The ITB Journal

Volume 3 | Issue 2 Article 4

2002

# GlobalCom

Declan Barber

Conor Gildea

Gavin Byrne

# GlobalCom

**Declan Barber, Conor Gildea, Gavin Byrne**
**Institute of Technology Blanchardstown**

## Abstract

*The objective of this developed application is to provide a company with a means of communicating with its employees no matter where they are physically located and what communication resources they may have at any particular time. The core focus of this application is to provide a Unified Messaging System using synchronous and asynchronous forms.*

## Introduction

*Is there any application that currently provides this service?*

There are many systems which currently offer services in a particular area of communications. None of these however offer an all-inclusive communications system. It is this niche in the market that we hope to aim our product at. In the market place to date, service providers provide some of the functionality that our application has to offer but the associated costs are high. What is novel about this project is it provides an autonomous enterprise platform with toll-free access for intranet and optimised extranet costs.

*Why is this project a good idea and why is it useful?*

This project is aimed at a niche in the market which will mean that for the near future the product will have no direct competition. With the increase in popularity of the internet, more and more people are seeking out more cost effective ways of communicating with each other. The internet provides a fast, efficient, and inexpensive medium of communicating. People do not always have access to a computer terminal connected to the internet and with this application it will provide them with a means of communication over the internet using their mobile phones. Mobile phone networks have become so advanced in the last couple of years that they provide a reliable world-wide medium of communication. Another aspect of this

application is that it provides a GPS telemetry service. This feature will allow the end-user to remotely track their GPS beacons and process the resulting telemetry data.

**Typical Usage Scenarios:**

1. *An employee out of reach of a computer terminal receives a critically important e-mail. The employee may not have an opportunity to read this for several days.*

This application will give the employee a means of accessing their e-mail without the need for a computer, but by using their mobile phone.

2. *An employee is away from the office and his resources are limited to a laptop and a mobile phone. The employee needs to be able to carry out every day functions as if he were actually connected to his company's local network.*

This application will provide the user with the means to communicate with their company's local network over a number of different mediums.

3. *As part of a distribution company's business, it is highly important that they know were their deliveries are at all times. This may include an employee who needs to know the location of a certain cargo at any time, whether they are physically present at a computer on the companies campus, or out in the field with nothing but a mobile phone.*

As part of this communications package, their will be the ability to obtain the location of any of the company's resources, i.e. delivery trucks, world-wide shipments, employee's cars, etc through the use of either a computer terminal connected to the internet or on a lesser level a mobile phone.

4. *A network administrator is looking for a new product that will cut down the need for expensive resources such as proprietary devices such as PBX's (Public Branch Exchange), cabling for voice only communications, and the need for employees trained solely in telephony skills that are needed to run these.*

This application will cut out the need for a separation between voice and data on a network. With this application the administrator has the ability to treat voice data in the same manner as regular data across the network, and eliminates the need for dedicated telephony hardware, cabling or personnel.

5. *A company wishes their employees to have the ability to hold a conference call over a corporate local area network or the internet.*

This application will provide users with the ability to hold one-to-one or conference calls with any other user of the system whether they are physically connected to their company's local network or whether they are connected to the internet anywhere in the world.

6. *A company wishes to have the ability to send information between their employees securely no matter where they are physically located.*

This application will give users the ability to encrypt their communications securely and efficiently.

7. *A computer wishes to have a web-based message board where any of their employees world-wide can add messages and read messages.*

This application will provide all users access to a central message board, to which anyone can add messages and read messages.

8. *A company needs the ability to provide real-time text-based chat between their employees using their corporate local network or the internet.*

This application will provide all users access to a virtual chat room to which anyone can hold a text-based conversation.

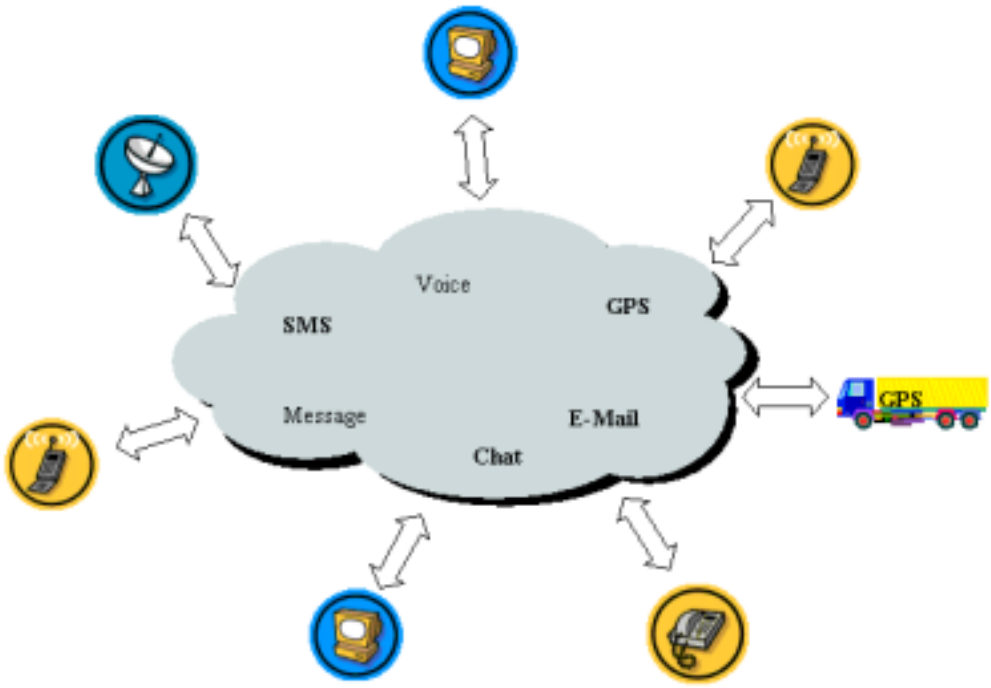
9. *A company wishes to have the ability to provide their employees with a secure company voice-mail system.*

As part of the application, there will be the option to send and receive secure voice-mails.

10. *A company wants the ability to convert easily from one medium to another providing them with a complete means of communications.*

The application will support the translation of communication mediums be it text or voice without the loss of information.

## Application Topology



This application will provide communication through the following services:

- Text-based: These systems will incorporate functions for communicating through some sort of text-based application.
- Mobile: These systems will provide the ability to communicate with the main application through the use of a mobile phone.
- Internet: These systems will be accessible through any web browser connected to the internet.
- Intranet: All of the systems services will be accessible through a company's local network.
- Computer Telephony: These systems will give users the ability to use all the services usually related with normal telephones, from their computer terminals.
- The Global Positioning System: Theses systems will provide geographical Information about specific company resources.

*Which aspects of JAVA technologies did we implement?*

For our application we took full advantage of the various packages that JAVA has to offer. It is only through these packages, and some we developed ourselves, that it was possible to develop our system to its full potential. These include:

- Java Servlets
- Java Server Pages(JSP)
- Java Communications API
- Java Remote Method Invocation(RMI)
- Java Media Framework(JMF)
- Java Telephony API

**Packages we developed during the course of our project:**

- JAVA PGP encryption facility
- SMS package
- GPS package

**How we used these technologies?**

**Java Servlets**

The main benefit of using Java Servlets is that they are server-centric meaning that any changes that are made are propagated to the connected clients. These clients that are

connected can be stripped down terminals we low processing power accessing the system using a web browser. All of the processing is carried out on the server end and the processed requested is sent back to the requesting clients terminal. Java Servlets were very useful in generating dynamic web pages and retrieving information from a database. Servlets were also used in the PGP encryption system, were a client-run applet could make a call to Servlets running on the application server to pass encrypted data back and forth. For the SMS system, the use of Servlets enabled us to provide all users logged onto the system, the ability to send and receive SMS messages through a mobile phone module which was connected to the server.

**JSP's**

This was used for the parts of the system where JAVA was only needed to generate dynamic web pages as oppose to Servlets which were used for the systems requiring heavy processing. The main difference between Servlets and JSP's is that with JSP's JAVA code is embedded in the HTML content were Servlet classes are called from HTML pages, do some kind of processing and then generate web pages based on this processing. The main part of our application which used JSP's is the multi-roomed Chat-room.

**JAVA Communications API**

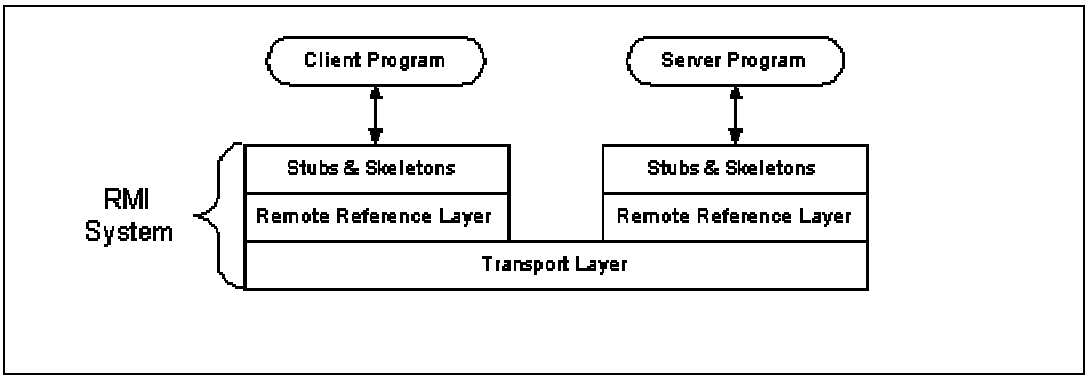
There are three levels of classes in the Java communications API: High-level classes like CommPortIdentifier and CommPort manage access and ownership of communication ports. Low-level classes like SerialPort and ParallelPort provide an interface to physical communications ports. The current release of the Java communications API enables access to serial (RS-232) and parallel (IEEE 1284) ports. Driver-level classes provide an interface between the low-level classes and the underlying operating system. Driver-level classes are part of the implementation but not the Java communications API. They should not be used by application programmers. This package provides an Enumeration of the available ports on the system. The static method `CommPortIdentifier.getPortIdentifiers` returns an enumeration object that contains a `CommPortIdentifier` object for each available port. This `CommPortIdentifier` object is the central mechanism for controlling access to a communications port, to resolve port ownership contention between multiple Java applications. Events are propagated to notify interested applications of ownership contention and allow the port's owner to relinquish ownership.

As an extension to this package we developed our own package that would handle all of the Serial IO communication with added respected to concurrent access to the Serial port. This consideration was very important, as the hardware would be connected to the server with the connecting clients competing for access to this device. The Serial IO can also be used when a telecommuter wishes to access the system remotely and there is no other form of communication other than the wireless mobile network. For the Serial IO package we decided to implement the singleton design pattern which prevents the class from being instantiated but all the method to be accessed directly and these methods are defined as been synchronised which means that once executed they have to finish and through the use of static variable it maintains that there is only ever one copy in memory. [1]

**Java Remote Method Invocation (RMI)**

RMI allows the code that defines the behaviour and the code that implements the behaviour to remain separate and to run on separate JVMs. This fits nicely with the needs of a distributed system where clients are concerned about the definition of a service and servers are focused on providing the service. Specifically, in RMI, the definition of a remote service is coded using a Java interface. The implementation of the remote service is coded in a class. Therefore, the key to understanding RMI is to remember that interfaces define behaviour and classes define implementation. With an understanding of the high-level RMI architecture, take a look under the covers to see its implementation. The RMI implementation is essentially built from three abstraction layers. The first is the Stub and Skeleton layer, which lies just beneath the view of the developer. This layer intercepts method calls made by the client to the interface reference variable and redirects these calls to a remote RMI service. The next layer is the Remote Reference Layer. This layer understands how to interpret and manage references made from clients to the remote service objects. The connection is a one-to-one (unicast) link. In the Java 2 SDK, this layer was enhanced to support the activation of dormant remote service objects via Remote Object Activation. The transport layer is based on TCP/IP connections between machines in a network. It provides basic connectivity, as well as some firewall penetration strategies. [2]

**RMI Breakdown**




**Java Media Framework (JMF)**

The Java Media Framework API (JMF) enables audio, video and other time-based media to be added to Java applications and applets. This optional package, which can capture, playback, stream and transcode multiple media formats, extends the multimedia capabilities on the JAVA platform, and gives the developers a powerful toolkit to develop scalable, cross-platform technology.

**Java Telephony API (JTAPI)**

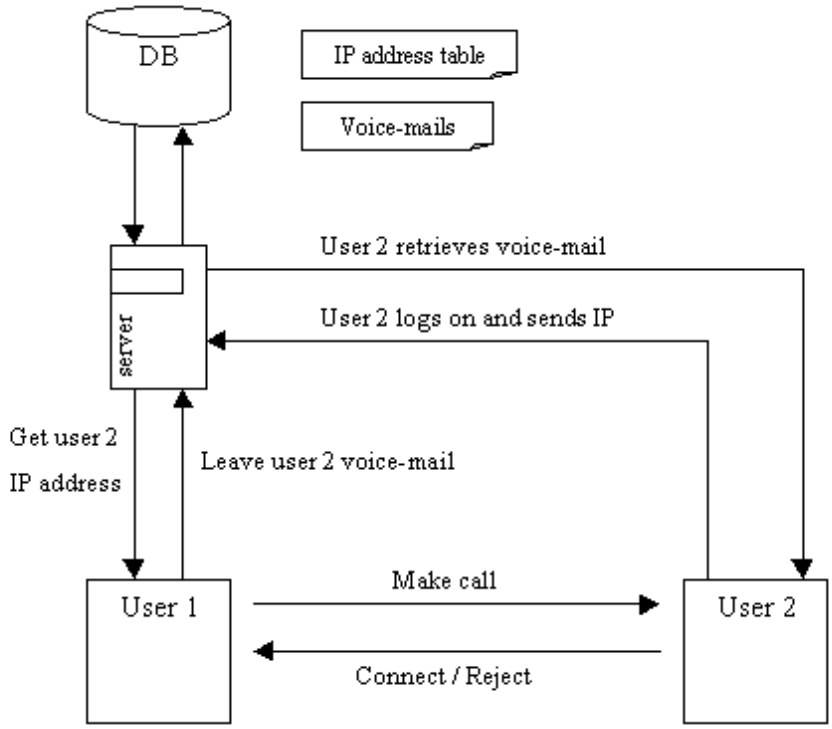
JTAPI is the set of classes, interfaces, and principles of operation that constitute a Java extension package in the *javax..** name space. JTAPI implementations are the interface between Java computer telephony applications and telephony services, whether those services are implemented as software, as in the case of a soft PBX, or hardware. JTAPI defines the access to one or more of the following areas of functionality: Call Control, Telephone Physical Device Control, Media Services for Telephony, and Administrative Services for Telephony.

The idea at the start of the project was to implement the telephony system as an applet which could be loaded up by the user from the web-page once they were logged in. However this proved more difficult than anticipated. Although the JTAPI model is described as being implemental as an applet, we found that any telephony program we developed needed access to certain system properties which are inaccessible to applets because of the use of the JAVA "sandbox" security model which runs on all JAVA Virtual Machines to stop programs downloaded from unknown sources causing malicious damage to the client's machine. We tried various techniques in trying to achieve this which included altering the client system's "java.policy" file to allow JAR files from our server's location all permissions to the client

machine (java.security.AllPermission). This was however only successful to the degree that we could run a JTAPI applet from appletviewer, which wasn't much good for the scope of the project. This is because appletviewer is not subject to the same security restrictions as applets loaded from a web browser. When this was unsuccessful, we tried to use signed JAR files to see if these would give our applet access to the required system properties. This however, was also unsuccessful.

Finally, we decided to implement a JTAPI application which would be downloadable from the server. This would run, but would only be fully functional once the user logged onto the main system. To do this we used JAVA RMI to pass encrypted messages between the client application and the main server where their information is stored. The IP address is retrieved by the server and used to access the client RMI application. This means that no matter were the user is logging in from, they can still access their missed/received/dialled calls history and any voice-mail they may have. Again the main idea behind this system is to have all components accessible from a web based e-mail style system incorporating folders for all systems-modules designed in a similar fashion. We also incorporated a secure telephone directory again using PGP encryption. All sending and receiving of audio data was controlled using the JMF. This toolkit provides multimedia features as described previously.

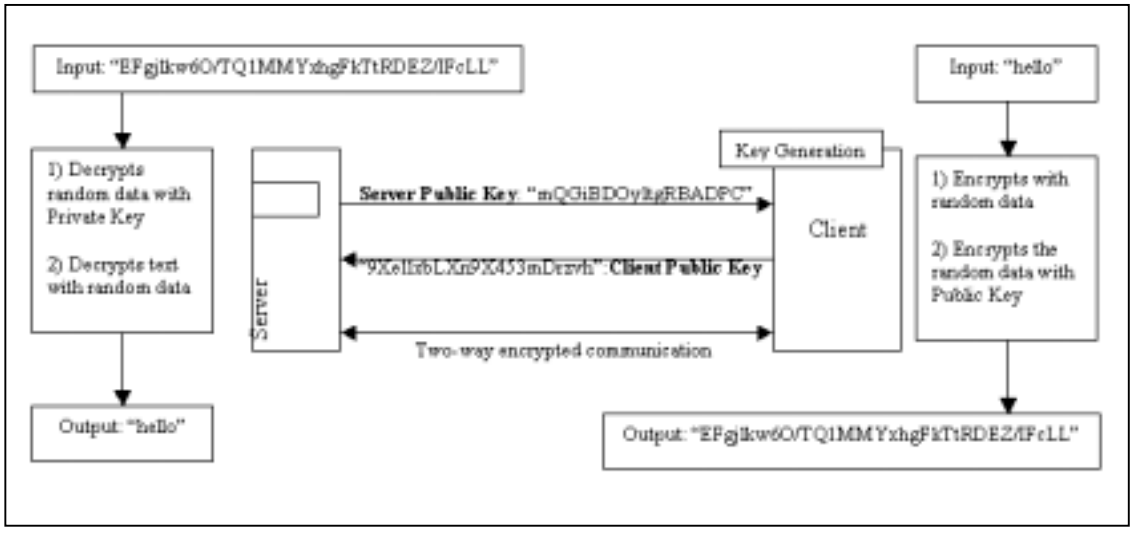
The basic Telephony Architecture:



## JAVA PGP encryption facility

We developed a package for the encrypting and decryption of text using Public/Private key pairs. We developed a mechanism for generating the random data needed for key generation, which took in an array of mouse coordinates from an applet window. This overcame the problem of computer generated random numbers, which are never really random. This means that a user can generate key pairs on a session basis, thus doing away with the need to remember any passphrase which is usually used to encrypt the private key for storage. The only key paires stored are on the server, where firewalls prevent access to the private key. Although the passphrase for this key would be needed for it to be of any good to anyone, nobody should get the chance to try and guess the passphrase. The only keys that go out on the wire are the public keys of the server and clients which are of no use to anyone as the private key cannot be obtained from this. This improves the security element of the system, as the only totally secure encryption system is one that you have wrote yourself. With PGP encryption, the longer the text which is to be encrypted is, the more efficient the encryption becomes. Random data is used to encrypt the text, and then the public key is used to encrypt this random data which generates what is called a "Key Block". This is sent along with the encrypted text where, at the other end, the private key is used to decrypt this key block giving back the random data which can then be used to decrypt the text. [1]

PGP Architecture:



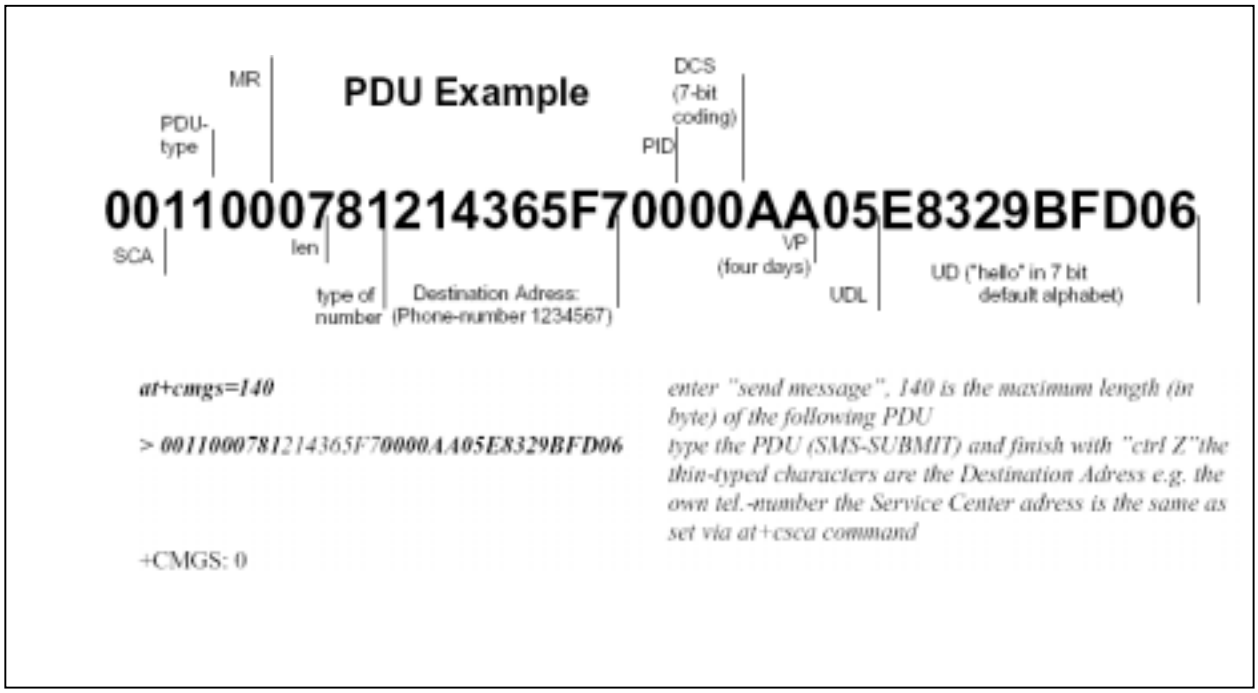

## SMS package

Short messaging is a low-cost transport used to communicate with mobile stations (e.g. cellular phones, pagers) across wireless networks such as GSM networks. The Short Message Service, or SMS, is a bi-directional service for short binary and alphanumeric messages (up

to 160 chars). Messages are transported in a store-and-forward fashion via a Short Message Service Centre (SMSC). Mobile stations (MSs) can send and received messages to and from other mobile stations. In a short messaging context, the mobile stations are known as SMEs or Short Message Entities.

This package that we developed provides a toolkit of methods that can be used to send and receive SMS messages. This package will extend from the Java Comm API which is used to send and receive the SMS Protocol Data Unit (PDU). This functionality is already previously mentioned.

The SMS package will build on the existing functionality of SMS and will make a number of alterations. One of the limitations of SMS is that you can only send up to 160 characters (7-bit). The SMS package improves this limitation through the use of a compression algorithm that will allow the user to send up to 380 characters. This is only available from (PC-PC) communication or for future JAVA enabled phones. The compression algorithm bases itself on a Huffman style dictionary compression method. Other functionality that the SMS package provides is the translation from "text" to "quicktext". For example "I will see you later" translates to "I will c u l8r". There is also the ability to send flash messages that are only suitable for certain phone models. Another aspect of the system is that it will remove the messages from your "SIM" and store them in a database for future viewing. The system will provide a mechanism of translating from one message media to another from example message board thread to SMS message. [3]



### GPS package

We have not got the GPS system working yet as we are still waiting to get the necessary hardware, i.e. GPS unit incorporating a GSM module. Once we have this, our aim is to provide users with access to positional tracking information about any vehicle(s) they wish to put one of these modules into. This will be done through data sent and received using data calls or SMS messages from these GPS modules to the GSM module connected to the server.

### Overall Application Architecture:

<table>
<tr><td colspan="4">Application Layer</td></tr>
<tr><td>Java Api's</td><td>Tomcat WebServer</td><td>JTAPI</td><td>ODBC</td></tr>
<tr><td colspan="4">Java Virtual Machine (JVM)</td></tr>
<tr><td colspan="2">Platform (OS)</td><td colspan="2">Telephony Platform (Peer)</td></tr>
</table>

## Conclusion

It was only through the use of JAVA as a technology was it possible for our application to reach its full potential. The computer world currently has many platforms, among them Microsoft Windows, Macintosh, OS/2, UNIX and NetWare; software must be compiled separately to run on each platform. The binary file for an application that runs on one platform cannot run on another platform, because the binary file is platform-specific. The Java Platform is a new software platform for delivering and running highly interactive, dynamic, and secure applets and applications on networked computer systems. But what sets the Java Platform apart is that it sits on top of these other platforms, and executes *bytecodes*, which are not specific to any physical machine, but are machine instructions for a *virtual machine*. A program written in the Java Language compiles to a bytecode file that can run wherever the Java Platform is present, on *any* underlying operating system. In other words, the same exact file can run on any operating system that is running the Java Platform. This portability is possible because at the core of the Java Platform is the Java Virtual Machine. It is this JVM that allows your application to run over heterogeneous system and allows for a "*Write Once, Run Anywhere*" capability.

## Links & References

[1] http://java.sun.com
[2] http://www.oreilly.com
[3] http://www.dreamfabric.com/sms