Making SMS Talk to J2ME™ Platform in the Real World

Shawn	Herb	Sam
Fitzgerald	**Vines III**	**Hacksisombath**
Author/Developer Ind. Consultant OnShoreSystems.com	Gamer/Developer Pier 1 Pier1.com	Developer Pier 1 Pier1.com

BOF-9941

BOF-9941 Session Objectives

Provide Information and facts on how to use SMS to communicate with Java ME™, within a real world setting.

Not just in a test lab.

Agenda

- **Fake world example & Tutorial Information**
  - Using WMA/PushRegistry API's
  - Testing code...in the fake world
- **Take the Red pill...Welcome to the real world**
  - What's missing from the examples
  - Writing J2EE™/J2SE™ code to talk to SMSC
- **SMSC, SMS Gateways and their protocols**
- **Working with Cell Providers**
  - Talk to the Trainman...He makes the rule here *(in the US)*
- **Using your new found knowledge**

Agenda

Fake world example & Tutorial Information

- Using WMA/PushRegistry API's
- Testing code...in the fake world

Take the Red pill...Welcome to the real world

- What's missing from the examples
- Writing J2EE™/J2SE™ code to talk to SMSC

SMSC, SMS Gateways and their protocols

Working with Cell Providers

- Talk to the Trainman...He makes the rule here *(in the US)*

Using your new found knowledge

WMA Code Example

```
 public class MessageMgr implements MessageListener, Runnable
{
  private Message inMsg;
  public MessageMgr() {
    try{   // setup to monitor port
        recCon=(messageConnection) Connection.open("sms://:5000”);
        recCon.setMessageListener(this);
    }catch(Exception e){ e.printStackTrace();}
  }

  public void notifyIncomingMessage(MessageConnection _inCon){
     Thread doMessage = new Thread(this);
     doMessage.start();
  }

  public void run(){ // load message to in message fields
        inMsg = recCon.receive();
  }

  public String getMsg(){
   return (inMsg != null?((TextMessage)inMsg).getPayloadText():null);
  }
}
```

PushRegistry Code Example

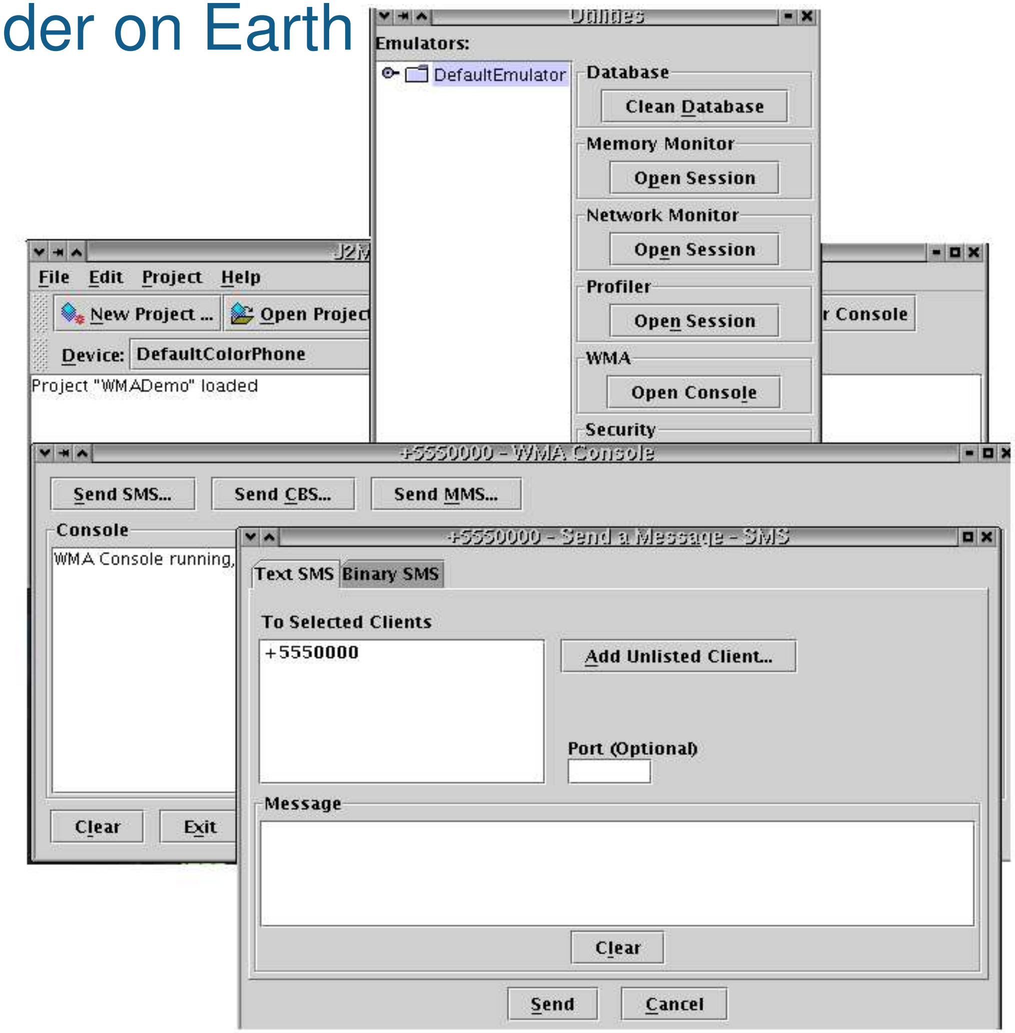
```
byte[] dataIn = null;
MessageConnection msgCon = null;
String[] conns = PushRegistry.listConnections(true);

if(connections == null || connections.length < 1){
         // not started by pushregistry
}else{
  try{
    try{
      msgCon = (MessageConnection)Connector.open(conns[0]);
      Message msg = msgCon.receive();
      if( msg instanceof TextMessage){
        dataIn = ((TextMessage)msg).getPayloadText().getBytes();
      }else{
        dataIn =((BinaryMessage)msg).getPayloadData();
      }
    }finally{
       if( msgCon != null) msgCon.close();
    }
  }catch(Exception e){
    e.printStackTrace();
  }
  // do something useful with dataIn
}
```

Testing with the WTK

The Friendliest Service Provider on Earth

- Start Phone Emulator
- WTK File->Utilities
- WMA (Open Console)
- Send SMS
- Select phone, port and text
- Send message



Device: DefaultColorPhone

Project "WMADemo" loaded

WMA Console running,

+5550000

Simple, unrealistic but simple

Sun's Wireless ToolKit 2.2 for linux

Example

Agenda

Fake world example & Tutorial Information

- Using WMA/PushRegistry API's
- Testing code...in the fake world

Take the Red pill...Welcome to the real world

- What's missing from the examples
- Writing J2EE™/J2SE™ code to talk to SMSC

SMSC, SMS Gateways and their protocols

Working with Cell Providers

- Talk to the Trainman...He makes the rule here *(in the US)*

Using your new found knowledge

What to look for?

Some Terminology

- UDH
  - User Defined Header – sets the protocol's port#
- SMSC
  - SMS Center – provider to adapt API calls to SMS
- SMS Gateway
  - What the SMSC talks through to pass SMS to a provider
- HTTP(S) Gateway
  - A format to call into an SMSC over the Net
- SMPP
  - Short Message Peer-to-Peer – Always connected SMS protocol
- ShortCodes
  - Short phone number to send server messages to.

Finding the missing parts

Google is of little help here

- Lists of SMS providers that send text
- Unclear terminology
- Confusion by the Cellproviders

If not Google then who could possibly help?

- MicroJava.com
  - WMA/PushRegistry(simplewire) article by Jason Edward Brown
- JavaRanch.com
  - A thread that I provide some of my initial findings
- SimpleWire.com, AQL.com, ClickATell.com
  - SMSC's that I have used (varying degrees of success)

What's Missing?

- Server code
  - User Defined Header (UDH) message settings
  - API's to Simulate SMSC access or SMPP
  - Impossible to do because there is no standards
    - SMSC (proprietary format)
    - SMPP (a standard but not available without contract)
- Easy access to SMS service for developers

Hacks and Workarounds

- Cellphone as SMS gateway
- Special Cell device (Nokia – M30 series)
- Issues: Slow response time & Low load support

J2EE™/J2SE™Code

```
 // create a provider object with needed API account number
Provider prov = new ClickATell("<yourID code>");
// Provider prov = new AQS2E();   // or this for AQL's provider object
// build SMS management object passing in the provider object to be used.
OssSMS sms = new OssSMS(prov);
// set message UserID and PassCode
sms.setSubscriberID("<your uid>");
sms.setSubscriberPassword("<yourpassword>");
/* Comment out next 2 lines to send a standard SMS message*/
sms.setSourcePort((short) 0x1388);
sms.setDestPort((short) 0x1388);
// set Message text and remember to pad the message
sms.setMsgText("test message"+"              ");
// set the phone number
sms.setMsgPin("<phone number>");
sms.submit();  // post message
// print out response from SMSC.
System.out.println("response\n"+sms.getResponse());
```

Footnote: Excerpt from “Making IT work – SMS for MIDP2.0”   www.cafepress.com/sms_midp2/

J2EE™/J2SE™Code

```
public boolean submit() {
   responseBuf = null;
   try {
      String postMsg = provider.buildPostMsg();
      URLConnection uConn = getUrlConnection(provider.getUrl());
      PrintStream postOut = null;
      try {
         postOut = new PrintStream( uConn.getOutputStream());
         postOut.println(postMsg);
         responseBuf =  getResponse(uConn);
         // parse and check message for validity
         // if okay continue else indicate failed submit.
       } finally  {
         if( postOut != null)
            postOut.close();
      }
   }catch(Exception e) {
      return false;
   }
   return true;
}
```

Footnote: Excerpt from “Making IT work – SMS for MIDP2.0”   www.cafepress.com/sms_midp2/

J2EE™/J2SE™Code

```
 public String buildPostMsg( ) throws Exception  {
   if( phone == null || phone.length() < 1  )   // invalid phone number is a show stopper
      throw new Exception("invalid phone");

   if(text == null || text.length() < 1 )        // invalid text is also a show stopper.
      throw new Exception("invalid SMS text");

   // format the correct post values
   StringBuffer outMsg = new StringBuffer("username=");
   outMsg.append(licenseKey).append("&password=").append(authKey);
   outMsg.append("&destination=").append(phone);

   if( from != null)     // ignore optional values if not provided
      outMsg.append("&originator=").append(from);

    if( useUdh ) { // ignore optional values if not provided
      outMsg.append("&udh=060504").append(new String(udhPort));
      outMsg.append("&data=").append(encodeMsg(text.getBytes()));  // encode text as binary
    } else  // encode text as text using URL encoding.
       outMsg.append("&data=").append(URLEncoder.encode(text));

   return outMsg.toString();    // return POST request messages
 }
```

Footnote: Excerpt from “Making IT work – SMS for MIDP2.0”   www.cafepress.com/sms_midp2/

J2EE™/J2SE™Code

```
public void setDestPort(short _port)
{
    useUdh = true;
    byte[] hexPort = ((String)Integer.toHexString(_port)).getBytes();
    for(int i=7, max=hexPort.length-1; i >=4; i--, max--)
    {
        if( max < 0)
            udhPort[i] = 0;
        else
            udhPort[i] = hexPort[max];
    }
}

 /** used to encode the message into hex, different than URLEncoding or UUencoding */
private String encodeMsg( byte[] _data)
{
    StringBuffer codedData = new StringBuffer(_data.length+_data.length+5);
    for(int i=0; i< _data.length; i++)
        codedData.append(Integer.toHexString(_data[i]));
    return codedData.toString();
}
```

Footnote: Excerpt from “Making IT work – SMS for MIDP2.0”   www.cafepress.com/sms_midp2/

Agenda

Fake world example & Tutorial Information

- Using WMA/PushRegistry API's
- Testing code...in the fake world

Take the Red pill...Welcome to the real world

- What's missing from the examples
- Writing J2EE™/J2SE™ code to talk to SMSC

SMSC, SMS Gateways and their protocols

Working with Cell Providers

- Talk to the Trainman...He makes the rule here *(in the US)*

Using your new found knowledge

What the output looks like

HTTP format Examples

- Click-A-Tell

  http://api.clickatell.com/http/sendmsg?session_id=e74dee1bbed22ee3a39f9aeab606ccf9&to=1234567890&udh=06050415810000&text=024A3A5585E19.....

- AQL (http Post)

  username=<aql username>&password=<password>&destination=<dest#(ie 447740.., country code+area+prefix+num)>&originator=<not necessary>&udh=0605040B8423F0&data=01060403AE81EA020....

- !!Remember Safety!!

  Post your important data, or better yet use HTTPS

SMSC

- SMSC is a 3rd party service
    - Collect fees on each message sent or received
    - Different CellProviders cost more
    - Different Regions cost more
- One Way Service
    - MT – Mobile Terminated (terminates at the mobile device)
    - 120 – 160 char limit
    - Sent from server/app to SMSC to Cellprovider
- Two Way Service
    - MT & MO – Mobile Originated (sent from the phone back to SMSC)
    - Same type of limits as MT
    - Charged for both messages (Premium SMS charges back to user)
- Some CellProviders limit the number of channels in use
    - Voice OR HTTP, however Voice AND SMS = COOL.

SMS Gateway

- The gateway is the port to the other side.
  - From TCP/IP HTTP ---> to SMS
- Must be connected to the Service provider (ie. SprintPCS)
- All Providers networked together (well mostly)
  - Nextel does not use true SMS, but a wap app
  - A SMS message from one provider is transferred to other carriers.
    - May not be correct format....
- Rolling your own
  - Can be done using software like SMSj, or Kannel
  - Connecting Cellphones off of serial ports
  - Fine for small load, or testing, but not realistic for production
- A Linux link for more info:
  - http://tuxmobil.org/phones_linux_sms.html

Agenda

Fake world example & Tutorial Information

- Using WMA/PushRegistry API's
- Testing code...in the fake world

Take the Red pill...Welcome to the real world

- What's missing from the examples
- Writing J2EE™/J2SE™ code to talk to SMSC

SMSC, SMS Gateways and their protocols

Working with Cell Providers

- Talk to the Trainman...He makes the rule here *(in the US)*

Using your new found knowledge

CellService Providers (US Centric View)

- Talk to the Trainman...He makes the rule here.. (in the US)
  - When you think you understand how to dodge bullets, the rules change, and all that you think you know is void.
  - The CellProviders control:
    - Root CA support (X.509 Certs)
    - Device API support
    - Lines/types of simultaneous communication
    - No standard between Providers
  - Indy/Small shops tend not to be able to afford service contracts
    - Monthly fees for ShortCode rental
    - Fees to ShortCode authority
    - Minimum monthly usage (plus overage penalty)
- Anyone from Outside the US? Comments?

Agenda

- **Fake world example & Tutorial Information**
  - Using WMA/PushRegistry API's
  - Testing code...in the fake world
- **Take the Red pill...Welcome to the real world**
  - What's missing from the examples
  - Writing J2EE™/J2SE™ code to talk to SMSC
- **SMSC, SMS Gateways and their protocols**
- **Working with Cell Providers**
  - Talk to the Trainman...He makes the rule here *(in the US)*
- **Using your new found knowledge**

Why bother

- Sometimes it is the only path from point A to B
  - Datagrams or Sockets must have fixed IP to work
  - Server initiated communication (must be Static not DHCP)
- Less costly than data plans. (for the user)
  - Unlimited Text messaging (in and/or out bound)
  - Data plans based on bytes, socket keep-alive big $$
  - SMS a fixed price per message (Note: SMS 120-160 chars)
- Your servers can be off the net
  - No more script kiddies
  - No spybots, zombie bots, or mis-keyed URLs

Where to use SMS

Some basic ideas

- Simple Chat Application
- Traffic Alert Application
- Trivia Game

- Ways to implement messaging
  - Device to Device
  - (One-way) Server to Device
  - (Two-way) Device to Server then back.

Simple Chat Application

- Communication Model
  - Device to Device
- Things to understand
  - No server needed
  - Potential cost for MO and MT messages
  - May pass from one provider to another
  - Only need to know phone number

Traffic Alert Application

- Communication Model
  - (One-way) Server to Device
- Things to understand
  - Device must be known.
  - Delivery time of message can verify based on QOS.
  - MT message based.

Trivia Game

- Communication Model
  - (Two-way) Device to Server and Back
- Things to understand
  - Server must be known
  - Delivery time of messages can verify based on QOS
  - Short codes must be used for MO messages
  - Message character limits must be considered

The Cost of Messaging

- Who gets charged?
  - Developer pays for short codes
  - Developer pays for MO's or MT's
  - Developer pays for SMSC service if needed.
  - User may pay for MO's or MT's based on plan
  - Users are subject to charges from built in developer message charges.

Summary

The key take-away should be that it's not easy, but it can have great benefits to your project.

Know the provider scope. *(remember Nextel is different)*

Understand security restrictions with signed app, and APIs

Keep number of connection small to limit user overages

For More Information

- Articles
  - Real World Experiences with the WMA and the Push Registry ***by*** Jason Edward Brown
    - http://www.microjava.com/articles/techtalk/WMA20
  - The MIDP 2.0 Push Registry ***by*** Enrique Ortiz
    - http://developers.sun.com/techtopics/mobility/midp/articles/pushreg/
- Books
  - Making IT work: SMS for MIDP2.0 ***by*** Shawn Fitzgerald
    - www.cafepress.com/sms_midp2/
- WebSite
  - OnShoreSystems.com/smsapi/

Q&A

iSERVERS.COM

Pier 1 imports®
Life more interesting.

Making SMS Talk to J2ME™ Platform in the Real World

Shawn
Fitzgerald
Author/Developer
Ind. Consultant
OnShoreSystems.com

Herb
Vines III
Gamer/Developer
Pier 1
Pier1.com

Sam
Hacksisombath
Developer
Pier 1
Pier1.com

BOF-9941