CHAPTER - 2

INTRODUCTION TO MOBILE CLOUD COMPUTING

2.1 Mobile Computing
2.2 Data Communication
2.3 Web Services
2.4 Cloud Computing
2.5 Mobile Cloud Computing Applications
2.6 Consuming CWS from Mobile Users
2.7 Introduction to Mobile Cloud Computing

References
2.1 Mobile Computing

Mobile Computing technology is that allows transmission of data, via a computer device, without consuming to be connected to a fixed physical link.

Mobile based audio visual communication is broadly recognized throughout the world and has a very quick increase in the number of users to the various cellular network companies over the preceding few years. An allowance of this technology is the aptitude to send and receive data across these cellular networks. This is the norm of mobile computing.

Mobile based data communication has develop a very important and rapidly growing technology as it allows users to transmit their data from remote locations to additional remote centers or fixed locations. This demonstrates to be the solution to the key problem of IT persons or business peoples who are on the move or travel - mobility.

2.1.1 Existing Cellular Network Architecture

Mobile communication took off with the overview of cellular based technology which allowed the efficient use of frequencies enabling the connection of a numerous clients. During the 1980's analogue technology was used. Amongst the most of are well-known systems were the AMPS (Advanced Mobile Phone Service) and the NMT-900 and 450 (Nordic Mobile Tele phone). In the mid 1990's the digital cellular technology was announced with G S M (Global System Mobile) being the most usually accepted system around the world.
A cellular network consists of mobile things linked up together to switching between gear, which inter connect the differ fragments of the network and permit accessing to the Public Switched Telephone Network (PSTN). The technology is unseen from common-view; it is combined with a number of transceivers know as Base Stations (BS). Each BS is located at a tactically selected location and covers a specified cell or area - henceforth the name cellular transportations. A numeral of end-to-end cells grouped together form a specific area and the conforming BS interconnect through a MSC (Mobile Switching Centre). The MSC is the soul of a cellular based radio system. It is accountable for switching and or routing, calls from the maker to the destination. It can be assumed of managing the cell, being answerable for setup, routing control device and termination of the call, for accumulating information for charging and accounting. The MSC may be linked to other MSC on the PSTN or to the same network.

Figure 2.1 Mobile Switching Centers

2.2 Data Communication
Data Communications is the conversation of data consuming existing communication networks. The term data covers a wide range of apps including inter-connection between Wide-Area-Networks (W A N), File Transfer (F T), electronic mail (e-mail), access to the internet and facsimile (FAX), Mobile-to-Mobile conversation (M-2-M).

![Overview of Mobile Communication](image)

Figure 2.2 Overview of Mobile Communication

Data Communications had been achieved expending a diversity of networks such as leased-lines, P S T N (Public Switched Telephone Network), A T M (Asynchronous Transfer Mode) / F R (Frame Relay) and I S D N (Integrated Services Data Network). These networks are partially or entirely analogue or digital using technologies such as packet - switching, circuit - switching, etc.
Mobility instigated in data communications consumes an important difference as compared to voice communications. Mobile phones allow the user to move from one remote location to other remote locations and talk / surfing internet at the roughly same time, the loss of the connection for 400ms during the hand over is undetectable by the user. When it arises to data, 400 m sec is not only measurable but origins enormous misrepresentation to the message. Hence data can be transmitted from an M S (mobile station) under the supposition that it leftovers steady or within the matching cell.

2.2.1 CDPD TECHNOLOGY

Now-a-days, the mobile data infrastructures market is pleasant lead by a technology called C D P D (Cellular Digital Packet Data) [48].

There are other alternatives also available to this technology known as Circuit Switched Cellular, Wireless Data Networks and Specialized Mobile Radio. As can be seen from the table below the CDPD technology is much more advantageous than the others technologies.

<table>
<thead>
<tr>
<th></th>
<th>Cellular Digital Packet Data (C.D.P.D.)</th>
<th>Circuit Switched Cellular</th>
<th>Specialized Mobile Radio (Extended)</th>
<th>Proprietary Wireless Data or Networks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Speed</td>
<td>Best</td>
<td>Best</td>
<td>Good</td>
<td>good</td>
</tr>
<tr>
<td>Security</td>
<td>Best</td>
<td>better</td>
<td>Good</td>
<td>better</td>
</tr>
<tr>
<td>Ubiquity</td>
<td>Best</td>
<td>best</td>
<td>Good</td>
<td>better</td>
</tr>
<tr>
<td>Cost of Service</td>
<td>Best</td>
<td>better</td>
<td>Better</td>
<td>good</td>
</tr>
</tbody>
</table>
Chapter 2 Introduction to Mobile Cloud Computing

<table>
<thead>
<tr>
<th>Cost of Distribution</th>
<th>Best</th>
<th>best</th>
<th>Better</th>
<th>good</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mobility</td>
<td>Best</td>
<td>good</td>
<td>Better</td>
<td>good</td>
</tr>
<tr>
<td>Interoperability</td>
<td>Best</td>
<td>good</td>
<td>Good</td>
<td>better</td>
</tr>
</tbody>
</table>

Table 2.1 List out and comparison of mobile data communication technology

C D P D networks allow mobile or fixed users to connect to the network crossways a packet switched system and a fixed link respectively. Fixed physical link is available for fixed users to the C D P D network. In the instance of a mobile user, if user can, the C D P D network facilities are non-existing; connecting to existing circuit switched networks and transmits data thru these networks. This is namely known as Circuit Switched CDPD (CS-CDPD).

2.3 Web Services (WS)

A Web Services is a technology interconnected to the idea of Service based Computing [49]. A Web Service [50] is

“A web service is a collection of standards and open protocols used for exchanging data between mobile apps, applications or systems. Software apps written in numerous programming languages and running on many platforms can use web services to exchange data over computer networks like the intranet or the Internet in a manner similar to inter-process communication on a sole computer. This interoperability (e.g., amongst Java, PHP, Python, or Windows and Linux applications) is owing to the consumption of open standards.”

---

Performance Analysis and Design for Mobile Cloud Computing on Various Platforms 38
A comprehensive web service (WS) is, hence, any service that:

- Is available over the Internet or Intranet networks
- Uses a standardized XML based messaging system
- Is not tied to specific operating system (OS) or programming language (PL)
- Is self-describing via a common XML syntax / grammar
- Is discoverable thru a simple find appliance

### 2.3.1 Why Web Services

Here are the key benefits of using Web Services (WSs) as are below:

#### 2.3.1.1 Revealing the existing function on to the network

A WS is a component of managed code that can be remotely invoked using HTTP, that is, it can be started using HTTP requests. Thus, Web Services lets you to representation the functionality of your existing code over the internet network. Once it is showing on the network, other apps can use the functionality of your program code.

#### 2.3.1.2 Connecting Different Apps

Web Services permits different apps to exchange to each other and share data info and services between themselves. Other apps can also usage the services of the web services. For i.e. VB or .NET app can exchange to java web services and vice versa. Therefore, Web services are used to make the request platform and technology autonomous.
2.3.1.3 Uniform Protocol

Web Services usages standardized business standard protocol for the communication. All the four layers (Service Transport, Service Description, XML Messaging and Service Discovery layers) use the well-defined protocol in the WS procedure stack. This regulation of procedure heap gives the industry many benefits like varied range of choices, decrease in the cost due to opposition and growth in the quality measure.

2.3.1.4 Small cost of message

WSs usages SOAP over HTTP protocol for the message communication; accordingly, you can use your present small cost internet for executing Web Services. This kind of solution is much less expensive as compared to exclusive solutions like E D I / B-2-B. Alongside SOAP over HTTP WSs can also be applied on other consistent transference mechanisms like File Transfer Protocol (FTP), One standard protocol to other standard protocol like XML to JSON, etc.

2.3.2 Web Services Protocol Standards

There are two WS protocols standards [54], SOAP WS and RESTful WS. Figure 2.3 shows SOAP WS in a service-oriented architecture.
Figure 2.3 Web Service oriented Architecture

SOAP based Web Services has well-adopted protocols. Following is a characteristic state of intense SOAP WS. Here, note that service discovery (step-1 and step-2) is possible.

1. Service publishers broadcast services to the service registry subsequent the UDDI standard [52] [53].
2. Users also follow UDDI to determine the service they necessity.
3. Clients cause code for a specific SOAP WS from the WSDL [51].
4. Users conversation SOAP messages with the service using the HTTP protocol. Figure 2.4(a), 2.4(b) and 2.4(c) demonstrations an example of HTTP POST request and response contains SOAP message using .NET platform.

Following is, WS example which workings as a service publishers and exposes two methods (AddTwoValue & SayGreeting) as Web Services to be used by apps. This is a regular pattern for a Web Service. DOT
NET Web Services use the .asmx extension. Here, note that a method exposed as a WS had the [WebMethod] attribute which described as Web Service. Save this file as RWService1.asmx in the IIS virtual directory (for e.g., c:\r_ws_app).

RWSERVICE1.ASMX

```csharp
using System.Web.Services;

namespace RWebApp1
{
    [WebService(Namespace = "http://tempuri.org")]
    [WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1_1)]
    [System.ComponentModel.ToolboxItem(false)]
    {
        [WebMethod]
        public int AddTwoValue(int a, int b)
        {
            return a + b;
        }
        [WebMethod]
        public string SayGreeting()
        {
            return "Welcome Rushi";
        }
    }
}
```

Figure 2.4(a) Request a SOAP Web Services
To test a Web Service, it must be published or available as live. A WS can be available either on an Internet or an Intranet. Next phase will publish this Web Service on IIS running on a local machine. Follow these steps to configuring the IIS.

- Open or Click Start->Settings->Control Panel->Administrative tools->Internet Information Services Manager.
- Next, expand and right click on [Default Web Site]; select New->Virtual Directory.
- Then, The Virtual Directory Creation Wizard open, then Click Next.
- The "Virtual Directory Alias" screen will open. Here, type the virtual directory name—for e.g., r_ws_app—and click next.
- The "Web Site Content Directory" screen will open. Now, you have to enter the directory path name for the virtual directory—for e.g., c:\r_ws_app—and clicks next.
- Then "Access Permission" screen will open. Modify the settings as per your application or WS requirements. If you do not want to change then keep the default settings for this WS. Click the next button. Now, it completes the IIS configuration. Then after click finish to complete the configuration settings.

To test that IIS has been configured properly or not, copy an HTML file (for e.g., r.html) in the virtual directory (C: \r_ws_app) created above. Now, open Internet Explorer or any other browser application and type http://localhost:90/r_ws_app/r.html. It should open the r.html file. If it doesn’t work properly, try replacing with “localhost:port_no” with the IP address of your device / machine. If it still doesn't work appropriately, check whether IIS service is running accurately; if not, then you may need to re-configure IIS and Virtual Directory steps as above mention.

To test our Web Service, copy RWService1.asmx in the IIS virtual directory created above (C: \r_ws_app). Open the Web Service in
Internet Explorer or in any browser supported by .Net Applications (http://localhost:1142/RWSERVICE1.asmx). It should open your WS page. The page should have links to two methods which is exposed as WS by our application as described earlier. If you don’t want to enable directory browsing, confirm that a default document is configured and that the file exists at your defined path.

Testing the Web Service
Here, Researcher has just display the sample, writing Web Services with easy steps in the .NET Framework / SDK. Writing Web Service clients is also easy in the .NET framework; though, it is a little more involved. As said prior, Researcher will write two types of service clients, one Web based and another is for Windows or Console application based clients. Now, write my first Web Service clients.

Web-Based Service Clients

Write a Web-based client as given below. Call it RWForm1.aspx. Here, note that it is an ASP.NET based application. Next save this in the virtual directory of the Web Service (C:\r_ws_app\RWebApp1\RWebApp1\RWForm1.aspx).

This application has two text fields that are used to input numbers from the users to be added. It has one button, for e.g., “Run Web Service”, which, as soon as clicked by users, gets the AddTwoValue numbers and SayGreeting Web Services.

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="RWForm1.aspx.cs" Inherits="RWebApp1.RWForm1" %>
<html xmlns="http://www.w3.org/1999/xhtml">
```
Chapter 2 Introduction to Mobile Cloud Computing

Performance Analysis and Design for Mobile Cloud Computing on Various Platforms

```csharp
using System;

namespace RWebApp1
{
    public partial class RWForm1 : System.Web.UI.Page
    {
        // Code for RWForm1.aspx.cs
    }
}
```
```csharp
{  
  protected void btnRun_Click(object sender, EventArgs e)  
  {  
    RWService1 rws1 = new RWService1();  
    lblGreeting.Text = rws1.SayGreeting();  
    int a = Int16.Parse(txtFNumber.Text);  
    int b = Int16.Parse(txtSNumber.Text);  
    lblAddTwoValue.Text = rws1.AddTwoValue(a,b).ToString();  
  }  
}  
}
```

Figure 2.4(b) Response to a SOAP Web Services

After the clients are created, we require generating a proxy for the Web Service to be consumed. This task work is done automatically by VS.NET platform for us when referencing a WS that has been added. Here, are the steps to be followed as following manner:

- Construct a proxy for the Web Service to be consumed. The proxy is produced using the “wsdl” utility supplied with the .NET software development kit. This utility extracts info from the Web Service and generates a proxy. Therefore, the proxy created is binding only for a precise web services. If you requisite to consume other WS, you need to create a separate proxy for that service as well. Visual Studio.NET produces a proxy automatically for you when the reference for the WS is added. It will create RWService1.cs in the present dir. Then after, we pre-requisite to build and compiled it to create RWService1.dll (Alternate) for the Web Service.
Chapter 2 Introduction to Mobile Cloud Computing

Performance Analysis and Design for Mobile Cloud Computing on Various Platforms

C:> WSDL
http://localhost:90/r_ws_app/RWService1.asmx?WSDL
C:> csc /t:library RWService1.cs

- Place the compiled proxy in the application-path\bin directory of the virtual directory of the Web Service (C:\r_ws_app\RWebApp1\RWebApp1\bin). IIS aspects for the proxy in this directory.
- Create the service clients, which researcher have already done. Here, consider that researcher have instantiated an object of the Web Service proxy in the clients. These proxy takings care of interrelating with the service.
- Type the URL of the clients in browser, such as IE to test it (for example, http://localhost:1142/RWForm1.aspx).

Windows / Console Application-Based Web Service Clients

Writing Windows / Console application-based Web Service clients is the same as writing any other Windows or Console application as well. The only external work to be done is to create the proxy (which Researcher have already done) and reference this proxy at the time when compiling the application. Succeeding is this Windows application that uses the WS (Web Service). This application creates a Web Service object (known as, proxy/alternate) and calls the SayGreeting and AddTwoValue methods on it. Researcher has added reference follow this steps:
  - Create a console or windows based application project.
  - Right Click on project name and Add Service Reference. A new window will be open.
  - Next, click the "Advanced" button. A new window will be open.
• Over again click on the "Add Web Reference" button. Again a new window will be open.
• Copy the URL of your running web service application and paste to it at URL (here, http://localhost:1142/RWService1.asmx) and click the "Go" button.
• Set the web service reference name. The default name is "localhost", here you may also change the name according to your service name. Now click the "Add Reference" button.
• Now the web service has been added in our console application.
• Write the following code.

**RConsoleWS.cs**

```csharp
using System;
namespace RConsoleApp1
{
    class RConsoleWS
    {
        static void Main(string[] args)
        {
            localhost.RWService1 rws1 = new localhost.RWService1();
            Console.WriteLine("Hello Rushi, \nCalling Web Service Methods");
            Console.WriteLine("---------------------");
            Console.WriteLine("\n Calling SayGreeting Method");
            Console.WriteLine("  " + rws1.SayGreeting());
            Console.WriteLine("\n\n Calling AddTwoValue Method ");
            Console.Write(" Enter First Number:");
        }
    }
}
```
```csharp
int a = Convert.ToInt32(Console.ReadLine());
Console.Write(" Enter Second Number: ");
int b = Convert.ToInt32(Console.ReadLine());
Console.WriteLine("\n Addition of two value is:" + rws1.AddTwoValue(a, b).ToString());
Console.ReadLine();
}
}
```

Figure 2.4(c) Response to a Windows based SOAP Web Services

Compile it using `c:\>csc RConsoleApp1`. It will create `RConsoleApp1.exe` executable file. Run it to test the application as well as Web Service also.

An alternative to SOAP-based Web Services are REST Web Services. Restful WS were first presented by Fielding [55] in his doctoral thesis in the year 2002. They survey a resource-oriented computing model. Restful WS is introduced as resources which are recognized by a Uniform Resource Identifier (URI). Users interconnect with Restful WS over the HTTP protocol, but the message part or body can follow any kind of formats, for e.g., XML to CSV and XML to JSON, as long as the users and the service providers approve upon it. Restful WS also takes benefit of the semantics of the HTTP protocol. For e.g., HTTP-GET request is for obtaining a resource and HTTP-POST request is for generating a resource. HTTP header, URL queries, & request body can all be used as a service inputs.

### 2.4 Cloud Computing
Cloud computing is the cutting-edge addition to the numerous of distributed computing models. Inappropriately, there is no clear classification of the term “Cloud Computing” and yet its origin is also not clear so far, for e.g. some link it to the standard representation of the Internet, others to the 2006 Eric Schmidt presentation or the 2001 New-York Times article referring to Dot.Net. Vaquero [56] refers to cloud computing as a model, which moves the location of computing systems infrastructure to the network in direction to decrease the expenses related with the administration and supervision of hardware & software resources. In this thesis, Cloud Computing paradigm is divided into three parts, Cloud Platforms (CP), Cloud Models (CM) and Cloud Services (CS).

Cloud Platforms (CP) regularly mention to apps hosts that offer storage capacity, computational power and Web-access. Two distinguished Cloud Platforms are Google App Engine (GAE) [4] and Heroku platform (Heroku). Heroku is based on platform as service concept, where all Heroku instance is a Web Dynos (WD). Users can choose different different Server languages, such as PHP, Ruby-on-Rails, Node.js, and select appropriate Dyno model to run their CWSs app module. Users are indicted amount of fee monthly based on specific quota limitations for these Heroku web-dynos. Table 2.2 lists the rental fee.

<table>
<thead>
<tr>
<th>Types of Dyno</th>
<th>Cost (Per Dyno - Monthly) (INR)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Free</td>
<td>₹0</td>
</tr>
<tr>
<td>Hobby</td>
<td>₹456.05</td>
</tr>
<tr>
<td>Standard-1x</td>
<td>₹1628.75</td>
</tr>
<tr>
<td>Standard-2x</td>
<td>₹3257.5</td>
</tr>
<tr>
<td>Performance</td>
<td>₹32575</td>
</tr>
</tbody>
</table>

Table 2.2 lists the approximate fee for Dyno
GAE is mainly a Web application based platform. Users can upload their Web API or Apps to run on GAE. Presently, it supports some programming languages, Python, Java, PHP and Go. GAE is free as long as the requests and or apps do not go beyond the “free quota”. Table 2.3 shows some of the limits of the “free quota” on quota and or daily basis and Table 2.4 list out some of the free available quota for search query.

<table>
<thead>
<tr>
<th>Resources</th>
<th>Free Quota (Default)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Daily Limit</td>
</tr>
<tr>
<td><strong>Outgoing Bandwidth</strong></td>
<td></td>
</tr>
<tr>
<td>(include HTTP &amp; HTTPS)</td>
<td>1GB</td>
</tr>
<tr>
<td><strong>Incoming Bandwidth</strong></td>
<td></td>
</tr>
<tr>
<td>(include HTTP &amp; HTTPS)</td>
<td>1GB / 14,400 GB</td>
</tr>
</tbody>
</table>

Table 2.3 lists some of the GAE “free-quota”

<table>
<thead>
<tr>
<th>Resource or API Call</th>
<th>Free Quota</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Total Storage (Indexes &amp; Documents)</strong></td>
<td>0.25 GB</td>
</tr>
<tr>
<td><strong>Simple Queries</strong></td>
<td>1000 queries per day</td>
</tr>
<tr>
<td><strong>Complex Queries</strong></td>
<td>100 queries per day</td>
</tr>
<tr>
<td><strong>Adding documents to Indexes</strong></td>
<td>0.01 GB per day</td>
</tr>
<tr>
<td><strong>Other API Calls (billed as operations)</strong></td>
<td>1000 operations per day</td>
</tr>
</tbody>
</table>

Table 2.4 lists specific of the GAE “free quota” for search query
Cloud Models refers to service models or layers, in cloud computing includes in mostly three different models, likely Infrastructure-as-a-Service (IaaS), Platform-as-a-Service (PaaS), and Software-as-a-Service (SaaS). If a cloud user accesses models on the infrastructure layer, for e.g., she/he can run her/his own apps on the resources of a cloud infrastructure and persist responsible for the maintenance, support, security and backup of these apps herself/himself. If she/he accesses a service on the application layer, these jobs are usually taken care of by the cloud service provider.

Figure 2.5: list the types of Cloud Deployment Models

Cloud Services mentions to software functions wide-open as W S on the Internet, also called as Web API or Web App. For e.g., services that provide information about the nearby hospital or famous place, city based on geo-coordinates. According to “Programmable web” [5], there are above more than 1808 available Web APIs related to e-commerce, database, accessibility, shopping, automation, data-storage, blogging, entertainment, etc. Several of them provide both SOAP and Restful W
S and are free under some limits, like as bandwidth usage, call manager, traffic limit. Service mash up is a widespread term in Web which means essential a customized or merges service using other services or API.

In this thesis, researcher has propositions a Mobile Cloud Computing (M C C) architecture which uses Cloud-based middle ware to sustenance mobile users / clients consuming Web Services (WS - Cloud Services). The design enhances the inter action among mobile users and Cloud Web Services CWS and provides a personal / private service mash up platform for mobile users. The architecture also provide peer-to-peer layer security on cloud computing.

2.5 Mobile Cloud Computing Applications

In numerous field of work the ability to keep on the move is dynamic in order to utilize time efficiently. In the small time they have been around, Mobile cloud computing has become a requisite part of business, education, and showbiz. The importance of Mobile Cloud Computers has been highlighted in different fields of which a few are described as below:

2.5.1 Emergency Services

Capacity to receive information on the move is vigorous where the emergency services are involved. Information regarding the location, address, name and other details of an occurrence can be dispatched as soon as possible.

2.5.2 Apps for Builders / Estate Agents
Estate agents or Builders can work either at home-based or out in the field. With the use of mobile cloud computers apps they can be supplementary productive. They can find current real-estate information by retrieving multiple listing Web Services (WS), which they can do from office, resident or car when available with clients. They can provide clients with immediate feed-back as regards specific homes or localities, since apps can be submitted on the advert. Consequently, mobile cloud computers permit them to give more time to the clients.

2.5.3 Credit / Debit Card Verification

At P O S (Point of Sale) terminuses in Shopping malls and Supermarkets, when clients’ usage credit cards for transactions, the intercommunication mandatory between the bank central computer and the P O S terminal, in order to result verification of the card usage, can take place fast and securely over cellular networks using a mobile cloud computer element.

2.5.4 Mobile E-mail

With use of a mobile client to send and read e-mails is a very useful plus for any persons or business individual, as it lets him/her to keep-in-touch with any contemporaries as well as any vital progresses that may touch their exertion, without accessing an office or home network.

2.5.5 At Hospitals

Now-a-days there are numerous devices and apps are available in a hospitals or at home usage, with the help of that devices patient can check their health as well as with the use of mobile cloud computers
medical apps they can even save or track their health records time-to-time, inter-connects at the hospital, organize diet schedule, any many more things they do with use of mobile cloud computers apps.

### 2.5.6 Stock Information Services

Trendy environments where access to stock is very much limited i.e.: at office hours, some place where other media is not present. The usage of transportable electronic databases read via a mobile cloud computer would be best.

### 2.5.7 At Companies

Representative or managers can use mobile devices to give presentations to key clients. They can access the up-to-the-minute marketplace share information. In a small break, they can review the presentation to take benefit of this info. They can link with the office about conceivable new offers and manage meetings for discussing responds to the new-fangled proposals. Hence, mobile cloud computers can force modest gains.

### 2.6 Consuming CWS from Mobile Users

Consuming cloud web services (CWS) from a mobile client (see figure 2.6) is diverse compared to the standard web services (WS) states, due to the following aspects.

- Mobile devices have limited resources (e.g. CPU power, screen size).
- The communication between client-users and service is established through wireless or cell network.
- Existing web services (WS) in the Cloud doesn’t support mobile clients.
There are numerous challenges in the process of consuming Cloud Web Services from mobile devices or clients. The following three are the focus of this thesis.

**Challenge 1.** Loss of connectivity: The interface between clients and services requires a steady connection. Though, due to the agility of the clients and the wire-less network setup, mobile clients can be temporarily removed from the preceding connected network and later may enter to another network. In such incidences, either service responses or service requests may fail to be distributed to their end point.

**Challenge 2.** Bandwidth/Latency: Cellular / Mobile networks have inadequate bandwidth and are often billed based on the amount of data transferred cell networks to mobile clients. Nevertheless, even a simple SOAP message frequently holds a large amount of XML data, which consumes a lot of bandwidth and the broadcast can cause foremost network latency. In addition, the SOAP message encompasses usually XML tags that aren’t all essential for the mobile clients.
Challenge 3. Inadequate resources: Mobile users are likely as “Thin Clients” [6] thru limited process power as user demand always more. The boundaries are fundamental to agility in addition not just the inadequacies of present-day technology [7]. For e.g., a service mash up implicate parsing and linking different WS results necessitates a lot of computations. The tasks are mimimizing the data and facts processing on mobile clients and outspreading processing power elsewhere mobile clients. Moreover, various mobile platforms do not include necessary libraries for communicate with SOAP WS.

2.7 Introduction to Mobile Cloud Computing

To overcome these challenges, researcher has propose a Mobile Cloud Computing (MCC) architecture (see figure 2.7) which connects mobile users to the Cloud Computing. The MCC architecture includes a mobile client and a middleware design.

There are two approaches to implement the mobile user: native applications and embedded browser (web-work) applications. Native applications are constructed with detailed programming languages supported by the mobile platforms. However, embedded browser (Web work) applications can run HTML and JavaScript (JS) in the embedded browser and use interfaces exposed by native application.
The middleware [57] turns as a proxy that is hosted on the Cloud platforms which offer mobile clients access to Cloud services. The middleware advances relations between the mobile clients and Cloud Services, for e.g., caching, optimization and adaptation. The middleware correspondingly make available comprehensive functions to the mobile clients, such as service mash up. In general way, the middleware improves the reliability, compatibility and functionality of the interface between mobile clients and Cloud based Services. In order to overawe the challenges listed in the previous section, the Mobile Cloud Computing (MCC) architecture provides the following features.

(C1) Loss of connectivity
- Mobile client and middleware caching – Copies of service results are stored on both the middleware and on the mobile clients. When the mobile clients / users are not able to connect to the middleware service, the client-side cache portion is used. When the middleware to
CWS connection is not available due to connectivity, the middleware returns its cached data to the mobile clients as they connected.

- Middleware immediate push – When the middleware receives any update of service result, it proximately sends the update to mobile clients that are connected to the middleware. When the mobile clients perceive an available network connection, they automatically launch a connection to the middleware services.

(C2) Bandwidth/Latency

- Protocol revolution – Protocol revolution decreases the bandwidth as well as latency of the client to the service interface. The middleware transmutes SOAP CWS to RESTful CWS. SOAP is a long-winded protocol which also involves XML parsing, while RESTful CWS can use light-weight format like as JSON for the communicate messages. Transmitting SOAP CWS to light weight protocols, like wise RESTful CWS, shrinks size of the messages as well as the processing time over the communication between mobile clients and the service.

- Optimization result – Optimized result diminishes the size of the service outcomes, therefore reduces the bandwidth used to interrelate with CWS. The middleware transforms the format of service results from XML to JSON or and eliminates needless data from the original service result. Fewer data transferring also diminishes network latency.

(C3) Inadequate resources

- Cloud Computing – Linking Cloud Computing (CC) to mobile clients outspreads the resources of mobile clients in a cost-efficient method. Cloud Computing Services encompasses the functionalities of mobile clients, while Cloud Platforms offer very much computational power to the mobile clients. The middleware is premeditated to be hosted on Cloud platforms, like Google App Engine (GAE) and Amazon Elastic Cloud Computing (EC2). Scalability is the uppermost concern of the
middleware. Cloud platforms deliver spontaneous scaling for the middleware.

- Personal Mash up Platform – Service application mash up permit mobile client to combine different services among different available category. Though, application mash up necessitates interface with processing power and CWS. Since of the resources constraint (processing power, energy, software libraries) of mobile clients, it is ineffective to do service mash up on the mobile clients. The middleware runs a Personal Mash up Platform which ensures service mash up for the mobile clients. The platform has standard interfaces for significant and consuming CWS. The services are kept on the middleware and can be allied to form a work-flow (a mash up service application) which offers prospect to caching in-between service results.
REFERENCES

[57]. D.E. Bakken, M. Api, Middle-Ware API, 2001.