MOBILE COMPUTING: Smart Applications using J2ME Technology
(SUCHANA SAMVAHAK)

Ankita Lathey*
ankita.lathey@gmail.com

Mr. Akash Tayal*
akashtayal786@gmail.com

Shweta*
shweta.igit@gmail.com

Lecturer, Indira Gandhi Institute of Technology
Guru Gobind Singh Indraprastha University, Delhi, India

ABSTRACT
This paper deals with building offline applications for handheld devices using Java 2 Micro Edition (J2ME). The technology used is capable of withstanding against Wireless Application Protocol (WAP), which has limitations in network infrastructure, Low wireless connection bandwidth, High Cost of wireless service and Data synchronization only when connectivity is available. The research applied to two databases, and the results for this method is robust against offline synchronization of the data using J2ME wireless Toolkit and Point base/Oracle lite as synching servers.

KEYWORDS:
Mobile Computing, WAP, Point Base Micro Unisync server.

1. INTRODUCTION:
Professionals are increasingly working away from their desks and require access to corporate data normally found only on desktop devices connected to enterprise networks. Wireless connections for mobile users offer the promise of remote access to enterprise data but persistent wireless connections are not always possible, practical or desirable. Developers require an end-to-end infrastructure with application services that enables the development, delivery and operation of secure, personalized applications on mobile or embedded devices. Persons, having LAN connectivity and those having intermittent connectivity both can access data and applications uninterruptible, without the unrealistic expectation of continuous wireless connectivity, if they're equipped with an "always-available" mobile solution.

Java provides the foundation for always-available mobile applications. For example, J2ME-enabled mobile device can operate in both wireless and disconnected modes. Unlike browser-based WAP or HTML applications, J2ME applications are persistent; a J2ME application can function on a cellular phone, laptop, or PDA without a wireless connection. When a wireless connection is established, however, the J2ME application can communicate with the corporate network, sending and receiving data changes, additions, and deletions. Java’s ability to work in all modes makes this technology invaluable for all types of environments and mobile applications. [1]

2. OFFLINE APPLICATION ARCHITECTURE
Experiments showed all the offline-enabled applications had the following design issues that needed to be addressed:
1. Isolating the data layer
2. Deciding which features to implement offline (connection strategy)
3. Deciding on the app's modality
4. Implementing data synchronization

2.1 ARCHITECTURE WITH A DATA LAYER
In general, isolating the data layer is a good first step.

![Data Layer Diagram](https://via.placeholder.com/150)

Figure 1: Data Layer
Through local data store all data storage and retrieval requests pass.
For example, if AJAX [2] application issues a request directly to the server to get all the accounts for a user, we might change this to instead ask an intermediate object for all the accounts for the user. This object could then decide whether to retrieve the data from the server, the local store, or some combination...
of both. Similarly, when the application wants to update the user's accounts, the app does so by calling the intermediate object. The intermediate object can then decide whether to write the data locally, whether to send the data to the server and it can schedule synchronization.[3]

The intermediate object act as a data switch layer that implements the same interface as the data layer as follows:

FIRST STEP: Make the data switch forward all calls to the data layer that interacts with the server. This step is useful to switch between the modes.

SECOND STEP: To create a new local data layer that uses a database instead of going to the web server for data. It's simpler if this data layer has the same interface as the existing data layer used to communicate with the server. If the interface is different then some translation needs to be done.

![Figure 2: Data Switch Layer](image)

SECOND STEP: To create a new local data layer that uses a database instead of going to the web server for data. It's simpler if this data layer has the same interface as the existing data layer used to communicate with the server. If the interface is different then some translation needs to be done.

![Figure 3: Local Data Layer](image)

Local store is faster than the remote connection.

2.2 DATA SYNCHRONIZATION

No matter which connection and modality strategy used, the data in the local database will get out of sync with the server data. For example, local data and server data get out of sync when: The user makes changes while offline, Data is shared and can be changed by external parties, Data comes from an external source.

Resolving these differences so that the two stores are the same is called "synchronization".

2.3 SYNCHRONIZATION STRATEGY

2.3.1 Manual Sync

It's manual because the user decides when to synchronize.

Manual sync requires that the amount of data is small enough to download in a reasonable amount of time and the user explicitly indicates when he or she is going offline, typically via a button in the user interface.

The problems with this method and with the offline mode it creates are that the Users don't always know the state of their network connections. Internet connections may die unexpectedly or be intermittent.

2.3.2 Background Sync

In "background sync", the application continuously synchronizes the data between the local data store and the server.

The benefits of background synching are that the Data is ready at all times, whenever the user chooses to go offline, or is accidentally disconnected and the performance is enhanced when using a slow Internet connection. But sync engine may slow the online experience if not using WorkerPool.

![Figure 4: background sync architecture](image)

3. ALGORITHM TO BUILD AN OFFLINE SYNCHING APPLICATIONS

3.1 ENVIRONMENT:

The development and testing the technology requires:

On MIDP client: JDK 1.5, CLDC 1.0, MIDP 1.0.3, J2ME wireless toolkit, RMS, Oracle9i lite, Mobile Development kit.

On server side: Jakarta Tomcat 5.5, MS SQL 5, ODBC: JDBC driver, PointBase micro5.6 (Unisync server), jdbc: poinbase: micro driver and Oracle9i lite.
Design Flow for making Offline Applications

3.2 SOLUTIONS:
During experimentation a database is created on the mobile phone using the RMS technique that builds sequential records and files. The same data will be residing at the remote database. The client device can synchronize at the data access-layer level by communicating with industry-standard interfaces. [9] Client’s mobile should support both real-time (e.g., browser-based) and offline access (e.g., client application and synchronization) to corporate back-end applications and content.

We have used two approaches in order to make the offline application. These two solutions are:
**SOLUTION 1:** synchronizing using Point Base and Unisync server [7]
**SOLUTION 2:** Building Offline Mobile Applications using Oracle lite.

3.3 EXPERIMENTAL RESULTS:
Following is the result interpreted, after comparing the two database approaches based upon given attributes:

<table>
<thead>
<tr>
<th>ATTRIBUTES</th>
<th>POINTBASE MICRO UNISYNCK SERVER</th>
<th>ORACLE 9i LITE MOBILE DATABASE</th>
</tr>
</thead>
<tbody>
<tr>
<td>VERSION USED</td>
<td>Point base micro 5.6 (Windows compatible)</td>
<td>Oracle Database 10g, Mobile Server (oracle 9i lite), Mobile Development kit</td>
</tr>
<tr>
<td>EXTERNAL SUPPORT</td>
<td>MS SQL/MS ACCESS databases, J2-ME wireless toolkit, Apache Web server, JDBC connectivity</td>
<td>Above three softwares (windows compatible)</td>
</tr>
<tr>
<td>LANGUAGE</td>
<td>J2ME and client MIDP applications</td>
<td>Mostly uses C/C++</td>
</tr>
<tr>
<td>PROVISION TO STORE DATA FOR SYNCRONIZATION</td>
<td>No such provision</td>
<td>Stores the data to be saved sync later</td>
</tr>
<tr>
<td>ARCHITECTURE FOLLOWED</td>
<td>HUB(server) and spokes (clients) architecture</td>
<td>offline architecture as described above</td>
</tr>
<tr>
<td>COMPLEXITY</td>
<td>Less</td>
<td>More</td>
</tr>
</tbody>
</table>

Table 1: Comparisons of two approaches

4. SOCIAL ASPECTS:
Maintenance of:
1. The land/corporal records in the remote area.
2. The databases in educational institutions constructed in the remote areas.
3. Databases in courts.
4. Stock Info Collation/Control via central database.
5. Credit Card Verification by speeding up the transactions.
6. Taxi/Truck Dispatch by uploading their information to the central dispatch office.[4]

5. CONCLUSION:
We described the technology of developing the offline application in J2ME enabled handheld devices. We have given two approaches having offline capabilities with the background as well as foreground synchronization. Thus using J2ME with MIDlets combines excellent online (WAP) and offline (J2ME)
capabilities that are useful for the wireless environment, which suffers from low bandwidth and network disconnection. The following table depicts a comparative study of the above mentioned technologies:

<table>
<thead>
<tr>
<th>FEATURES</th>
<th>WAP</th>
<th>J2ME</th>
</tr>
</thead>
<tbody>
<tr>
<td>EXTERNAL ENVIRONMENT SUPPORT</td>
<td>Present</td>
<td>Needs external support</td>
</tr>
<tr>
<td>LANGUAGES</td>
<td>WML &amp; WML scripts</td>
<td>JAVA and MIDP</td>
</tr>
<tr>
<td>MODE OF DISPLAY</td>
<td>Single screen</td>
<td>Multiple screens</td>
</tr>
<tr>
<td>NAVIGATION</td>
<td>Tag line based</td>
<td>Command based</td>
</tr>
<tr>
<td>INTERACTION</td>
<td>Lower (no interaction model)</td>
<td>Standard are defined for interaction commands</td>
</tr>
</tbody>
</table>

Table 2: Comparative study of the technologies

6. FUTURE IMPLEMENTATION:
Extension to this work will include the development of more complex database and bringing forth the universal technology that support offline mode with synchronization for real time devices.

7. REFERENCES:
[1] Java developer’s journal by Jeff Capone, PhD, CTO of Aligo.
[3] Yun Yang “Supporting Online Web-based Teamwork in Offline Mobile Mode too” School of Information Technology, Swinburne University of technology, Australia, 2000 IEEE
[4] Internet report “MOBILE COMPUTING: PAST, PRESENT AND FUTURE” By Vasilis Koudounas and Omar Iqbal
[5] David Hemphill “Give Your Mobile Workforce Offline Muscle with the MIDP Record Management System” December 8, 2004
[7] “High availability mobile applications Mobile databases and J2ME tools” by Michael Juntao Yuan
[8] Translation Server for a Mobile Phone, Clare Scully, May 2005