SRI VENKATESHWARAA COLLEGE OF ENGINEERING & TECHNOLOGY
(Approved by AICTE, New Delhi & Affiliated to Pondicherry University, Puducherry.)
13-A, Villupuram – Pondy Main road, Ariyur, Puducherry – 605 102.
Phone: 0413-2644426, Fax: 2644424 / Website: www.svcetpondy.com

Department of Computer Science and Engineering

Subject Name: **MOBILE COMPUTING** Subject Code: **CS E84**

UNIT V

Mobile Computing Models: Client Server model – Client/Proxy/Server Model – Disconnected Operation Model – Mobile Agent Model – Thin Client Model – Tools: Java, Brew, Windows CE, WAP, Sybian, and EPOC.

2 Marks

1. What are the Different types of mobile computing models? (April'13)

The following models of computing in the mobile environment are currently being researched and investigated:

Client/Server

Client/Proxy/Server

Disconnected Operation

Mobile Agents

The Thin Client model

2. Write about client-server model.

In this model, neither the client nor the servers a r e aware of the client (or server) mobility. The conventional client/server model is used without any modifications made in the application or the transport layer. The wireless media is transparent since it is handled in the data link layer. The mobility, on the other hand, is made transparent by handling the variable client/server location through location-based routing in the network layer. Mobile IP is an example of a network protocol that hides the C/S mobility.

3. What is the advantage in client server model? (UQ April'13)

The advantage of this model is its portability. The client or the server need not be changed in any way. Simply, the client (server) is ported to the mobile host, with a fixed address specified for the server (on the fixed network or on a mobile host).

4. What are the disadvantages of client server model? [April 2013][April 2014]

The mobile client may suffer from a slow and unpredictable response time, especially when large server replies such as query results are transmitted without any considerations to the limited wireless bandwidth.

The server caching strategy may not work properly in this model because the majority of the fixed network caching algorithms use call backs to invalidate the client cache. Most invalidation algorithms rely on the continuous availability of the client. Since the client could be disconnected or temporarily inaccessible (during hand-offs for example), the cache invalidation process could fail.

5. What is meant by client/proxy/server model?

To overcome the shortcomings of the conventional client/server model, the client/proxy/server (C/P/S) model introduces a mobility-aware middle layer to mediate the interactions between the client and the server. The basic idea behind this model was introduced in the Mowgli architecture [72], which is to split the communication path between the client and the server into two parts by using a store-and-forward interceptor.

6. What are the advantages of proxy model?

The main advantage of this model is that the proxy allows the client and the server to be designed without any built-in mobility assumptions. The proxy assumes that the client is mobile and the server is in the fixed network. The result from the server is sent back to the proxy. The proxy filters the results according to the limitations of the wireless media and/or the client's mobile unit. The proxy may also store the filtered results until the client is connected.

7. What is disconnected operation model.

Mobile clients may face wide variations in network conditions and local resource availability when accessing remote data. This is true in the C/S and the C/P/S models. Disconnected operations are a variation of the C/S model where, instead of working under the extreme case of weak-connectivity, the mobile client effectively switches to use a network of zero bandwidth and infinite latency.

8. Explain briefly about mobile agent's model. [April 2014]

Mobile agents are mobile scripts with associated execution state information. A mobile agent could either be relocated along with the user, or it could be relocated during the execution of the agent. The relocation of the agent involves saving the state before initiating relocation and later restarting the mobile agent at the new location.

9. Write about thin client model. [NOV 2013]

The thin client computing model attempts to offload most application logic and functionality from mobile clients to stationary servers. In this model, applications in stationary servers are usually mobile-aware and optimized for mobile client devices. This model is especially suitable for dumb terminal or small PDA applications.

10. Write short notes about the architecture of thin client model.

The thin client architecture from CITRIX Corporation allows a variety of re- mote computers, regardless of their platform, to connect to a Windows NT terminal server to remotely access a powerful desktop and its applications. A server called Meta Frame runs under Windows NT in the desktop machine

11. What is a mobile agent?

The mobile agent is an emerging new model that provides an alternative to the C/P/S model. A mobile agent i s an active entity that is knowledgeable of both the limitations of the mobile environment and the mobile user.

12. Classify the mobile agents.

Agents can be classified as static, mobile scripts, or mobile objects. Static agents are those which execute just on a single site, either as a client or as a server. A static agent could be carrying out some activity like mail filtering. Mobile scripts are those that are downloaded from a server and executed on a client. Java applets, perl, or python scripts can be classified as mobile scripts.

13. Expand and explain MDCS.

In the proxy architecture, the mobile host is provided with a specialized transport service, the Mowgli Data Channel Service (MDCS). It provides prioritized data channels with flow control between the mobile host and the base station. Existing TCP/IP protocols are used between the base station and a fixed host so that the protocol software in the fixed network remains unmodified.

14. When does the agent act as a local client?

The mobile agent is an execution context initially loaded with the queries or data access requests. Once the agent moves to the data source (server), it acts as a local client to the server.

15. What is Wireless Application Protocol (WAP)? Wireless-

that does not require wires, enabling radio transmission.

Application- software designed to complete a particular task.

Protocol- means a set of rules.

These three simply means a set of rules governing the transmission and reception of signals through computer applications. Computer may include mobiles, tablets also. WAP has spearheaded today's Internet communication and advanced telephony services. WAP allows allow the devices to view pages from Internet. But, the pages displayed have only a plain text and the images are black and white. WAP is similar to HTML, except that it has been optimized for-

- ✓ Low-display capability.
- ✓ Low-memory.
- ✓ Low band width devices like mobile phones etc.

16. Why do we need WAP?

Long before the first WAP devices came, Internet was limited to your computer only. Now with WAP, you can communicate to your friends using Internet through your mobile phone too. Thus globally expanding massive communication and sharing of data.

17. What is a Micro WAP Browser?

Similar to your own internet browser, there is a browser for WAP too. This browser is called Micro WAP Browser that is used for visiting web-sites through a WAP device. What's special about it is, it makes minimal demands on hardware, memory and CPU and it displays the information in WML (it a restricted mark-up language).

18. What are the different layers of WAP architecture?

The whole WAP architecture has been divided into 5 main layers, namely:-

1. Application Layer
2. Session Layer
3. Transaction Layer
4. Security Layer
5. Transport Layer

19. What are the various protocols in a WAP protocol suite?

The WAP protocol suite consists of following protocols-

1. Wireless Application Environment (WAE)
2. Wireless Session Protocol (WSP)
3. Wireless Transaction Protocol (WTP)
4. Wireless Transport Layer Security (WTLS)
5. Wireless Datagram Protocol (WDP)

20. What is WAP 2.0?

WAP 2.0 is simply a blended mixture of XHTML, end to end HTTP, which was released in 2002. It has dropped the gateway and custom protocol suite used to communicate with it.

21. On which Networking Model is WAP based upon?

Open System Interconnection (OSI) model.

22. Describe the Transport Layer of WAP

The Transport Layer consists of Wireless Datagram Protocol (WDP).WDP allows WAP to be bearer-independent by adapting the transport layer of the underlying bearer. The WDP presents a consistent data format to the higher layers of the WAP protocol stack, thereby offering the advantage of bearer independence to application developers.

23. Define in brief, "Wireless Application Environment (WAE)"

The Wireless Application Environment, or WAE, provides architecture for communication between wireless devices and Web servers. It consists of device specifications and the content development programming languages like WML

24. How you define WAP

WAP is stands for **Wireless Application Protocol**. Using WAP protocol we display internet contents on wireless users. Example: mobile phone, i-pod etc.

I have given you some basic information about WAP are given below:

1. WAP is used as an application communication protocol.
2. Using WAP we can access services and information
3. We can inherited WAP from Internet standards.
4. WAP is designed for handheld devices Like: mobile phones, i-pod etc.

25. Give examples of WAP

1. Using WAP we can get information of train time-table.
2. Using WAP can purchase tickets. Like: movie, journey ticket etc.
3. Using WAP we perform task like that Flight check in
4. We can also viewing traffic information.
5. We can get information about current weather condition.
6. Using WAP we can do also trading of shares.
7. We can also display sport results on our small wireless devices

26. Does WAP runover GPRS

Yes, it can do. GPRS is a new over-the-air service that transmits data packets to hand-held devices. It will allow much faster WAP transmission than currently available over SMS or CSD when using GSM.

27. How secure is WAP

One of the layers of the WAP stack, known as WTLS, provides encryption and authentication for server-to-client security. This prevents fraudulent access to WAP transactions and opens the way for e-commerce- and intranet-type applications.

29.What does Windows CE means?

Windows CE is a Windows operating system, also called Windows Embedded Compact. It is mainly used for mobile devices like Smart Phones and PDAs. Windows CE is an operating system created by Microsoft for entrenched systems. It is a different operating system and kernel, rather than a trimmed down version of desktop Windows. It is not to be confused with Windows XP Embedded which is NT-based.

30. What is EPOC? [NOV 2013]

EPOC[Electronic Piece Of Cheese] is an operating system designed for small, portable computer-telephones with wireless access to phone and other information services. operating system from Psion Software, designed specifically for mobile, ROM -based computing devices. *EPOC16* is a 16-bit version of the operating system that has been available for several years and is embedded in many handheld devices. *EPOC32* is a newer, 32-bit operating system that supports preemptive multitasking

11 Marks

1. Explain mobile computing models (11)][April 2014] [April 2013][NOV 2013]

Computing in the mobile environment is different from the conventional fixed- network computing. This is partially due to the movement of the mobile hosts that require remaining connected from different access points while moving. The difference also stems from the nature of the wireless links that are relatively unreliable and offer low communication bandwidth. Furthermore, mobile hosts equipped with rechargeable batteries suffer from limited operation time constraints. As a consequence, new models in the mobile environment are needed to support information access for mobile users.

The following models of computing in the mobile environment are currently being researched and investigated:

1. Client server model
2. The Client/Proxy/Server Model
3. The Thin Client Model
4. The Disconnected Operation Model
5. The Mobile Agent Model

THE CLIENT/SERVER MODEL

In this model, neither the client nor the server are aware of the client (or server) mobility. The conventional client/server model is used without any modifications made in the application or the transport layer. The wireless media is transparent since it is handled in the data link layer. The mobility, on the other hand, is made transparent by handling the variable client/server location through location-based routing in the network layer. Mobile IP is an example of a network protocol that hides the C/S mobility. The advantage of this model is its portability. The client or the server need not be changed in any way. Simply, the client (server) is ported to the mobile host, with a fixed address specified for the server (on the fixed network or on a mobile host). The disadvantages of this model are listed below:

The mobile client may suffer from a slow and unpredictable response time, especially when large server replies such as query results are transmitted without any considerations to the limited wireless bandwidth.

The server caching strategy may not work properly in this model because the majority of the fixed network caching algorithms use call backs to invalidate the client cache. Most invalidation algorithms rely on the continuous availability of the client. Since the client could be disconnected or temporarily inaccessible (during hand-offs for example), the cache invalidation process could fail.

THE CLIENT/PROXY/SERVER MODEL

To overcome the conventional client/server model, the client/proxy/server (C/P/S) model introduces a mobility-aware middle layer to mediate the interactions between the client and the server. The basic idea behind this model was introduced in the Mowgli architecture], which is to split the communication path between the client and the server into two parts by using a store-and-forward interceptor.

The main advantage of this model is that the proxy allows the client and the server to be designed without any built-in mobility assumptions. The proxy assumes that the client is mobile and the server is in the fixed network. The result from the server is sent back to the proxy. The proxy filters the results according to the limitations of the wireless media and/or the client's mobile unit. The proxy may also store the filtered results until the client is connected. Examples of data filtering include: color and resolution reduction, audio file removal, and file size reduction. The programming of the proxy involves knowledge of the mobile host hardware specifications. For example, a mobile host that does not have audio capability will benefit from audio file removal filtering. In addition to the mobile host, the mobile user profile can be useful in providing the proxy with user preferences such as no-images and no-colors.

THE DISCONNECTED OPERATION MODEL

Mobile clients may face wide variations in network conditions and local resource availability when accessing remote data. This is true in the C/S and the C/P/S models. A disconnected operation is a variation of the C/S model where, instead of working under the extreme case of weak-connectivity, the mobile client effectively switches to use a network of zero bandwidth and infinite latency. The operations that enable a client to continue accessing critical data during the disconnection (switch off) period are called disconnected operations. The ability to operate when disconnected can be useful even when connectivity is available. For example, disconnected operation can extend battery life by avoiding wire- less transmission and reception. It allows radio silence to be maintained, a vital capability in military applications. And, it is a viable fallback position when network characteristics degrade beyond usability. Voluntary disconnection can be treated as planned failures which can be anticipated and prepared.

THE MOBILE AGENT MODEL

Agents can be classified as static, mobile scripts, or mobile objects. Static agents are those which execute just on a single site, either as a client or as a server. A static agent could be carrying out some activity like mail filtering. Mobile scripts are those that are downloaded from a server and executed on a client. Java applets, perl, or python scripts

can be classified as mobile scripts. Mobile agents are mobile scripts with associated execution state information. A mobile agent could either be relocated along with the user, or it could be relocated during the execution of the agent. The relocation of the agent involves saving the state before initiating relocation and later restarting the mobile agent at the new location. The mobility of agents raises a large number of issues like security, authorization mechanisms, access mechanisms, and relocation mechanisms.

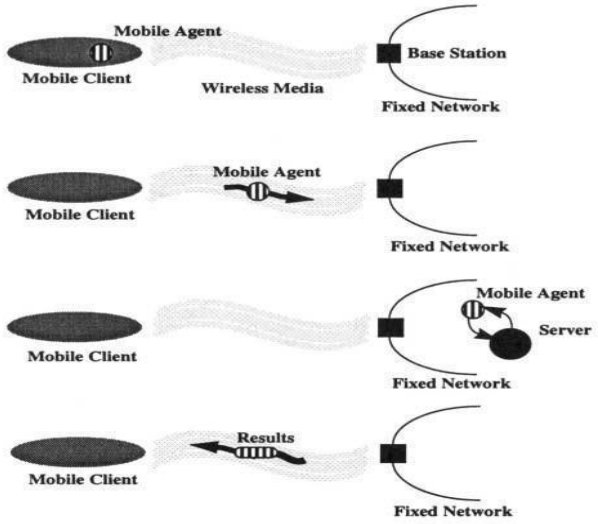

Figure 7.8 Mobile agents for mobile computing

The mobile agent is an emerging new model that provides an alternative to the C/P/S model. A mobile agent is an active entity that is knowledgeable of both the limitations of the mobile environment and the mobile user. To access remote data, the mobile user sends a mobile agent on his behalf to the data source in the fixed network. The mobile agent is an execution context initially loaded with the queries or data access requests.

THE THIN CLIENT MODEL

The thin client computing model attempts to offload most application logic and functionality from mobile clients to stationary servers. In this model, applications in stationary servers are usually mobile-aware and optimized for mobile client devices. This model is especially suitable for dumb terminal or small PDA applications.

The thin client architecture from CITRIX Corporation allows a variety of re- mote computers, regardless of their platform, to connect to a Windows NT terminal server to remotely access a powerful desktop and its applications. A server called MetaFrame runs under Windows NT in the desktop machine and communicates with the thin clients executing at the remote computers using the Independent Computing Architecture protocol (ICA). The ICA client and the MetaFrame server collaborate to display the virtual desktop on the remote computer screen. They also collaborate to process mouse and keyboard events and to

execute programs and view data stored at the server. All executions are remote and none take place at the client portable computer.

2. Write in detail about JAVA (11)

Java is a programming language offers the most portable commercial environment for writing software applications. The success of Java has been mostly in providing standard Application Program Interfaces (APIs), a very thoughtfully designed infrastructure for OOP that prohibits many bad design and implementation habits such as multiple inheritances. Standard and open APIs offer a process of evolving a language that is open to many vendors. There are three major categories of Java APIs and virtual machines, namely J2ME, J2SE, and J2EE.

Java offers three distinct features as a mobile application platform:

1. Java is an object oriented programming language. As any other programming language, it can be used to write applications.

2. Java offers complete code mobility and weak mobile agent ability. Java allows for platform-independent programming.

3. Java is a platform.

First, Java, as with any other programming language, is just that: a programming language. It allows us to program a set of instructions. Perhaps just as importantly,

Java is somewhat of a *vendor-neutral language-based platform*."

Java Database Connectivity (JDBC) APIs present the same interface to the developers regardless of what database is being used.

Java, as a platform and programming language, offers mobile code. But, the standard Java Virtual Machine was designed for desktop computers and requires far too many resources for the typical cell phone, PDA, or mobile device. The standard Java Virtual Machine is packaged, along with accompanying tools and class libraries, into Java 2 Standard Edition (J2SE).

J2ME

J2ME is a specification for a virtual machine and some accompanying tools for resource-limited devices. J2ME specifically addresses those devices that have between 32 kB and 10 MB of memory. J2ME addresses the needs of two categories of devices [Sun Micro J2ME Spec 2000]:

1. *Personal, mobile, connected information devices*. This portion of J2ME is called CLDC for Connected, Limited Device Configuration. These types of devices include cell phones, PDAs, and other small consumer devices. CLDC addresses the needs of devices with 32 to 512 kB of memory. The virtual machine for the CLDC is called KVM for K-Virtual Machine.

2. *Shared, fixed, connected information devices*. Internet-enabled appliances, mobile computers installed in cars, and similar systems that have a total memory of 2 to 16 MB and can have a high bandwidth and continuous connection to the network are in this group. CDC, or Connected Device Configuration, is the part of J2ME that addresses such devices. CDC is a superset of CLDC.

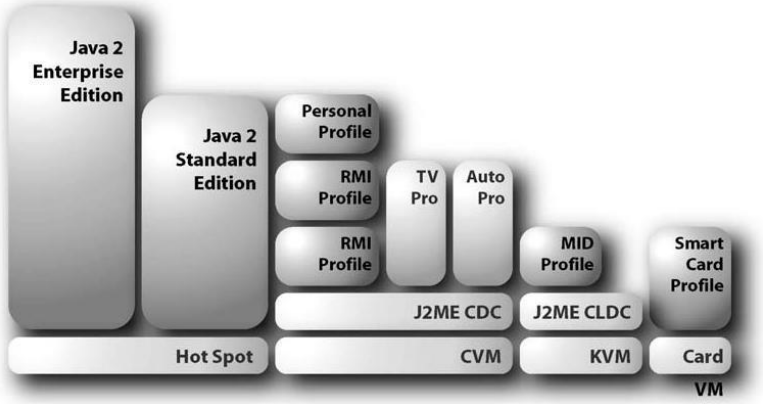

FIGURE 2.2. J2ME Stack (CLDC/CDC and MIDP).

CLDC addresses the following features:

1. *Providing a virtual machine for providing language features*. Perhaps the most important thing to keep in mind for those who have built applications using the Java Virtual Machine on desktops and servers is that the J2ME/CLDC Virtual Machine is not at all like the version that comes with J2SE. Some features not offered on the KVM are the following:

a. *Floating point arithmetic*: Floating point operations are expensive or require the chipset on the device to have specific implementations for them.

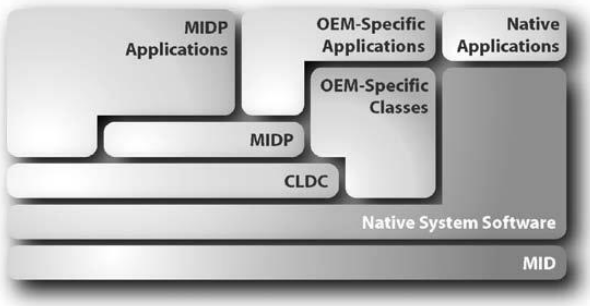

FIGURE 2.3. Layering of Functionality between CLDC and MIDP.

b. *Support for JNI*: Java Native Interfaces (JNI) allow developers to write applications that use C/C++ programming languages along with Java in providing Java APIs to modules or applications not written in Java.

c. *Thread grouping*: Advanced threading features are not offered on the KVM and CLDC. Multithreading requires a baseline amount of resources to be dedicated to creating, maintaining, and destroying threads. Each thread takes up a certain amount of resources by simply existing, even if it never does any actual work.

d. *Full-blown exception handling*: Exception and error handling seems to be one of the first places that platform providers trim when building frameworks and tools for limited devices.

e. *Automatic garbage collection of unused objects*: Though the KVM does offer some of the memory management features of the J2SE Virtual Machine, it does not offer *finalization* of objects. This means that you have to tell the KVM when you are done with an object.

f. *Weak references*: The J2SE Virtual Machine does not allow finalization of an object until all weak and strong references to that object are cleared. The KVM does not provide this functionality for weakly referenced objects.

2. *Providing a security framework for tasks such as downloading MIDlets (J2ME CLDC/MIDP applications)*. Security is one of the most troublesome and complicated features for providers of mobile application frameworks and tools

3. *Providing a reasonable amount of functionality for input and output*. Most pro- grams need a persistence mechanism. CLDC provides a very limited and yet sufficient set of APIs to read and write to the nonvolatile memory provided by devices

4. *Providing some internationalization capabilities*. CLDC's input/output (I/O) pack- age provides input and output stream readers that can handle different character encoding schemes. This allows internationalization in two ways:

a. *Dynamic*: The program can determine the required character set dynamically and use the proper character set at run time.

b. *Static*: There can be multiple versions of the J2ME application ready to be loaded onto the device.

5. *Providing a reasonable amount of networking capabilities*. CLDC provides a con- nection framework to provide basic networking capabilities. The areas addressed by profiles are the following:

1. Download and installation of applications,

2. life-cycle management of applications,

3. User interface feature,

4. Database functionality, and

5. Event handling.

Sun Microsystems has a free tool kit offers the components for development of J2ME applications:

1. *KToolbar*

2. *Preverifier*

3. *Compiler*

4. *Emulators*

5. *Emulation of Performance*

3. Explain BREW (11)

Qualcomm's **BREW** (**Binary Run-time Environment for Wireless**) gives application developers a new and different approach in producing mobile applications. BREW is built directly into the hardware. It is offered as an API to access the CDMA, GSM/GPRS, or UMTS chip sets that provide the support for it. But, it is primarily intended for the variations of CDMA, a technology owned and licensed by Qualcomm. BREW applications can be written on a PC using the BREW Software Development Kit (SDK). Once the application is developed, it must be tested, and then deployed. Deployment of BREW applications is a process done jointly by Qualcomm and telecommunications carriers and not just the developer.

Though the creators of BREW say that they first came up with the acronym BREW and then found the words to fit the acronym, the platform is somewhat biased toward wireless applications that run on phones. And this may be the only weakness of BREW as a mobile development platform. Although today developing mobile applications means targeting cell phones or PDAs, this is changing rapidly with new devices being introduced to the market.

BREW applications, also referred to as BREW applets, are written in C though some support for C++ is provided (although some fundamental things such as extending the base API through inheritance are not possible) and, using code generation or virtual machine technologies, other languages such as Java can be supported. One of the most impressive things about BREW is its near-full treatment of dimensions of mobility in its architecture, feature implementation, and SDK. Let us look at the various components that allow the developer to build a BREW application.

BREW SDK Overview

Once you have installed the BREW SDK, you will have the following set of applications available for development:

1. *BREW MIF Editor*: Every BREW module, defined as the classes that make up one or more BREW applications, has an associated Module Information File (MIF). MIFs are *required*. Every BREW module must have a MIF. The MIF Editor provides a GUI tool for editing the MIF file associated with the classes that make up a module. The MIF Editor that comes with BREW SDK version2.0 can be started as a wizard inside Visual C++ 6.0 or independently as a stand- alone application. We will look at the use of the MIF Editor and building a simple application.

2. *BREW Device Configurator*: This is a stand-alone application that allows developers to make up their own handset by configuring a vanilla mobile phone and specifying the behavior of the keys, the look and feel of the screen, and other specifics of the device. This development tool addresses the large variety of existing devices by allowing developers to create their own device emulator and testing the application. Remember, also, that because BREW is a platform for writing application for the handset,

it is possible to use the application to build some adaptive behavior to adapt to each type of device. Still, the Device Configurator is invaluable in that it allows developers to test the application on their own emulated device environment.

3. *BREW Emulator*: For those who have designed and implemented any mobile application, it is obvious that one of the most difficult steps in the development process is the incremental unit testing. Although most platforms provide some sort of a generic emulator, most do not allow for custom configuration of a device (done by the Device Configurator) or using the custom configuration to simulate running an application **4.** *Register as a BREW developer*: This is not a simple sign up for notifications and other news about BREW. You will have to have the VeriSign Class 3 certificate before you can become an "Authenticated Developer." Once you are an authenticated developer, you are ready to work.

5. *Obtain a Class ID for your application*: During the development process, you can use a dummy Class ID. But, to get the application out on a real device, you need to get a Class ID for the application. This Class ID uniquely distinguishes your application from all other BREW applications. Every BREW application has to have a Class ID and the Class IDs are issued and provisioned by Qualcomm centrally to avoid ID collision.

6. *Perform a unit test and send it to a testing lab*: If you are done with steps 1–5, you are ready to submit your application for testing. To get the application onto your phone, a Qualcomm-approved testing center needs to test your application. It should be obvious that you want to bullet-proof the application before submitting it to the testing center. This process is put in place to avoid "crashing" the mobile device.

7. *Perform a pricing and carrier evaluation*: Once the testing lab approves an application, it is ready to be provisioned. Deployment of an application is done by Qualcomm and the carriers supporting BREW. Therefore, the application developer must submit the software to Qualcomm and the carriers for actual deployment. Of course, deployment is done after the carrier and Qualcomm approve of the application.

Unfortunately, deploying a BREW application onto a device is not free. Before you get an application up and working on a device, you need to pay various types of fees for the VeriSign certificate and, in practicality, become a Qualcomm developer, requiring a significant

membership fee depending on what type of membership you want to sign up for. The positive spin on this, of course, is that the membership fee pays for some marketing and weeds out those developers who do not have a product and are just playing around with the platform. If you want to just learn the platform, the best thing is probably just to download the SDK and the tools after reading this section and then experiment with it. Now, let us go on to the actual code.

Hello BREW

Once you have downloaded and installed the BREW SDK, you can get started. Here is the procedure:

1. Click on File, New, and then Projects. You will see the BREW Application Wizard. Choose it.

2. The wizard will ask you if you want File, Network, Database, TAPI, or Sound functionality. These selections correspond to the organization of the BREW standard libraries:

a. *Files*: BREW provides an API that allows storage of small amounts of information in structures with which application developers are familiar: files and directories of files.

b. *Database*: Information is often better stored in a database instead of files if the data must be searched, sorted, or indexed. BREW provides a set of APIs to store, manipulate, and retrieve data that have a small amount of database- like functionality. What BREW offers, it should be noted, is not nearly as complete as a full-blown database system. However, it offers enough for useful functionality.

c. *TAPI (Telephony API)*: Because BREW is built on a wireless telephony platform (CDMA), it is natural that it provides telephony functionality. At the time of release of BREW SDK 2.0, functionality is limited to sending SMS messages and switching back and forth between incoming and outgoing telephony calls. However, being built on a telephony platform, it is almost certain that BREW will offer functionality that provides control and manipulation of the audio over the telephony channel, integration with voice recognition, and other useful functionality.

d. *Sound functionality*: Sound functionality is provided through a set of multimedia APIs. Sound can be stored on the device in BREW's own format of QCELP (a Qualcomm technology).

3. Once you have selected which libraries you will be using in your application, you will need to create a MIF file for it. If you do not yet have a true valid Class ID, you have to get the Verisign certificate, become a Qualcomm developer, and go through the steps that we

mentioned previously. Once again, every BREW application must have an MIF file. Click on the MIF Editor. You will see these different tabs on the wizard:

a. *Applets*: This is where you generate a test Class ID (or if you have already become a Qualcomm developer, get a real Class ID from Qualcomm). The other basic properties of the BREW application (as we mentioned bef or e , interchangeably referred to as BREW applet) are set in this pane. Note that if you click on the *advanced* button on this pane, another window pops up with some features that seem to be programmatic. In BREW, certain behaviors of the application such as its treatment of incoming telephony events (when a call comes in while the application is running) have to be specified. Some behaviors are fundamental to the behavior of the application and, therefore, are required to have a Footprint in the MIF file.

b. *General*: This pane is for entering the security-related information. Every BREW application m a y provide access to other BREW applications a n d modules or require a particular access level for certain functionality on the device. It is important for this information to be on the MIF file because other applications must know whether they are usable by other applications or not and so that the application container (the BREW environment running on the BREW device) knows whether it can load and execute the application.

c. *Extensions and Dependencies*: Because BREW applications can come in several modules or have interdependencies among themselves, the MIF files allow for specifying these dependencies. The Extensions and Dependencies panes provide a graphical way of manipulating these dependencies.

4. Explain WINDOWS CE (11)

Windows CE is a Windows operating system, also called Windows Embedded Compact. It is mainly used for mobile devices like Smart Phones and PDAs. Windows CE is an operating system created by Microsoft for entrenched systems. It is a different operating system and kernel, rather than a trimmed down version of desktop Windows. It is not to be confused with Windows XP Embedded which is NT-based. An operating system is the master control program that enables the hardware by abstracting it to the application via drivers [Development tools for Mobile and Embedded Applications]. Microsoft's various products revolve around different versions of an operating system.

These two operating systems are designed for two different purposes. There are different flavors of the Windows CE operating systems, of course, depending on the hardware platform. Some of these flavors are the Pocket PC, Windows CE .NET, and Pocket PC 2002. These flavors largely depend on the commercial bundling of different feature sets and

hardware platforms with which they are shipped (such as Compaq's IPAQ). Embedded Windows XP, in contrast, is a subset of the desktop version of Windows XP components. Development for Embedded Windows XP is a bit more straightforward than developing for Windows CE.

Mobile application frameworks that are based on an operating system treat developing mobile applications in the same way as they treat their stationary counterparts on PCs. As we mentioned previously, the operating system provides basic access to the hardware such as I/O, networking, etc. So, the applications that run on Windows CE and Embedded Windows XP are controlled by them, respectively. Microsoft provides tools to build applications for each environment too. These are as follows:

1. *Embedded Visual C++*: This is a tool set separate from Visual Studio, the typical development environment for PC-based Windows applications. It allows for authoring mobile applications in C++.

Emulators and a debugger are provided. The latest version of this tool provides advanced features such as exception handling and run-time debugging, features you will cherish if you are actually developing an application in C++ for mobile devices.

2. *Embedded Visual Basic*: This tool provides the ability to write applications using Visual Basic. Visual Basic applications can be developed faster but do not offer the developer the ability to tune and optimize the application for resource- starved mobile devices. Therefore, Embedded Visual Basic is really not a suitable tool for developing large commercial applications, but it does well for proof-of- concept and prototype applications.

3. *Smart Device Extensions for .NET*: The .NET application programming platform, the newest set of tools for building Microsoft Windows-based applications, can be complemented with a set of extensions that allow developers to author .NET applications for mobile devices.

4. *Microsoft Mobile Internet Toolkit*: This is really a server-side framework.

As in the other Microsoft Windows platforms, Windows CE allows the use of COM and ActiveX components in addition to the Win32 API. The other significant features are markup language processing (HTML, XML, XSL, etc.), security (e.g., SSL), a subset of the Windows ADO database access framework in ADOCE (ADO for Windows CE), and limited functionality in coupling with the Microsoft messaging queue in MSMQ.

Just like Windows 2000, Windows CE utilizes protected memory architecture. When a Windows CE machine first boots, it creates a single 4 GB virtual address space [Introduction to eVC++]. This, however, does not mean that there is 4 GB of random-access memory (RAM) available! In fact, currently most Windows CE devices are limited to under 64 MB. Moreover, although we expect mobile devices to continue to grow in their processing ability and

memory, it is unlikely that we will ever want to run memory-intensive applications on mobile devices because physical size is a limiting factor and battery life is not growing as fast as processing power. This virtual address space is then divided into 33 different "slots," each of which is available for use by a process. The maximum size of each slot is 32 MB. This is simply the model with which the memory is managed; it does not mean that the device is required to have 4 GB of memory or that it offers 32 MB of memory per process. Also, keep in mind that Windows CE does not allow paging (file swaps), so you can exceed the available RAM. File swapping is something that is typically not implemented as a strategy for improving the memory limitations of mobile devices as it is cost prohibitive to the battery life and takes considerable processing power.

When building eVC applications, keep the following in mind:

1. Graphics are expensive. Whether you decided to use GDI or another method of rendering graphics, delivering them takes more memory, more CPU, more time, and more power from the battery. So, try to avoid graphics when possible.

2. Use events instead of polling when possible. Figure 2.8 shows some sample code that involves events in Windows CE environment. Polling is expensive for the same set of reasons as graphics are. Sleeps and event notifications are both features available to the eVC programmer to produce efficient applications.

3. Be very frugal in the use of RAM in your applications. Remember that persistent and RAM memory are typically handled by one set of hardware on mobile devices. Today, most mobile devices, including Windows CE devices, do not have hard drives (though this is changing).

4. As mentioned in item 3, because RAM and persistent memory often share the same hardware, being frugal in persisting data or handling data in memory pays off in reducing power consumption as well.

5. There is some functionality provided to the application developer to get the status of the power consumption. You can use this functionality in two ways. First, you can use it while designing and testing to see the power consumption during the life cycle of the usage of the application. Alternatively, you may want to use the power status (called up by GetPowerStatusEx function) to change the behavior of the application. For example, if the battery is getting low, you might want to persist the data to the network or locally and shut down the application after warning the user.

6. Make sure that you clean up memory resources whenever you get a WM HIBERNATE event (which sends the device into hibernation). Failing to do a good memory cleanup there will lead to memory leaks and application instability.

5. Explain WAP in detail.

Wireless Application Protocol (WAP) is the single framework most used in building mobile applications today. Despite all of its initial high promises, its lack of meeting those promises, and being written off for dead, WAP seems to have survived the critics and continues to improve. WAP, which was initially intended to be as pervasive for wireless and mobile applications as HTTP has been for the Web, never achieved the level of success initially expected. However, to date, WAP has the largest install base of all open application development platforms (second to NTT Docomo's closed and proprietary i-mode system) on mobile phones, meaning that WAP is installed on more mobile phones than any other software.

1. *WAP is intended for thin clients*. Like HTTP, the designers of WAP 1.x were thinking about a thin-client technology: a case where nearly all logic is calculated on the server and very simple display instructions are bundled in some markup language to be displayed by the client.

2. *WAP is built on its own lower level communication protocol*. The HTTP assumes the existence of TCP/IP (which in turn provides persistent connections), WAP is built on its own set of communication protocols that wrap around TCP, UDP, or a variety of other possible protocol implementations.

3. *Typical deployment of WAP includes a proxy or a gateway*. Wireless carriers (also referred to as *bearer networks*) used to control every single incoming and outgoing bit of data that travels on their network. Interoperability or functionality.

4. *WAP is a complete framework for mobile applications*. Most tools created for development of applications treat a part of the mobile application chain; WAP treats, or at least attempts to treat, all parts of the mobile equation. The fact that WAP has various parts, however, does not mean that some of its parts cannot be used individually. For example, J2ME applications can use WAP as a communication protocol while not using the WAP browser (instead using a client-side J2ME application to provide a richer and better user experience).

WAP Architecture

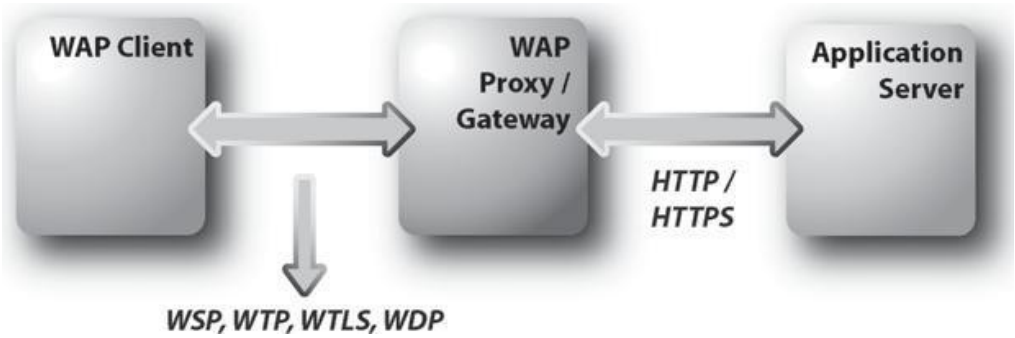

FIGURE 2.11. Basic Communication Architecture in WAP.

WAP architecture shows the functionality to the communication between the client and the server (WAP Gateway or WAP Proxy):

1. *Handling of Telephony on the Device*: Currently, devices and networks treat voice and data differently. Most voice is processed, and will continue to be processed for a very long time, through a telephony system that does not provide much in the way of complex operations.

2. *Push*: Push offers some degree of treatment for one of the dimensions of mobility, namely support for active behavior. It needs to be reiterated that the WAP architecture was designed with a subset of the mobile application arena in mind: wireless applications.

WAP Proxies and Gateways

The difference between a proxy and a gateway is that a client determines when it will use a proxy. Some WAP-enabled devices do have the ability to change their proxy settings; however, this feature is typically disabled from access by the user by the network provider or the device manufacturer as it can circumvent billing mechanisms (especially for the United States compared to Europe). Therefore, whether used as a proxy or a true gateway, WAP intermediaries are typically referred to as WAP Gateways.

WAP gateways provide six important features:

1. *Security*: The WAP gateway provides a secure handoff point between WTLS (Wireless Transport Layer Security) to external security mechanisms such as SSL for HTTP in the form of HTTPS.

2. *Network Access*: The WAP gateway is the access point for the WAP client devices. Network providers are able to restrict access to users and connect their usage to billing systems by using the WAP gateways.

3. *Protocol Conversion*: The gateway is responsible for converting Wireless Session Protocol (WSP) to HTTP. This allows WAP to be based on a non- TCP/IP application layer protocol (WDP) and yet interact with the HTTP-based

4. *Caching*: The WAP gateways provide a caching mechanism equivalent to (and even surpassing) that provided by HTTP.

5. *Preparation of Content and Scripts*: WML is textual, and text is not a very efficient format for transfer of data. Therefore, the gateway encodes WML into "Compiled WML" (WMLC) before shipping it to the WAP-enabled device.

6. *Functionality Offered through WAP 2.x and Higher*: Although WAP 2.x is not yet deployed in the United States and is just beginning to be rolled out in Europe, it offers a variety of changes to the role of the proxy/gateway in WAP deployments.

WAP 2.x offers WAP push, support for UAProf ,additional functionality for WTA (Wireless Telephony Application), an External Functionality Interface (EFI) offering an extensibility

mechanism for WAP APIs, data synchronization using SyncML, a persistent storage interface for storing data on the device, a multimedia messaging service, pictograms (small images used in messages), and provisioning.

The Multimedia Messaging Service (MMS) is the more mature child of the ever popular Short Messaging Service (SMS).

6. Explain SYMBIAN EPOC (5)

Symbian, one of the most powerful and popular platforms for mobile development, was created jointly by Ericsson, Nokia, Panasonic, Psion, Samsung Electronics, and Siemens. The effort in creating this new operating system targeted at mobile devices started in 1998 and the first Symbian phones became available in 2001. The majority of the user base of Symbian devices is in Europe with very little user base in the United States; however, the market share in Europe is large and growing, with other markets wide open between the various contenders of mobile operating systems including Symbian. The Symbian OS 7.0 comes with considerable basic functionality for mobile applications: support for MMS, HTTP communication, SyncML synchronization, SMS, support for Mobile IP (through support for IPv6), and short-range wireless networking with IrDA and Bluetooth.

Symbian started as an operating system that supported primarily C++, but it evolved to providing support for Java as well. Like the other tools that we have looked at, you can download the development SDK for free from the Symbian site and there are a variety of commercial IDE (Integrated Development Environment) that support application development for Symbian. The Java Virtual Machine implementation of EPOC is based on the Personal Java standard.

Deploying Java applications to Symbian is much easier than deploying BREW or J2ME and more like deploying them onto a Windows CE device. This is because Symbian is designed more as a PDA operating system than as an ultra-light mobile environment. Symbian's latest operating system (Symbian OS 7.0) supports multithreading.

7. Briefly explain about The Tools used in Mobile Computing Model.[NOV 2013]

Mobile Computing is "taking a computer and all necessary files and software out into the field". Mobile computing is any type of computing which use Internet or intranet and respective communications links, as WAN, LAN, WLAN etc. Mobile computers may form a wireless personal network or a piconet.

There are at least three different classes of mobile computing items:

Portable computers, *compacted* lightweight units including a full character set keyboard and primarily intended as hosts for software that may be parameterized, as laptops, notebooks, notepads, etc.

Mobile phones including a restricted key set primarily intended but not restricted to for vocal communications, as cell phones, smart phones, phonepads, etc.

Wearable computers, *mostly* limited to functional keys and primarily intended as incorporation of software agents, as watches, wristbands, necklaces, keyless implants, etc.

The existence of these classes is expected to be long lasting, and complementary in personal usage, none replacing one the other in all features of convenience..

Limitations:

Range & Bandwidth: Mobile Internet access is generally slower than direct cable connections, using technologies such as GPRS and EDGE, and more recently HSDPA and HSUPA 3G and 4G networks. These networks are usually available within range of commercial cell phone towers. Higher speed wireless LANs are inexpensive but have very limited range.

Security standards: When working mobile, one is dependent on public networks, requiring careful use of VPN. Security is a major concern while concerning the mobile computing standards on the fleet. One can easily attack the VPN through a huge number of networks interconnected through the line.

- **Power consumption**: When a power outlet or portable generator is not available, mobile computers must rely entirely on battery power. Combined with the compact size of many mobile devices, this often means unusually expensive batteries must be used to obtain the necessary battery life.
- **Transmission interferences**: Weather, terrain, and the range from the nearest signal point can all interfere with signal reception. Reception in tunnels, some buildings, and rural areas is often poor.

Potential health hazards: People who use mobile devices while driving are often distracted from driving and are thus assumed more likely to be involved in traffic accidents (Cell phones may interfere with sensitive medical devices. Questions concerning mobile phone radiation and health have been raised.

- **Human interface with device**: Screens and keyboards tend to be small, which may make them hard to use. Alternate input methods such as speech or handwriting recognition require training.