Mobile Software Engineering

Unit 2

Red: indicates important

Contents

- GSM
- J2ME
- Java Device Test Suite

Introduction to GSM

- GSM is the architecture on which mobile devices are based on.
- GSM stands for Global System for Mobile Communication
- It's a digital wireless standard.
- Features of GSM:
  - If a mobile device is based on GSM, then it can be used in all those countries where this particular standard is prevailing.
  - Almost all services that are existent in a wireline network are provided by GSM to all the users of mobile devices which are based on it.
  - Quality of voice telephony is not excellent.
  - It provides good security as there is an option to encrypt the information that is being exchanged.
  - There is no need for significant modification of wireline networks due to the establishment of networks based on GSM standard.

Architecture of GSM

- There are three different parts in a mobile device. They are SIM card, Mobile equipment and Terminal equipment.
- **SIM** stands for Subscriber Identity Module. A mobile device can be loaded by any SIM. The number of mobile device is associated with SIM. Size of a SIM is similar to smart card. Every Sim has a PIN(Personal Identity Number) associated with it. After loading the SIM into the mobile device, the user is asked to enter the PIN. Only on entering the correct PIN, the user will be able to start using the services offered by the mobile operator. Initially PIN is loaded by the operator. The PIN can be changed by the user. If the user is unable to enter the PIN correctly for the first time, then he/she is given an opportunity to re-enter it for a fixed number of times. In case he/she is not able to enter it correctly and exhausted all the tries, then the PIN is blocked. So, that particular SIM cannot be used unless the network operator activates it. The SIM can be unblocked on entering the correct PUK(PIN Unblocked Key). Information regarding subscriber, PIN and PUK codes are present in SIM.

- Architecture of GSM consists of Mobile devices, Base Station Systems(BSS), Mobile Switching Center(MSC) etc.. The communication between Mobile device and BSS is through radio interface. BSS communicates with MSC by connecting to Network and Switching Subsystem(NSS). An MSC is a telephone exchange that is specifically configured for mobile applications. Mobile devices and Public Switched Telephone Network(PSTN) interface through Base stations. BSS consists of a Base Transceiver Station(BTS) and Base Station Controller(BSC). Equipment related to transmission, reception and signaling is part of BTS and this equipment is used by it to contact Mobile devices. BSC deals with the allocation of radio channel and its release apart from hand off management. Using ISDN protocols, BSC communicates with BTS.

Wireless Application Development Using J2ME

- Java supports mobile application development using its J2ME.
- J2ME stands for Java 2 Platform, Micro Edition.
- J2ME provides an environment under which application development can be done for mobile phone, personal digital assistants and other embedded devices.
- J2ME includes API's and Java Virtual Machines.
- It includes a range of user interfaces, provides security and supports a large number of protocols.
- J2ME also supports the concept of write once, run anywhere concept.
- Application can be developed using J2ME targeting a specific type of devices.

- J2ME is one of the popular platforms that is being used across the world for a number of mobile devices, embedded devices.
- Some of the wireless applications are Games, Applications that access databases, Location based services etc..
- The architecture of J2ME consists of a number of components that can be used to construct a suitable Java Runtime Environment(JRE) for a set of mobile devices.
- When right components are selected, it will lead to a good memory, processing strength and I/O capabilities.
- There are two configurations for J2ME. They are Connected Limited Device Configuration(CLDC) and Connected Device Configuration(CDC). Each configuration consists of a virtual machine and a set of libraries. Each configuration provides functionalities. One feature is network connectivity. To provide a full set of functionalities a profile and an additional set of APIs should be added. Additional APIs will enable the usage of native properties and suitable user interface. Each profile will support a smaller set of devices.

- CLDC is combined with MIDP. MIDP stands for Mobile Information Device Profile. This combination provides a Java based environment for cell phones in which applications can be developed. Optional packages can be added to the configuration to support additional services. Optional packages consist of set of APIs to support latest technologies like Bluetooth, multimedia etc.. Optional packages are not mandatory.
- CDC is the larger of the two configurations.
- CLDC is designed for devices with less memory, slow processor and unstable network connections. Its suitable for cell phones. It has memory that ranges from 128KB to 512 KB.
- CDC is designed for devices with larger memory, faster processor and significant network bandwidth. Its suitable for TV set top boxes and PDAs(Personal Digital Assistant). CDC consists of JVM and Java software that includes a portion of J2SE(Java 2vStandard Edition) platform.

- MIDP is designed for cell phones and other lower level PDAs. User interface, network connectivity, application management are offered by MIDP. The combination of MIDP and CLDC provides JRE for mobile devices. This leads to better power consumption and efficiency in memory utilization.
- Different profiles for CDC :
  - Function Profile(FP): It's the lowest level profile for CDC. It provides embedded devices which does not have a user interface with network capability. Two more profiles are available: Personal Basis Profile(PBP) and Personal Profile(PP). FP can be combined with these profiles to provide GUI for the devices that require it. All the profiles are layered in manner. This facilitates adding as well as removing profiles.
  - PP is useful for all the devices that require a GUI, Internet support etc. Devices like higher level PDAs, Communicators etc. require such features. Java AWT (Abstract Window Toolkit) is contained in this profile. It also offers web fidelity, applets etc.

- PBP is a subset of PP. It supports application environment to those devices in the network that support at least a basic graphical representation. PP as well as PBP are layered to the top of CDC and FP.
- Reasons for the usage of Java Technology for wireless application development :
  - Java platform is secure. Its safe. It always works within the boundaries of JVM. Hence, if something goes wrong, then only JM is corrupted. The device is never damaged.
  - Automatic garbage collection. This is provided by Java.
  - Java offers exception handling mechanism. It helps in creating robust application.
  - Java is portable.
- J2ME wireless toolkit for development of wireless application using J2ME.

- Soft wares required to setup a development environment:
  - J2SE(Java 2 Standard Edition) SDK(Software Development Kit) of v1.4.2 or upwards.
  - J2ME(Java 2 Micro Edition) Wireless Toolkit. With this toolkit MIDlets can be developed.
  - Any text editor
- MIDlets are programs that use the MIDP.
- The first step is to install J2SE SDK. J2ME wireless tool kit runs on the Java platform provided by J2SE. J2SE must be installed as it consists of Java compiler as well as other requisite software which is used by J2ME wireless tool kit to build programs.
- The second step is to install J2ME wireless toolkit. Its possible to develop MIDP applications using J2ME wireless toolkit. J2ME wireless toolkit uses the concept of projects rather than files. Once J2ME wireless toolkit is installed and run, we can start creating projects. At the end of creation of projects, you have an MIDP suite. Once the project is created, its properties can be changed. Also, the project can be build and executed in the device emulator.

Creation of new project

- Let the title of project be project1. MIDlet class name be projectlet1.
- Once the names are given, the process of creating the project commences. At the end of creation, a directory titled project1 will be created in which the following subdirectories are there:
  - bin
  - lib
  - res
  - Src
- A compiled MIDlet suite(a .jar file) and the MIDlet suite directory(a .jad file) sre created in the bin directory. Extra JAR that are to be used in the project are placed in the lib directory. Any resource files like images are placed in the res directory. The source code created will be in src directory. The following subdirectories are also created. These are not used by the developer explicitly:
  - Tmpclasses
  - tmplib

- Once the MIDlet projectlet1 is developed, it should be saved as projectlet1.java file in the SRC directory. Build the project. After successful build, run it. On running, a mobile phone emulator will pop up. In the emulator, the title of the project will be displayed along with a button labelled Launch. On clicking it, the MIDlet developed will be executed and the result is displayed on the screen. There is a button labelled Exit at the bottom of the display on which the MIDlet is executed. If the Exit button is clicked, then we exit the MIDlet that is executed. There are two ways to exit the emulator. They are:
  - Emulator window can be closed
  - ESC key can be pressed
- There are a number of emulators provided by J2ME wireless toolkit. One of them can be selected from the KT tool bar. After selecting, we need to execute the project again.
- When the project is build, the toolkit will compile all the java files in the src directory. The compilation is performed in the MIDP environment. There were different APIs in the MIDP and J2SE environment. When the project is being compiled in the MIDP environment, then the APIs from MIDP should be considered for the classes that are used in MIDlets. All MIDP classes are verified in two phases before they are loaded and executed. At build time, the first verification is done by J2ME wireless toolkit. When classes are loaded, the second verification is done by device's runtime system. Finally, all MIDlets are packaged into MIDlet suites. These suites are distributed to actual devices.

- Along with the J2ME wireless toolkit, information about the following is provided:
  - Application development cycle
  - Attributes of MIDlet
  - Files in the installed directories
  - Device types
  - Portability
  - Guidelines on the process of configuring an emulator
  - Usage of J2ME wireless toolkit
- MIDlets can establish network connections using servlets.
- For developing servlets we need server. Tomcat is the server that can be used for this purpose.

- The first step is to install and run Tomcat. Latest version can be downloaded from the website Apache Jakarta Project. Tomcat is written in Java. Tomcat needs the location of J2SE to run. Hence, the location of J2SE should be set in the JAVAHOME environment variable. Once these steps are done, then open the command window. Change to the bin directory of Tomcat. Type startup to start running the Tomcat. To shutdown Tomcat, open the command window ,change to bin directory and type shutdown.

Developing a servlet

- Suppose we create a servlet named counter. Name of file will be "counter.java". It should be saved in the root directory of Tomcat. Once its saved, it can be compiled. The servlet can be compiled using javac.

  Eg: D:\javac counter.java

- Steps to connect MIDlet to a servlet :
  - Start Ktoolbar. This is a part of J2ME wireless toolkit.
  - Open the project1.
  - Write the code for MIDlet(countmidlet.java) that connects to counter servlet.
  - Screen of countmidlet.java has two commands: exit and connect. On click of connect the connect() is invoked and a network connection is established. It also transfers the result.
  - Countmidlet.java should be saved in apps/project1/src directory under J2ME wireless toolkit directory.
  - Now, J2ME wireless toolkit should know that a new MIDlet is added to it. This is done by going to Settings->MIDlets->Add. Enter countmidlet for both MIDlet name and class name. Click OK.
  - Go to Settings-> User Defined. Add the property name as countmidlet.URL. This URL will invoke counter servlet.

Java Device Test Suite

- Used to test the J2ME applications.
- Implementation of CLDC and MIDP can be evaluated using the test suite.
- They can also be validated and verified.
- Different types of tests that can be conducted using JDTS :
  - Functional tests
  - Stress tests
  - Performance tests
  - Security tests
- In function testing, the application is tested whether it behaves as indicated or not. Its behavior with both internal and external components are tested.
- In stress test, the application will be run under the upper limits of the memory and the processing power.
- In performance tests, response time etc. are evaluated.

- In security tests, the MIDlets are checked for their security model.
- In JDTS, there is a Test Console as well as test manager . This helps developers to test their J2ME applications.
- Features of JDTS :
  - More than one test suite can be executed at once.
  - Test Manager can accept connections from multiple test consoles concurrently.
  - Each test console might be testing several devices with the same configuration.
  - Test reports are generated in HTML format.