario, the Cluster transaction model comes out to be the most suitable, given the user preferences for model characteristics and their importance.

Table 6: Transaction Model Evaluation Scenario 4 - Cluster

Model Characteristics	Transaction Models				User Inputs	
	Kangaroo	Pro-Motion	Multi-Database	Cluster	Preference	Importance (1-10)
Online Transaction Management	Yes	No	No	No	No	10
Offline Transaction Management	No	Yes	No	No	No	10
Works in Multi-tier Networks	Yes	Yes	Yes	No	No	10
Allows User Interaction	No	Yes	No	No	No	4
Consumes More Power	No	Yes	No	No	No	10
Uses More Bandwidth	No	Yes	No	No	No	6
Requires Fixed and Reliable Network	No	Yes	Yes	No	Yes	10
Local Transaction Management on Device	No	Yes	No	No	No	1
Automatic Transaction Timeout	No	Yes	Yes	No	No	1
High Data Transfer	No	Yes	No	No	Yes	9
Works in Distributed Database Environment	No	Yes	Yes	Yes	Yes	2

Model Characteristics	Transaction Models				User Inputs	
	Kangaroo	Pro-Motion	Multi-Database	Cluster	Preference	Importance (1-10)
Support for Short Lived Transactions	Yes	Yes	Yes	Yes	Yes	10
Score	54	41	63	64		
Preferred Transaction Model				Cluster		

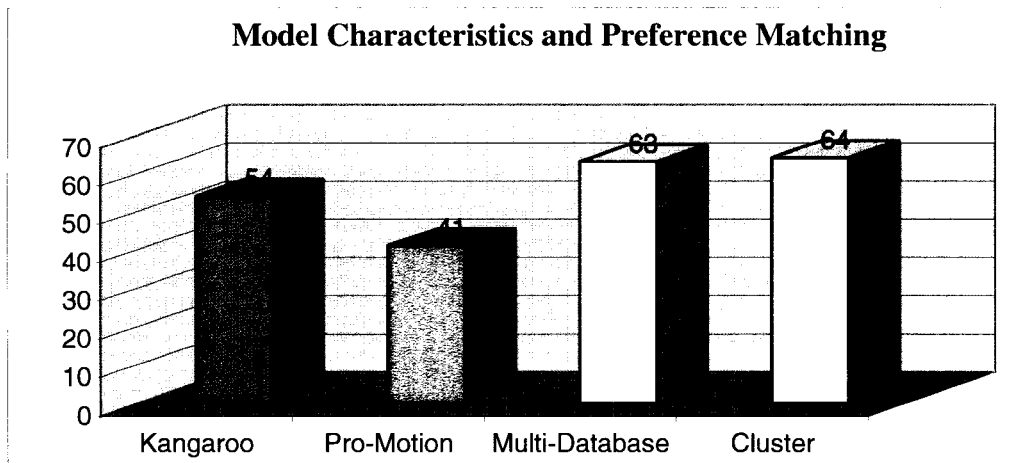


Figure 13: Model Characteristics and Preference Matching – Cluster

2.2 Optimization Techniques

As mentioned earlier, mobile devices are constrained with respect to their storage capacity, computing power, and display capabilities. Some of the techniques developed to overcome these constraints are compression, summarization, and caching. This section briefly discusses each of these techniques.

2.2.1 Compression

This technique is used to effectively manage the limited storage capacity on mobile devices. While compression reduces the amount of required storage, it requires

additional computing for decompression of the data needed by an application. The biggest disadvantage of this technique, however, according to Chan and Roddick (2003), is that it does not take data semantics into consideration.

2.2.2 Summarization

Mobile devices have limited storage, which requires the best use of available space by any mobile application. The space can be better managed by summarizing the relevant data and storing it on the mobile device. A summary database can help provide the information a mobile user may be looking for, even when reliable and secure connections to the central database are not available.

According to Chan and Roddick (2003), “Database summarization involves the reduction of the size and information capacity of a database while maximizing the usability of the resultant dataset” (p. 4). The following figure depicts the architecture of summary databases on mobile devices:

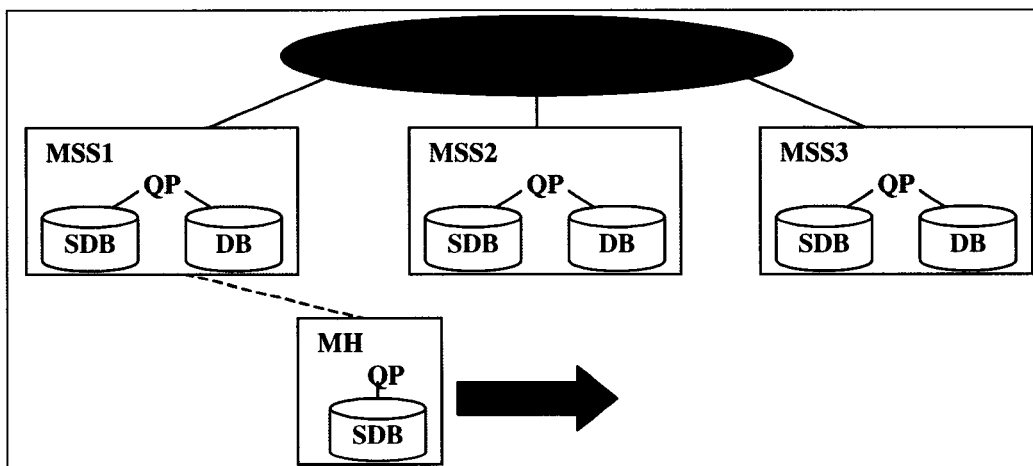


Figure 14: Mobile Database Architecture with Summary Databases

Source: Chan, D. & Roddick, J. (2003). *Context-Sensitive Mobile Database Summarisation*. Retrieved December 29, 2003, from <http://crpit.com/confpapers/CRPITV16Chan.pdf>

As shown in the figure above, the summary database on a mobile device is derived from the central database. In addition, a query processor on a mobile support station can make use of both the main and summary databases whereas it uses only the summary database on a mobile host. If a summary database cannot provide the required information for a query issued at a mobile host, the query is sent to the main database. It could also happen that for a particular query, part of the information is retrieved from the summary database located on mobile host or mobile support station and the rest from the central database.

Chan and Roddick (2003) list the following as the key characteristics of a good summarization technique:

Responsiveness – This is one of the key reasons for implementing any summarization technique. Therefore, it should be aimed at providing a timely response to the request of a mobile user.

Accuracy – It should be able to provide accurate information to the fullest extent possible according to the user needs. For example, some users may be interested in general information whereas others may need very specific information on a certain topic. This is possible only when the data is consistent. However, in a mobile computing environment, a number of mobile devices may be updating data while disconnected. This will lead to changes that will be applied to the main database when these mobile devices are connected back to the network. Keeping these issues in mind, proper care needs to be taken to ensure data consistency.

Adaptability and Graceful Degradation – Since a mobile device may need to work in different kinds of environments, a good summary database should have the ability to adapt to new and differing environments. The contents in a summary database may need to change depending on the location, since a mobile user may need to query data based on the location. For example, a user may be interested in finding nearby restaurants, theaters, shopping malls, etc. In addition, degradation of the response quality should take place gradually. The wireless bandwidth may also vary from one cell to another, so the contents of a summary database may need to be changed accordingly.

A good summary database should utilize a list of criteria and their respective priorities to determine what data needs to be replicated from the central database.

Madria, Mohania, and Roddick (1998) suggest the following approaches that can be used to update main and summary databases:

Update Summary and Main Database Online – In this approach, the summary database is updated in real-time providing up-to-date information to the mobile users. However, this can cause delays in create and update operations, as the data needs to be updated in main and summary databases simultaneously.

Update Summary and Main Database Offline – The summary databases can be refreshed from the main databases at predefined intervals. The data refresh can be scheduled at times when the system usage is low. The drawback of this method is that it does not provide up-to-date information.

Update Summary Database Offline and Main Database Online – This approach allows access to the main database, while the summary database is being

updated. The summary database may not be updated with all of the transactions taking place on the main database. Therefore, queries may give incorrect information in certain circumstances.

All of the techniques described above are suitable for updating the summary database on mobile devices. Use of a specific technique will depend on the nature of mobile application. For example, in a banking application, the summary and main databases should be updated online. On the other hand, in an application where a certain amount of downtime is acceptable, the summary and main databases should be updated while offline.

2.2.3 Caching

Caching is a technique that is used widely for query optimization in mobile database systems. Caching refers to storing some data on a mobile device that needs to be accessed quite often. Queries on a mobile device can use the cached data rather than accessing the central database, which will provide better response time.

Hopfner and Sattler (2003) proposed a caching mechanism that provides an efficient way to access the data from mobile devices. According to them, the following should be taken into consideration while designing a cache:

Query Rewriting – Queries that need to use the cache need to be rewritten such that they refer to the cache instead of the central database. In addition, the cached data may not provide full response to certain queries. In such circumstances, parts of the query that cannot be answered by the cache are sent to the central database server. The

results of those parts are then accumulated to provide a complete result to the mobile user.

Cache Replacement – Due to the memory limitations on mobile devices, only a subset of data can be stored at any point in time. If the new data needs to be loaded on a mobile device, old data needs to be deleted to free up the space. Proper care needs to be taken while removing data from the cache, such that the overall cached data is relevant and can be used efficiently by future queries.

3.0 MOBILE DATABASE

Most mobile applications and services have to be supported by some sort of database. The mobile computing environment involves communication between the central database server and the mobile database application platforms, such as laptops, PDAs, and mobile phones. A central database server is typically a powerful computer that hosts large databases. A mobile database application platform contains a mobile database and the applications using that database. Mobile users access the data residing in a central database through the database applications. The central database can be accessed by a variety of users that are local, remote, and mobile. For example, if a cell phone user wants to find restaurants in a given area, the mobile application first connects to the central database. Then, the mobile application retrieves the list and formats it to display on the cell phone. The mobile application may not even have to connect to a central database if the information is already stored in the mobile device because the user had accessed the same information earlier. This is where the mobile databases come into the picture.

Mobile and central databases play a significant role in creating a mobile computing environment. The database management for mobile applications represents a big challenge. The exchange of data takes place between mobile devices and the central database server to keep the mobile database up-to-date, as well as to upload the data that is changed by the mobile user to the central database server. Most mobile devices come with the software to help synchronize the data between the mobile database and the central database.

The difference between a static database and a mobile database is that the static database is tied to a specific host that is located in a fixed physical location. It does not offer any flexibility as far as access to such a database is concerned. On the other hand, a mobile database offers a great deal of flexibility by providing access to it from anywhere and at anytime by means of mobile devices. For this very reason, mobile databases are gaining popularity every day. One of the major benefits of mobile databases is that many users can use them at any time from any location.

While mobile databases provide flexibility, they require special attention in terms of database design so that the mobile device users can enjoy the flexibility they provide.

An effective mobile database design attempts to specify an optimal load between stationary and mobile units. Caching, or temporarily storing data at the mobile station, is closely related to distribution, and also requires an optimal balance between mobile and stationary database units. (Johnson, 2002, p. 14)

Another critical requirement for mobile databases is high availability. It is vital that mobile and central databases are robust to ensure around the clock availability of mobile applications and services, otherwise customer satisfaction and service provider revenue will suffer. This becomes even more important when the mobile applications are providing critical services, such as emergency assistance and security.

3.1 Architecture

A mobile database environment consists of a host database server where the central database resides, mobile databases that contain only a part of the central database, and the network through which host database and the mobile databases communicate.

The following figure demonstrates the role of mobile databases in a mobile computing environment:

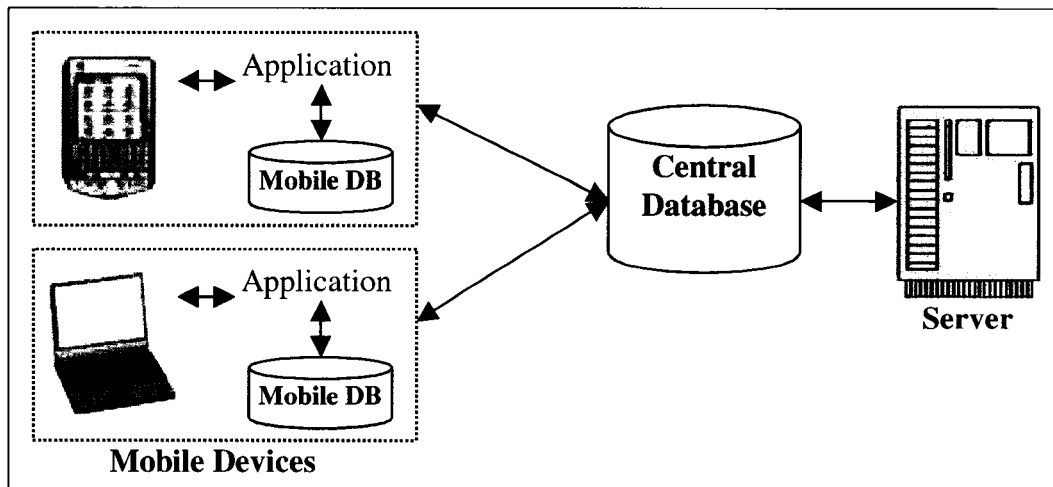


Figure 15: Mobile Database Environment

The database architecture depends on the capability of the client device and application, along with the services it is supposed to provide. For example, for an Order Entry database application running on a powerful personal computer that needs to validate and process the data in real-time before sending it to the central database server, the Client-Server Architecture would be ideal. On the other hand, for a Field Service application running on a mobile device, which is expected to handle transaction even while not connected to the central database server, the Distributed Database Systems Architecture would be more appropriate.

In client server architecture, the central database runs on the server and the applications that use this database run on the client. A single central database provides data access to multiple clients. The following figure represents typical client-server database architecture:

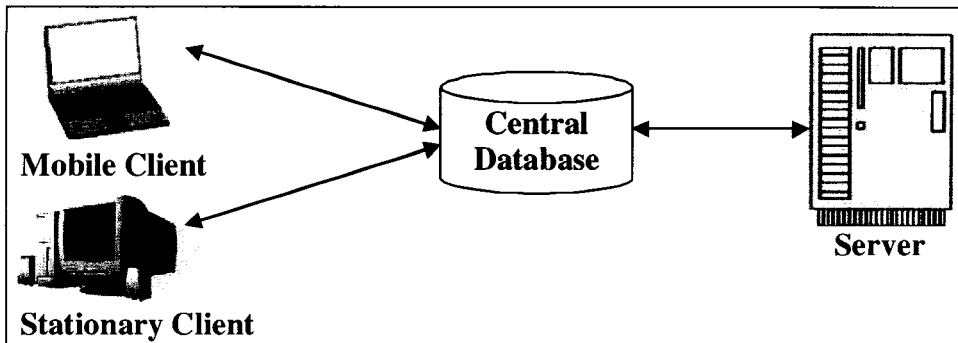


Figure 16: Client-Server Database Architecture

In distributed database systems, the database is distributed across different physical locations. In a mobile computing environment, this provides high availability, as multiple instances of the database exist and even if one fails, the others are still available. In addition, depending on their location, the clients can connect to any of these instances to access the database. The following figure represents distributed database system architecture:

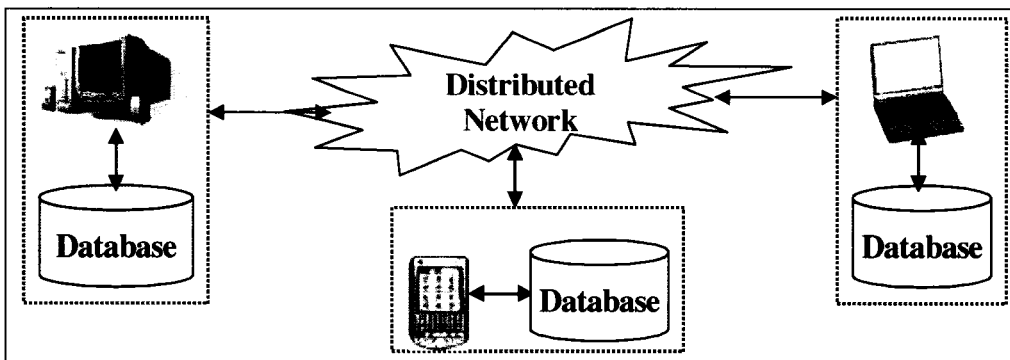


Figure 17: Distributed Database Architecture

Mobile databases can be distributed under two possible scenarios:

1. The entire database is distributed among wired and wireless components, possibly with full or partial replication. A base station or fixed host manages its own database with a DBMS-like functionality, with additional functionality for locating mobile units and additional query and transaction management features to meet the requirements of mobile environments.
2. The database is distributed among wired and wireless components. Data management responsibility is shared among base stations or fixed hosts and mobile units. (Elmasari, 2004, p. 5)

As mentioned earlier, a mobile database can be thought of as a distributed database that resides on mobile devices and a fixed network. The central database can be accessed through the network and mobile devices keep all or part of such a database. The mobile devices can be connected to the network at some times and disconnected from it at other times. The mobile devices have the capability to manipulate the information contained in the database even while they are disconnected from the central database. Changes performed through such offline transactions are propagated to the central database when the mobile devices are connected to the network. The database on the mobile device is also synchronized with the central database whenever the mobile device connects to the network. In other words, keeping the data up-to-date requires two-way communication – mobile database to central database and vice versa.

Although mobile devices currently in the market are more powerful and feature-rich than they were couple of years ago, they still have limitations in terms of memory and display size. Keeping these storage and display constraints in mind, only a limited amount of data can be stored on the mobile device. Therefore, a strategy has to be developed that makes best use of the available resources.

The two approaches that can be used to synchronize data between the mobile database and the central database server are: *push* and *pull*. The push approach can be used whenever the server wants to send some data to the mobile device. The pull approach can be used whenever mobile device needs some data from the server. The following figure illustrates the two approaches.

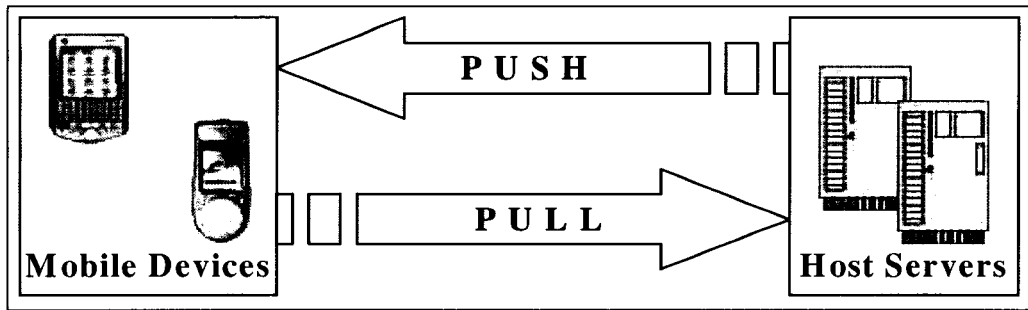


Figure 18: Push-Pull Approach

Both the push and pull approaches are used for data synchronization between mobile databases and central database servers (Jago, 2003).

3.2 Database Management Systems

Many mobile database management systems are already available in the market. The following are the mobile database management systems from some major players:

3.2.1 IBM DB2 Everyplace

This relational database has approximately 200 kilobytes footprint and boasts of industry leading indexing and query performance. It supports mobile devices running on various platforms including MS Windows CE, Pocket PC, Palm, Linux, and Symbian. DB2 Everyplace Sync Server is the component that is used to synchronize data between the mobile databases and the central database. It offers quick and secure bi-directional synchronization through compression and encryption. It also provides data partitioning and conflict resolution (IBM Corporation, 2003).

The following figure illustrates how the IBM DB2 Everyplace Sync Server synchronizes the data between the central database and mobile clients:

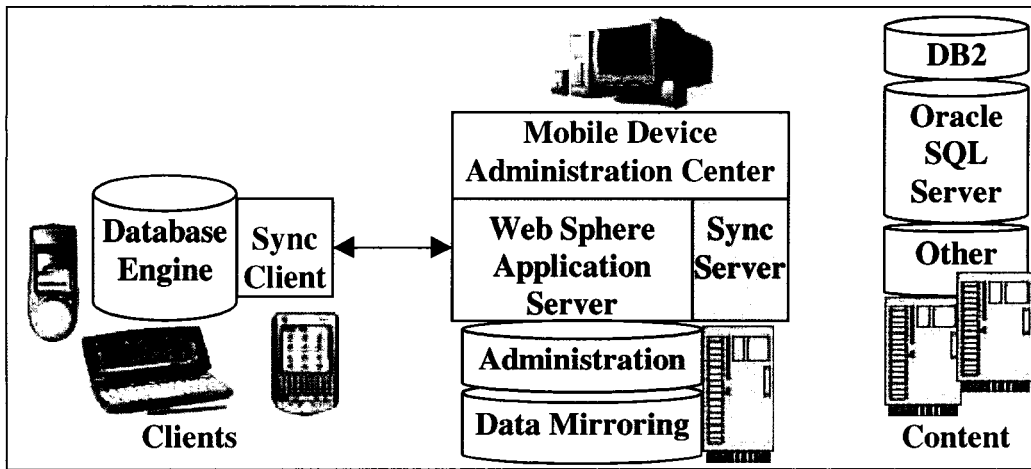


Figure 19: IBM DB2 Everyplace Architecture

Source: IBM Corporation (2003). *IBM DB2 Everyplace Performance Tuning Guide*. Retrieved December 1, 2003, from <ftp://ftp.software.ibm.com/software/data/db2/everyplace/perftuning.pdf>

3.2.2 Oracle9i Lite

Oracle9i Lite Database has 50 to 750 kilobytes footprint that is very much suitable for mobile computing. Platforms supported by Oracle9i Lite include MS Windows CE, Palm OS, and Symbian EPOC. Another component of Oracle9i Lite is the Mobile Server, which is a robust offline mobile application server. The Mobile Server offers a comprehensive set of features to deploy, manage, and synchronize offline applications on a large number of mobile devices. The Mobile Server supports the bi-directional synchronization between the Oracle9i Database and mobile devices (Oracle Corporation, 2001).

The following figure provides an overview of the Oracle9i Lite Architecture. As shown in the figure, a number of applications are running on the central database and are accessible on mobile devices through the Oracle Application Server.

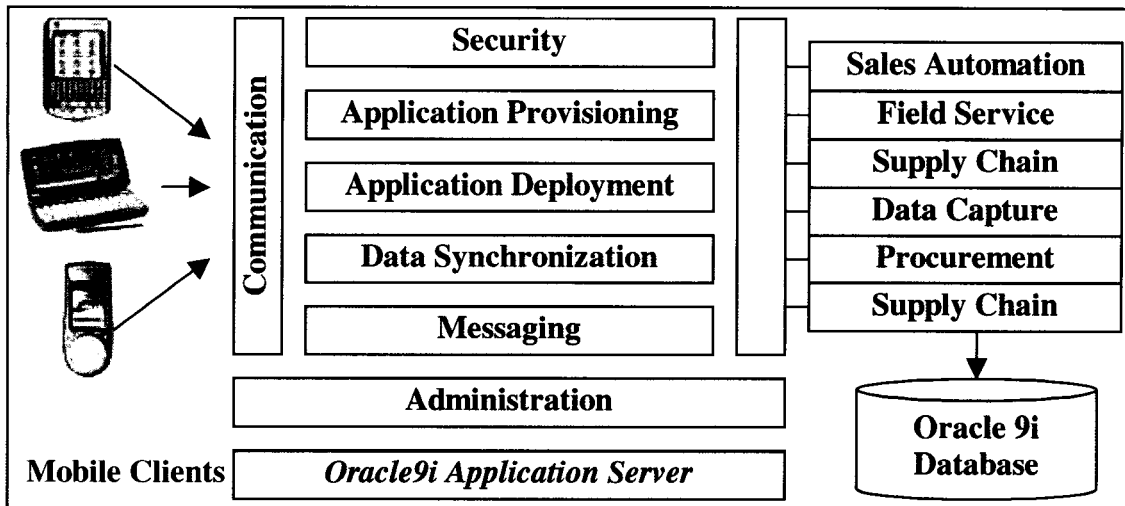


Figure 20: Oracle9iLite Architecture

Source: Oracle Corporation (2001). *Oracle9iLite; Technical White Paper*. Retrieved November 15, 2003, from http://otn.oracle.com/products/lite/pdf/o9ilite_bwp.pdf

3.2.3 SQL Server for Windows CE

SQL Server for Windows CE provides the complete functionality of a relational database in approximately one-megabyte footprint. In addition, its query processor optimizes the queries for better performance. Database security is provided through 128-bit encryption for the database file on the device (Microsoft Corporation, 2002).

According to Seshadri and Garrett (2000), SQL Server for CE has three major components: Storage Engine, Query Processor, and Connectivity. The Storage Engine deals with the data storage and access. The Query Processor helps in processing the requests made for retrieving data stored on the mobile device. The Connectivity feature offers an easy and secure way to setup the connectivity options for data replication or synchronization.

The following figure provides an overview of the SQL Server for CE Architecture. As shown in the figure, the interaction between SQL Server CE and SQL

Server database is facilitated through the SQL Server CE Service Agent and SQL Server Client Agent.

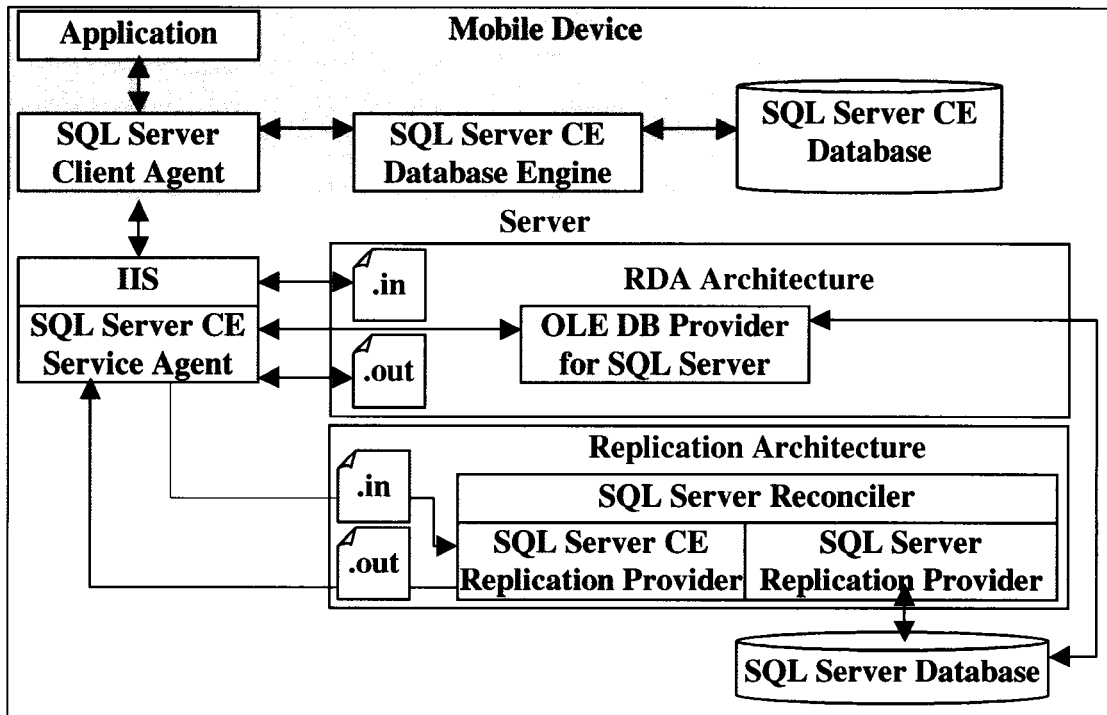


Figure 21: SQL Server for Windows CE Architecture

Source: Brown, M. & Meunier, D. (2003). *Create Compact, Robust Mobile Apps with SQL Server CE 2.0 and the .NET Compact Framework*. Retrieved October 17, 2003, from <http://msdn.microsoft.com/msdnmag/issues/03/01/SQLServerCE20>

3.2.4 Sybase UltraLite

The Sybase UltraLite database is part of SQL Anywhere Studio and is targeted toward memory-constrained mobile devices. The footprint of UltraLite can be as small as 50 kilobytes, but it still provides referential integrity and bi-directional synchronization. It supports both Microsoft Windows CE and Palm OS, two of the major operating systems for mobile devices. The MobiLink Synchronization Server, which is also part of the SQL Anywhere Studio, provides easy and secure bi-directional synchronization of

UltraLite with central database servers including Sybase Adaptive Server Enterprise, Oracle, IBM DB2, and Microsoft SQL Server (Sybase Inc., 1999).

The following figure describes how the MobiLink Synchronization Server facilitates the bi-directional synchronization of data between the Enterprise Database and the Mobile Devices.

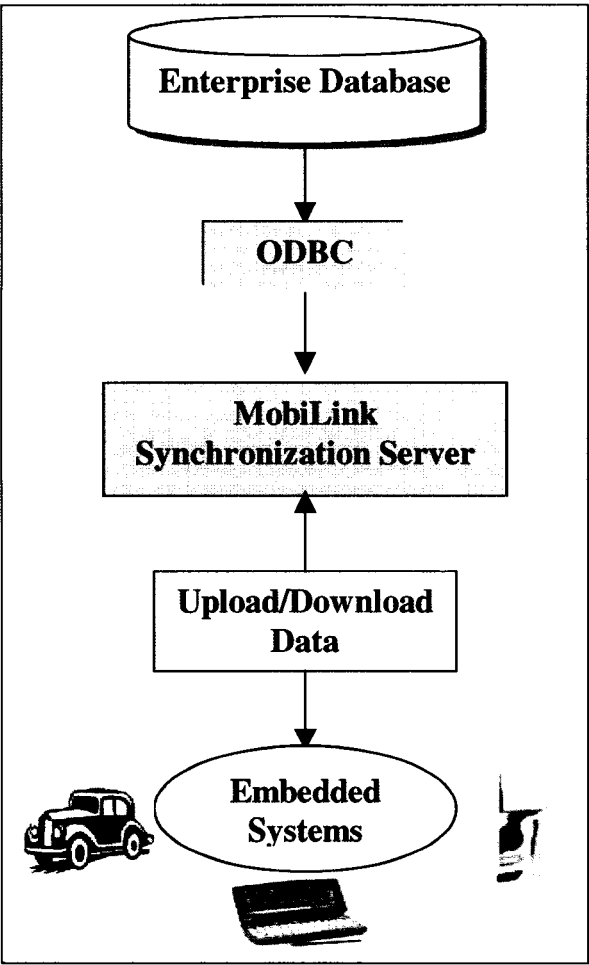


Figure 22: Sybase UltraLite Architecture

Source: Sybase, Inc. (1999). *The Next Generation Database for Embedded System*. Retrieved September 12, 2003, from <http://www.sybase.com/content/1002378/embedded.pdf>

3.2.5 Mobile DBMS Comparison

The following is a side-by-side comparison of the four mobile database management systems discussed earlier:

Table 7: Mobile Database Management System Comparison

Attributes	Mobile Database Management Systems			
	Microsoft SQLCE	Oracle 9iLite	IBM DB2 Everyplace	Sybase UltraLite
Footprint	1000 KB	50 – 1000 KB	180 KB	50 KB
Platform	Microsoft Windows CE based devices - Pocket PC, Handheld PC, and embedded devices	Palm, Symbian, Microsoft Pocket PC, and Microsoft Windows	Microsoft Pocket PC, Palm, Symbian, and QNX Neutrino	Microsoft Windows CE, WindRiver VxWORKS, QNX Neutrino, and Symbian
Encryption	Yes	Yes	Yes	Yes
Support for other database vendors	No	No	Yes	No
Conflict detection	Yes	Yes	Yes	Yes
Synchronization	Yes	Yes	Yes	Yes
Support for stored procedures	No	Yes	No	Yes
Easy installation and setup	Yes	No	No	Yes
Limited platform support	Yes	No	No	No

Mobile Database Management Systems				
Attributes	Microsoft SQLCE	Oracle 9iLite	IBM DB2 Everyplace	Sybase UltraLite
Major limitations	Supports only Windows platform and SQL Server	Works only with Oracle central database server	Does not support sub queries, views	Limit on size and number of database objects

3.3 Mobile Database Design Methodology

The database is one of the key components of any data centered application.

Therefore, care must be taken when designing databases.

Trademarks of a good database design include

- A functional database is generated
- The database accurately represents the business's data
- The database will be easy to use and maintain
- Acceptable response time exists for the end users
- Modifications are easily made to the structure
- Data can be retrieved and modified easily
- Down time because of poor design is minimized
- Very little maintenance is needed
- Data is kept safe by planning the security
- Redundant data is minimized or nonexistent
- Data can be easily backed-up or recovered
- The actual structure of the database will be virtually transparent to the end user (Stephens and Plew, 2001)

For mobile database applications, a good database design is even more important as it must strive to achieve the above characteristics while taking the storage, processing power, battery life, and display constraints of mobile devices into consideration. The following flow chart illustrates the database design process:

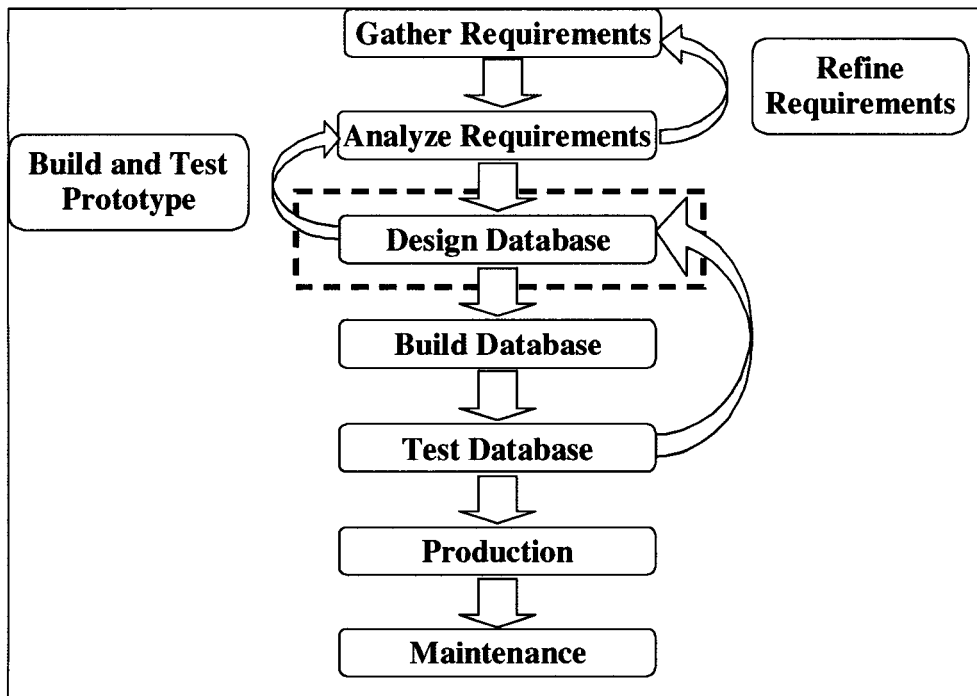


Figure 23: Database Design Process

Some of the major database design steps as identified above are as follows:

Gather and Analyze Business Requirements – This step involves talking to customers and end users to find out what they expect from the application. The requirements should be analyzed and refined until there is some level of agreement among all the stakeholders. It is vital to have clearly documented requirements so that needs and scope of the application can be defined.

Design Database – This is the main step that seeks to answer a number of infrastructure related questions including the following:

- What mobile platform will the application support?
- What mobile database management system will be most suitable for the application?

- What central database will be appropriate?
- What transaction model will best fit the needs?
- What optimization techniques will improve responsiveness?

It uses the finalized requirements and identifies the database objects that will store the relevant data. The entities, attributes, and their relationships can be determined by answering some questions including the following:

- What data types will be used?
- Who will have access to the data?
- What will be the data retention policy?
- Will data auditing be required?
- How will the data be updated or retrieved?
- What will the data depend on?

Build and Test Prototype – Once the database objects have been identified, a prototype needs to be created using them. If the prototype conforms to the business requirements, then the process continues to the next step. Otherwise, the requirements need to be analyzed again to get better understanding.

Build and Test Database – Once the prototype has been tested and accepted, the complete database needs to be created for development and testing. At this stage, it is imperative to ensure that all the user requirements have been implemented. Feedback received at this stage should be carefully evaluated and implemented, if appropriate.

Production and Maintenance – This is by far the most crucial step as the database application is finally going into production. Once the application is live, it moves into the maintenance phase, which will include enhancements and fixes to the issues identified in production.

As mentioned earlier, the *design database* step in the database design process is the most critical one. This is the step that involves making a number of important decisions. The following flow chart provides a detailed view of the design database step:

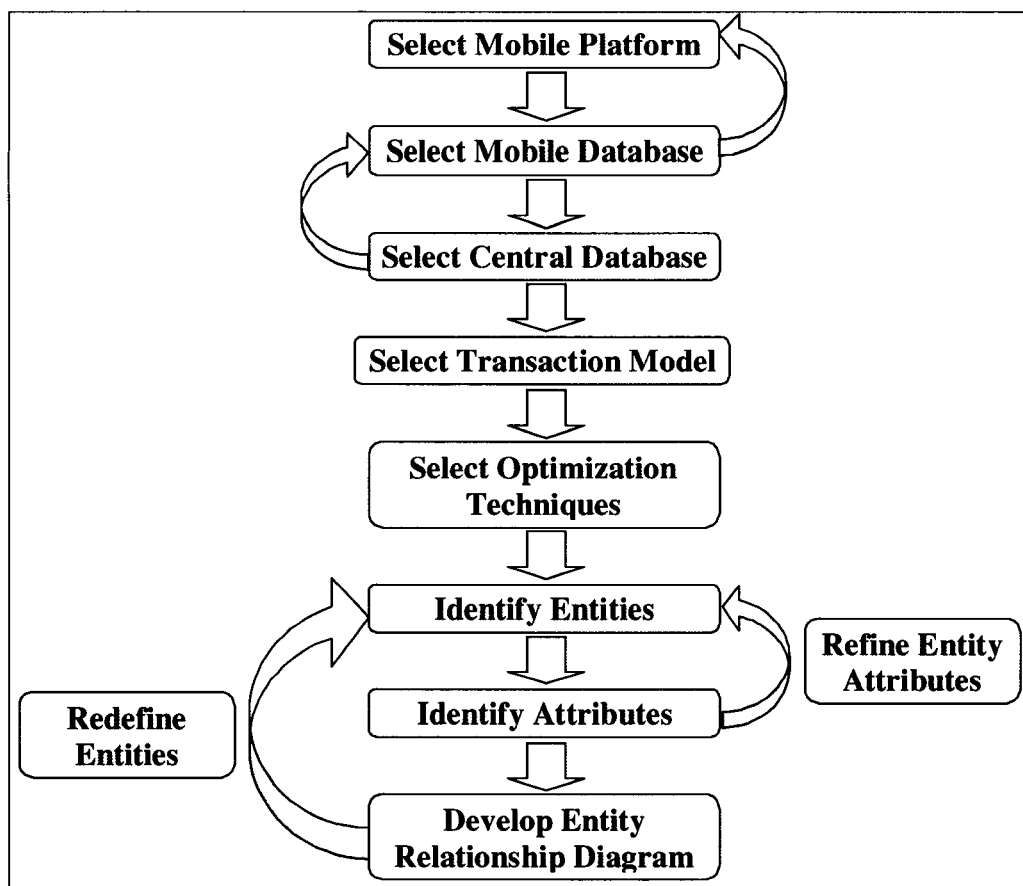


Figure 24: Detailed Design Database Step

The steps identified in this figure have been described in detail below:

Mobile Platform – Since the platform or operating system will decide which mobile devices will be able to use the database, it is very important to perform a side-by-side comparison for the possible choices. For example, if a decision to select the Pocket PC platform is made, then the users of Palm or other operating systems may not be able to use the database application. Some of the factors that may be considered when choosing a platform include, ease of use, ease of development, ease of deployment, price, support, and popularity.

Mobile Database – Once the platform has been chosen, an appropriate mobile database management system will need to be selected. As identified in side-by-side comparison of mobile databases earlier, not all the commercially available mobile database management systems run on every platform. For example, SQL Server for Windows CE runs only on Windows platforms. On the other hand, Oracle9i Lite runs on many platforms including Palm, Pocket PC, and Linux. Other factors that may be considered when choosing a mobile database management system include, support for stored procedures, support for other database vendors, ease of installation, synchronization, and conflict resolution. At this point, if it is determined that the mobile databases supported on the mobile platform selected earlier, do not meet the requirements, then it may be necessary to select another mobile platform.

Central Database – Selection of a central database goes hand-in-hand with the selection of a mobile database, since both need to coexist in one system while sharing the data. For example, if SQL Server for Windows CE is used as the mobile database, SQL Server could be used as the central database to ensure interoperability, since both the

databases are from the same vendor. Moreover, the selection of the central database will be restricted by the mobile database that is selected. For example, IBM DB2 Everyplace is the only mobile database that supports central databases from other database vendors. At this point, if it is determined that the central database supported by the mobile databases selected earlier does not meet the requirements, then it may be necessary to select another mobile database.

Transaction Model – An appropriate transaction model should be chosen depending on the model characteristics and user preferences. The evaluation model developed in the earlier section can be used for this purpose. The evaluation model takes the user preference and the importance for each of the transaction model characteristics and suggests the preferred transaction model. For example, if a mobile application needs to provide support for online transaction management on fixed and reliable networks while consuming low power, the Kangaroo transaction model would be the preferred choice. Similarly, if a mobile application needs to provide offline transaction management with user interface irrespective of the amount of power or bandwidth consumed, the Pro-Motion transaction model would be the preferred choice.

Optimization Techniques – Since mobile devices have limited resources, optimization techniques need to be used to make the most out of the available resources.

To make the best use of available bandwidth, the data transferred between mobile devices and the central database can be compressed. This will not only improve the performance, but will also make the bandwidth available to the other users. The

compression technique provides the capability to move large amount of data between the mobile device and the central database server.

Since mobile devices have limited storage, only the most frequently used data should be stored to make the best use of available space. This will allow most of the mobile user requests to be fulfilled using the data stored in the mobile database. Since mobile application will not have to issue queries to the central database for most of the user requests, the response time will be improved.

The resource intensive computations should be performed at the central database and results should be stored in summary columns. The mobile user will simply need to query this preprocessed information from the central database instead of trying to process on its own. For example, if multiple tables need to be joined in order to get some information or some calculations need to be performed, the results should be saved in some table columns in the central database that can be accessed by mobile users.

The main point to understand is that the performance of mobile applications depends on not only the mobile device and database, but also other factors, such as the central database and transaction model.

Database Entities – After the infrastructure related decisions have been made, decisions pertaining to storing the data need to be made. This is a very important task as it involves identifying the database entities and the data they will maintain. For example, if user information needs to be stored, then the question will be what user information should be stored? Should a separate table exist to store information for different types of users or should a single table be used to store information for all types of users, including

an attribute signifying the type of user? To some extent, normalization techniques will also dictate where the data will be stored.

Entity Attributes – While database entities are being identified, attributes contained in each of those entities will need to be considered. For each of those attributes, the data type and data size also need to be determined. Depending upon the requirements, some attributes may have to be de-normalized. In addition, some summary attributes may be added, which will be updated by database processes. These changes may require redefining some of the database entities.

Entity Relationship Diagram – Once the details about database entities and their attributes are clear, an entity relationship diagram needs to be developed. This diagram will provide the relationships between different entities. At this point if some issues are identified, it may become necessary to review the database entities and attributes.

There are many issues that need to be considered while following the above steps for designing the database. Some of the main issues are listed in the following section.

3.4 Mobile Database Design Factors and Potential Solutions

According to Gupta and Tang (2003), the main issues faced in a mobile computing environment are data management, query processing, and transaction management. In addition, Xia and Helal (2003) have identified device connectivity, access to limited resources, and adapting to location sensitive data as some of the challenges for mobile database design. Furthermore, Rennhackkamp (1998b) suggests that data types, response time, backup and recovery, concurrency control, and security are some of the issues that need to be considered while designing mobile database

applications. These issues identified by the above authors can be categorized in five different areas, as illustrated in the figure below:

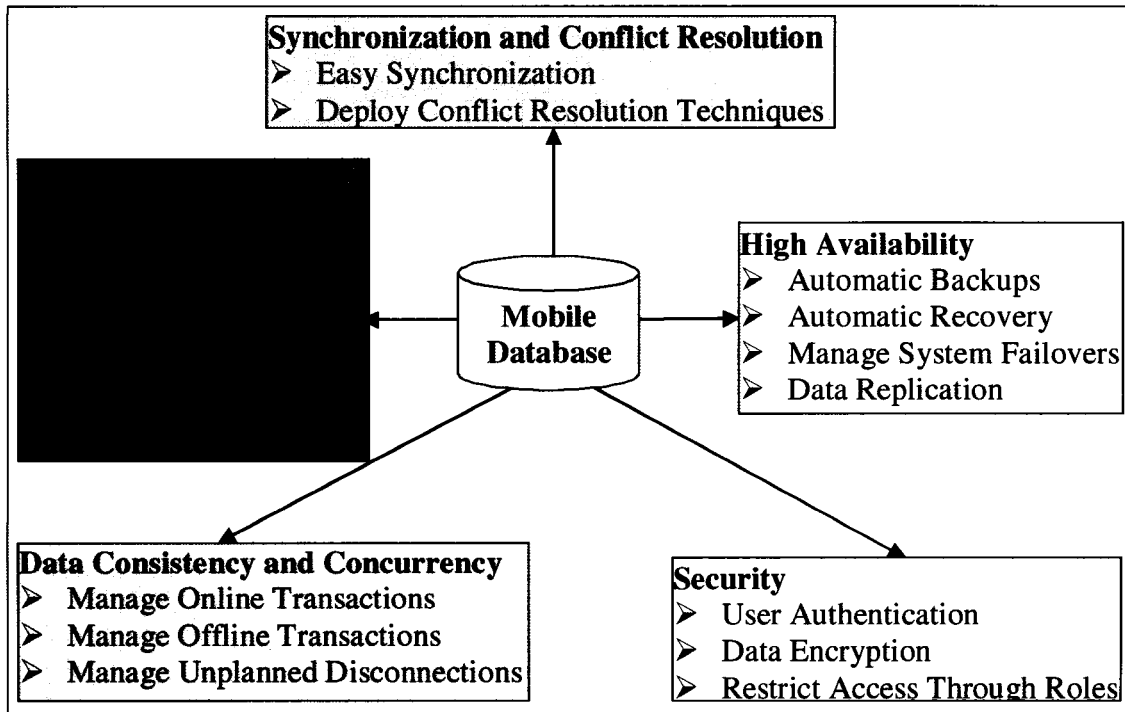


Figure 25: Mobile Database Design Problems and Potential Solutions

3.4.1 Responsiveness

Mobile applications are expected to provide access to information and services anytime and anywhere. To fulfill this promise, these applications should be highly responsive to end-user requests. Since mobile devices have limited storage and resources, it is essential that they be used very effectively and efficiently.

The amount of time required to access the relevant data should be reasonable. There are two methods to reduce access latency. The first method is to organize data such that related data items are placed together. This can be achieved by using the Cluster mobile transaction model. The second method is to improve the cache to hit ratio

of the data. This can be achieved by caching frequently used data on the mobile device. In addition, broadcast history can be analyzed using the data mining techniques to determine the request patterns and either caching or doing predictive pre-fetching of the data to improve performance (Daniel, 1999).

Querying and displaying only the relevant data can improve responsiveness. Special attention should be given to data types that require large amount of storage space. Data summarization can be used to provide pre-processed data, which can be readily used by the mobile client. Data can also be de-normalized for faster data access without having to join multiple tables. When transferring large amount of data, data compression techniques can be used to improve performance.

Another unknown factor in mobile computing environments is the number of users accessing an application at a given time. The system should be able to handle peak usage without having a major impact on the performance.

3.4.2 Data Consistency and Concurrency

Mobile devices are expected to work both online and offline. It is vital for a mobile application to ensure data consistency and concurrency irrespective of the type of transaction. While managing online transactions may be easy, managing both online and offline transactions can be rather difficult. There should be a robust mechanism to handle planned as well as unplanned disconnections between mobile devices and the central database so that data consistency and concurrency are maintained. As explained in the earlier chapter on mobile transaction models, an appropriate model needs to be chosen based on the requirements. For example, if mostly online transactions are expected, then

the Kangaroo transaction model can be used; if offline transactions are expected most of the time, then the Pro-Motion transaction model can be used.

The mobile clients should be able to read or write to any available server that holds a consistent view of the database. For example, a mobile client can issue writes at one server and reads at another server later. If these two servers are not synchronized, the read operation will fetch inconsistent data (Daniel, 1999).

3.4.3 Synchronization and Conflict Resolution

Synchronization of data between the mobile database and the central database is very important. This can be done either on a regular basis or as needed. The key is to maintain a true picture of data between all of the mobile devices and the central database server. Most mobile databases offer some synchronization tool, which makes it easier to synchronize the data between the mobile device and the central database server.

During the synchronization process, application may encounter some conflicts. All possible conflict situations must be thoroughly investigated and a conflict resolution technique for each must be developed. Most data synchronization tools also provide automatic conflict resolution techniques.

3.4.4 Security

Many applications and services, from commerce to banking to emergency services, are now available to mobile device users. While performing mobile commerce, it is important to keep personal and account information secure. While banking or trading, it is important to keep account and trade information secure. For emergency

services, such as medical services, it is important to keep the patient information secure. Therefore, security is of utmost importance in mobile computing environments.

Proper safeguards such as user authentication and data encryption must be established to ensure that the personal and other critical information of a mobile user is securely maintained.

3.4.5 High Availability

Since they are tools used to improve productivity, mobile devices and applications must be available around the clock. This is possible only if there are well-established strategies to manage database backup, recovery, and system failovers. A well-designed and well-implemented backup and recovery strategy is a cornerstone of every database deployment and must be incorporated into the database design.

Some of the techniques that can be used to ensure high availability include, regularly scheduled automatic backups and automatic recovery in the event of a failure. To minimize the impact of a system failure, data replication can also be used.

4.0 PROBLEM-BASED LEARNING ENVIRONMENT

In problem-based learning, students are encouraged to investigate real-life problems and find solutions. To achieve this, students are provided with the resources and guidance as necessary. Problem-based learning is different from traditional classroom learning in many ways. In problem-based learning, a few students work together in a group to achieve their goals of learning by investigating and solving a specific topic or problem. A faculty member works as a mentor for the students and helps them as needed. The faculty member also works as a facilitator and works with the students to create a schedule to ensure that the project is completed in a timely manner. In addition, certain resources are allocated for the students to accomplish their goals. The students communicate with each other on a regular basis and learn from each other in the process. The problem-based learning methodology is targeted towards faster and efficient learning where students drive the learning effort (Jones, 1996b).

The following are the main steps involved in problem-based learning:

- A problem is identified and defined.
- The problem or the project is assigned to a team of students with some faculty member as their guide.
- Students and the faculty member agree on the scope of the project.
- The project is divided into a number of activities.
- Each activity is assigned to one or more students depending on its complexity and efforts involved.
- A checklist is created for each activity, which has one or more checklist items.

- A schedule is put into place for all of the activities involved.
- Resources such as labs, computers, textbooks, conferences, and seminars are allocated to the project.
- Regular meetings are scheduled to track the progress and share any ideas.
- Some or all of the activities may have dependencies. For example, a student working on a certain activity may require regular input from other students.
- If a student asks a question to the faculty member, all other students should be able to see the response.
- Students submit their questions to a forum so that other students get a chance to respond to it.
- All students should be able to communicate with one another.
- At the completion of each activity, the corresponding checklists should be validated to make sure that all of the students are making expected progress.

The following process diagram illustrates the steps involved in problem-based learning environment:

4.1 Process Diagram

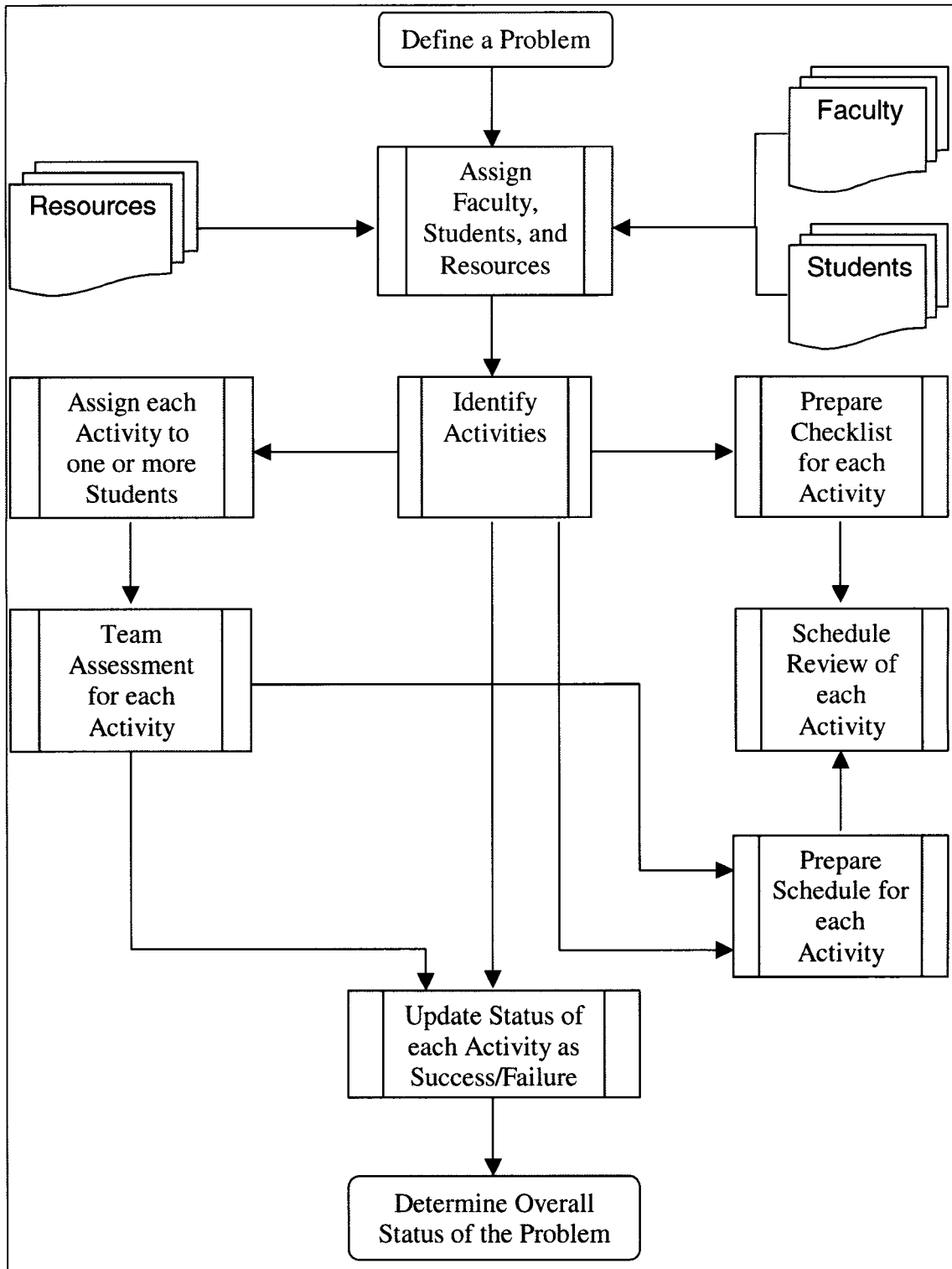


Figure 26: Problem-Based Learning Process Diagram

4.2 Advantages

Learning by Doing – Most of the time, people learn more by actually doing something rather than reading about it. In traditional classroom teaching there is more emphasis on facts and theory. However, the problem-based learning approach allows the students to explore the concepts and apply them in practice. This helps students get a deeper understanding of the subject (Jones, 1996a).

Motivation – In traditional classroom teaching, assignments are mostly preset and students have to work on them whether they like it or not. On the other hand, problem-based learning gives students the opportunity to work on subjects of interest to them. If the subject is of interest, students feel more motivated and get the work done more efficiently (Jones, 1996a).

Teamwork – Problem-based learning environments give students an opportunity to work in a team without any supervision. In the process, they learn how to coordinate and work with other people in a group. This also helps them improve their communication skills. These are very important skills, which will be of tremendous benefit to students in their professional life (Jones, 1996a).

Problem Solving – In problem-based learning environments, students need to acquire and manage resources needed for their projects. This helps them develop skills to identify and make better use of resources to fully accomplish the tasks. Moreover, since students work on real life problems rather than textbook scenarios, it helps them develop better problem-solving skills (Jones, 1996a).

Faculty-Student Relationship – Problem-based learning environment is a win-win for both students and faculty. On one hand, students get a course curriculum, which is interesting and inline with reality. On the other hand, the faculty gets an opportunity to work closely with the students on new subject areas (Jones, 1996a).

4.3 Disadvantages

Course Objectives – In a problem-based learning environment, course objectives are not as clear as in traditional classrooms. Hence, it becomes challenging for students to come up with the learning objectives. In a traditional classroom teaching environment, the faculty provides students with the objectives of the course and the resources to accomplish them (Leung, 2001).

Basic Concepts – In a traditional classroom teaching environment, first the basic concepts of the subject are taught, which form the basis for solving more complex future problems. Problem-based learning does not necessarily follow the same format. Therefore, it is up to the students to acquire this knowledge on their own. If students start working on a problem without having the knowledge and understanding of the basic concepts, it may cause confusion, frustration, and a delay in completing the assigned work (Leung, 2001).

Faculty Involvement – In problem-based learning environment, students drive the learning effort through active guidance from the faculty as needed. Although there is faculty involvement, it is not as much as in traditional learning environment. Due to this lack of faculty involvement, students may lose focus on their work (Leung, 2001).

Learning Style – While team learning is a more life-like experience, it can be a problem for students who are used to studying on their own. It may take longer for such students to get used to the new learning environment, which may affect the overall performance of the team (Leung, 2001).

Resource Management – In a traditional classroom teaching environment, all of the required resources are made available to the students. In a problem-based learning environment, availability and manageability of resources can become difficult, as it needs to suit varying requirements of the students (Leung, 2001).

4.4 Challenges

The problem-based approach to learning faces various challenges, some of which are discussed below:

Lack of Awareness – The students need to be educated to make sure that they understand what problem-based learning actually is and how it can benefit them. The students are so accustomed to the classroom style of learning that it may become difficult for them to work on their own to reach their learning objectives. In the initial stages, the faculty needs to work closely with students to make sure that they are on the right track. Once students understand the process, they can be given more flexibility to continue on their own.

Changing the Curriculum – Changing the curriculum requires extensive effort on the part of administration and faculty, in addition to their current responsibilities. Therefore, it is very important to offer some incentives so that everybody involved is willing to make extra effort (San Diego State University, 1996a).

Resistance from Faculty – Since most faculty members worldwide are accustomed to the traditional classroom teaching, they are often hesitant toward the idea to use new teaching methodologies, such as problem-based learning. Therefore, faculty members need to be convinced that problem-based learning is, in fact, an effective method of teaching. One way to achieve this could be to teach a specific topic to a group of students using both techniques and then give them the same assessment exams at the end of the course. The results of this exam could be used to determine the effectiveness of problem-based learning versus traditional classroom teaching (San Diego State University, 1996b).

Resource Constraints – As problem-based learning requires students to work in small groups, there may be a problem assigning resources to all of the groups. For example, in the traditional learning methodology, one faculty member provides guidance to all of the students in a class. However, with problem-based learning, more faculty members may be needed for the students. Alternatively, one faculty member may need to work with many groups, which would require schedule management. The resources may need to be shared among different groups, so ensuring availability of the resources when needed may also become a challenge (San Diego State University, 1996a).

The following figure summarizes the advantages, disadvantages, and challenges of problem-based learning environment identified above:

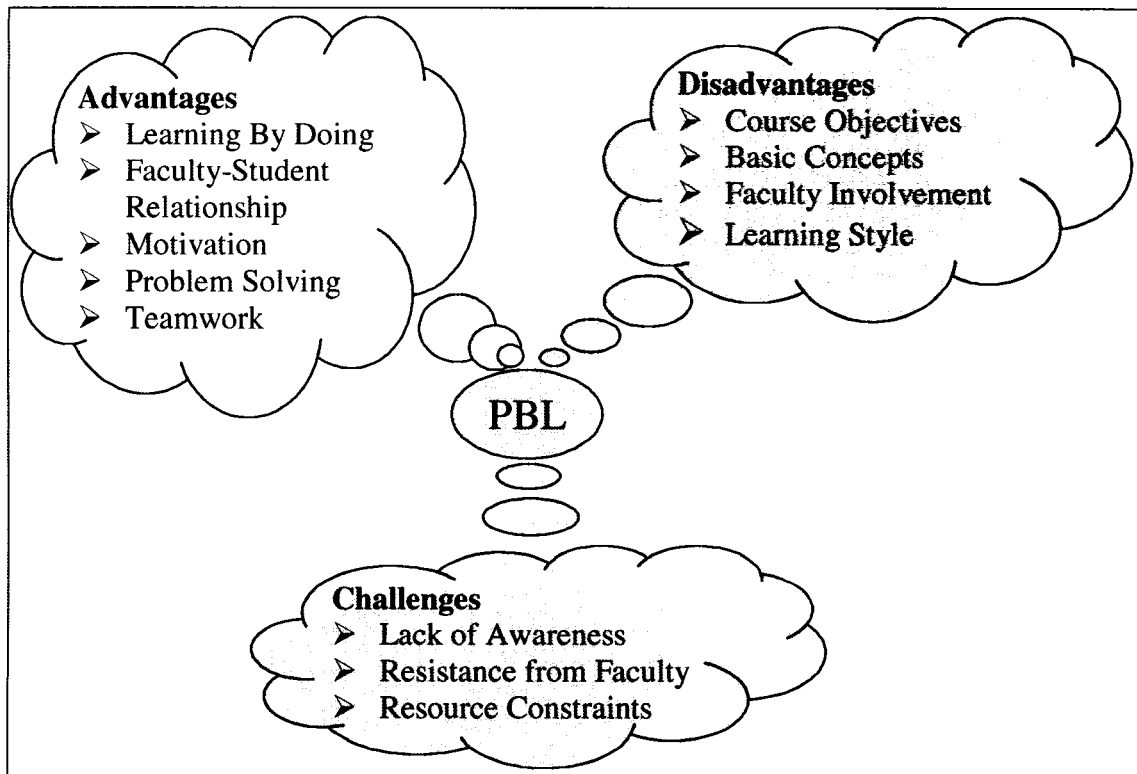


Figure 27: PBL - Advantages, Disadvantages, and Challenges

The problem-based learning approach allows students to work on real-life problems and find solutions with the minimal guidance from the faculty. Problem-based learning offers many benefits to students and faculty members. It prepares students with the practical knowledge that they can use in their professional life. In addition, it helps the faculty to offer projects that are more focused towards the interests of students.

It is inevitable that mobile devices will change the education methodologies. In the current fast-paced environment, it is not always feasible for individuals to earn their degrees through traditional classroom teaching. Mobile devices provide the ease and flexibility to study anytime and anywhere by enabling collaboration with other team members and the faculty.

4.5 High Level Design

This section provides the high-level design for the problem-based learning application.

Architecture – For the PBL application, SQL Server for Windows CE was selected as the mobile database and SQL Server 2000 was selected as the central database. Some of the main reasons for selecting SQL Server were as follows:

- Popular platform
- Support for embedded devices
- Rapid development using Microsoft .NET, which also provides a pocket PC simulator for immediate testing
- Easy deployment
- Automatic synchronization and conflict resolution techniques available between SQL CE and SQL Server databases

The following figure illustrates the architecture that will be employed in the problem-based learning application. The data will be stored in a SQL CE database on the mobile device. The changes performed on the mobile database will be synchronized with the host SQL Server database using the synchronization software.

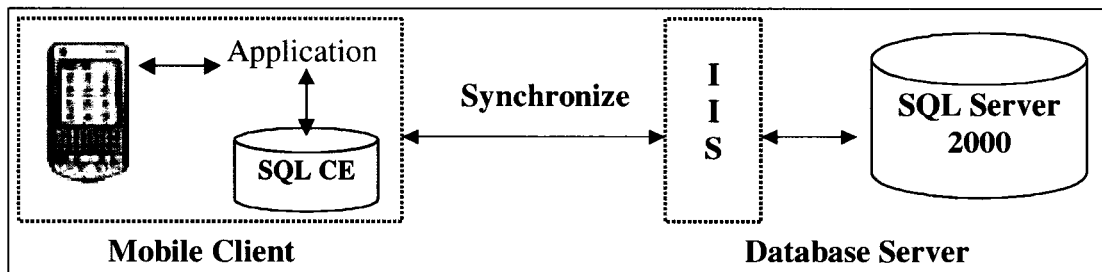


Figure 28: Basic SQL Server CE Architecture

Database Schema – As part of this research, a database has been designed to create an application that can support a problem-based learning environment. Mobile database design problems and solutions identified in the earlier sections have been applied while designing this database. The database entities needed to support the problem-based learning application can be divided into two main categories – setup and transactional. The following table lists the setup related database entities:

Table 8: PBL Application Database Entities – Setup

Table name	Description
PBL_DEPARTMENTS	This table stores the information on all the departments in a school. This is a setup table that must have some data before data in some other PBL tables can be created. Data synchronization for this table is not necessary, so it does not exist on the mobile device.
PBL_USERS	This table stores the information for all of the PBL users – Student, Faculty, and administrator. This table has been de-normalized to store the address and contact information so that user-related data is easily available without having to join with other tables. This table is synchronized with the central database, but the data is limited to project team members of the user.
PBL_COURSES	This table stores the information on all of the courses that are offering problem-based learning projects. This table is synchronized with the central database, but the data is limited to the courses for which the user may be registered.
PBL_COURSE_SECTIONS	This table stores the section information for each of the courses. There is a one-to-many relationship between the PBL_COURSES and PBL_COURSE_SECTIONS tables. This table is synchronized with the central database, but the data is limited to the courses for which the user may be registered.

Table name	Description
PBL_PROJECTS	This table stores the information on all of the projects for a particular course section. There is a one-to-many relationship between the PBL_COURSE_SECTIONS and PBL_PROJECTS tables. This table is synchronized with the central database, but the data is limited to the projects for which the user may be registered.
PBL_RESOURCES	This table stores the information on all of the resources available for problem-based learning projects. Data synchronization for this table is not necessary, so it does not exist on the mobile device.
PBL_LOOKUP_TYPES	This table stores different Lookup Types used throughout the application. Data synchronization for this table is not necessary, so it does not exist on the mobile device.
PBL_LOOKUP_CODES	This table stores different Lookup Codes for each of the Lookup Type. Data synchronization for this table is not necessary, so it does not exist on the mobile device.

The following table lists the transaction related database entities:

Table 9: PBL Application Database Entities – Transactional

Table name	Description
PBL_STUDENT_PROJECTS	This table stores the list of students registered for a particular project. There is a one-to-many relationship between the PBL_PROJECTS and PBL_STUDENT_PROJECTS tables. This table is synchronized with the central database, but the data is limited to the projects for which the user may be registered.

Table name	Description
PBL_ACTIVITIES	This table stores the information on all of the activities for a project. There is a one-to-many relationship between the PBL_PROJECTS and PBL_ACTIVITIES tables. This table is synchronized with the central database, but the data is limited to the projects for which the user may be registered.
PBL_ACTIVITY_DEPENDENCIES	This table stores the dependencies between an activity and other activities. There is a one-to-many relationship between the PBL_ACTIVITIES and PBL_ACTIVITY_DEPENDENCIES tables. This table is synchronized with the central database, but the data is limited to the projects for which the user may be registered.
PBL_STUDENT_ACTIVITIES	This table stores the activities assigned to a particular student. There is a one-to-many relationship between the PBL_USERS and PBL_STUDENT_ACTIVITIES tables. This table is synchronized with the central database, but the data is limited to the projects for which the user may be registered.
PBL_CHECKLIST	This table stores the checklists for each activity. There is a one-to-many relationship between the PBL_ACTIVITIES and PBL_CHECKLISTS tables. This table is synchronized with the central database, but the data is limited to the projects for which the user may be registered.
PBL_CHECKLIST_ITEMS	This table stores the checklist items for each of the activity checklists. There is a one-to-many relationship between the PBL_CHECKLISTS and PBL_CHECKLIST_ITEMS tables. This table is synchronized with the central database, but the data is limited to the projects for which the user may be registered.

Table name	Description
PBL_PROJECT_RESOURCE_ALLOCATIONS	This table stores the resource allocation for a given project. There is a one-to-many relationship between the PBL_PROJECTS and PBL_PROJECT_RESOURCE_ALLOCATIONS tables. This table is synchronized with the central database, but the data is limited to the projects for which the user may be registered.

In addition to the above tables, there are some views such as PBL_STUDENTS, PBL_FACULTY, and PBL_LOOKUPS that have been created to provide relevant information. For the detailed database entity structure, please refer to Appendix D. The following figure represents the entity relationship diagram for the problem-based learning application:

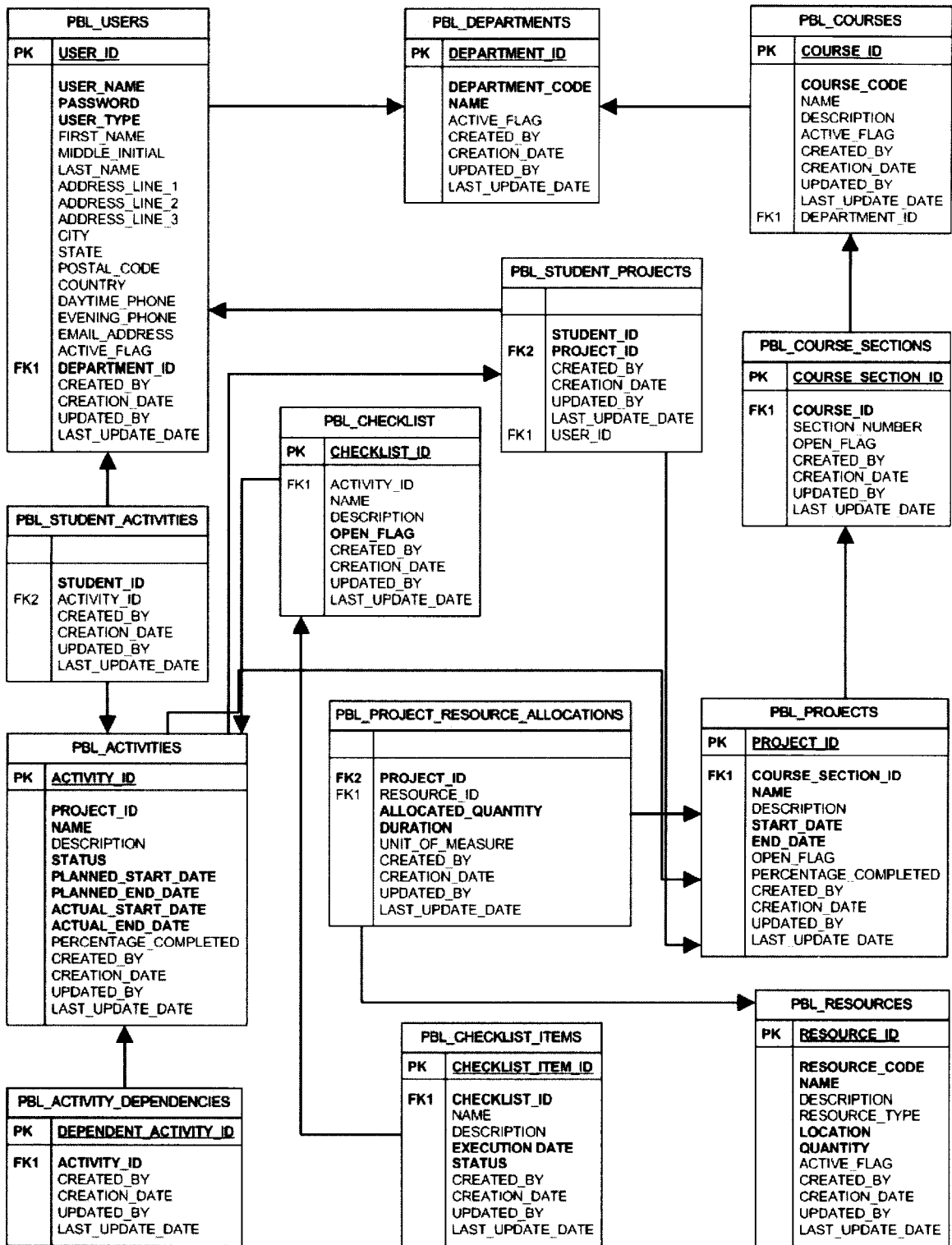


Figure 29: PBL Entity Relationship Diagram

User Interface – The user interfaces for the problem-based learning application can be divided into two categories – setup and transactional. The setup user interfaces will not be accessible through the mobile devices, since only the administrator and the faculty will be allowed to create, update, and query the setup related data. The following table provides the names and descriptions of each of the setup related user interfaces. It also provides the type of user to which a specific interface is available.

Table 10: PBL Application Setup User Interfaces

Name	Description	User Type
Create, Update, or Query Departments	This user interface allows the user to create and maintain departments that are offering problem-based learning courses.	Administrator
Create, Update, or Query Courses and Course Sections	This user interface allows the user to enter course details and section details for a course that is offering problem-based learning projects. A department must be setup before a course can be created.	Administrator and Faculty
Create, Update, or Query Projects	This user interface allows the user to setup different projects being offered through a course. A course must have been setup before the projects can be created here.	Faculty
Create, Update, or Query Lookups	This user interface allows the user to create or update lookup types and lookup codes that are being used in the PBL application.	Administrator

Unlike setup user interfaces, transactional user interfaces will be accessible through mobile devices, as they allow the students and faculty to create, update, and query the transactional data. The following table provides the names and descriptions of

each of the transactional user interfaces. It also provides the type of user to which a specific interface is available.

Table 11: PBL Application Mobile User Interfaces

User Interface	Description	User Type
Setup	This user interface allows the user to specify application setup parameters such as Subscriber, Publisher, and Internet URL. It also allows the user to enter appropriate login information for the central database and the Internet Information Server.	Student and Faculty
Login	This user interface allows an existing user to log into the PBL application. This provides the third level of security, apart from the central database and the Internet Information Server.	Student and Faculty
Register	This user interface allows a new user of the PBL application to register for an account.	Student and Faculty
My Profile	This user interface allows the user to create, update, or query their personal information such as name, phone number, and postal address.	Student and Faculty
Quick Menu	This user interface allows the user to directly jump to a specific transactional user interface such as Project Registration, and My Projects.	Student and Faculty
Project Registration	This user interface allows the user to register for courses offering problem-based learning projects given the course code or name.	Student
Course Information	This user interface provides additional details about a specific course that a user may be registering for or may have already registered.	Student and Faculty
My Team	This user interface provides names of the team members that a user may be working with on a specific project.	Student

User Interface	Description	User Type
My Projects	This user interface provides details on a project that a user may be working on. From here, the user can go to another interface that allows creating or updating project activities.	Student and Faculty
Project Activities	This user interface allows the users to create, update, or query the activities for a project. From here, the user can go to another interface that allows creating or updating activity checklists.	Student and Faculty
Assignees	This user interface provides names of one or more team members that are assigned to a specific activity. This interface is only accessible through the project activities user interface.	Student and Faculty
Dependent Activities	This user interface allows the user to create dependencies for a particular activity on one or more activities. This interface is only accessible through the project activities user interface.	Student and Faculty
Activity Checklists	This user interface allows the user to setup the checklist for each of the project activity. This interface is only accessible through the project activities user interface.	Student and Faculty
Reserve Resources	This user interface allows the user to search for available resources using parameters such as resource name, code, and location. It then allows the user to reserve those resources.	Student and Faculty

As mentioned earlier, not all of the user interfaces are meant for mobile device users. The same logic applies to the database tables. In other words, only a limited number of tables and a specific set of data will be stored on the mobile devices when synchronized with the central database. In addition, each user interface may be

manipulating data in one or more tables depending upon the transaction involved. The following table provides the user interface and database table mapping:

Table 12: PBL User Interface and Entity Mapping

User Interface	Database Table
Create, Update, or Query Departments	PBL_DEPARTMENTS
Create, Update, or Query Courses and Course Sections	PBL_COURSES, PBL_COURSE_SECTIONS
Create, Update, or Query Projects	PBL_PROJECTS
Create, Update, or Query Lookups	PBL_LOOKUP_TYPES, PBL_LOOKUP_CODES
Login or Register	PBL_USERS
My Profile	PBL_USERS
Project Registration	PBL_PROJECTS, PBL_STUDENT_PROJECTS
Course Information	PBL_COURSES, PBL_COURSE_SECTIONS
My Team	PBL_STUDENT_PROJECTS
My Projects	PBL_STUDENT_PROJECTS
Project Activities	PBL_ACTIVITIES, PBL_STUDENT_ACTIVITIES
Assignees	PBL_STUDENT_ACTIVITIES
Dependent Activities	PBL_ACTIVITY_DEPENDENCIES
Activity Checklists	PBL_CHECKLISTS, PBL_CHECKLIST_ITEMS
Reserve Resources	PBL_RESOURCES, PBL_PROJECT_RESOURCE_ALLOCATIONS

Please refer to Appendix C for detailed information on the mobile user interfaces identified above.

Use Cases – Please refer to Appendix E for the Use Cases for problem-based learning application.

Seed Data – Some of the tables will be populated with basic data to allow the administrator to setup the application.

Administrator Setup

Setup a department for the PBL administrator in the PBL_DEPARTMENTS table.

Table 13: PBL Departments Setup

Column	Value
DEPARTMENT_ID	1
DEPARTMENT_CODE	ADMIN
NAME	ADMIN
ACTIVE_FLAG	Y
CREATED_BY	1
CREATION_DATE	SYSTEM DATE
UPDATED_BY	1
LAST_UPDATE_DATE	SYSTEM DATE

Setup the administrator for PBL application in the PBL_USERS table.

Table 14: PBL Users Setup

Column	Value
USER_ID	1
USER_NAME	PBLADMIN
PASSWORD	PBLADMIN
USER_TYPE	ADMINISTRATOR
FIRST_NAME	PBLADMIN
LAST_NAME	PBLADMIN
ACTIVE_FLAG	Y
DEPARTMENT_ID	1
CREATED_BY	1
CREATION_DATE	SYSTEM DATE
UPDATED_BY	1
LAST_UPDATE_DATE	SYSTEM DATE

Lookup Setup

Create some commonly used lookup types in the PBL_LOOKUP_TYPES table.

Table 15: PBL Lookups Setup

Type	Name	Description
YES_NO	Yes or No	Status of records
USER_TYPE	User Types	Types of PBL users
DEPARTMENT_CODE	Department Code	Code for each department

Type	Name	Description
RESOURCE_TYPE	Resource Types	Type of resources available to reserve
ACTIVITY_CHECKLIST_STATUS	Activity and Checklist Status	Statuses for activities and checklists
UOM	Unit of Measure	UOM used for resources

The following are some lookup codes for each of the lookup type listed above:

LOOKUP_TYPE = 'YES_NO'

Column	Value
CODE	'YES', 'NO'
NAME	'Yes', 'No'

LOOKUP_TYPE = 'USER_TYPE'

Column	Value
CODE	'STUDENT', 'FACULTY', 'ADMINISTRATOR'
NAME	'Student', 'Faculty', 'Administrator'

LOOKUP_TYPE = 'DEPARTMENT_CODE'

Column	Value
CODE	'ADMIN'
NAME	'Admin'

LOOKUP_TYPE = 'RESOURCE_TYPE'

Column	Value
CODE	'BOOK', 'COMPUTER', 'SOFTWARE', 'LAB'
NAME	'Book', 'Computer', 'Software', 'Lab'

LOOKUP_TYPE = 'ACTIVITY_CHECKLIST_STATUS'

Column	Value
CODE	'PENDING', 'PARTIAL', 'COMPLETE'
NAME	'Pending', 'Partially Completed', 'Completed'

LOOKUP_TYPE = 'UOM'

Column	Value
CODE	'EACH', 'DOZEN', 'POUND'
NAME	'Each', 'Dozen', 'Pound'

5.0 ANALYSIS AND CONCLUSIONS

Database is the most critical component for any data dependent application. This is even more critical for mobile applications and services, given the small display, storage, and processing power of mobile devices. Therefore, the database design is of utmost importance in mobile computing environments. A well-designed mobile database should follow the design methodologies identified in chapter three. The problem-based learning environment was selected to demonstrate the effectiveness of these database design methodologies.

One of the important aspects of a database design is to identify the appropriate transaction model to support mobile transactions. The evaluation model developed as part of this research can be used to determine the preferred transaction model, given the requirements and their importance for a specific mobile application. For example, the evaluation model suggested Pro-Motion as the preferred transaction model for the problem-based learning application.

Another aspect is to ensure data consistency and synchronization between the mobile device and the central database server for which adequate algorithms need to be developed. Many mobile database management systems available in the market today provide the resources and framework to develop good mobile applications. The side-by-side comparison developed in this research can be used to select the most suitable mobile database management system. For example, SQL Server for Windows CE was selected as the best mobile database management system, given the needs of problem-based learning application.

In addition, five major problem areas were identified that must be considered while designing mobile databases. Each of these problem areas and their potential solutions were applied while designing problem-based learning application and are described below:

Responsiveness – The mobile database being used for the problem-based learning application, SQL Server for Windows CE, stores the data queried by the user. It keeps it up-to-date through automatic conflict resolution and synchronization, thereby providing faster response.

The mobile database stores only the data that is relevant to that particular mobile user. This has been achieved in two ways. First, only those tables that are actually needed on the mobile database have been selected for synchronization with the central database. For example, some setup tables such as PBL_DEPARTMENTS and PBL_LOOKUP_TYPES are not synchronized at all. Second, only a subset of data, partitioned based on the user id, is synchronized with the central database. For example, only the projects, activities, and checklists related to the mobile user are synchronized between the mobile database and the central database. Therefore, for a student, the mobile database will store data related to projects that the student may be currently working on. Similarly, for a faculty member, the mobile database will store data related to projects that the faculty may be mentoring. The synchronization of a subset of data from specific entities can be easily accomplished by setting appropriate data replication rules in SQL Server for Windows CE.

The database has not been fully normalized so that faster access to the data can be provided. For example, currently, the PBL_USERS table stores address information as well. This could have been further normalized to have a separate table for storing just the addresses. In that case, the address information would have been available by joining user and address tables. This unnecessary join would have really negatively impacted the performance of the queries involved.

Some of the tables have summary columns that will be updated by background processes. Since it will be time and resource consuming to process this information on the device, the data is processed on the central database server to provide easy access to information. For example, there are columns named PERCENTAGE_COMPLETED and STATUS in the PBL_PROJECTS, PBL_ACTIVITIES, and PBL_CHECKLISTS tables. The percentage completed column is used to indicate the progress of a project, activity, or checklist. Similarly, the status column is used to indicate if a project, activity, or checklist is open or closed. The background process will compute values for these columns and store them in the table so that students or faculty members can determine the progress of a project, activity, or checklist.

Data Consistency and Concurrency – Ensuring data consistency and concurrency are very critical for mobile applications. The good thing is that most databases already available in the market handle it well, including SQL Server for Windows CE that is being used for the problem-based learning application.

It is also important to select the correct transaction model to manage both online as well as offline transactions. Using the transaction model evaluation tool that was

developed as part of this research, the Pro-Motion model was selected for the PBL application since the course information is not expected to change frequently. Therefore, the mobile users may not need to be connected to the central database server; they should be able to perform transactions even in offline mode. Some of these transactions could be the addition of a new activity, addition of a new checklist, or the addition of dependent activities for an activity.

Since multiple mobile users may be involved in modifying the shared data, database level constraints and checks should be in place to make sure that the data committed to the central database is proper. In addition, some business rules may also need to be added to ensure that only the validated data is stored in the tables.

In addition, multiple mobile users may be engaged in modifying the same piece of data. Locking mechanisms can also be used to ensure that only one user can modify the critical data at a given point in time. For instance, to update the schedule for a project or an activity, an exclusive lock should be obtained on that row so that other users cannot modify the same data.

Synchronization and Conflict Resolution – Synchronization and conflict resolution is another aspect that mobile databases handle automatically with little or no help from the application developer or the end user. SQL Server for Windows CE provides priority-based data merging and conflict resolution when multiple users are trying to modify certain data. Some of the main reasons for selecting SQL Server for Windows CE as the mobile database for PBL application are as follows:

- Automatic synchronization

- Conflict resolution techniques
- Automatic data replication

High Availability – The fact that data also resides on the mobile device allows users to work offline. When the users come online, data is synchronized, thereby giving an “always connected” feeling. However, this does not mean that the central database need not be available around the clock.

A backup and recovery strategy needs to be developed to avoid extended downtimes due to system failures. Automated backup and recovery techniques can ensure continuous database availability. Data can also be replicated across multiple sites. If one site fails for some reason, the other site can switch over and respond to the user requests. This way, the users experience a minimal downtime.

Security – The problem-based application provides controlled access through different user types - student, faculty, and administrator. The students will have access to user interfaces that are applicable only to the students. The faculty members will have access to the user interfaces that students have access to and more. The administrator, naturally, would have access to the complete system. Additional security related features available in the PBL application are as follows:

- Multiple levels of user authentication - database level, IIS level, and application level
- The sensitive data is encrypted to ensure user privacy

- Data can be encrypted during transfer between the mobile device and central database server

Further research needs to be done in the area of high availability to provide continuous support for mobile applications and services. The amount of data is increasing so rapidly that it has become important to have strategies in place to perform automated backup and recovery operations. The distributed database architecture can also be used to provide database connectivity in case of server failure.

The following figure illustrates how load balancing could be used in a distributed architecture to minimize the downtime and thus provide high availability. The data can be replicated across multiple servers to provide load balancing. The requests from mobile users can be directed to the group of servers in a round robin fashion to distribute the workload.

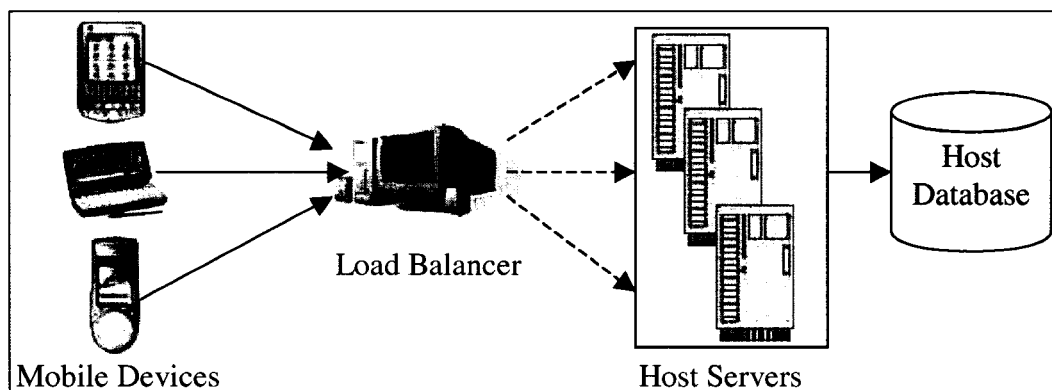


Figure 30: Load Balancing

In addition to this, work needs to be done in the area of data and network security. Although not very easy, it is possible for hackers to intercept wireless transmissions and acquire critical and sensitive information. Therefore, the mobile user needs to be educated in ways to ensure security of personal data.

REFERENCES

- Appelbe, H. (2003). *Mobile Location Based Services & Privacy*. Retrieved December 18, 2003, from http://www.irlogi.ie/pdf/Mobile_LBS_IRLOGI_2003.pdf
- AT&T Wireless (2004). *mMode Syncs you to the world*. Retrieved February 13, 2004, from <http://www.attwireless.com/personal/features/mmode/index.jhtml>
- Bloor Research (2001). *Oracle 9i from Oracle*. Retrieved May 26, 2004, from www.it-analysis.com/researcharchivepdf.php?id=374
- Brown, M. & Meunier, D. (2003). *Create Compact, Robust Mobile Apps with SQL Server CE 2.0 and the .NET Compact Framework*. Retrieved October 17, 2003, from <http://msdn.microsoft.com/msdnmag/issues/03/01/SQLServerCE20>
- Cahners In-Stat Group (2001, August 24). *Wireless Internet Panel Findings*. Retrieved July 6, 2003, from http://www.instat.com/panels/pdf/location_based_survey.pdf
- Chan, D. & Roddick, J. (2003). *Context-Sensitive Mobile Database Summarisation*. Retrieved December 29, 2003, from <http://crpit.com/confpapers/CRPITV16Chan.pdf>
- Clark, L. & Demir, O. (2003). *Transaction Management in Mobile Distributed Databases*. Retrieved April 30, 2004, from <http://sirius.cs.ucdavis.edu/teaching/265-SQ03/demir.pdf>
- Daniel, B. (1999). Mobile Computing and Databases-A Survey [Electronic version]. *IEEE Transactions on Knowledge and Data Engineering*, 11 (1), 108-117.
- Detecon (2003). *Multimedia-Messaging Services*. Retrieved November 30, 2003, from <http://www.detecon.com/load.php?url=L211ZGhL3BkZi9maW5hbFRyZW5kbGV0dGVyX01NU19fMTAwMjA0X2UucGRm>
- Dirckze, R. & Gruenwald, L. (2000). A pre-serialization transaction management technique for mobile multidatabases [Electronic version]. *Mobile Networks and Applications*, 5(4), 311-321.
- DSstar (2000). *Microsoft Details SQL Server 2000 Windows CE Edition*. Retrieved April 8, 2004, from <http://www.hpcwire.com/dsstar/00/0502/101587.html>

- Dunham, M., Helal, A., & Balakrishnan, S. (1997). A mobile transaction model that captures both the data and movement behavior [Electronic version]. *Mobile Networks and Applications*, 2, 149-162.
- Elmasari, N. (2004). *Mobile Databases*. Retrieved December 31, 2003, from http://www.cc.gatech.edu/classes/AY2004/cs4440-fall/ch27_mobile_final.pdf
- FCC (2001). *FCC Wireless 911 Requirements*. Retrieved November 15, 2003, from http://www.fcc.gov/911/enhanced/releases/factsheet_requirements_012001.pdf
- Forman, G. & Zahorjan, J. (1994). The Challenges of Mobile Computing [Electronic version]. *Computing Milieux*, 27 (4), 38-47.
- GIS Monitor (2003). *What are Location based Services Anyway?* Retrieved December 2, 2003, from <http://www.gismonitor.com/news/newsletter/archive/020603.php#What>
- GITUS (2000). *Mobile Communication*. Retrieved December 24, 2003, from <http://www.gitus.cz/en/mobcomm.html>
- Gupta, M. & Tang, Na (2003). *Query Processing Issues in Mobile Databases*. Retrieved December 19, 2003, from <http://sirius.cs.ucdavis.edu/teaching/265-SQ03/gupta.pdf>
- Hill, L. (2000). Alexandria Digital Library Project. *Core Elements of Digital Gazetteers: Placenames, Categories, and Footprints*. Retrieved July 6, 2003, from http://www.alexandria.ucsb.edu/~lhill/paper_drafts/ECDL2000_paperdraft7.pdf
- Hirsch, R., Coratella, A., Felder, M., & Rodriguez, E. (2001). A Framework for Analyzing mobile Transaction Models [Electronic version]. *Journal of Database Management*, 12(3), 36.
- Holliday, J., Agrawal, D., & Abbadi, A. (2002). Disconnection Modes for Mobile Databases [Electronic version]. *Wireless Networks*, 8(4), 391-402.
- Hopfner, H. & Sattler, K. (2003). *Cache-supported Processing of Queries in Mobile DBS*. Retrieved January 05, 2004, from http://www.witi.cs.uni-magdeburg.de/~hoepfner/paper/mDBIS03_Cache.pdf
- Hurson, A. & Lim, J. (2002). Transaction Processing in Mobile, Heterogeneous Database Systems [Electronic version]. *IEEE Transactions on Knowledge and Data Engineering*, 14(6), 1330-1346.

- IBM Corporation (2002). *Bring scalability to your mobile computing environment*. Retrieved March 29, 2004, from <http://www.appforge.com/products/ibm/DB2EPV8ss.pdf>
- IBM Corporation (2003). *IBM DB2 Everyplace Performance Tuning Guide*. Retrieved December 1, 2003, from <ftp://ftp.software.ibm.com/software/data/db2/everyplace/perftuning.pdf>
- IBM Corporation (2004). *DB2 Everyplace -- small footprint relational database*. Retrieved April 20, 2004, from <http://www.linuxdevices.com/products/PD8311509562.html>
- Jagoe, A. (2003). *Mobile Location Services: The Definitive Guide*. New Jersey: Prentice Hall.
- Jing, J., Helal, A., & Elmagarmid, A. (1999). Client-Server Computing in Mobile Environments [Electronic version]. *ACM Computing Surveys*, 31(2), 117-157.
- Johnson, K. (2002). *The Design Of Secure Mobile Databases: An Evaluation Of Alternative Secure Access Models*. Retrieved January 6, 2004, from <http://neoref.ils.unc.edu/2795.pdf>
- Jones, D. (1996a). *The Advantages of PBL*. Retrieved January 30, 2004, from <http://edweb.sdsu.edu/clrit/learningtree/PBL/PBLadvantages.html>
- Jones, D. (1996b). *What is PBL?*. Retrieved January 30, 2004, from <http://edweb.sdsu.edu/clrit/learningtree/PBL/WhatisPBL.html>
- Kan, S. (2003). *Metrics and Models in Software Quality Engineering*. Boston: Addison-Wesley.
- Krishnamurthi, G., Chessa, S., & Somani, A. (2000). Fast recovery from database/link failures in mobile networks [Electronic version]. *Computer Communications*, 23(5), 561-574.
- Larman, C. (2002). *Applying UML and Patterns*. New Jersey: Prentice Hall.
- Lee, K., leong, H., & Si, A. (1999). Semantic Query Caching in a Mobile Environment [Electronic version]. *Mobile Computing and Communications Review*, 3(2), 28-36.
- Lee, W. (2003). *Using Remote Data Access with SQL Server CE 2.0*. Retrieved January 6, 2003, from <http://www.ondotnet.com/pub/a/dotnet/2003/01/06/sqlce20.html>

- Leung, W. (2001). Is PBL better than traditional curriculum? [Electronic Version]. *STUDENTBMJ*, 9, 306-310.
- Lim, J., Hurson, A. (2001). Transaction Processing in a Mobile, Multi-Database Environment [Electronic version]. *Multimedia Tools and Applications*, 15, 161–185.
- Madria, S., Mohania, M., & Roddick, J. (1998). *Approximate Query Processing Model for Mobile Computing using Summary Databases*. Retrieved January 29, 2004, from http://www.cse.iitk.ac.in/users/cs634/www/Docs/query_processing.doc
- Microsoft Corporation (2002). *SQL Server CE 2.0 Product Overview*. Retrieved December 31, 2003, from <http://www.microsoft.com/sql/ce/productinfo/overview.asp>
- Mobile IN (2001). *Location-based Services*. Retrieved February 2, 2004, from http://www.mobilein.com/location_based_services.htm
- Morgan, B. (2001). *Resting On Their Shoulders*. Retrieved May 26, 2004, from http://www.wirelessinternetmag.com/news/0108/0108_devbiz_shoulders.htm
- Morgan Hill Unified School District (2003). *Advantages of PBL*. Retrieved February 5, 2004, from <http://www.lospaseos.mhu.k12.ca.us/McRodWeb/pbl/advantage.htm>
- OnStar Corporate Information. (2003). *About Us: Background*. Retrieved December 23, 2003, from <http://onstar.internetpressroom.com/pressroom.cfm>
- Oracle Corporation (2001). *Oracle9i Lite; Technical White Paper*. Retrieved November 15, 2003, from http://otn.oracle.com/products/lite/pdf/o9ilite_bwp.pdf
- Oracle Corporation (2004a). *Oracle Database Lite Overview*. Retrieved April 22, 2004, from http://otn.oracle.com/products/lite/lite_datasheet_10g.pdf
- Oracle Corporation (2004b). *Oracle9i Lite*. Retrieved April 22, 2004, from http://otn.oracle.com/products/lite/htdocs/o9ilite_datasheet.htm
- PC Magazine (2002). *On the Road to Mobile Databases*. Retrieved December 24, 2003, from http://www.pcmag.com/print_article/0,3048,a=23371,00.asp
- Pitoura, E. & Bhargava, B. (1995). *Maintaining consistency of data in mobile distributed environments*. Retrieved February 4, 2004, from <http://jotm.objectweb.org/related/icdcs95.pdf>

- Porta, T. (2002). Introduction to the IEEE Transactions on Mobile Computing [Electronic version]. *IEEE Transactions on Mobile Computing*, 1(1), 2-9.
- Racherla, G. & Das, A. (1996). Mobile Computing [Electronic version]. *IEEE Potentials*, 13-15.
- Rennhackkamp, M. (1997). Mobile Database Replication. 10(11), 81.
- Rennhackkamp, M. (1998a). Access your data anywhere, anytime [Electronic Version]. *E-Business Advisor*, 16 (10).
- Rennhackkamp, M. (1998b). *Mobile database is the way to empower a mobile workforce!*. Retrieved December 26, 2003, from <http://www.dba.co.za/MobileDatabase.htm>
- Rhem, J. (1998). *Problem-based Learning: An Introduction*. Retrieved January 23, 2004, from http://www.ntlf.com/html/pi/9812/pbl_1.htm
- Saygin, Y. & Ulusoy, O. (2002). Exploiting Data Mining Techniques for Broadcasting Data in Mobile Computing Environments [Electronic version]. *IEEE Transactions on Knowledge and Data Engineering*, 14 (6), 1387-1399.
- San Diego State University (1996a). *The Barriers to PBL*. Retrieved January 18, 2004, from <http://edweb.sdsu.edu/clrit/learningtree/PBL/PBLBarriers.html>
- San Diego State University (1996b). *How to Overcome Barriers and Implement PBL*. Retrieved January 18, 2004, from <http://edweb.sdsu.edu/clrit/learningtree/PBL/PBLimplementing.html>
- Segun, K., Hurson, A., Desai, V., Spink, A., & Miller, L. (2003). *Transaction Management in a Mobile Data Access System*. Retrieved April 23, 2004, from <http://www.comp.nus.edu.sg/~yuenck/35.pdf>
- Seshadri, O. & Garrett, P. (2000). SQLServer For Windows CE – A Database Engine for Mobile and Embedded Platforms [Electronic version]. *IEEE*, 642-644.
- Seydim, A. (1999). *An Overview of Transaction Models in Mobile Environments*. Retrieved December 29, 2003, from http://engr.smu.edu/~yasemin/mobile_trans.pdf
- Stephens, R. & Plew, R. (2001). *Database Design*. Sams Publishing.

- Sybase, Inc. (1999). *The Next Generation Database for Embedded System*. Retrieved September 12, 2003, from <http://www.sybase.com/content/1002378/embedded.pdf>
- Sybase, Inc. (2003). *UltraLite Database User's Guide*. Retrieved May 28, 2004, from <http://download.sybase.com/pdfdocs/awg0900e/ulfoen9.pdf>
- Thews, D. (2003). *Create Mobile Database Apps*. Retrieved December 24, 2003, from http://www.ftponline.com/vsm/2003_10/magazine/features/thews/default_pf.aspx
- Thomas, D. (2004). *Mobile computing improves productivity*. Retrieved March 15, 2004, from <http://www.vnunet.com/News/1152053>
- Wired News (1999). *The New Road Rage*. Retrieved March 20, 2003, from <http://www.wired.com/wired/archive/7.07/gm.html>
- Woo, A. (2001) *An Overview of Mobile Database Access*. Retrieved August 20, 2003, from <http://www.wookieweb.com/palm/case/mobiledb.htm>
- Xia, Y. & Helal A. (2003). *A Dynamic Data/Currency Protocol for Mobile Database Design and Reconfiguration*. Retrieved December 24, 2003, from <http://www.harris.cise.ufl.edu/projects/publications/HELLAL-MobileDB.pdf>

APPENDIX A: GLOSSARY

Term	Definition
Cell	A predefined geographical area, which forms the mobile network
Central Database	A database that is located on a fixed host
Compact	The smallest unit of data that can be cached on a mobile device for local transaction management
Distributed Database	A database that is located on multiple hosts
Entity Relationship Diagram	A diagram that provides information on database entities, their attributes, and relationships between them
Fixed Host	A server whose location remains fixed
Footprint	The size of mobile database
Global Transaction	A transaction applicable to a multi-database environment
Global Database Management System	A system in which mobile users can connect to a common interface without having to know the structure of local databases
Joey Transaction	In the Kangaroo transaction model, when a mobile device moves from one cell to another, a new instance of the transaction gets created, which becomes the parent of other transactions
Kangaroo Transaction	It represents one complete transaction that takes place in the Kangaroo Transaction model
Local Transaction	A transaction that occurs locally to a network
Mobile Computing	Ability of users to use wireless devices to perform mobile transactions
Mobile Database	A database that resides on a mobile device
Mobile Device	A wireless device that is capable of initiating mobile transactions
Mobile Host	Same as Mobile Device
Mobile Support Station	Mobile support station or MSS acts as a bridge between the mobile units and the fixed network and creates a mobile network

Term	Definition
Mobile Transaction	A transaction in which at least one mobile device is involved
Mobile Unit	Same as Mobile Device
Multi-database system	A system where multiple databases are setup together to be accessed by mobile devices
Seed Data	Basic data needed to setup the application
Transaction	A set of read and write operations on a database

APPENDIX B: ABBREVIATIONS

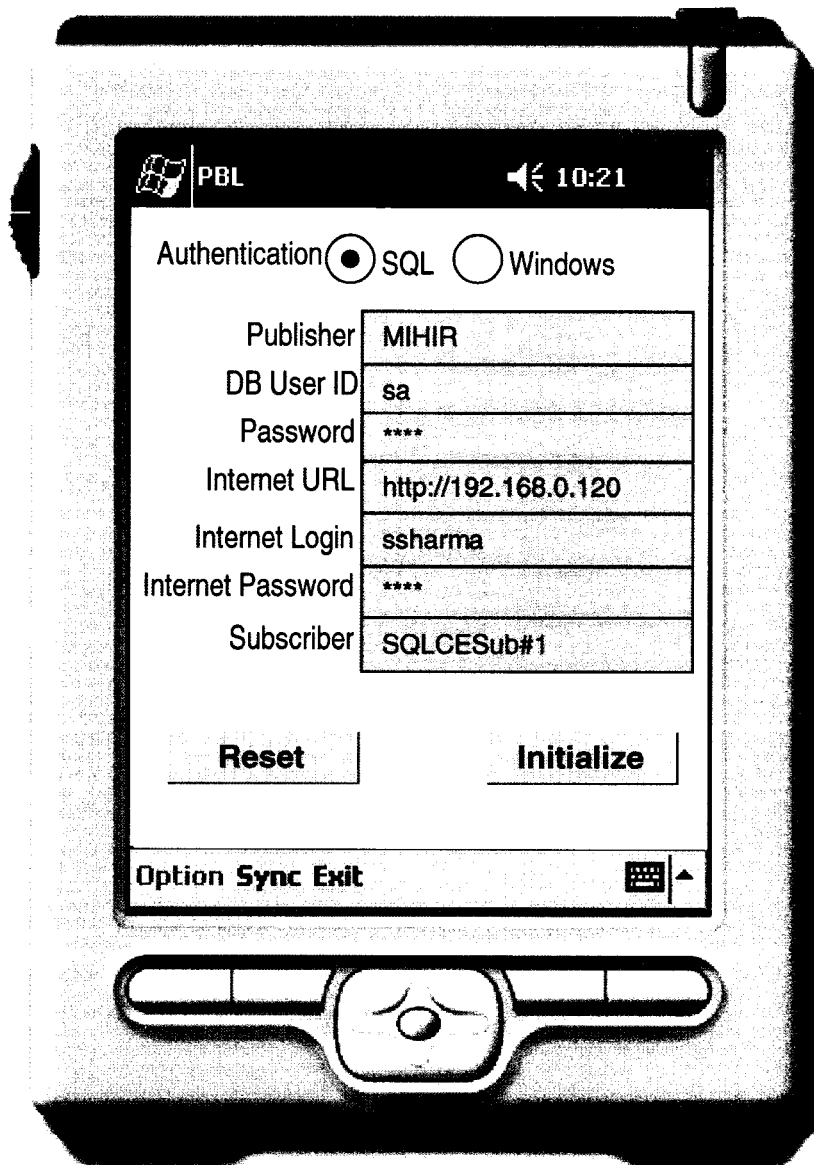
Abbreviation	Description
ACID	Atomicity, Consistency, Isolation, and Durability
DB	Database
DBMS	Database Management System
ERD	Entity Relationship Diagram
GPS	Global Positioning System
GT	Global Transaction
GTC	Global Transaction Coordinator
GTM	Global Transaction Manager
IIS	Internet Information Server
JT	Joey Transaction
KB	Kilobyte
KT	Kangaroo Transaction
LT	Local Transaction
MB	Megabyte
MH	Mobile Host
MMDBS	Mobile multi-database system
MSS	Mobile Support Station
MU	Mobile Unit
PBL	Problem-Based Learning
PDA	Personal Digital Assistant

Abbreviation	Description
QP	Query Processor
SDB	Summary Database
STM	Site Transaction Manager
URL	Uniform Resource Locator

APPENDIX C: USER INTERFACE

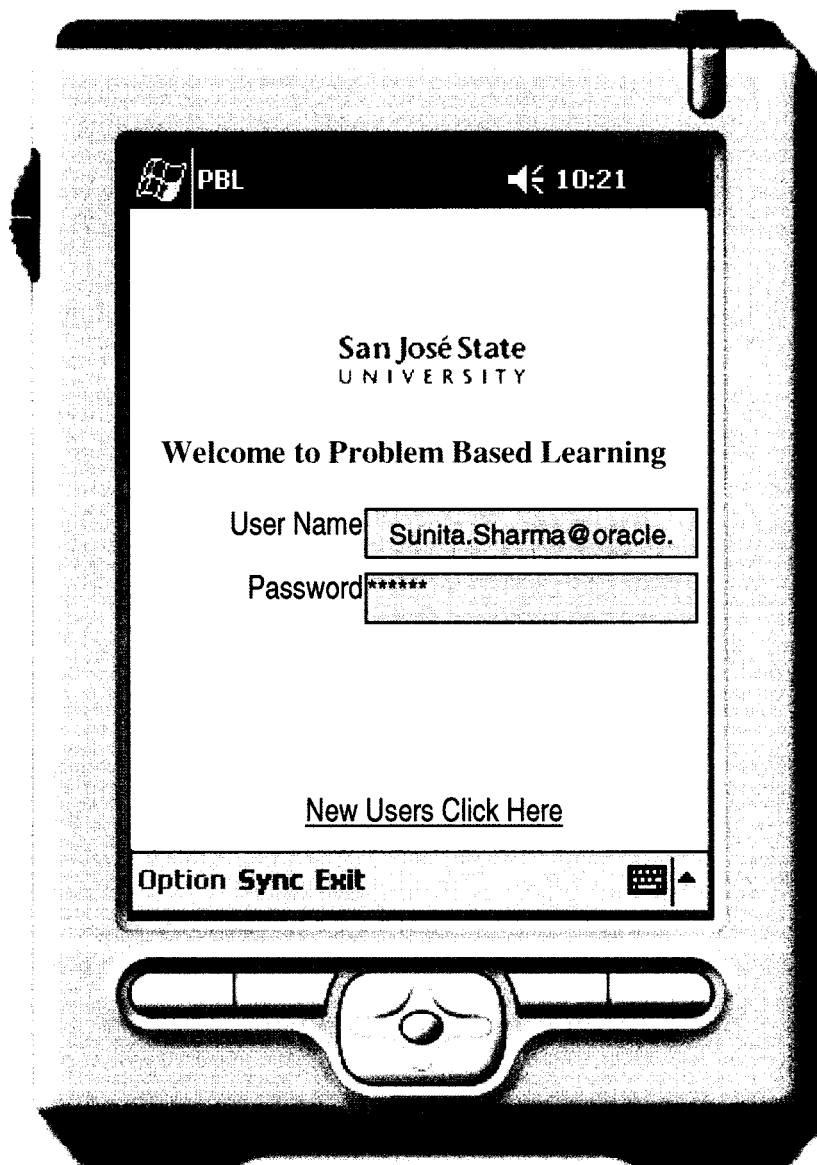
Setup

This is the initial user interface that is displayed when the PBL application is started. This allows the user to setup the initialization parameters such as the publisher, subscriber, and Internet URL. It is also used to perform the user authentication for accessing the central database and the IIS.



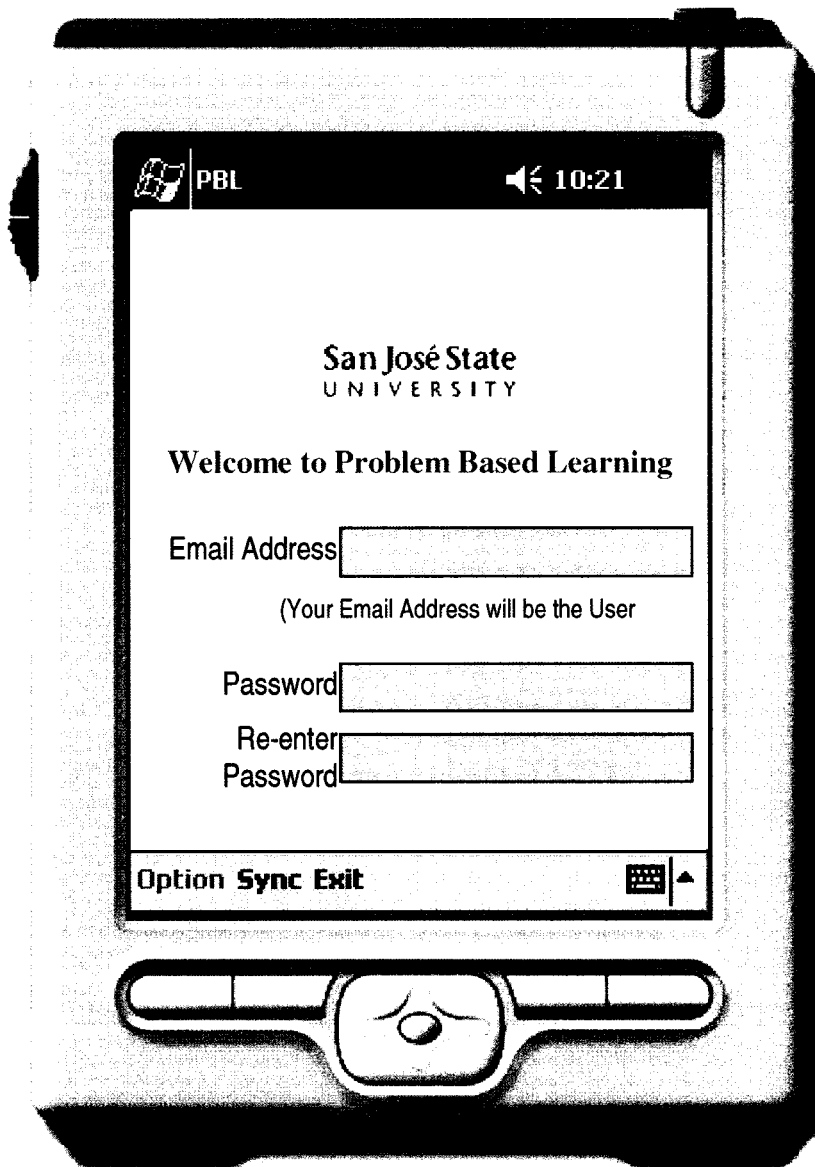
Login

This user interface allows the users to log into the PBL application using their user name and password. This is the third level of authentication, which comes after the database and IIS level authentication done as part of setup and initialization. Once the user is logged into the application, the Options menu item becomes enabled, which provides access to the other user interfaces.



Register

This user interface allows new users to register for the PBL application. The email address that the user specifies here is used as the user name for logging into the application. The email address was chosen as the user name for the application since it is unique and also provides contact information for the user.



The image shows a handheld device screen with a registration form. The screen is framed by a dark border. At the top left, there is a small icon of a computer monitor and the text "PBL". At the top right, there is a speaker icon and the time "10:21". The main content area is white and contains the following text and form elements:

San José State
UNIVERSITY

Welcome to Problem Based Learning

Email Address

(Your Email Address will be the User)

Password

Re-enter Password

At the bottom of the screen, there is a navigation bar with the text "Option Sync Exit" on the left and a small icon of a keyboard and an arrow on the right. Below the screen is a physical keypad with a central circular button.

My Profile

This user interface allows the users to update their personal information such as name, phone number, and address. At the time of registration, the user is automatically asked to provide this information.

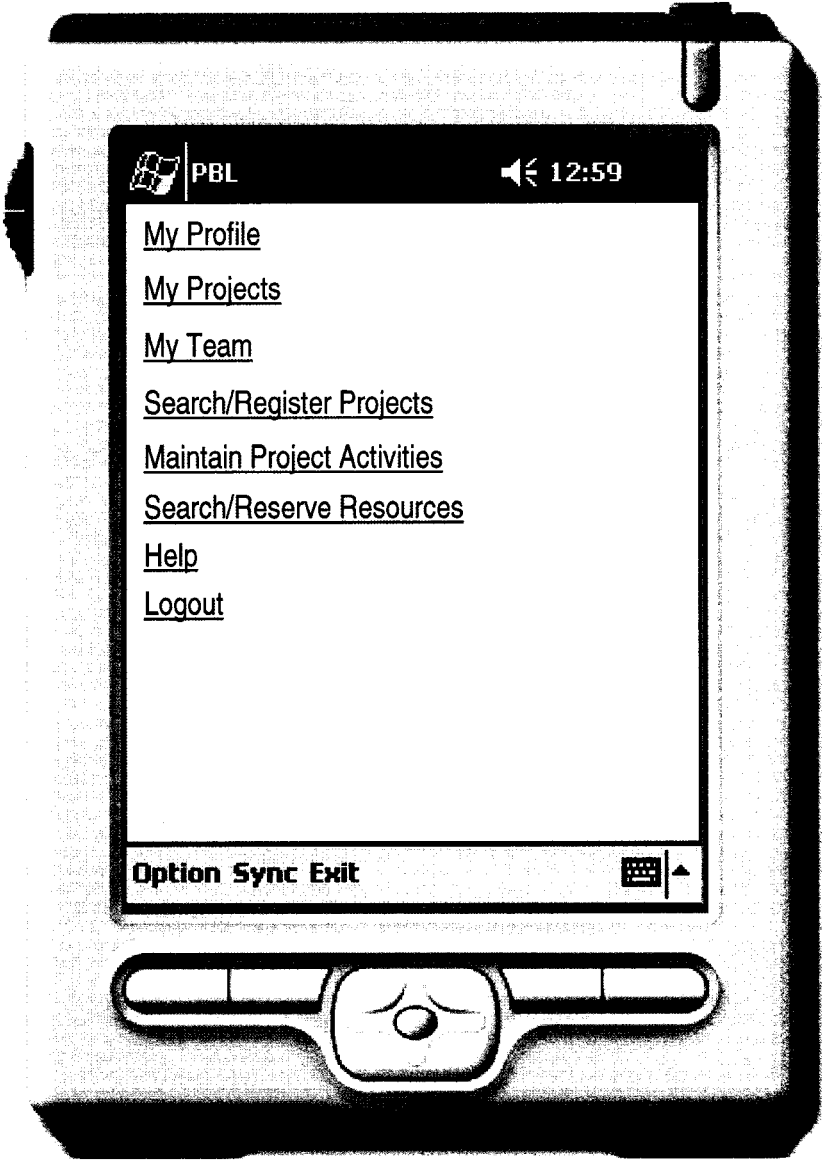
The image shows a mobile device screen displaying a 'My Profile' form. The form is titled 'My Profile' and contains the following fields and values:

First Name	Sunita
Middle Initials	
Last Name	Sharma
Department	Computer Engineering
Email	Sunita.Sharma@oracle.com
Day Phone	650-999-9999
Evening Phone	408-999-9999
Address	123 Sunnyvale Ave # 11
City	San Jose
State	CA
Postal Code	95133
Country	USA

At the bottom right of the form, there is a **Save** button. Below the form, there are navigation options: **Option Sync Exit** and a keyboard icon with an arrow pointing up.

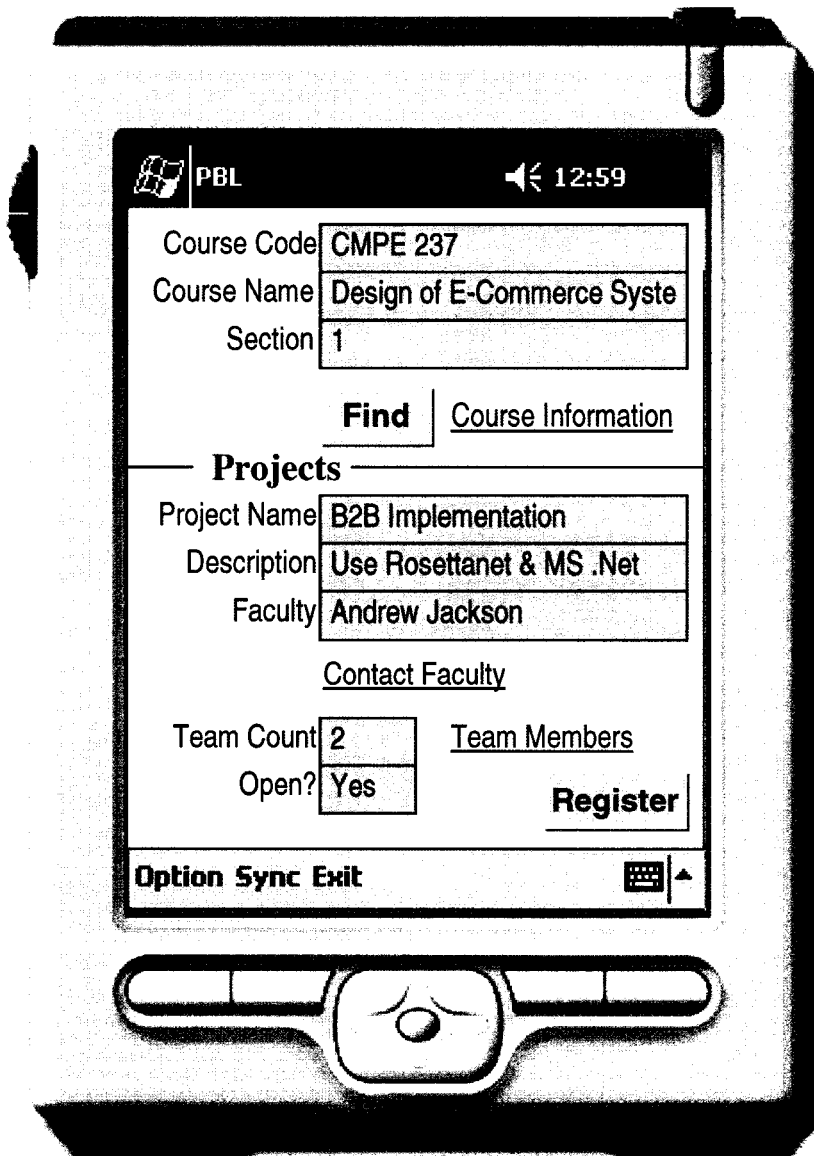
Quick Menu

This user interface provides a quick link to all main user interfaces such as My Profile, My Projects, Register Projects, and Reserve Resources.



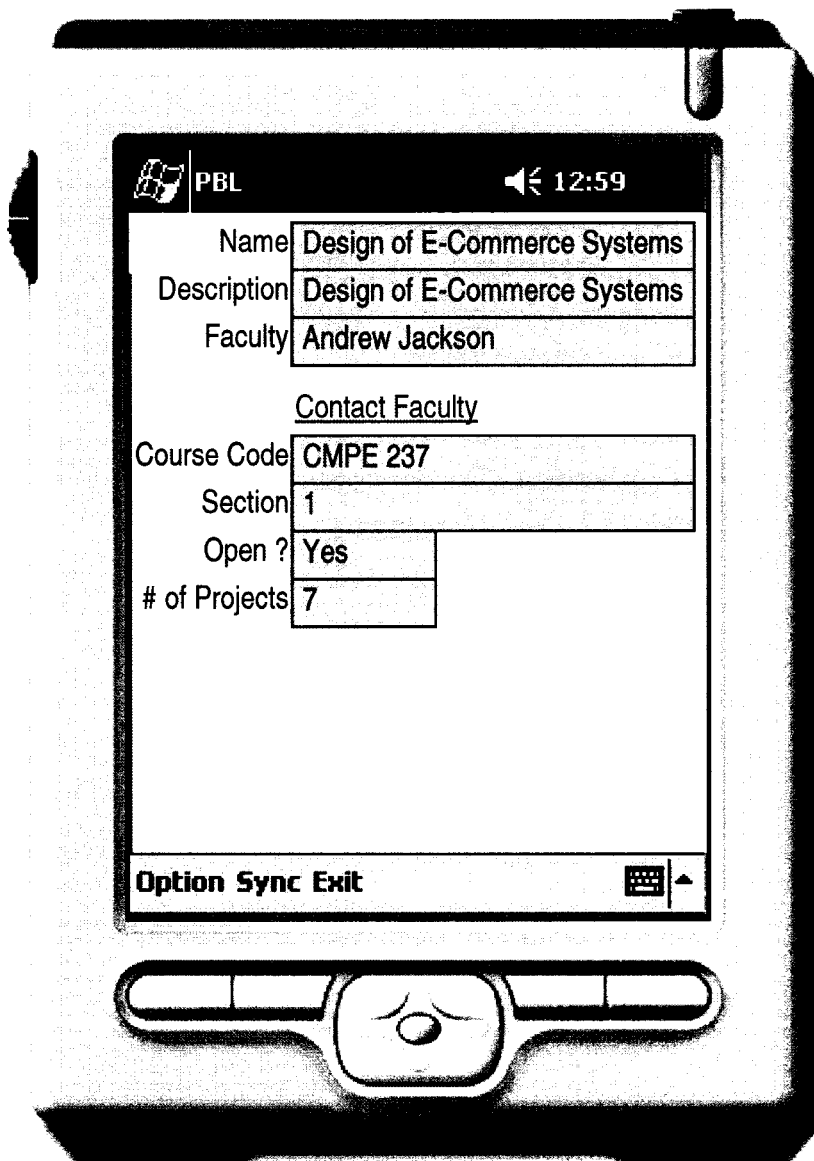
Project Registration

This user interface allows the user to search for projects, given the course code or name. The user can also register for a project if it is still open. Additional details on the course or project can be obtained by clicking on the Course Information link. An email can be sent to the faculty by clicking on the Contact Faculty link. The user can also view the list of existing team members by clicking on the Team Members link.



Course Information

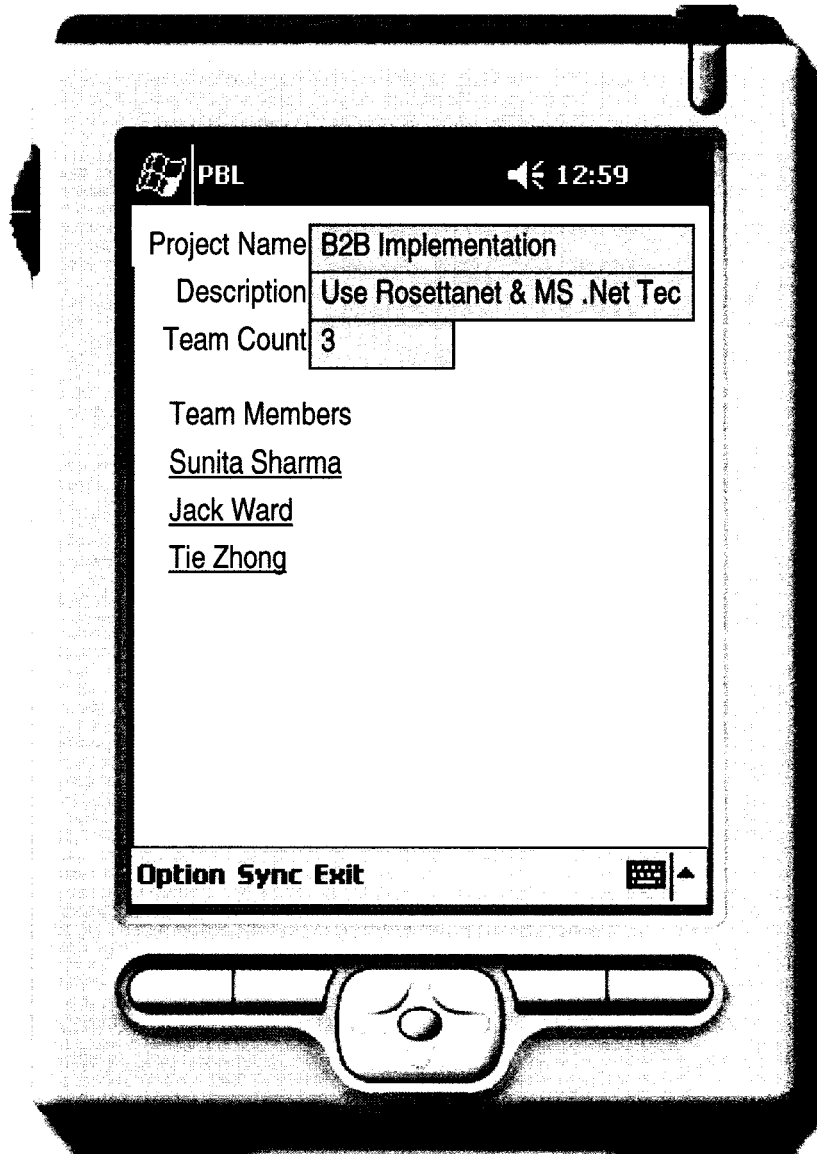
This user interface provides details on a particular course and allows the user to send an email to the faculty to get additional information. This user interface is accessible from other user interfaces such as the Course Information link on project registration user interface.



My Team

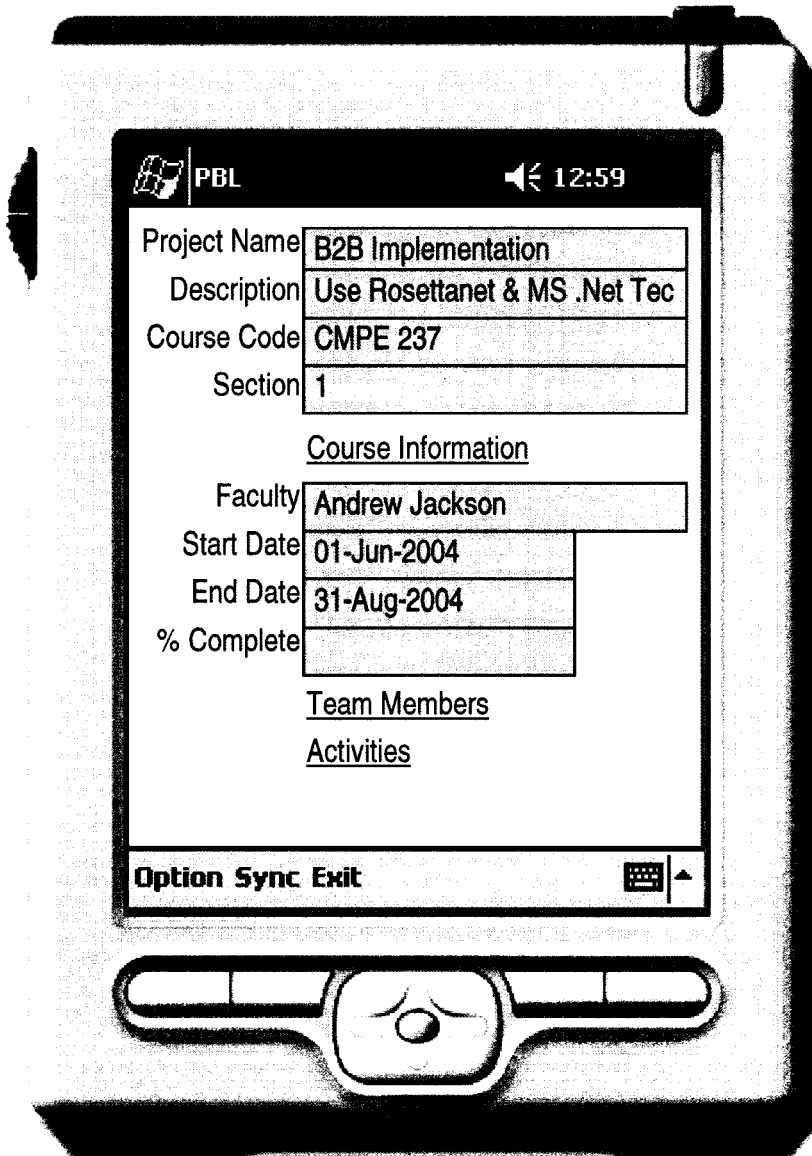
This user interface provides information on team members for a particular project.

This user interface is accessible from other user interfaces such as the Team Members link on project registration user interface.



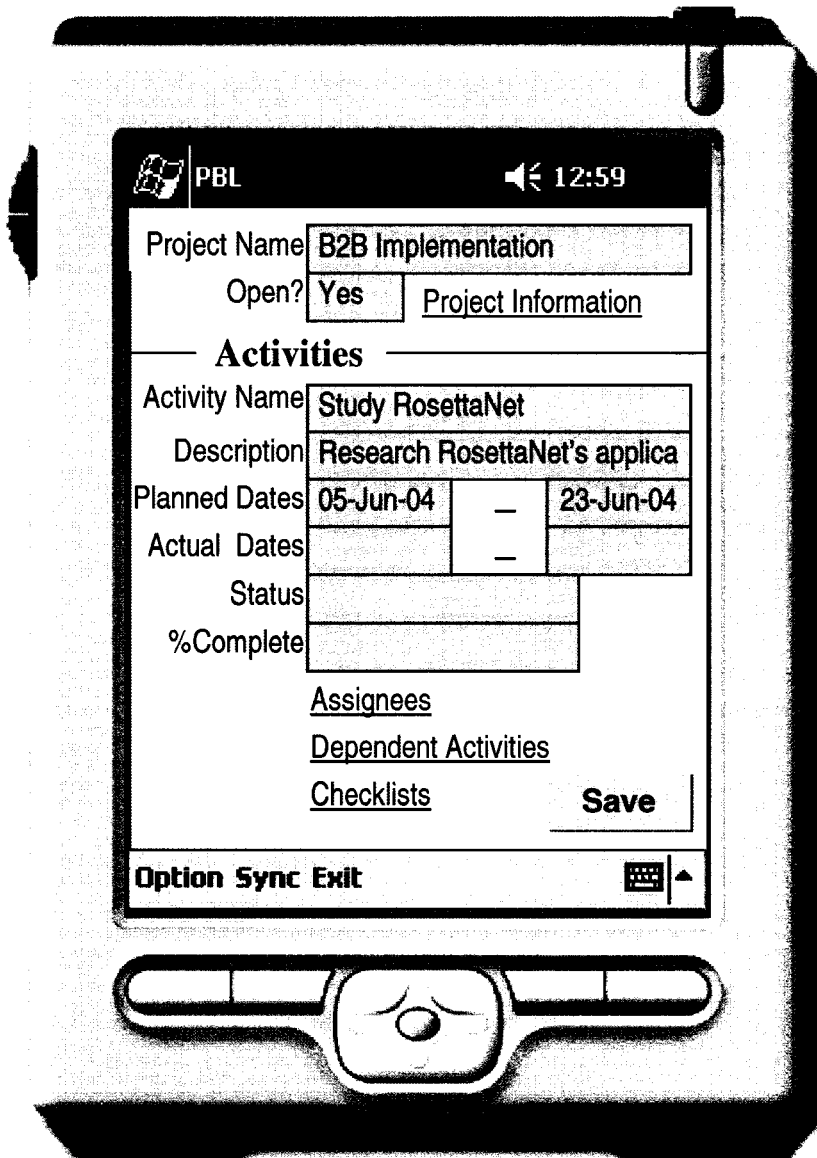
My Projects

This user interface provides details on the projects for which the user may have registered. This user interface also allows access to the Course Information and Team Members link. In addition, it provides a link to the activities for this particular project.



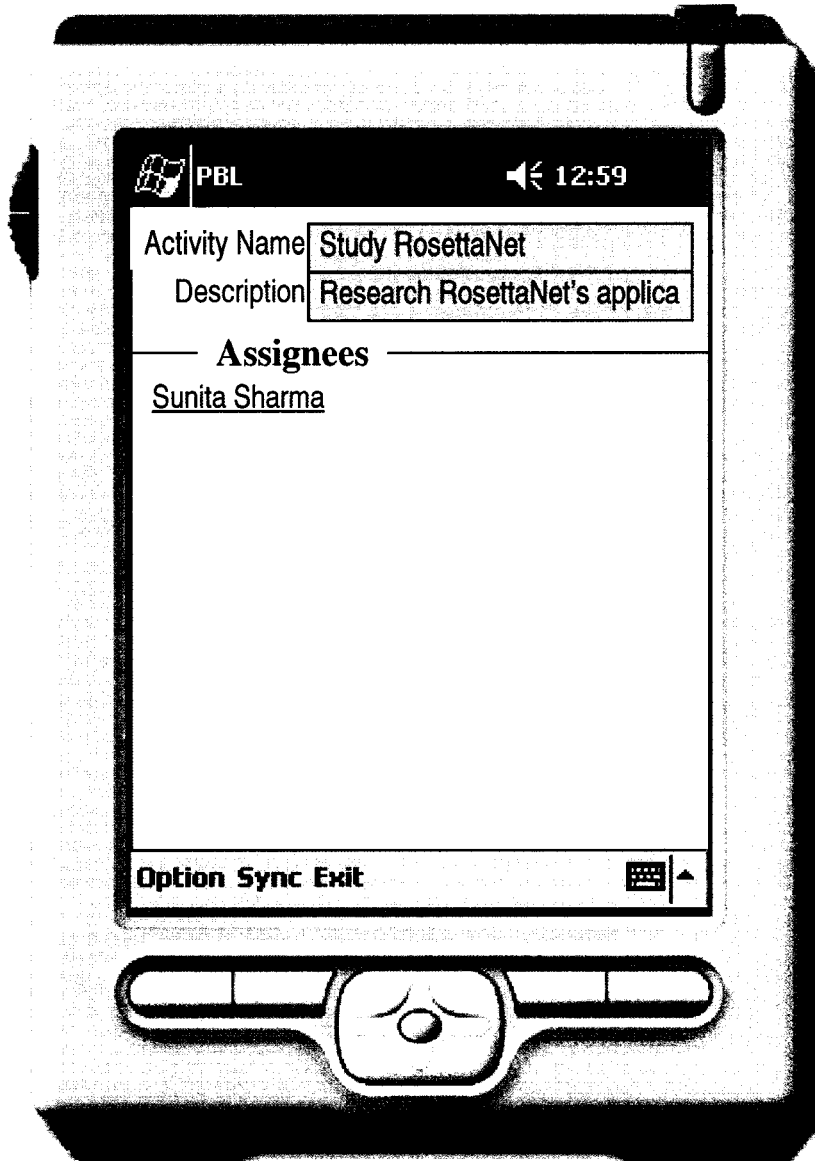
Project Activities

When the user clicks on the Activities link on My Projects user interface, the Project Activities user interface is displayed. This user interface allows the user to view and maintain activities for a particular project. It provides a link to view the assignees for a particular activity. It also allows the user to view the dependent activities for a particular activity. In addition, it provides a link to the activity checklists.



Assignees

This user interface provides the list of assignees for a particular activity. This user interface is accessible through the Assignees link on the Project Activities user interface.



Dependent Activities

This user interface allows the user to view and maintain the list of dependent activities for a particular activity. Unless all of the dependent activities are completed, the main activity cannot be completed. This user interface is accessible through the Dependent Activities link on the Project Activities user interface.

The screenshot shows a handheld device screen with the following content:

Activity Name: Implement RosettaNet for B2B
Description: RosettaNet Implementation
Status: Open
%Complete: []

Dependent Activities

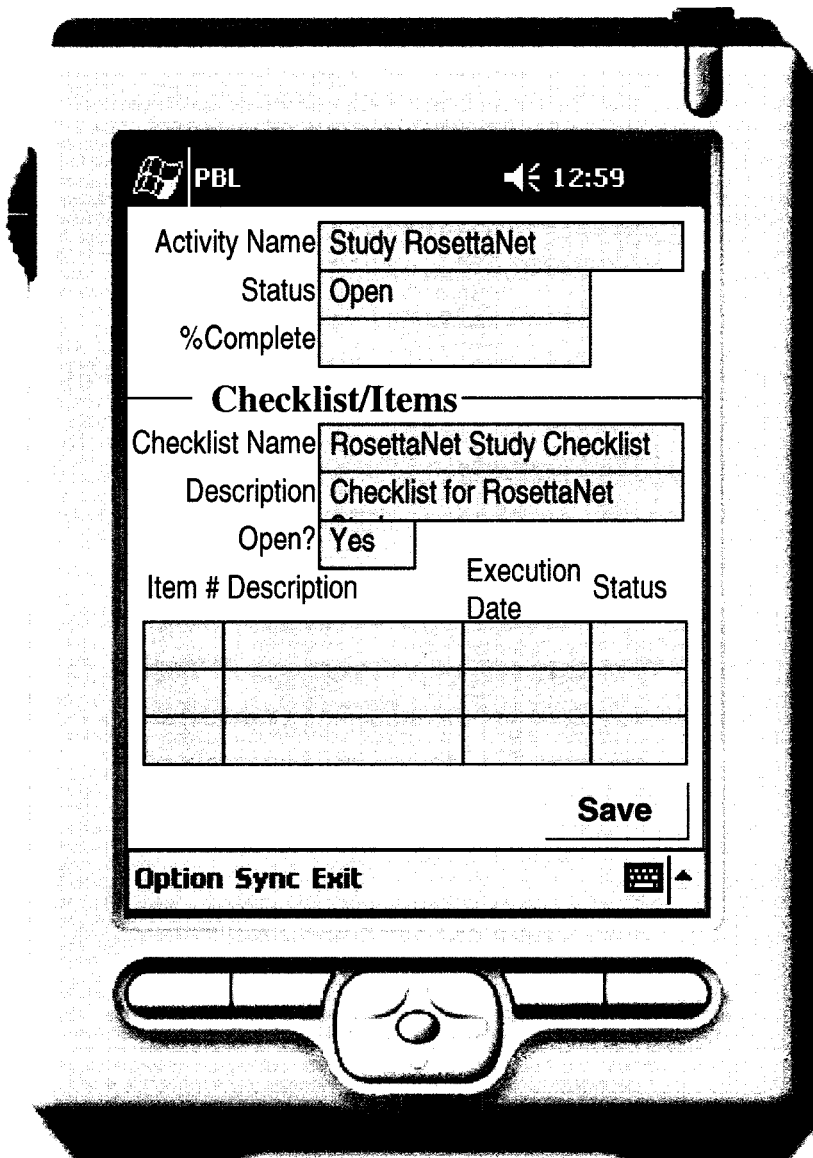
Activity#	Name
1	Study RosettaNet
2	Study Microsoft .Net Technology

Save

Option Sync Exit

Activity Checklists

This user interface allows the user to view and maintain checklist and checklist items for a particular activity. Unless all of the checklist items are marked as Complete, the activity cannot be completed. This user interface is accessible through the Checklists link on the Project Activities user interface.



The screenshot shows a handheld device screen with a software interface. At the top left is a logo and the text 'PBL'. At the top right is a back arrow and the time '12:59'. The main form contains the following fields:

- Activity Name: Study RosettaNet
- Status: Open
- %Complete: [Empty field]

Below these fields is a section header 'Checklist/Items'. Under this header are the following fields:

- Checklist Name: RosettaNet Study Checklist
- Description: Checklist for RosettaNet
- Open?: Yes

Below these fields is a table with the following columns: Item #, Description, Execution Date, and Status. The table is currently empty.

At the bottom right of the form is a 'Save' button. At the bottom left of the screen is the text 'Option Sync Exit' and a keyboard icon.

Reserve Resources

This user interface allows the user to search for resources and then reserve the resources if they are available.

The screenshot shows a handheld device screen with the following content:

Name	Computer Lab
Description	Computer Lab
Code	COMP
Type	LAB
Location	Engineering Building
Quantity	3

Find

Reservation

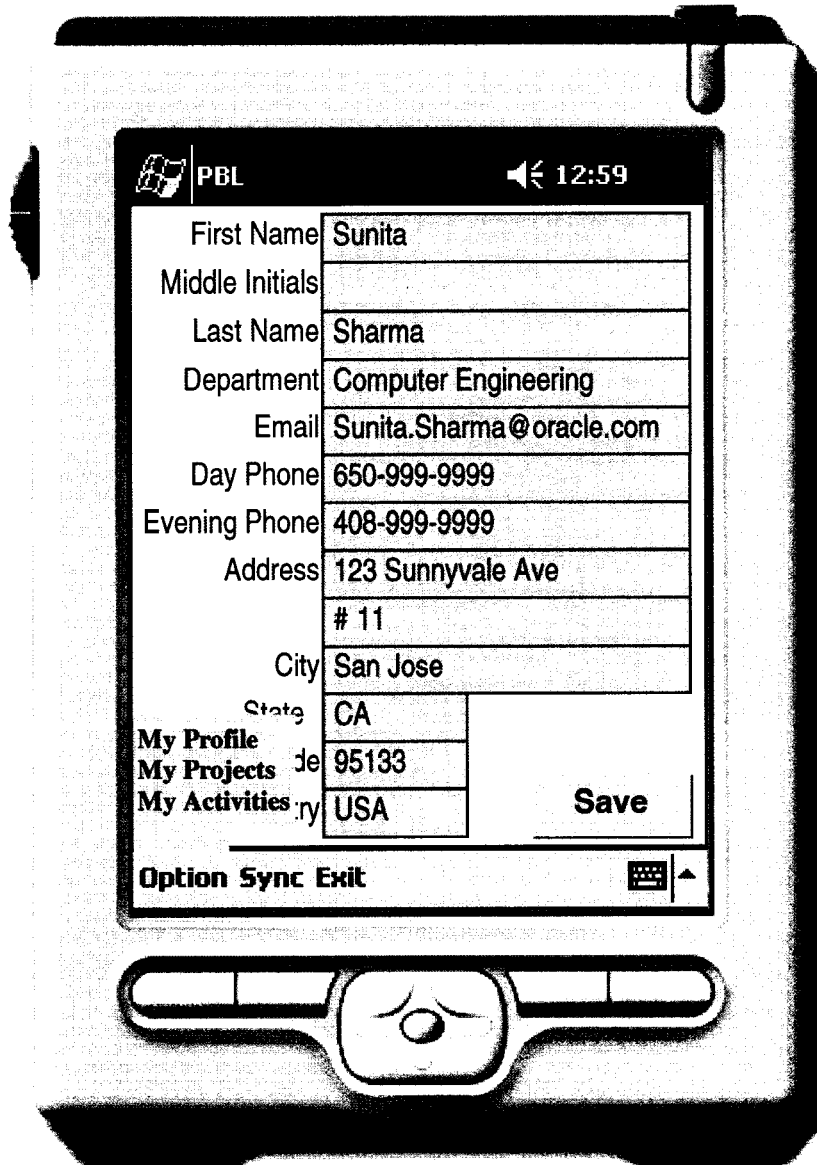
Project Name	B2B Implementation
Quantity	1
UOM	Each
Duration-Days	0.5

Reserve

Option Sync Exit

Options Menu

The options menu provides access to My Project, My Profiles, and My Activities user interfaces.



APPENDIX D: DATABASE SCHEMA

Table: PBL_DEPARTMENTS

This table stores information related to departments that offer PBL courses.

Column	Data Type	Comments
DEPARTMENT_ID	NUMBER	Unique department identifier
DEPARTMENT_CODE	VARCHAR (50)	Department code
NAME	VARCHAR (100)	Department name
ACTIVE_FLAG	CHAR (1)	Active or Inactive
CREATED_BY	NUMBER	User who created the record
CREATION_DATE	DATE	Date when the record was created
UPDATED_BY	NUMBER	User who updated the record
LAST_UPDATE_DATE	DATE	Date when the record was updated

Table: PBL_USERS

This table stores information about the registered PBL users.

Column	Data Type	Comments
USER_ID	NUMBER	Unique user identifier
USER_NAME	VARCHAR (250)	User name for authentication
PASSWORD	VARCHAR (100)	Password for authentication
USER_TYPE	VARCHAR (10)	Student, Faculty, or Administrator
FIRST_NAME	VARCHAR (100)	First name of the user
MIDDLE_INITIAL	VARCHAR (100)	Middle initial of the user
LAST_NAME	VARCHAR (100)	Last name of the user

Column	Data Type	Comments
ADDRESS_LINE_1	VARCHAR (250)	Line one of the user address
ADDRESS_LINE_2	VARCHAR (250)	Line two of the user address
ADDRESS_LINE_3	VARCHAR (250)	Line three of the user address
CITY	VARCHAR (100)	City
STATE	VARCHAR (100)	State
POSTAL_CODE	VARCHAR (50)	Postal code
COUNTRY	VARCHAR (100)	Country
DAYTIME_PHONE	VARCHAR (100)	Daytime phone number
EVENING_PHONE	VARCHAR (100)	Evening phone number
EMAIL_ADDRESS	VARCHAR (250)	Email address
ACTIVE_FLAG	CHAR (1)	Active or Inactive
DEPARTMENT_ID	NUMBER	Department id
CREATED_BY	NUMBER	User who created the record
CREATION_DATE	DATE	Date when the record was created
UPDATED_BY	NUMBER	User who updated the record
LAST_UPDATE_DATE	DATE	Date when the record was updated

Table: PBL_COURSES

This table stores information on the courses offered in the PBL environment.

Column	Data Type	Comments
COURSE_ID	NUMBER	Unique course identifier
COURSE_CODE	VARCHAR (50)	Course code

Column	Data Type	Comments
NAME	VARCHAR (100)	Name of the course
DESCRIPTION	VARCHAR (250)	Description of the course
ACTIVE_FLAG	CHAR (1)	Active or Inactive
DEPARTMENT_ID	NUMBER	Department id
FACULTY_ID	NUMBER	User id of the faculty
CREATED_BY	NUMBER	User who created the record
CREATION_DATE	DATE	Date when the record was created
UPDATED_BY	NUMBER	User who updated the record
LAST_UPDATE_DATE	DATE	Date when the record was updated

Table: PBL_COURSE_SECTIONS

This table stores information on the available sections for each course.

Column	Data Type	Comments
COURSE_SECTION_ID	NUMBER	Unique course section identifier
COURSE_ID	NUMBER	Course id
SECTION_NUMBER	NUMBER	Section number
OPEN_FLAG	CHAR (1)	Open or Closed
CREATED_BY	NUMBER	User who created the record
CREATION_DATE	DATE	Date when the record was created
UPDATED_BY	NUMBER	User who updated the record
LAST_UPDATE_DATE	DATE	Date when the record was updated

Table: PBL_PROJECTS

This table stores information on the PBL projects. Depending upon the number of students and groups, there can be a multiple projects in one section.

Column	Data Type	Comments
PROJECT_ID	NUMBER	Unique project identifier
COURSE_SECTION_ID	NUMBER	Course section id
NAME	VARCHAR (100)	Project name
DESCRIPTION	VARCHAR (250)	Project description
START_DATE	DATE	Start date of the project
END_DATE	DATE	End date of the project
OPEN_FLAG	CHAR (1)	Open or Closed
PERCENTAGE_COMPLETED	NUMBER	Percentage of the project completed, which can either be updated by the faculty or a background process
CREATED_BY	NUMBER	User who created the record
CREATION_DATE	DATE	Date when the record was created
UPDATED_BY	NUMBER	User who updated the record
LAST_UPDATE_DATE	DATE	Date when the record was updated

Table: PBL_STUDENT_PROJECTS

There is a many-to-many relationship between students and projects, which is maintained in this table.

Column	Data Type	Comments
STUDENT_ID	NUMBER	Student id
PROJECT_ID	NUMBER	Project id
CREATED_BY	NUMBER	User who created the record
CREATION_DATE	DATE	Date when the record was created
UPDATED_BY	NUMBER	User who updated the record
LAST_UPDATE_DATE	DATE	Date when the record was updated

Table: PBL_RESOURCES

This table stores information on the resources available to the PBL users.

Column	Data Type	Comments
RESOURCE_ID	NUMBER	Unique resource identifier
RESOURCE_CODE	VARCHAR (100)	Resource code
NAME	VARCHAR (100)	Name of the resource
DESCRIPTION	VARCHAR (250)	Description of the resource
RESOURCE_TYPE	VARCHAR (50)	Book, Software, or Lab
LOCATION	VARCHAR (250)	Location code
QUANTITY	NUMBER	Total quantity of the resource
ACTIVE_FLAG	CHAR (1)	Active or Inactive
CREATED_BY	NUMBER	User who created the record
CREATION_DATE	DATE	Date when the record was created
UPDATED_BY	NUMBER	User who updated the record
LAST_UPDATE_DATE	DATE	Date when the record was updated

Table: PBL_PROJECT_RESOURCE_ALLOCATIONS

This table stores resource allocations for a particular project.

Column	Data Type	Comments
PROJECT_ID	NUMBER	Project id
RESOURCE_ID	NUMBER	Resource id
ALLOCATED_QUANTITY	NUMBER	Quantity of the resource allocated to the project
UNIT_OF_MEASURE	VARCHAR (50)	Meters, Pounds, Hour, or Day
DURATION	NUMBER	Duration for which the resource is allocated to the project
CREATED_BY	NUMBER	User who created the record
CREATION_DATE	DATE	Date when the record was created
UPDATED_BY	NUMBER	User who updated the record
LAST_UPDATE_DATE	DATE	Date when the record was last updated

Table: PBL_ACTIVITIES

Each project can be divided into one or more activities. This table stores information on the activities that a project involves.

Column	Data Type	Comments
ACTIVITY_ID	NUMBER	Unique activity identifier
PROJECT_ID	NUMBER	Project id
NAME	VARCHAR (100)	Activity name
DESCRIPTION	VARCHAR (250)	Activity description

Column	Data Type	Comments
STATUS	VARCHAR (10)	Open, Closed, or Pending
PLANNED_START_DATE	DATE	Planned start date for the activity
PLANNED_END_DATE	DATE	Planned end date for the activity
ACTUAL_START_DATE	DATE	Actual start date for the activity
ACTUAL_END_DATE	DATE	Actual end date for the activity
PERCENTAGE_COMPLETED	NUMBER	Percentage of the activity completed, which can either be updated by the student or a background process
CREATED_BY	NUMBER	User who created the record
CREATION_DATE	DATE	Date when the record was created
UPDATED_BY	NUMBER	User who updated the record
LAST_UPDATE_DATE	DATE	Date when the record was updated

Table: PBL_ACTIVITY_DEPENDENCIES

This table stores information on the dependent activities.

Column	Data Type	Comments
ACTIVITY_ID	NUMBER	Activity id
DEPENDENT_ACTIVITY_ID	NUMBER	Dependent activity id
CREATED_BY	NUMBER	User who created the record

Column	Data Type	Comments
CREATION_DATE	DATE	Date when the record was created
UPDATED_BY	NUMBER	User who updated the record
LAST_UPDATE_DATE	DATE	Date when the record was updated

Table: PBL_STUDENT_ACTIVITIES

There is a many-to-many relationship between students and activities, which is maintained in this table.

Column	Data Type	Comments
STUDENT_ID	NUMBER	User id of the student
ACTIVITY_ID	NUMBER	Activity id
CREATED_BY	NUMBER	User who created the record
CREATION_DATE	DATE	Date when the record was created
UPDATED_BY	NUMBER	User who updated the record
LAST_UPDATE_DATE	DATE	Date when the record was updated

Table: PBL_CHECKLISTS

This table stores the checklist for each activity.

Column	Data Type	Comments
CHECKLIST_ID	NUMBER	Unique checklist identifier
ACTIVITY_ID	NUMBER	Activity id
NAME	VARCHAR (100)	Name of the checklist
DESCRIPTION	VARCHAR (250)	Description for the checklist

Column	Data Type	Comments
OPEN_FLAG	CHAR (1)	'Y' or 'N'. The value of this field is determined using the STATUS column of the PBL_CHECKLIST_ITEMS table
CREATED_BY	NUMBER	User who created the record
CREATION_DATE	DATE	Date when the record was created
UPDATED_BY	NUMBER	User who updated the record
LAST_UPDATE_DATE	DATE	Date when the record was updated

Table: PBL_CHECKLIST_ITEMS

This table contains the items for each checklist.

Column	Data Type	Comments
CHECKLIST_ITEM_ID	NUMBER	Unique checklist item identifier
CHECKLIST_ID	NUMBER	Checklist id
NAME	VARCHAR (100)	Name of the checklist item
DESCRIPTION	VARCHAR (250)	Description of the checklist item
EXECUTION_DATE	DATE	Date the item was executed
STATUS	VARCHAR (10)	Pending, Partially Completed, or Completed
CREATED_BY	NUMBER	User who created the record
CREATION_DATE	DATE	Date when the record was created
UPDATED_BY	NUMBER	User who updated the record
LAST_UPDATE_DATE	DATE	Date when the record was updated

Table: PBL_LOOKUP_TYPES

This table stores information on PBL lookup types.

Column	Data Type	Comments
TYPE	VARCHAR (100)	Type of the lookup
NAME	VARCHAR (100)	Name of the lookup type
DESCRIPTION	VARCHAR (250)	Description
CREATED_BY	NUMBER	User who created the record
CREATION_DATE	DATE	Date when the record was created
UPDATED_BY	NUMBER	User who updated the record
LAST_UPDATE_DATE	DATE	Date when the record was updated

Table: PBL_LOOKUP_CODES

This table stores the codes for each lookup type.

Column	Data Type	Comments
TYPE	VARCHAR (100)	Type of the lookup
CODE	VARCHAR (10)	Lookup code
NAME	VARCHAR (100)	Name of the lookup code
DESCRIPTION	VARCHAR (250)	Description of lookup code
ENABLED_FLAG	CHAR (1)	"Y" or "N"
CREATED_BY	NUMBER	User who created the record
CREATION_DATE	DATE	Date when the record was created
UPDATED_BY	NUMBER	User who updated the record
LAST_UPDATE_DATE	DATE	Date when the record was updated

View: PBL_STUDENTS

This view is based upon the PBL_USERS table for active PBL students.

View Query

```
SELECT
PU.FIRST_NAME,
PU.MIDDLE_INITIAL,
PU.LAST_NAME,
PU.ADDRESS_LINE_1,
PU.ADDRESS_LINE_2,
PU.ADDRESS_LINE_3,
PU.CITY,
PU.STATE,
PU.COUNTRY,
PU.POSTAL_CODE,
PU.EMAIL_ADDRESS,
PU.DAYTIME_PHONE,
PU.EVENING_PHONE,
PD.NAME DEPARTMENT_NAME
FROM PBL_USERS PU, PBL_DEPARTMENTS PD
WHERE PU.USER_TYPE = 'STUDENT'
AND PU.ACTIVE_FLAG = 'Y'
AND PU.DEPARTMENT_ID = PD.DEPARTMENT_ID
AND PD.ACTIVE_FLAG = 'Y'
```

View: PBL_FACULTY

This view is based upon the PBL_USERS table for active PBL faculty members.

View Query

```
SELECT
PU.FIRST_NAME,
PU.MIDDLE_INITIAL,
PU.LAST_NAME,
PU.ADDRESS_LINE_1,
PU.ADDRESS_LINE_2,
PU.ADDRESS_LINE_3,
PU.CITY,
PU.STATE,
PU.COUNTRY,
PU.POSTAL_CODE,
PU.EMAIL_ADDRESS,
PU.DAYTIME_PHONE,
PU.EVENING_PHONE,
PD.NAME DEPARTMENT_NAME
FROM PBL_USERS PU, PBL_DEPARTMENTS PD
WHERE PU.USER_TYPE = 'FACULTY'
AND PU.ACTIVE_FLAG = 'Y'
AND PU.DEPARTMENT_ID = PD.DEPARTMENT_ID
AND PD.ACTIVE_FLAG = 'Y'
```

View: PBL_LOOKUPS

This view is based upon the PBL_LOOKUP_TYPES and PBL_LOOKUP_CODES tables for active PBL lookups.

View Query

```
SELECT
PLT.TYPE LOOKUP_TYPE
PLT.NAME LOOKUP_NAME
PLC. CODE LOOKUP_CODE
PLC.NAME MEANING
FROM PBL_LOOKUP_TYPES PLT, PBL_LOOKUP_CODES PLC
WHERE PLC.TYPE = PLT.TYPE
AND PLD.ENABLED_FLAG = 'Y'
```

APPENDIX E: USE CASES

Use Case ID:	1
Use Case Title:	Manage Departments
Primary Actors:	Administrator
Stakeholders and Interests: Student, Faculty	
Preconditions: 1. The department code and name is available.	
The department information is stored in the database.	
Post-conditions: 1. The department information has been stored successfully in the database.	
Main Success Scenario: 1. The department code and name are retrieved for each department. 2. The code and name for each department is stored.	
Alternative Flow: 1. If any required information is unavailable, an appropriate error message is displayed to the user	
Special Requirements: None	
Technology and Data Variations List: None	

Use Case ID:	2
Use Case Title:	Manage Users
Primary Actors:	Administrator
Stakeholders and Interests: Student, Faculty, and Administrator	
Preconditions:	
<ol style="list-style-type: none"> 1. The department information for each user is available. 	
The user information is stored in the database.	
Post-conditions:	
<ol style="list-style-type: none"> 1. The user information has been stored successfully in the database. 	
Main Success Scenario:	
<ol style="list-style-type: none"> 1. Each user is assigned a user name and password. Since email address for each user is supposed to be unique, it is set as the user name. 2. The type is set for each user. 3. The department information for each user is stored. 4. The address information for each user is also stored. 	
Alternative Flow:	
<ol style="list-style-type: none"> 1. If any required information is unavailable, an appropriate error message is displayed to the user 	
Special Requirements: None	
Technology and Data Variations List: None	

Use Case ID:	3
Use Case Title:	Manage Courses
Primary Actors:	Administrator
Stakeholders and Interests: Student, Faculty	
Preconditions:	
<ol style="list-style-type: none"> 1. The course is divided into different sections. 2. The faculty and department information is available. 	
For each course, the section, department, and faculty information is stored.	
Post-conditions:	
<ol style="list-style-type: none"> 1. The course information has been stored successfully in the database. 	
Main Success Scenario:	
<ol style="list-style-type: none"> 1. The course code, name, and section number are retrieved for each course. 2. The information for each course is stored. 3. A faculty member is assigned to each course. 4. The department id is associated with each course. 	
Alternative Flow:	
<ol style="list-style-type: none"> 1. If any required information is unavailable, an appropriate error message is displayed to the user 	
Special Requirements: None	
Technology and Data Variations List: None	

Use Case ID:	4
Use Case Title:	Manage Projects
Primary Actors:	Administrator
Stakeholders and Interests: Student, Faculty	
Preconditions:	
<ol style="list-style-type: none"> 1. The project information is available. 2. The faculty and course information is available. 	
For each project, the course, section, and faculty information is stored.	
Post-conditions:	
<ol style="list-style-type: none"> 1. The project details have been stored successfully in the database. 	
Main Success Scenario:	
<ol style="list-style-type: none"> 1. The course and section information is retrieved for each project. 2. The faculty information for each project is stored. 3. The scheduled start and end date for each project is stored. 	
Alternative Flow:	
<ol style="list-style-type: none"> 1. If any required information is unavailable, an appropriate error message is displayed to the user. 	
Special Requirements: None	
Technology and Data Variations List: None	

Use Case ID:	5
Use Case Title:	Login
Primary Actors:	Student, Faculty
Stakeholders and Interests: Student, Faculty	
Preconditions:	
1. The User should have a valid user name and password.	
The Student or Faculty is identified and authenticated	
Post-conditions:	
1. Student or Faculty has successfully logged on to the PBL system.	
Main Success Scenario:	
1. The student or faculty logs in using the user name and password.	
2. The user is authenticated and the relevant project information is displayed.	
3. This list is displayed on the screen.	
Alternative Flow:	
1. If the user name or password is incorrect, an appropriate error message is displayed to the user.	
2. The student can search for a particular project independent of the projects displayed on the screen.	
Special Requirements: None	
Technology and Data Variations List: None	

Use Case ID:	6
Use Case Title:	Create Project Details
Primary Actors:	Student, Faculty
Stakeholders and Interests: Student, Faculty	
Preconditions:	
<ol style="list-style-type: none"> 1. The students and faculty have successfully logged on to the system. 	
The project details have been created and stored in the database.	
Post-conditions:	
<ol style="list-style-type: none"> 1. The project has successfully been created and details have been generated. 	
Main Success Scenario:	
<ol style="list-style-type: none"> 1. A team of students is created for a specific project. 2. A faculty member is assigned for the project to guide the students. 3. The project is divided into activities. 4. Each activity is assigned to one or more students. 	
Alternative Flow: None	
Special Requirements: None	
Technology and Data Variations List: None	

Use Case ID:	7
Use Case Title:	Create Project Schedule
Primary Actors:	Student, Faculty
Stakeholders and Interests: Student, Faculty	
Preconditions:	
<ol style="list-style-type: none"> 1. A project is created and assigned to a team of students and a faculty member. 	
The project schedule is created and agreed upon by involved students and faculty.	
Post-conditions:	
<ol style="list-style-type: none"> 1. The project has successfully been created and details have been generated. 	
Main Success Scenario:	
<ol style="list-style-type: none"> 1. The start and end date for the project is determined. 2. The overall schedule is divided into different slots and one slot is allocated for each activity of the project. 3. A checklist exists for each activity to verify whether it has been completed successfully or not. 	
Alternative Flow: None	
Special Requirements: None	
Technology and Data Variations List: None	

Use Case ID:	8
Use Case Title:	Manage Activities
Primary Actors:	Student
Stakeholders and Interests: Student, Faculty	
Preconditions:	
<ol style="list-style-type: none"> 1. A project is divided into activities and a schedule is available for each activity. 	
The activities of a project are managed primarily by the students with the guidance of faculty.	
Post-conditions:	
<ol style="list-style-type: none"> 1. The activities for a project are managed and tracked as per the schedule. 	
Main Success Scenario:	
<ol style="list-style-type: none"> 1. Each activity is assigned to one or more students. 2. An activity can depend on one more activities. 3. Students collaborate with each other to complete the assigned activity. 4. The faculty guides the students as needed. 5. The checklist is created for each activity that needs to be verified to determine the completion status. 	
Alternative Flow: None	
Special Requirements: None	
Technology and Data Variations List: None	

Use Case ID:	9
Use Case Title:	Allocate Resources
Primary Actors:	Administrator
Stakeholders and Interests: Student	
Preconditions:	
<ol style="list-style-type: none"> 1. All of the resources needed for a project are available. 	
The resources are allocated to the projects on first come first serve basis and resources are tracked against each project.	
Post-conditions:	
<ol style="list-style-type: none"> 1. The desired resources have been allocated and inventory is calculated accordingly. 	
Main Success Scenario:	
<ol style="list-style-type: none"> 1. The discussion takes place among students to decide which resources are needed for a specific project. 2. The students may seek advice from their faculty on the resources. 3. The students checkout the required resources from the repository. 4. The resource inventory is calculated accordingly for future allocations. 	
Alternative Flow:	
<ol style="list-style-type: none"> 1. If the requested resources are not available, an appropriate message is displayed to the user. 	
Special Requirements: None	
Technology and Data Variations List: None	

Use Case ID:	10
Use Case Title:	Create Checklist
Primary Actors:	Faculty
Stakeholders and Interests: Student, Faculty	
Preconditions:	
<ol style="list-style-type: none"> 1. All of the activities for a project are available and students are assigned for each activity. 	
The checklist is created for each activity to ensure the timely progress of the project.	
Post-conditions:	
<ol style="list-style-type: none"> 1. The checklist for each activity has been created. 	
Main Success Scenario:	
<ol style="list-style-type: none"> 1. A checklist is prepared for each activity by the faculty in coordination with the students involved. 2. Each checklist is divided into separate items that can individually be tracked. 3. The execution date is set for each item. 	
Alternative Flow: None	
Special Requirements: None	
Technology and Data Variations List: None	