Interactive SMS

Plain HTTP Connection (MO Outbound)

8/18/2011
v1.1

Table of Contents

Scope

This document has been designed for ACTEL’s clients.

Whereby a client is any content owner wishing to use ACTEL’s MO SMS services via HTTP to collect messages or income from his end-users.

Introduction

This document explains the messaging integration points between Actel's Large Scale Competition systems and mobile messaging clients. It goes into the details of system integration topology, connection protocols, Actel's requirements and recommendation, and step by step guidelines for safe and successful integration.

Description

Connectivity to mobile messaging clients will be established by using Actel's Carrier grade messaging gateway. The clients will provide Actel with access to their system so that MO and MT messages may be exchanged. The following diagram describes how the connection flow is arranged:

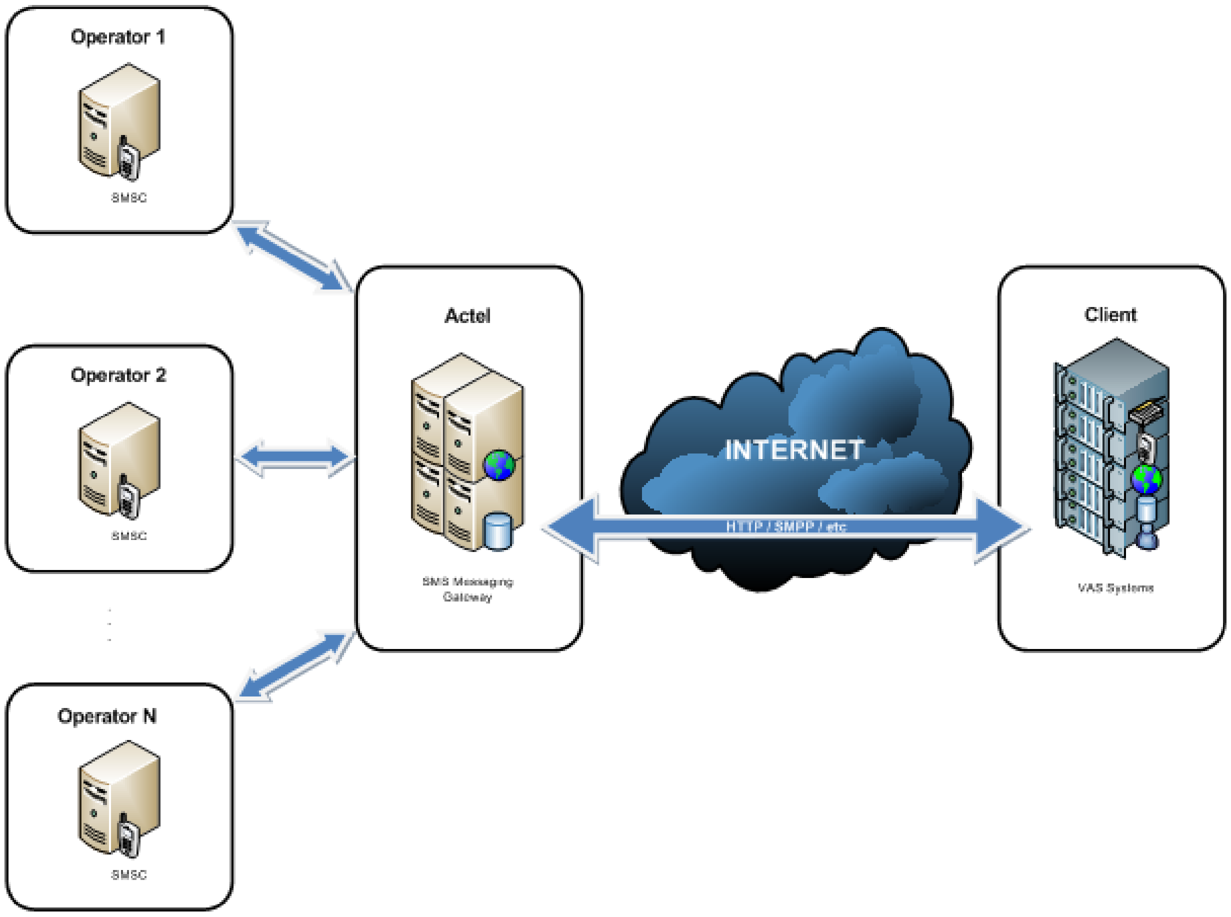

Figure 1 - Connection flow between Actel & the Client

Message Delivery

The connection between a client and Actel will be established over the Internet. The two entities will communicate using a single protocol.

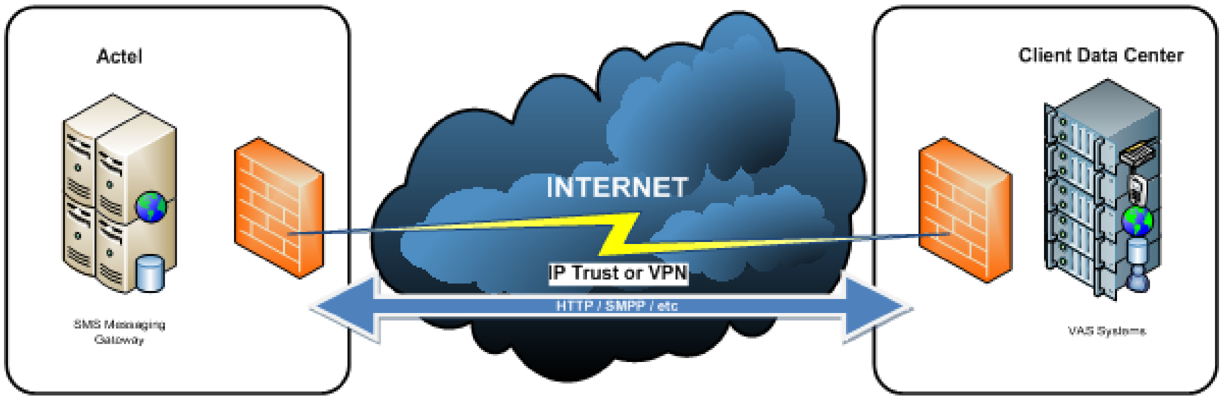

Figure 2 - Connection flow between Actel & the Client

Actel has defined a HTTP protocol that it recommends clients to use. Details may be found in the Annex 2 of this document. Actel is able to support tailored HTTP, SMPP messaging protocols if required but there may be an extra deployment cost involved.

Occasionally the client may not have sufficient infrastructure to handle large MO queues in their gateways. A large scale competition as run by Actel may generate a significant MO traffic to the promotion's short code. If such traffic is in progress and network connectivity between Actel's and the client's data center fails for a few hours, Actel will be able to queue the MO messages until the connection gets reestablished. Queued messages will be then submitted by Actel.

Outbound Connection

We define the Outbound MO Connection as the stream of MO message bound from ACTEL to its clients.

In Outbound MO Connections, ACTEL is the Sender (reporter).

For these types of connections, we issue recommendations rather than strict and ridged API specifications. We require the evaluation and presence of certain key variable but we won't be enforcing their names or orders. We leave it up to the client to determine the latter based on his system's needs.

Protocol and Parameters

Protocol	**HTTP**
URL	http://IP/getsms.aspx

HTTP Parameters		**Type**
signature	MD5 signature **refer to the Annex 3**	Alphanumeric String – Upper case letters
destination	Short code	Integer
smssender	Phone number	Integer
smstext	Body of the message (Plain text for English , UCS2 for Arabic)	Alphanumeric String
smsid	Unique ID for each Request.	Alphanumeric String
idlang	0 for UCS2 (Arabic messages) , 1 for ACSII	Integer
opid	Operator ID **refer to the Annex 1**	Integer

Return values

Returned Values	Description	Type
Invalid signature	Signature is missing or incorrect	Alphabetic String
Invalid destination	Destination is missing	Alphabetic String
Invalid smssender	SMS Sender is missing	Alphabetic String
Invalid idlang	ID Lang is missing or incorrect	Alphabetic String
Invalid opid	OPID is missing or incorrect	Alphabetic String
Invalid SMSID	SMS ID is missing	Alphabetic String
Invalid Request duplicates	duplicate (for a second duplicate request)	Alphabetic String
Invalid Request Error & "Error Description"	Unknown error	Alphabetic String
Ok	Request is received Successfully	Alphabetic String

Samples

Below are the acceptable request formats/structures including all valid HTTP parameters:

1) Valid format including all HTTP parameters for a Text request:

Text Request	http://IP/getsms.aspx?signature=test&destination=1081&smssender=9613793475&idlang=1&opid=2&SMSID=123&smstext=test%20message

2) Valid format including all HTTP parameters for a Unicode request:

Unicode Request	http://IP/getsms.aspx?signature=test&destination=1081&smssender=9613793475&idlang=0&opid=2&SMSID=123&smstext=066106620663

ANNEXES

Annex 1 – Operator IDs

opid	Operator Name	Country Name
1	Alfa	Lebanon
2	MTCTouch	Lebanon
3	Djezzy	Algeria
4	batelco	Bahrain
5	Zain	Bahrain
6	Mobinil	Egypt
7	Vodafone	Egypt
8	asiacell	Iraq
9	Iraqna	Iraq
10	Zain	Iraq
11	Zain	Jordan
12	Orange	Jordan
13	Umniah	Jordan
14	Xpress	Jordan
15	MOBILY	KSA
16	STC	KSA
17	Zain	kuwait
18	wataniya	kuwait
19	MAROCTEL	Morocco
20	MEDITEL	Morocco
21	Nawras	Oman
22	omanMobile	Oman
23	Jawal	Palestine
24	qtel	Qatar
25	Zain	Sudan
26	AREEBA	Syria
27	MTN	Syria
28	syriatel	Syria
29	Tunisiana	Tunisia
30	Etisalat	UAE
31	sabafon	yemen
32	MTN	yemen
33	yemenMobile	yemen
34	Etisalat	Egypt
41	DU	UAE
42	korectel	Iraq
43	Sanatel	Iraq
44	Smartcom	International
48	Tunistelecom	Tunisia

49	MTN	Sudan
50	Mobilis	Algeria
51	Nedjma	Algeria
52	Wana	Morocco
53	Libyana	Libya
54	elmadar	Libya
55	Etisaluna	Iraq
56	Sudatel	Sudan
57	Y	yemen
83	Zain	KSA
84	Jordan Telecom	Jordan
85	viva	kuwait
86	ZainIQ	Iraq
87	OmanTel	Oman
88	Zain	Palestine
92	Vodafone	Qatar
93	Wataniya	Palestine

Annex 2 – Glossary

Aggregators/mobile messaging aggregators: Entities that provide access to mobile operators. Often referred to as service providers, an aggregator connects to the mobile operators and though one connection to its clients, aggregates access to multiple operators.

Gateway/messaging gateway: This is a system that handles receiving and sending messages between other messaging entities. They provide the single point of access and implement the notions of queuing, retry failed messages and implementation of required communication protocols.

Large scale competition: Large scale competitions, often referred to as super promotions, are services that Actel specializes in. Large scale competitions are by definitions services accessible by a large user base and generate big amounts of traffic.

VAS Promotion Systems: The system that implement the large scale competitions. VAS (Value Added Service) Promotion Systems refers to the application layer that Actel implements.

MO: Mobile Originating messages are messages sent by the user.

MT: Mobile Terminating messages are messages sent by Actel to the user.

Dialogue/MO-MT Dialogue: Represents a text based SMS dialogue that Actel's Systems holds with a user. A user sends and MO, Actel answers with an MT triggering the user to send another MO, thus establishing a dialogue with the user.

Base Bulk: Broadcasts to subscribers of the operators who have not yet joined the promotion.

Teaser Bulks: Broadcasts to subscribers who have joined the promotion and are being invited to play some more.

MO Inbound: The MO traffic that is sent to ACTEL by any peer.

MO Outbound: The MO Traffic that is sent from ACTEL to any other peer.

MT Inbound: The MT Traffic that is received by ACTEL from any peer.

MT Outbound: The MT traffic that is sent by ACTEL to any peer.

Submission, Hit, Transmission: These terms designate a single HTTP PDU sent from or to ACTEL

Annex 3 – MD5 encryption function

DISCLAIMER: Please be advised that the provided code samples are provided for your convenience. Use them at your own risk.

VB.Net

```
Public Function MD5Encrypt(ByVal EncString As String) As String
        'Variable Declarations
        Dim MD5String As String
        Dim EncStringBytes() As Byte
        Dim Encoder As New UTF8Encoding
        Dim MD5Hasher As New MD5CryptoServiceProvider
        'Converts the String to bytes
        EncStringBytes = Encoder.GetBytes(EncString)
        'Generates the MD5 Byte Array
        EncStringBytes = MD5Hasher.ComputeHash(EncStringBytes)
        'Create MD5 hash
        MD5String = BitConverter.ToString(EncStringBytes)
        MD5String = MD5String.Replace("-", "")
        'Returns the MD5 encrypted string to the calling object
        Return MD5String
    End Function
```

Usage

```
Signature=MD5Encrypt("ClientPassword@" & smssender)
```

PHP

```
$Signature= md5("clientpassword@".$smssender);
```

Java

```
import java.security.MessageDigest;

public class Main {
    public static void main(String[] args) {
        try{
            MessageDigest digest = java.security.MessageDigest.getInstance("MD5");
            //Put in here the String that you want to has its MD5 sig
            String sig = new String("actel");
            digest.update(sig.getBytes());
            byte[] hash = digest.digest();
            String result = "";
            for (int i=0; i < hash.length; i++) {
                result += Integer.toString( ( hash[i] & 0xff ) + 0x100, 16).substring( 1 );
            }
            System.out.println(result);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

VB 6.0

```
Private Const BITS_TO_A_BYTE = 8
Private Const BYTES_TO_A_WORD = 4
Private Const BITS_TO_A_WORD = 32

Private m_lOnBits(30)
Private m_l2Power(30)

   m_lOnBits(0) = CLng(1)
   m_lOnBits(1) = CLng(3)
   m_lOnBits(2) = CLng(7)
   m_lOnBits(3) = CLng(15)
   m_lOnBits(4) = CLng(31)
   m_lOnBits(5) = CLng(63)
   m_lOnBits(6) = CLng(127)
   m_lOnBits(7) = CLng(255)
   m_lOnBits(8) = CLng(511)
   m_lOnBits(9) = CLng(1023)
   m_lOnBits(10) = CLng(2047)
   m_lOnBits(11) = CLng(4095)
   m_lOnBits(12) = CLng(8191)
   m_lOnBits(13) = CLng(16383)
   m_lOnBits(14) = CLng(32767)
   m_lOnBits(15) = CLng(65535)
   m_lOnBits(16) = CLng(131071)
   m_lOnBits(17) = CLng(262143)
   m_lOnBits(18) = CLng(524287)
   m_lOnBits(19) = CLng(1048575)
   m_lOnBits(20) = CLng(2097151)
   m_lOnBits(21) = CLng(4194303)
   m_lOnBits(22) = CLng(8388607)
   m_lOnBits(23) = CLng(16777215)
   m_lOnBits(24) = CLng(33554431)
   m_lOnBits(25) = CLng(67108863)
   m_lOnBits(26) = CLng(134217727)
   m_lOnBits(27) = CLng(268435455)
   m_lOnBits(28) = CLng(536870911)
   m_lOnBits(29) = CLng(1073741823)
   m_lOnBits(30) = CLng(2147483647)

   m_l2Power(0) = CLng(1)
   m_l2Power(1) = CLng(2)
   m_l2Power(2) = CLng(4)
   m_l2Power(3) = CLng(8)
   m_l2Power(4) = CLng(16)
   m_l2Power(5) = CLng(32)
   m_l2Power(6) = CLng(64)
   m_l2Power(7) = CLng(128)
```

```
  m_l2Power(8) = CLng(256)
  m_l2Power(9) = CLng(512)
  m_l2Power(10) = CLng(1024)
  m_l2Power(11) = CLng(2048)
  m_l2Power(12) = CLng(4096)
  m_l2Power(13) = CLng(8192)
  m_l2Power(14) = CLng(16384)
  m_l2Power(15) = CLng(32768)
  m_l2Power(16) = CLng(65536)
  m_l2Power(17) = CLng(131072)
  m_l2Power(18) = CLng(262144)
  m_l2Power(19) = CLng(524288)
  m_l2Power(20) = CLng(1048576)
  m_l2Power(21) = CLng(2097152)
  m_l2Power(22) = CLng(4194304)
  m_l2Power(23) = CLng(8388608)
  m_l2Power(24) = CLng(16777216)
  m_l2Power(25) = CLng(33554432)
  m_l2Power(26) = CLng(67108864)
  m_l2Power(27) = CLng(134217728)
  m_l2Power(28) = CLng(268435456)
  m_l2Power(29) = CLng(536870912)
  m_l2Power(30) = CLng(1073741824)

Private Function LShift(lValue, iShiftBits)
  If iShiftBits = 0 Then
    LShift = lValue
    Exit Function
  ElseIf iShiftBits = 31 Then
    If lValue And 1 Then
      LShift = &H80000000
    Else
      LShift = 0
    End If
    Exit Function
  ElseIf iShiftBits < 0 Or iShiftBits > 31 Then
    Err.Raise 6
  End If

  If (lValue And m_l2Power(31 - iShiftBits)) Then
    LShift = ((lValue And m_lOnBits(31 - (iShiftBits + 1))) * m_l2Power(iShiftBits)) Or &H80000000
  Else
    LShift = ((lValue And m_lOnBits(31 - iShiftBits)) * m_l2Power(iShiftBits))
  End If
End Function

Private Function RShift(lValue, iShiftBits)
  If iShiftBits = 0 Then
```

```
        RShift = lValue
        Exit Function
    ElseIf iShiftBits = 31 Then
        If lValue And &H80000000 Then
            RShift = 1
        Else
            RShift = 0
        End If
        Exit Function
    ElseIf iShiftBits < 0 Or iShiftBits > 31 Then
        Err.Raise 6
    End If

    RShift = (lValue And &H7FFFFFFE) \ m_l2Power(iShiftBits)

    If (lValue And &H80000000) Then
        RShift = (RShift Or (&H40000000 \ m_l2Power(iShiftBits - 1)))
    End If
End Function

Private Function RotateLeft(lValue, iShiftBits)
    RotateLeft = LShift(lValue, iShiftBits) Or RShift(lValue, (32 - iShiftBits))
End Function

Private Function AddUnsigned(lX, lY)
    Dim lX4
    Dim lY4
    Dim lX8
    Dim lY8
    Dim lResult

    lX8 = lX And &H80000000
    lY8 = lY And &H80000000
    lX4 = lX And &H40000000
    lY4 = lY And &H40000000

    lResult = (lX And &H3FFFFFFF) + (lY And &H3FFFFFFF)

    If lX4 And lY4 Then
        lResult = lResult Xor &H80000000 Xor lX8 Xor lY8
    ElseIf lX4 Or lY4 Then
        If lResult And &H40000000 Then
            lResult = lResult Xor &HC0000000 Xor lX8 Xor lY8
        Else
            lResult = lResult Xor &H40000000 Xor lX8 Xor lY8
        End If
    Else
        lResult = lResult Xor lX8 Xor lY8
    End If

    AddUnsigned = lResult
End Function

Private Function F(x, y, z)
    F = (x And y) Or ((Not x) And z)
End Function

Private Function G(x, y, z)
    G = (x And z) Or (y And (Not z))
End Function

Private Function H(x, y, z)
    H = (x Xor y Xor z)
End Function

Private Function I(x, y, z)
    I = (y Xor (x Or (Not z)))
End Function

Private Sub FF(a, b, c, d, x, s, ac)
    a = AddUnsigned(a, AddUnsigned(AddUnsigned(F(b, c, d), x), ac))
    a = RotateLeft(a, s)
    a = AddUnsigned(a, b)
End Sub

Private Sub GG(a, b, c, d, x, s, ac)
    a = AddUnsigned(a, AddUnsigned(AddUnsigned(G(b, c, d), x), ac))
    a = RotateLeft(a, s)
    a = AddUnsigned(a, b)
End Sub

Private Sub HH(a, b, c, d, x, s, ac)
    a = AddUnsigned(a, AddUnsigned(AddUnsigned(H(b, c, d), x), ac))
    a = RotateLeft(a, s)
    a = AddUnsigned(a, b)
End Sub

Private Sub II(a, b, c, d, x, s, ac)
    a = AddUnsigned(a, AddUnsigned(AddUnsigned(I(b, c, d), x), ac))
    a = RotateLeft(a, s)
    a = AddUnsigned(a, b)
End Sub

Private Function ConvertToWordArray(sMessage)
    Dim lMessageLength
    Dim lNumberOfWords
    Dim lWordArray()
    Dim lBytePosition
    Dim lByteCount
    Dim lWordCount

    Const MODULUS_BITS = 512
    Const CONGRUENT_BITS = 448

    lMessageLength = Len(sMessage)

    lNumberOfWords = (((lMessageLength + ((MODULUS_BITS - CONGRUENT_BITS) \ BITS_TO_A_BYTE)) \
(MODULUS_BITS \ BITS_TO_A_BYTE)) + 1) * (MODULUS_BITS \ BITS_TO_A_WORD)
    ReDim lWordArray(lNumberOfWords - 1)

    lBytePosition = 0
    lByteCount = 0
    Do Until lByteCount >= lMessageLength
        lWordCount = lByteCount \ BYTES_TO_A_WORD
        lBytePosition = (lByteCount Mod BYTES_TO_A_WORD) * BITS_TO_A_BYTE
        lWordArray(lWordCount) = lWordArray(lWordCount) Or LShift(Asc(Mid(sMessage, lByteCount + 1,
1)), lBytePosition)
        lByteCount = lByteCount + 1
    Loop

    lWordCount = lByteCount \ BYTES_TO_A_WORD
    lBytePosition = (lByteCount Mod BYTES_TO_A_WORD) * BITS_TO_A_BYTE

    lWordArray(lWordCount) = lWordArray(lWordCount) Or LShift(&H80, lBytePosition)

    lWordArray(lNumberOfWords - 2) = LShift(lMessageLength, 3)
    lWordArray(lNumberOfWords - 1) = RShift(lMessageLength, 29)

    ConvertToWordArray = lWordArray
End Function

Private Function WordToHex(lValue)
    Dim lByte
    Dim lCount

    For lCount = 0 To 3
        lByte = RShift(lValue, lCount * BITS_TO_A_BYTE) And m_lOnBits(BITS_TO_A_BYTE - 1)
        WordToHex = WordToHex & Right("0" & Hex(lByte), 2)
    Next
End Function

Public Function MD5(sMessage)
    Dim x
    Dim k
    Dim AA
    Dim BB
    Dim CC
    Dim DD
    Dim a
    Dim b
    Dim c
    Dim d

    Const S11 = 7
    Const S12 = 12
    Const S13 = 17
    Const S14 = 22
    Const S21 = 5
    Const S22 = 9
    Const S23 = 14
    Const S24 = 20
    Const S31 = 4
    Const S32 = 11
    Const S33 = 16
    Const S34 = 23
    Const S41 = 6
    Const S42 = 10
    Const S43 = 15
    Const S44 = 21

    x = ConvertToWordArray(sMessage)

    a = &H67452301
    b = &HEFCDAB89
    c = &H98BADCFE
    d = &H10325476

    For k = 0 To UBound(x) Step 16
        AA = a
        BB = b
        CC = c
        DD = d

        FF a, b, c, d, x(k + 0), S11, &HD76AA478
        FF d, a, b, c, x(k + 1), S12, &HE8C7B756
        FF c, d, a, b, x(k + 2), S13, &H242070DB
        FF b, c, d, a, x(k + 3), S14, &HC1BDCEEE
        FF a, b, c, d, x(k + 4), S11, &HF57C0FAF
        FF d, a, b, c, x(k + 5), S12, &H4787C62A
        FF c, d, a, b, x(k + 6), S13, &HA8304613
        FF b, c, d, a, x(k + 7), S14, &HFD469501
        FF a, b, c, d, x(k + 8), S11, &H698098D8
        FF d, a, b, c, x(k + 9), S12, &H8B44F7AF
        FF c, d, a, b, x(k + 10), S13, &HFFFF5BB1
        FF b, c, d, a, x(k + 11), S14, &H895CD7BE
        FF a, b, c, d, x(k + 12), S11, &H6B901122
        FF d, a, b, c, x(k + 13), S12, &HFD987193
        FF c, d, a, b, x(k + 14), S13, &HA679438E
        FF b, c, d, a, x(k + 15), S14, &H49B40821

        GG a, b, c, d, x(k + 1), S21, &HF61E2562
        GG d, a, b, c, x(k + 6), S22, &HC040B340
        GG c, d, a, b, x(k + 11), S23, &H265E5A51
        GG b, c, d, a, x(k + 0), S24, &HE9B6C7AA
        GG a, b, c, d, x(k + 5), S21, &HD62F105D
        GG d, a, b, c, x(k + 10), S22, &H2441453
        GG c, d, a, b, x(k + 15), S23, &HD8A1E681
        GG b, c, d, a, x(k + 4), S24, &HE7D3FBC8
        GG a, b, c, d, x(k + 9), S21, &H21E1CDE6
        GG d, a, b, c, x(k + 14), S22, &HC33707D6
        GG c, d, a, b, x(k + 3), S23, &HF4D50D87
        GG b, c, d, a, x(k + 8), S24, &H455A14ED
        GG a, b, c, d, x(k + 13), S21, &HA9E3E905
        GG d, a, b, c, x(k + 2), S22, &HFCEFA3F8
        GG c, d, a, b, x(k + 7), S23, &H676F02D9
        GG b, c, d, a, x(k + 12), S24, &H8D2A4C8A

        HH a, b, c, d, x(k + 5), S31, &HFFFA3942
        HH d, a, b, c, x(k + 8), S32, &H8771F681
        HH c, d, a, b, x(k + 11), S33, &H6D9D6122
        HH b, c, d, a, x(k + 14), S34, &HFDE5380C
        HH a, b, c, d, x(k + 1), S31, &HA4BEEA44
        HH d, a, b, c, x(k + 4), S32, &H4BDECFA9
        HH c, d, a, b, x(k + 7), S33, &HF6BB4B60
        HH b, c, d, a, x(k + 10), S34, &HBEBFBC70
        HH a, b, c, d, x(k + 13), S31, &H289B7EC6
        HH d, a, b, c, x(k + 0), S32, &HEAA127FA
        HH c, d, a, b, x(k + 3), S33, &HD4EF3085
        HH b, c, d, a, x(k + 6), S34, &H4881D05
        HH a, b, c, d, x(k + 9), S31, &HD9D4D039
        HH d, a, b, c, x(k + 12), S32, &HE6DB99E5
        HH c, d, a, b, x(k + 15), S33, &H1FA27CF8
        HH b, c, d, a, x(k + 2), S34, &HC4AC5665

        II a, b, c, d, x(k + 0), S41, &HF4292244
        II d, a, b, c, x(k + 7), S42, &H432AFF97
        II c, d, a, b, x(k + 14), S43, &HAB9423A7
        II b, c, d, a, x(k + 5), S44, &HFC93A039
        II a, b, c, d, x(k + 12), S41, &H655B59C3
        II d, a, b, c, x(k + 3), S42, &H8F0CCC92
```

```
        II c, d, a, b, x(k + 10), S43, &HFFEFF47D
        II b, c, d, a, x(k + 1), S44, &H85845DD1
        II a, b, c, d, x(k + 8), S41, &H6FA87E4F
        II d, a, b, c, x(k + 15), S42, &HFE2CE6E0
        II c, d, a, b, x(k + 6), S43, &HA3014314
        II b, c, d, a, x(k + 13), S44, &H4E0811A1
        II a, b, c, d, x(k + 4), S41, &HF7537E82
        II d, a, b, c, x(k + 11), S42, &HBD3AF235
        II c, d, a, b, x(k + 2), S43, &H2AD7D2BB
        II b, c, d, a, x(k + 9), S44, &HEB86D391

        a = AddUnsigned(a, AA)
        b = AddUnsigned(b, BB)
        c = AddUnsigned(c, CC)
        d = AddUnsigned(d, DD)
    Next

    MD5 = UCase(WordToHex(a) & WordToHex(b) & WordToHex(c) & WordToHex(d))
End Function
```

Annex 4 – Convert from UCS2 to Arabic (or UTF8) function

VB Script and VB.net :

```
  Function Arabize(ByVal text As String) As String
        Try
            Dim thetext, char1, arabictext As String
            arabictext = ""
            thetext = text
            For i As Integer = 1 To Len(thetext) / 4
                char1 = Left(thetext, 4)
                arabictext = arabictext & ChrW(CInt("&H" & char1))
                thetext = Right(thetext, Len(thetext) - 4)
            Next
            Arabize = arabictext
        Catch ex As Exception
            Arabize = " invalid message"
        End Try
    End Function
```

PHP 5

```
header('Content-Type: text/html; charset=UTF-8');
mb_http_output('UTF-8');
echo '<html><head>';
echo '<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />';
echo '</head><body>';

$UCS2 = "06230643062A064A0644002006470064A00200627064406230642064806 49";
$ucs2string = pack("H*" , $_REQUEST['ucs2']);
//$ucs2string = pack("H*" , $UCS2);
$utf8string = mb_convert_encoding($ucs2string , 'UTF-8', 'UCS-2');
echo 'UTF8: '.$utf8string.'<br />';

echo '</body></html>';
```

Annex 5 – Convert from Arabic (or UTF8) to UCS2 function

VB Script and VB.net:

```
  Public Function UniEnc(ByVal text As String) As String

        Dim i, j, count_char As Integer, nchar, hexval, arabicunicode As
String
        arabicunicode = ""
        For i = 1 To Len(text)
            count_char = count_char + 1
            nchar = Left(text, 1)
            hexval = Hex(AscW(nchar))
            If Len(hexval) < 4 Then
                For j = 1 To 4 - Len(hexval)
                    hexval = "0" & hexval
                Next
            End If
            arabicunicode = arabicunicode & hexval
            text = Right(text, Len(text) - 1)
        Next
        UniEnc = arabicunicode
    End Function
```

PHP 5

```
header('Content-Type: text/html; charset=UTF-8');
mb_http_output('UTF-8');
echo '<html><head>';
echo '<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />';
echo '</head><body>';

$bodyar = "الأقوى هي أكتيل";
$arr = unpack('H*hex', @iconv('UTF-8', 'UCS-2BE', $bodyar));
echo "UCS2 : ".$arr['hex'];
echo "<br><br>";

echo '</body></html>';
```