APÉNDICE 1: CÓDIGO DE LA CLASE NODO

```
Option Explicit
Option Base 1

Public Nombre As String
Dim Nnodos
Dim Conexiones() As String
Dim Distancias() As String


Sub AgregaNodos(ByVal arr As Variant) 'arr debe de ser un arreglo
Dim i As Integer

ReDim Conexiones(1 To UBound(arr))

For i = 1 To UBound(arr)
   Conexiones(i) = arr(i)
Next

Nnodos = UBound(Conexiones)

End Sub

Sub AgregaDist(ByVal arr As Variant) 'arr debe de ser un arreglo de double
Dim i As Integer

ReDim Distancias(1 To UBound(arr))

For i = 1 To UBound(arr)
   Distancias(i) = arr(i)
Next

Nnodos = UBound(Distancias)

End Sub

Property Get Contar()

Contar = Nnodos

End Property

Property Get ObtenDist(N As String) As Double
Dim i As Integer
Dim Va As Variant
```

```
For i = 1 To Nnodos
   If Conexiones(i) = N Then
      Va = Distancias(i)
      Exit For
   End If
Next

If IsNumeric(Va) Then
   ObtenDist = Va
Else
   ObtenDist = Interpreta(Va)
End If

End Property

Property Get ArrDist() As Variant
ArrDist = Distancias
End Property

Property Get ArrNodos() As Variant
ArrNodos = Conexiones
End Property

Private Function Interpreta(ByVal Dist As String) As Double
Dim i As Integer
Dim p As Integer
Dim contProb As Integer
Dim contPunto As Integer
Dim Clave As String
Dim cadena As String
Dim Actual As String
Dim TocaProb As Boolean
Dim ran As Single
Dim Miu As Double
Dim Sigma As Double


p = 1
contProb = 0
contPunto = 0
Clave = UCase(Mid(Dist, 1, 3))
Select Case Clave
   Case "DIS"
      Dim Prob() As Single
      Dim Punto() As Double
      i = 1
      TocaProb = True
      Actual = Mid(Dist, 4 + i, 1)
      While Actual <> ")"
```

```
    If Actual = "," Then
       Select Case TocaProb
          Case True
             contProb = contProb + 1
             ReDim Preserve Prob(1 To contProb)
             If contProb = 1 Then
                Prob(contProb) = Val(cadena)
             Else
                Prob(contProb) = Val(cadena) + Prob(contProb - 1)
             End If
          Case False
             contPunto = contPunto + 1
             ReDim Preserve Punto(1 To contPunto)
             Punto(contPunto) = Val(cadena)

       End Select
       cadena = ""
       TocaProb = Not (TocaProb)

    Else
       cadena = cadena & Actual
    End If

i = i + 1
Actual = Mid(Dist, 4 + i, 1)
Wend
contPunto = contPunto + 1
ReDim Preserve Punto(1 To contPunto)
Punto(contPunto) = Val(cadena)
Interpreta = Dis(Punto, Prob)

Case "EXP"
    Dim Beta As Double
    cadena = ""
    i = 1
    Actual = Mid(Dist, 4 + i, 1)
    While Actual <> ")"
       cadena = cadena & Actual
       i = i + 1
       Actual = Mid(Dist, 4 + i, 1)
    Wend
    Beta = Val(cadena)
    Interpreta = Exponencial(Beta)
Case "NOR"
    i = 1
    cadena = ""
    Actual = Mid(Dist, 4 + i, 1)
    While Actual <> ")"
       Select Case Actual
          Case ","
```

```
                Miu = Val(cadena)
                cadena = ""
            Case Else
                cadena = cadena & Actual
        End Select
        Sigma = Val(cadena)
        i = i + 1
        Actual = Mid(Dist, 4 + i, 1)
    Wend
    Interpreta = Normal(Miu, Sigma)
Case "GAM"
    i = 1
    cadena = ""
    Actual = Mid(Dist, 4 + i, 1)
    While Actual <> ")"
        Select Case Actual
            Case ","
                Miu = Val(cadena)
                cadena = ""
            Case Else
                cadena = cadena & Actual
        End Select
        Sigma = Val(cadena)
        i = i + 1
        Actual = Mid(Dist, 4 + i, 1)
    Wend
    Dim alf As Integer
    alf = Int(Miu)
    Interpreta = Gamma(alf, Sigma)
Case "BET"
    i = 1
    cadena = ""
    Actual = Mid(Dist, 4 + i, 1)
    While Actual <> ")"
        Select Case Actual
            Case ","
                Miu = Val(cadena)
                cadena = ""
            Case Else
                cadena = cadena & Actual
        End Select
        Sigma = Val(cadena)
        i = i + 1
        Actual = Mid(Dist, 4 + i, 1)
    Wend
    Interpreta = BetaDist(Miu, Sigma)
End Select

End Function
```