



BREW 2005
conference

Telephony & SMS Interfaces

Praveen Yaramada
Engineer, Staff

Rajeev Singh
Engineer, Staff/Manager

QUALCOMM Incorporated

June 3, 2005



Telephony APIs

- **ITAPI**

- **Supported limited telephony functionality**

- Originate voice calls
 - Access to limited phone state information
 - Access to limited call state information

- **Second Generation Telephony APIs**

- **Extend wireless capability of device to BREW[®] applications (ability to implement a complete dialer application on BREW)**
 - **Ensure finer access restrictions**
 - **Uniform and Extendable API for existing & emerging underlying technologies (GSM, CDMA, 1x, EVDO, UMTS, WLAN)**



Telephony Interfaces

- **ITelephone**
 - Read access to phone information
 - AEECLSID_TELEPHONE also implements IModel
- **ICallMgr**
 - Interface to create & manage calls
- **ICall**
 - Represents a communication link
 - Implements IModel for event notifications
- **IPhoneNotifier**
 - Is a INotifier. To register & receive telephony events
- **IPhoneCtl**
 - Allows application to alter the phone behavior
- **ISuppsTrans**
 - Represents a supplementary service transaction



Phone and Serving System Information

- **AEETPh Info**

- Current phone state and capability
 - Operating Mode, Preferred Mode, Band preference, NAM selection, hybrid operation preference (eg : 1x and EvDO), Subscription, etc.
- ITELEPHONE_GetPhoneInfo()

- **AEETSS Info**

- Provides Serving Network Information (Capability, System Mode (physical layer protocols, Identity, etc.)
- Received Signal Strength
- ITELEPHONE_GetServingSystemInfo()

- **Asynchronous Notifications**

- AEET_NMASK_PHONE, AEET_NMASK_SS
- Add listener on ITelephone



Adding Listener

- **IModel enables asynchronous notifications for memory resident applications <AEEModel.h>**
- **Installing a listener on ITelephone**

```
LISTENER_Init(&piApp->lisPhone, My_PhoneListener,  
piApp);
```

```
if (SUCCESS == ITELEPHONE_QueryInterface(  
    piApp->piTelephone,  
    AEEIID_MODEL,  
    (void*)&piModel)) {  
    IMODEL_AddListener(piModel, &piApp->lisPhone);  
    IMODEL_Release(piModel);};
```

- **Uninstall listener**

```
LISTENER_Cancel(&piApp->lisPhone);
```



Telephone Call

- **Two Primitives to Represent a Phone Call**
 - **Call descriptor**
 - An unsigned number that uniquely identifies an active call known to the system
 - **ICall**
 - An object encapsulation of an active or terminated call
- **Call States:**
 - **Active call: A phone call known to the system & identified by call descriptor**
 - `AEET_CALL_STATE_ORIG`, `AEET_CALL_STATE_INCOM`, `AEET_CALL_STATE_CONV`, `AEET_CALL_STATE_ONHOLD`, `AEET_CALL_STATE_DORMANT`
 - **Terminated call: A phone call which is no longer tracked by system but exists only as an object**
 - `AEET_CALL_STATE_IDLE`, `AEET_CALL_STATE_ENDED`



Call Information

- **Synchronous retrieval of AEETCallInfo**
 - ITELEPHONE_GetCallInfo()
 - ICALL_GetInfo()
- **Partial**
 - ICALL_GetType()
 - ICALL_GetState()
- **Asynchronous notifications with AEETCallEventData**
 - Register based on call type
 - AEET_NMASK_VOICE_CALL, AEET_NMASK_DATA_CALL, AEET_NMASK_TEST_CALL, AEET_NMASK_OTHER_CALL
 - Register for new calls originated or incoming
 - AEET_NMASK_NEW_CALLDESC
 - Add listener on ICall instance



Originate Call

- **ICALLMGR_Originate**

- Originates a requested call type to a given address
- Returns an ICall instance
- Installs a listener

- **ICALLMGR_OriginateEx**

- Varargs using ICallOrigOpts
- Service type, service option, calling line Id specification, called number sub address, etc.

- **State Transition**

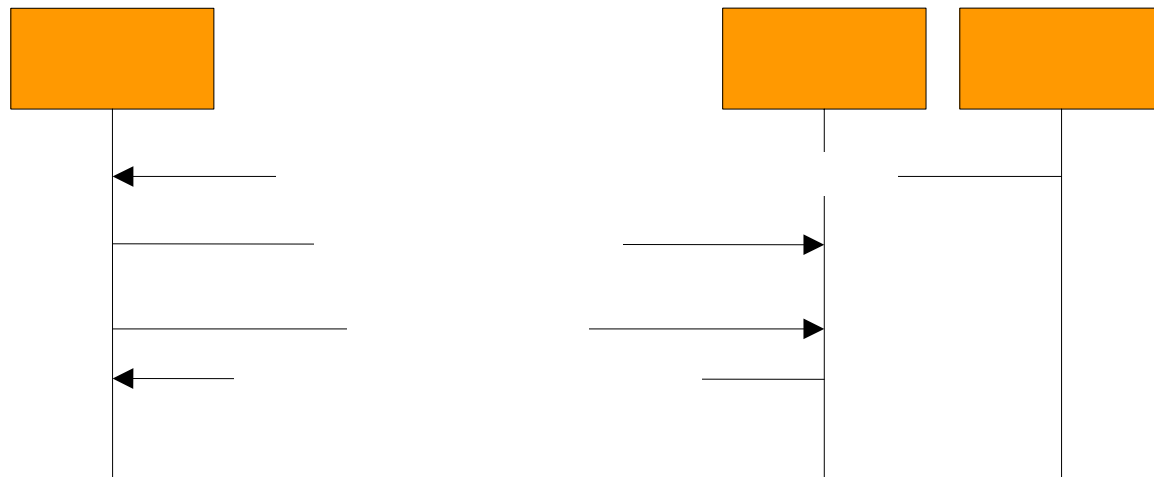
AEET_CALL_STATE_ORIG to AEET_CALL_STATE_CONV



Answer Call

- **Notification for incoming call**
 - AEET_NMASK_NEW_CALLDESC
 - Known as call descriptor
- **ICALLMGR_GetCall()**
 - ICall object wrapped on a call descriptor
- **ICALL_Answer()**

AEET_CALL_STATE_INCOM to AEET_CALL_STATE_CONV.



IPhoneCtl

- **Privileged Class**
- **Change Received Signal Strength Delta**
 - Affects notifications of RSSI
- **Change Operating Mode**
 - Offline, online, low power mode, reset
- **Change System Preferences**
 - Acquisition order, band preferences, roaming preferences, etc.
- **Change NAM selection, Active Line Selection**
- **Perform Call Related Supplementary Service Action**
- **Create a New Supplementary Service Transaction (ISuppsTrans)**



Supplementary Service Transaction

- **Starting a New Transaction**

- Use ID AEET_SUPPSVC_TRANSID_NEW
- Create ISuppsTrans from IPHONECTL_GetSuppsTrans()

- **Network Initiated Transaction**

- AEET_NMASK_NEW_NI_SUPPS_TRANS
- EVT_NOTIFY data contains transaction id
- Create ISuppsTrans from IPHONECTL_GetSuppsTrans()

- **Using ISuppsTrans**

- ISUPPSTRANS_LoadMessage() to load a AEETSuppsMessage
- ISUPPSTRANS_Run() asynchronous callback when done
- ISUPPSTRANS_GetResult() to retrieve response



SMS Interfaces

- Introduction
- SMS Support in BREW Client
- SMS Interfaces
- BREW directed SMS
- EMS
- WAP
- SMS Encodings



SMS Support in BREW Client

- **ITAPI Introduced in BREW Client 1.0.1 to Enable Applications Use Telephony & SMS**
 - **Telephony: Make calls & get device or call related telephony information**
 - **SMS: Receive SMS**
 - **BREW directed SMS**
 - **Mobile Terminated SMS**
- **ITAPI_SendSMS() Introduced in BREW Client 1.x**
 - **Enabled send SMS**
- **ITAPI Reference Implementation Enhanced to Cover GSM/WCDMA Networks in BREW Client 2.x**

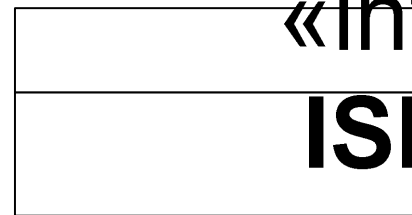
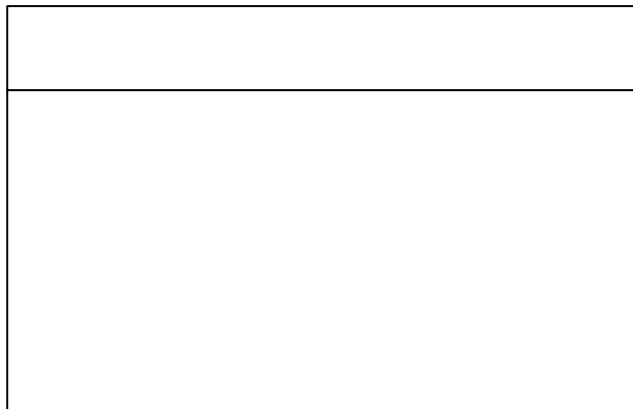
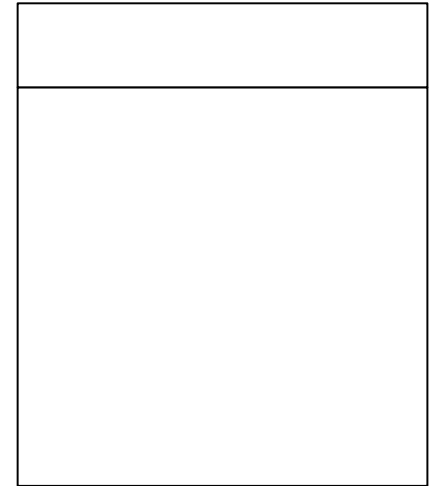
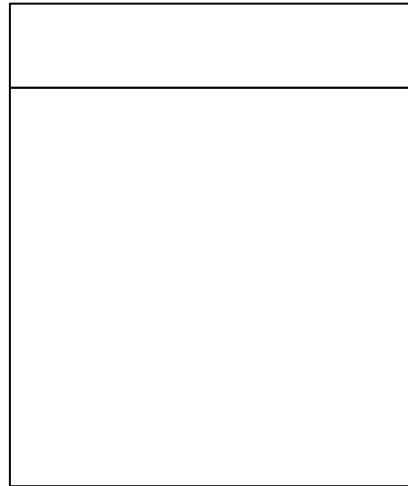
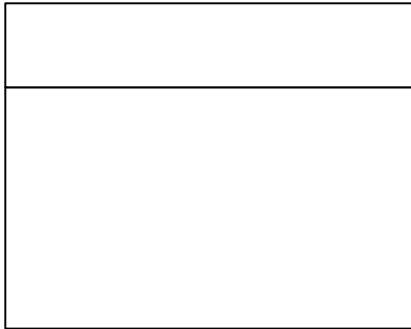


SMS Support in BREW Client

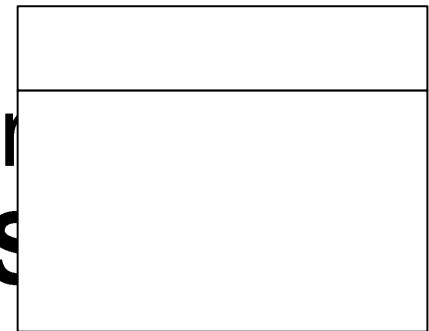
- **ISMSNotifier, ISMS, ISMSMsg, ISMSStorage, ISMSBCCConfig and ISMSBCSrvOpts Interfaces Introduced in BREW Client 3.1.x**
 - Refer BREW SDK® release notes for information about specific release supporting an interface
- **ITAPI SMS stuff re-implemented on SMS Interfaces on BREW Client Releases with SMS Interfaces**



SMS Interfaces



«interface»
ISMS



+AddRef() : unsigned long
+Release() : unsigned long
+QueryInterface() : int



ISMSNotifier

- **Inherits from INotifier**
- **Provides Event Notification for Mobile Terminated Messages**
- **Applications Register for Notifications on Arrival of SMS Message of the Specified Type**
 - **AEESMS_TYPE_TEXT**
 - **AEESMS_TYPE_PAGE**
 - **AEESMS_TYPE_VOICE_MAIL_NOTIFICATION**
 - **AEESMS_TYPE_BROADCAST**
 - **AEESMS_TYPE_WAP**
 - **AEESMS_TYPE_EMS**
- **Applications Notified by EVT_NOTIFY Event and Provided a 32 bit ID which Applications can use to Access Message**



ISMSMsg

- **Inherits from IWebOpts**
- **Provides an Abstract Representation of SMS message**
- **Stateless Interface which can be viewed as “Hash Dictionary” which lets user add, get & remove name Value Pairs**
- **Provides Air Interface Independent Representation of SMS message**
- **Provides Ability to add new Parameters without Compromising backward Compatibility**
- **Refer AEESMS.h for details on Options Supported**



ISMS

- **Provides Access to send & receive SMS**
- **Enables Applications calculate bytes available for Payload**
- **Enables Applications find encoding Types allowed for Mobile Originated SMS**
- **Enables Device Messaging Application Control how ISMS should acknowledge mobile terminated Messages**
- **Refer AEESMS.h for more Details on Methods**

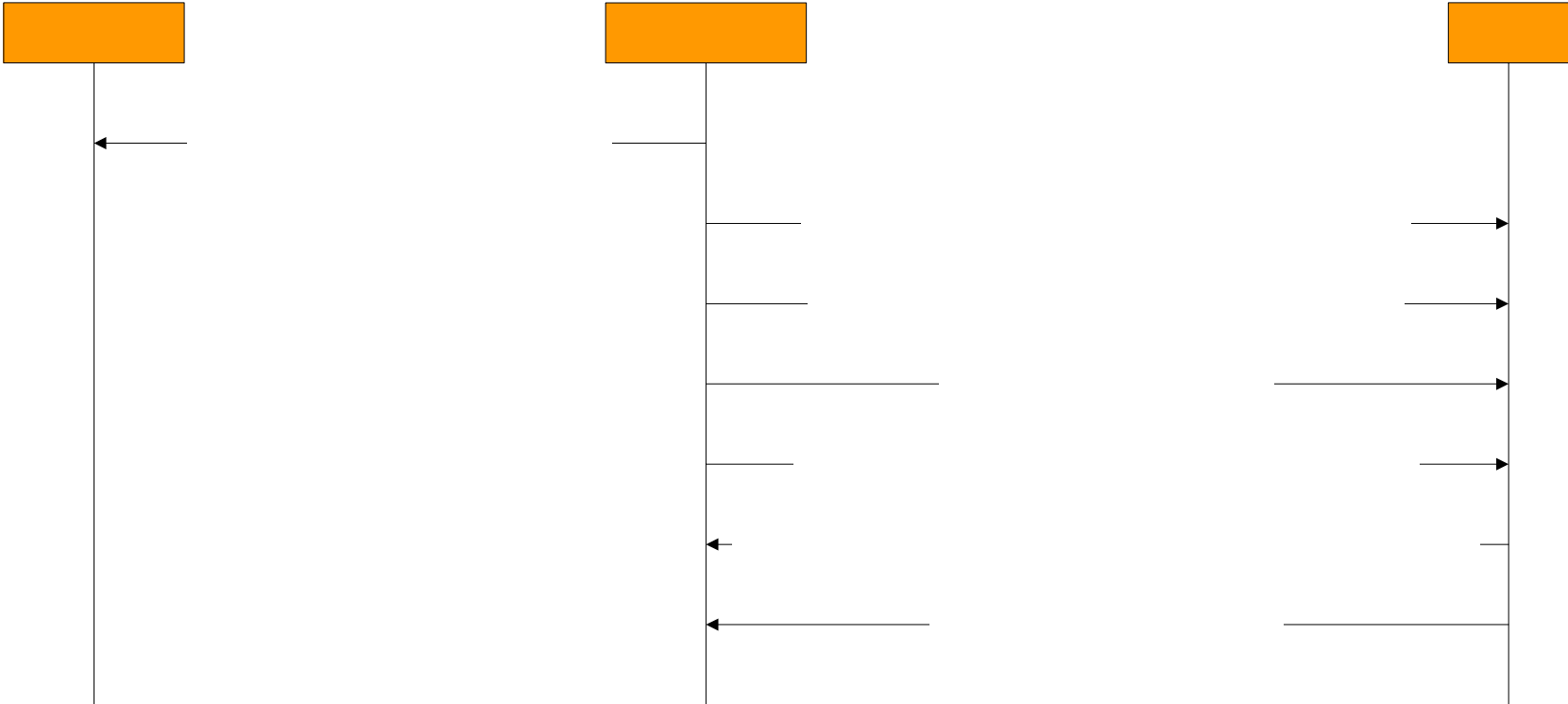


ISMS Model Listener

- **IModel enables Applications listen to ISMS Events**
- **ISMS sends EVT_MDL_SMS_MSG_SENT on successful sending of the Message**
- **EVT_MDL_SMS_MSG_SENT enables Applications Access Message ID, Message Alpha ID & other Information associated with the message**

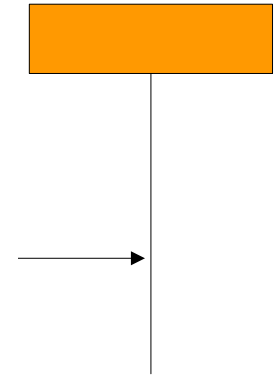
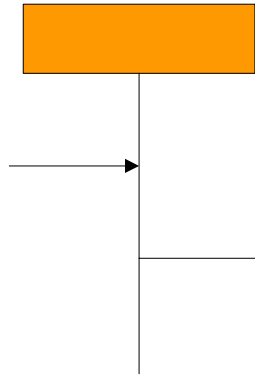
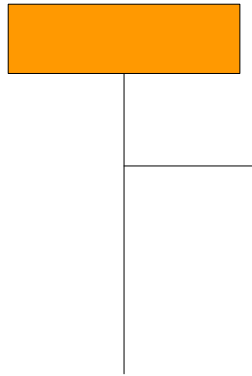


Sending a Message



ISMSMsg

Receiving a Message



ISMSNotifier

ISMSStorage

- **Supports Storage to RUIIM, SIM, NV CDMA & NV GW**
 - RUIIM & NV CDMA are 1x specific
 - SIM & NV GW are GSM/WCDMA specific
- **Provides Access to store & read SMS**
- **Provides Access to delete SMS**
- **Enables Applications enumerate Messages of a Specific Type on the Specified Storage Type**
- **Refer AEESMS.h for more Details on Methods**



ISMSStorage Model Listener

- IModel enables Applications listen to ISMSStorage events
- ISMSStorage sends EVT_MDL_SMSSTORAGE_STORE on successful storing of the Message
- ISMSStorage sends EVT_MDL_SMSSTORAGE_UPDATE on successful update of the stored Message

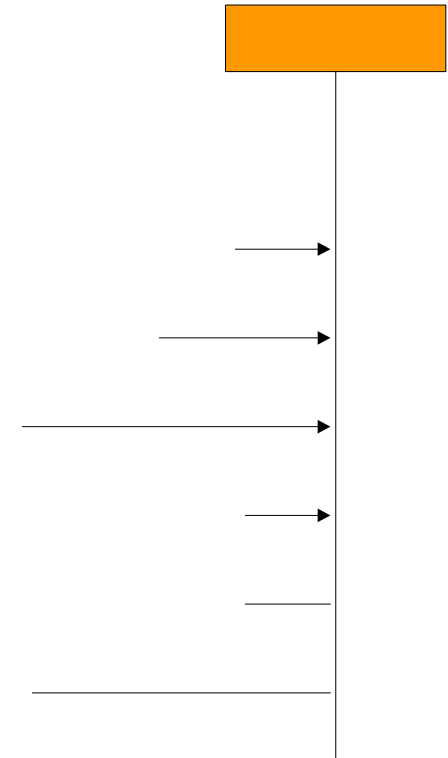
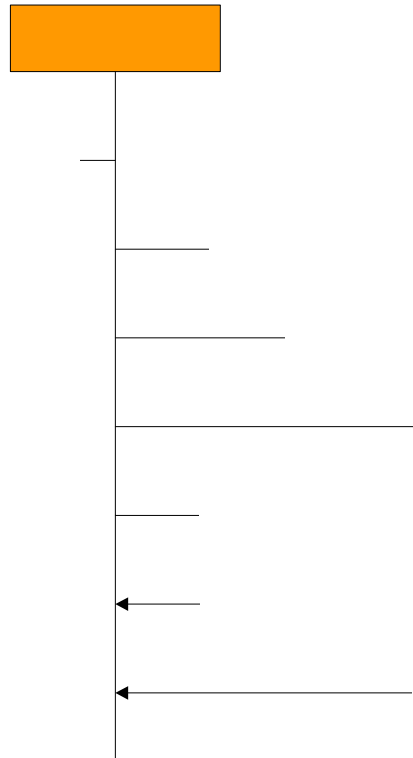
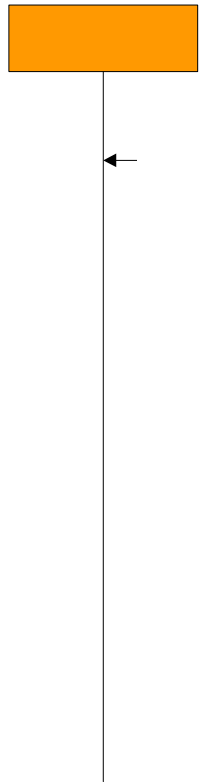


ISMSStorage Model Listener

- **ISMSStorage sends EVT_MDL_SMSSTORAGE_DELETE on successful deletion of the Message**
- **ISMSStorage sends EVT_MDL_SMSSTORAGE_DELETE_TAG on successful deletion of the messages of an specified Tag**
- **ISMSStorage sends EVT_MDL_SMSSTORAGE_DELETE_ALL on successful deletion of all of the Messages**



Storing a message



ISMSMsg

ISMSBCSrvOpts

- **Inherits from IWebOpts**
- **Provides an Abstract Representation of an SMS Broadcast Service**
- **Stateless Interface which can be viewed as “Hash Dictionary” which lets user add, get and remove name Value Pairs**
- **Refer AEESMS.h for Details on Options supported**



ISMSBCConfig

- **Enables Applications Access Broadcast Configuration**
- **Enables Applications Access & Specify Broadcast Preferences**
- **Enables Applications specify Priority Threshold, Alert Type and other Parameters associated with user Preference about an Specific Type of Broadcast SMS messages**
- **Refer AEESMS.h for more Details on Methods**



BREW Directed SMS

- **EVT_APP_MESSAGE Behavior is retained for Backward Compatibility**
- **EVT_APP_MESSAGE_EX added to provide 32 bit message ID to destination application**
- **Destination application can get full message by using ISMS_ReceiveMsg() with message ID**
- **BREW handler directed Behavior is retained for backward Compatibility**



EMS

- **Enhanced Messaging Service**
- **Enables insertion of picture, sound, text formatting information, etc in an SMS message**
- **Programmer's view – SMS messages with user data headers**
- **ISMSMsg enables add, get & remove on user data headers by MSGOPT_UDH_XXX options**
- **Use MSGOPT_SETUDHXXX() macros to set user data header buffer**
- **Use MSGOPT_GETUDHXXX() macros to get user data header & user data header length**



WAP

- **Specific type of EMS message or SMS with binary payload in specific format**
- **When sent as EMS, contains AEE SMS_UDH_PORT_16 with value 2948**
- **When sent as EMS, also contains AEE SMS_UDH_CONCAT_8 if message is concatenation of multiple messages**
- **When sent as SMS with binary payload, port & concatenation information is embedded in payload**



WAP Support in BREW Client

- ITAPI supports **NMASK_TAPI_WAP_PUSH** notification mask & provides WAP push messages as large as concatenation of the user data of 3 SMS messages
- Applications registered with **NMASK_TAPI_WAP_PUSH** mask are notified with **EVT_NOTIFY** & provided user data as **AEESMSTextMsg** structure
- **ISMSNotifier** supports notification for **AEESMS_TYPE_WAP** messages and provides the message ID for accessing the message



SMS Encodings

- Applications can specify message payload with **MSGOPT_PAYLOAD_SZ**, **MSGOPT_PAYLOAD_WSZ** & **MSGOPT_PAYLOAD_BINARY**
- **MSGOPT_PAYLOAD_XXX** specifies a byte buffer & means to find length of byte buffer
- **MSGOPT_PAYLOAD_ENCODING** tells how to interpret payload byte buffer
- **MSGOPT_MOSMS_ENCODING** tells what encoding to use to send message to network
- ISMS does conversion from **MSGOPT_PAYLOAD_ENCODING** to **MSGOPT_MOSMS_ENCODING** & gives error on failure

