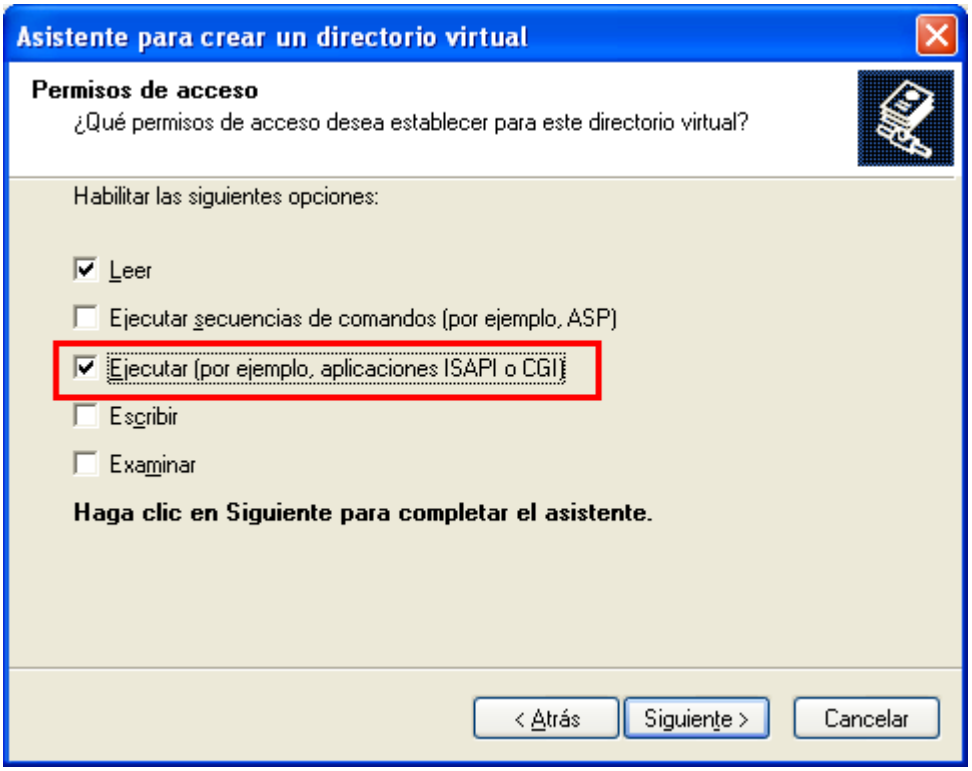
Web Service Sistema Interno

Contenido

Consideraciones:

Para poder brindar este servicio, la PC o Server que cumpla la función de alojar este componente debe tener instalado y habilitado el IIS 5.1 o superior. Dentro del IIS se debe crear un directorio virtual apuntando al lugar donde se encuentra el servicio llamado wsSistemaInternoSiopel.dll y cuando se genere este directorio virtual se debe tener en cuenta de seleccionar la opción Ejecutar (por ejemplo, aplicaciones ISAPI o CGI) dentro del Asistente de creación, como muestra la imagen a continuación



Luego de la creación del Directorio Virtual y de verificar que el IIS este ejecutándose, el WS ya esta disponible para su uso, se debe verificar también la posible configuración del firewall que pueda existir.

Para poder utilizar los métodos de valorización, el MC al cual se conecta el WS debe ser versión 8.30.2 o superior y debe tener entre sus componentes la librería CalcDll.dll, la cual no necesariamente está incluida en la Solución Siopel.

Métodos:

Importante: Si se apunta un navegador de Internet al Web Service, este devolverá todos los métodos habilitados y se podrá navegar esta pagina para encontrar los datos que se deben enviar y se recibirán en cada caso.
Todas las funciones que se enumeran a continuación devuelven como resultado un objeto del tipo TWSResponse
Todas las funciones, excepto Login, reciben como mínimo un parámetro el cual es la sesión devuelta en el método login.

Login(const Clave: String)

Se utiliza para autenticarse en Siopel. Si el resultado de este comando fue exitoso, en la propiedad RespMessage del objeto TWSResponse se informará el número de sesión que otorgo el WS a dicha conexión y que deberá ser utilizado en los comandos que se envíen a este. Si la respuesta a este comando no es positiva, no debe invocarse ningún otro método, ya que si se lo hace estos serán rechazados.

Logout(const Sesion: String)

Se utiliza para desregistrarse de Siopel.

StillAlive(const Sesion: String)

Este comando se debe enviar al WS solo cuando se tiene configurado el valor *SessionTimeOut* en el archivo de configuración del WS y sirve para indicarle a este que la sesión sigue activa. Se debe enviar en caso de que no haya ninguna petición para hacer al WS y se desee mantener la sesión. Se tiene que tener en cuenta que para que el WS no corte la sesión no debe pasar un tiempo mayor al valor antes mencionado (*SessionTimeOut*) sin realizar peticiones.

OrdenConsultarOrdenes(const Sesion: String; const Rueda: String; const Especie: String; const TipoOperacion: String; const TipoOrden: String; out Ordenes: TArrayOfIdOrden)

Este comando devuelve una lista de Identificadores de ordenes que cumplen con los parámetros que se envían

OrdenNueva(const Sesion: String; const Orden: TDatosOrdenNueva; out IdOrden: TIdOrden)

Sirve para generar una orden nueva

OrdenEliminar(const Sesion: String; const OrdenTipo: String; const OrdenNro: Double)

Este comando es utilizado para eliminar una orden ya existente

OrdenModificar(const Sesion: String; const Orden: TDatosOrdenModificacion)

Se utiliza para modificar una orden ya existente

OrdenActualizarConsumidos(const Sesion: String; const OrdenTipo: String; const OrdenNro: Double; const CantAsignada: Double; const CantEjecutada: Double)

Este comando actualiza los consumidos de ordenes externas

OrdenAsignar(const Sesion: String; const OrdenTipo: String; const OrdenNro: Double; const Operador: String)

Este comando sirve para asignar una orden a un operador

OrdenConsultarDatos(const Sesion: String; const OrdenTipo: String; const OrdenNro: Double; out Orden: TDatosConsultaOrden)

Este comando devuelve los datos de una orden determinada

OrdenConsultarConsumidos(const Sesion: String; const OrdenTipo: String; const OrdenNro: Double; out Consumidos: TDatosConsumidosOrden)

Este comando sirve para obtener los consumidos acumulados de una orden dada

OrdenConsultarMovimientos(const Sesion: String; const OrdenTipo: String; const OrdenNro: Double; out Movimientos: TDatosMovimientosOrden)

Con este comando se obtendrán los movimientos generados por una orden determinada

OrdenUsarNovedades(const Sesion: String; const Vigencia: Integer)

Con este comando se activa el almacenamiento temporal de las novedades de las ordenes para luego ser consultado con el método **OrdenNovedades**. El parámetro vigencia indica por cuanto tiempo (en segundos) mantiene las novedades en memoria el WS.

OrdenNovedades(const Sesion: String; out Novedades: TDatosNovedadesOrdenes)

Este comando devuelve todas las novedades que se produjeron con una orden desde la última vez que se invoco o desde la activación de la funcionalidad. Para que esta función devuelva datos, antes se debe utilizar el comando **OrdenUsarNovedades**

ControlarAplicarLimites(const Sesion: String; DatosCtrl: TDatosCtrlLimites; var RtaCtrl:TDatosCtrlLimitesOut)

Permite controlar y/o aplicar límites del operador

ValorizarOperacion(const Sesion: String; DatosValorizacion: TDatosValorizacionIn; var ValorizacionContado, ValorizacionFuturo: TDatosValorizacionOut)

Permite obtener la valorización de una oferta u operación

ControlarAplicarLimitesOperacion(const Sesion: String; DatosValorizacion :TDatosValorizacionIn; DatosCtrl: TDatosCtrlLimites; var RtaCtrl:TDatosCtrlLimitesOut)

Esta función recibe como parámetros datos de una operación u oferta para valorizar y el resultado de esa valorización, se utilizará para controlar y/o aplicar los limites de acuerdo a los parámetros informados

TradingNuevo(const Sesion: AnsiString; DatosTrading: TDatosAltaTrading; var Id: AnsiString) : TWSResponse; stdcall;

Esta función recibe como parámetros los datos para ingresar un trading al sistema y devuelve un ID que se debe utilizar para consultar el estado a través del método que se detalla a continuación. Un resultado exitoso de esta rutina no implica un ingreso correcto del Trading, para tal objetivo se implemento la siguiente función

TradingConsultarEstado(const Sesion: AnsiString; Id: AnsiString; var DatosEstado: TDatosEstadoTrading) :TWSResponse; stdcall;

Esta función recibe como parámetro el Identificador de la operación que se desea consultar. Y devolverá como respuesta el estado de la misma.

Tipos de datos

A continuación se detallan los tipos de datos utilizados en los métodos anteriores

TWSResponse

```
<xs:sequence>
 <xs:element name="Code" type="xs:int"/>
 <xs:element name="RespMessage" type="xs:string"/>
</xs:sequence>
```

Valores posibles del campo Code
0:Comando ejecutado exitosamente
Otro valor: El comando fallo, en el campo RespMessage se encontrará un texto informativo que indicará porque fallo el comando

TArrayOfIdOrden

```
<xs:complexContent>
 <xs:restriction base="soapenc:Array">
  <xs:sequence/>
  <xs:attribute ref="soapenc:arrayType" n1:arrayType="ns1:TIdOrden[]"/>
 </xs:restriction>
</xs:complexContent>
```

TIdOrden

```
<xs:sequence>
 <xs:element name="Tipo" type="xs:string"/>
 <xs:element name="Numero" type="xs:double"/>
</xs:sequence>
```

TDatosOrdenNueva

```
<xs:sequence>
 <xs:element name="Comitente" type="xs:string"/>
 <xs:element name="CantidadOrden" type="xs:double"/>
 <xs:element name="EstadoOrden" type="xs:string"/>
 <xs:element name="Observaciones" type="xs:string"/>
 <xs:element name="CompraVenta" type="xs:string"/>
 <xs:element name="Especie" type="xs:string"/>
 <xs:element name="PlazoVuelta" type="xs:int"/>
 <xs:element name="FechaOrden" type="xs:string"/>(*F)
 <xs:element name="HoraOrden" type="xs:string"/>(*H)
 <xs:element name="CodigoRueda" type="xs:string"/>
 <xs:element name="CantidadMinima" type="xs:double"/>
 <xs:element name="TipoLote" type="xs:string"/>
 <xs:element name="Enteliq" type="xs:string"/>
 <xs:element name="TipoLiq" type="xs:string"/>
 <xs:element name="TipoOpOrigen" type="xs:string"/>
 <xs:element name="PermisoCambio" type="xs:string"/>
 <xs:element name="OrdenTipo" type="xs:string"/>
 <xs:element name="OperadorSIOPEL" type="xs:string"/>
 <xs:element name="GrupoEconomico" type="xs:string"/>
 <xs:element name="DuenioCAP" type="xs:string"/>
 <xs:element name="ValorCAP" type="xs:double"/>
 <xs:element name="PrecioMinimo" type="xs:double"/>
```

```
    <xs:element name="PrecioMaximo" type="xs:double"/>
    <xs:element name="DiasValidez" type="xs:int"/>
  </xs:sequence>
```

Significado de los campos de TDatosOrdenNueva:

Campo	Tipo(long,dec.)	Significado / Valores posibles
Comitente	String(18)	Código del cliente que hace la orden
CantidadOrden	Double(18,4)	Cantidad de la orden
EstadoOrden	String(1)	Estado de la orden. Siempre “1”
Observaciones	String(200)	Observaciones
CompraVenta	String(1)	B:Bid (compra) O:Offer (venta)
Especie	String(15)	Código de la Especie (ver abajo)
PlazoVuelta	Integer(3)	Plazo de Vuelta si tiene. Si no tiene 000
FechaOrden	String(6)	Formato YYMMDD
HoraOrden	String(6)	Formato HHMMSS
CodigoRueda	String(4)	Código de la rueda si está predeterminada. Sino vacío
CantidadMinima	Double(18,4)	Cant. Mínima a operar si el Lote es Parcial
TipoLote	String(1)	“T”: Total “P”:Parcial
EnteLiq	String(1)	Código Entidad Liquidadora (tabla 016-XX)
TipoLiq	String(1)	Código del Tipo de liquidación (tabla 010-XX)
TipoOpOrigen	String(1)	Modalidad operativa (tabla 014-09)
PermisoCambio	String(1)	“T”-Puede cambiar todo “M”-Solo puede mejorar “N”-No puede modificar ningún valor
OrdenTipo	String(1)	“I”: Interna / “E”: Externa
OperadorSIOPEL	String(2)	Código operador al que se le asigna la orden o vacío si no está predeterminado
GrupoEconomico	String(18)	Grupo Económico al que pertenece si DueñoCAP=”G”
ValorCAP	Double(5,2)	Valor CAP
DuenioCAP	String(1)	“G”:Grupo Económico “C”:Comitente “X”:Comitente por Cuenta (comitente del comitente si el comitemte es un agente) Blanco:No tiene CAP
PrecioMinimo	Double(18,8)	Precio/Tasa Mínimo de la orden
PrecioMaximo	Double(18,8)	Precio/Tasa Máximo de la orden
DiasValidez	Integer(4)	Debe ser igual o mayor a 1

Formato campo Especie:

Los formatos posibles son dos (si se usa uno o el otro lo determina el Tipo de Negociación):

- Sin parte variable (con cupón y plazo): NTTTTTUNNNPPPMC
- Con parte variable: NTTTTTVVVVVVVMC

Donde:

- N: Código Tipo de Negociación (tablas 018-XX)
- TTTTT: Código de Título SIOPEL
- U: “C” Si negocia cupón. “ “ si no
- NNN: Nro de cupón
- PPP: Plazo (de ida o contado)

- M: Moneda
- C: Clase del título (ver en definición de títulos)
- VVVVVVV: Parte Variable

TDatosOrdenModificacion

```
<xs:sequence>
 <xs:element name="OrdenTipo" type="xs:string"/>
 <xs:element name="OrdenNro" type="xs:double"/>
 <xs:element name="Cantidad" type="xs:double"/>
 <xs:element name="GrupoEconomico" type="xs:string"/>
 <xs:element name="DuenioCAP" type="xs:string"/>
 <xs:element name="ValorCAP" type="xs:double"/>
 <xs:element name="PrecioMinimo" type="xs:double"/>
 <xs:element name="PrecioMaximo" type="xs:double"/>
</xs:sequence>
```

Formato del campo OrdenNro:
AAMMDDHHSSSSS (Fecha del día, solo la Hora y Secuencia, de 13 de longitud)

TDatosConsultaOrden

```
<xs:sequence>
 <xs:element name="Destinatario" type="xs:string"/>
 <xs:element name="OrdenTipo" type="xs:string"/>
 <xs:element name="NroOrden" type="xs:double"/>
 <xs:element name="Rueda" type="xs:string"/>
 <xs:element name="Especie" type="xs:string"/>
 <xs:element name="Tipo" type="xs:string"/>
 <xs:element name="Divulgacion" type="xs:int"/>
 <xs:element name="Vigencia" type="xs:int"/>
 <xs:element name="EnteLiquidador" type="xs:string"/>
 <xs:element name="TipoLiquidacion" type="xs:string"/>
 <xs:element name="Cantidad" type="xs:double"/>
 <xs:element name="CantidadMin" type="xs:double"/>
 <xs:element name="TipoOpOrigen" type="xs:string"/>
 <xs:element name="ClaseOferta" type="xs:string"/>
 <xs:element name="Cliente" type="xs:string"/>
 <xs:element name="TipoOpDestino" type="xs:string"/>
 <xs:element name="ClienteDestino" type="xs:string"/>
 <xs:element name="PlazoVuelta" type="xs:int"/>
 <xs:element name="EstadoOrden" type="xs:string"/>
 <xs:element name="Texto" type="xs:string"/>
 <xs:element name="PrecioXCta" type="xs:double"/>
 <xs:element name="GrupoEconomico" type="xs:string"/>
 <xs:element name="ValorCAP" type="xs:double"/>
 <xs:element name="DuenioCAP" type="xs:string"/>
 <xs:element name="PrecioMinimo" type="xs:double"/>
 <xs:element name="PrecioMaximo" type="xs:double"/>
 <xs:element name="DiasValidez" type="xs:int"/>
 <xs:element name="ConsumidoAcumulado" type="xs:double"/>
 <xs:element name="EjecutadoAcumulado" type="xs:double"/>
</xs:sequence>
```

Significado de los campos de TDatosConsultaOrden

Campo	Tipo(long,dec.)	Significado / Valores posibles
Destinatario	String(6)	Codigo de Mercado, Agente y Operador que tiene asignada la orden. Formato MAAAOO
OrdenTipo	String(1)	"I": Interna / "E": Externa
NroOrden	Integer(9)	Identificador de la orden en forma conjunta con Orden tipo.
Rueda	String(4)	Codigo de rueda en la que fue ingresada la orden
Especie	String(15)	Codigo de la Especie
Tipo	String(1)	T=Total. P=Parcial
Divulgacion	Integer(2)	01 a 99
Vigencia	Integer(3)	000 a 999 (minutos). Solo aplicable para algunas Clases de Ofertas. Sino informar 000.
EnteLiquidador	String(1)	Código Entidad Liquidadora (tabla 016-XX)
TipoLiquidacion	String(1)	Código del Tipo de liquidación (tabla 010-XX)
Cantidad	Double(18,4)	Cantidad de la orden
CantidadMin	Double(18,4)	Cant. Mínima a operar si el Lote es Parcial
TipoOpOrigen	String(1)	Modalidad operativa (tabla 014-09)
ClaseOferta	String(1)	No se informa en ordenes Internas/Externas
Cliente	String(18)	Cliente por cuenta
TipoOpDestino	String(1)	No se informa en ordenes Internas/Externas
ClienteDestino	String(18)	No se informa en ordenes Internas/Externas
PlazoVuelta	String(3)	Plazo de vuelta
EstadoOrden	String(1)	Estado de la orden
Texto	String(100)	Texto de la oferta relacionada
PrecioXCta	Double(18,8)	Precio por cuenta
GrupoEconomico	String(18)	Grupo Económico al que pertenece si DueñoCAP="G"
ValorCAP	Double(5,2)	Valor CAP
DuenioCAP	String(1)	"G":Grupo Económico "C":Comitente "X":Comitente por Cuenta (comitente del comitente si el comitemte es un agente) Blanco:No tiene CAP
PrecioMinimo	Double(18,8)	Precio/Tasa Mínimo de la orden
PrecioMaximo	Double(18,8)	Precio/Tasa Máximo de la orden
DiasValidez	Integer(4)	Días validez de la orden
ConsumidoAcumulado	ConsumidoAcumulado	Cantidad consumida (ofertas) acumulada
EjecutadoAcumulado	EjecutadoAcumulado	Cantidad concertada (operaciones) acumulada

TDatosConsumidosOrden

```
<xs:sequence>
  <xs:element name="OrdenTipo" type="xs:string"/>
  <xs:element name="NroOrden" type="xs:double"/>
  <xs:element name="ConsumidoAcumulado" type="xs:double"/>
  <xs:element name="EjecutadoAcumulado" type="xs:double"/>
</xs:sequence>
```

TDatosMovimientosOrden

```
<xs:complexContent>
 <xs:restriction base="soapenc:Array">
  <xs:sequence/>
  <xs:attribute ref="soapenc:arrayType" n1:arrayType="ns1:TDatosMovimientoOrden[]"/>
 </xs:restriction>
</xs:complexContent>
```

TDatosMovimientoOrden

```
<xs:sequence>
 <xs:element name="SecuenciaOrden" type="xs:string"/>
 <xs:element name="OrdenTipo" type="xs:string"/>
 <xs:element name="OrdenNro" type="xs:double"/>
 <xs:element name="Estado" type="xs:string"/>
 <xs:element name="Accion" type="xs:string"/>
 <xs:element name="OpSiopel" type="xs:string"/>
 <xs:element name="Cantidad" type="xs:double"/>
 <xs:element name="ConsumidoAcumulado" type="xs:double"/>
 <xs:element name="EjecutadoAcumulado" type="xs:double"/>
 <xs:element name="Motivo" type="xs:string"/>
 <xs:element name="DatoAdicional" type="xs:string"/>
 <xs:element name="PrecioMinimo" type="xs:double"/>
 <xs:element name="PrecioMaximo" type="xs:double"/>
 <xs:element name="DiasValidez" type="xs:int"/>
</xs:sequence>
```

Valores posibles del campo Estado

0:Cerrada
1:En Trámite
2:Cancelada
3:Asignada
4:Enviada
5:Rechazada
6:Ejecutada
7:Baja
8:Totalmente ingresada
9:Parcialmente ingresada
A:Parcialmente ingresada y ejecutada
C:Totalmente ingresada y parcialmente ejecutada
D:Parcialmente ingresada y pendiente de baja
E:Parcialmente ingresada y ejecutada y pendiente de baja
F:Totalmente ingresada, parcialmente ejecutada y pendiente de baja
G:Parcialmente ingresada y ejecutada y cancelada
H:Totalmente ingresada y pendiente de baja
I:Pendiente de baja

Valores posibles del campo Accion

A:Alta
B:Baja
M:Modificación

E:Cambio de Estado
O:Asignación de Operador
C:Cancelación

Valores posibles del campo Motivo
T:Trading
F:Oferta
O:Operación
S:Solicitud de Alta/Baja o Modificacion, dependiendo de la accion
R:Respuesta de Alta/Baja o Modificacion, dependiendo de la accion
C:Pedido cancelado por el destino

TDatosNovedadesOrdenes

```
<xs:complexContent>
 <xs:restriction base="soapenc:Array">
  <xs:sequence/>
  <xs:attribute ref="soapenc:arrayType" n1:arrayType="ns1:TDatosNovedadesOrden[]"/>
 </xs:restriction>
</xs:complexContent>
```

TDatosNovedadesOrden

```
<xs:sequence>
 <xs:element name="OrdenTipo" type="xs:string"/>
 <xs:element name="OrdenNro" type="xs:double"/>
 <xs:element name="Eventos" type="xs:string"/>
</xs:sequence>
```

TDatosCtrlLimites

```
<xs:sequence>
 <xs:element name="Destinatario" type="xs:string"/>
 <xs:element name="Valor" type="xs:double"/>
 <xs:element name="Accion" type="xs:string"/>
 <xs:element name="Destino" type="xs:string"/>
 <xs:element name="BidOffer" type="xs:string"/>
</xs:sequence>
```

TDatosCtrlLimitesOut

```
<xs:sequence>
 <xs:element name="Valor" type="xs:double" />
 <xs:element name="ConsumidoOfertas" type="xs:double" />
 <xs:element name="ConsumidoOperaciones" type="xs:double" />
 <xs:element name="DefinidoOfertas" type="xs:double" />
 <xs:element name="DefinidoOperaciones" type="xs:double" />
</xs:sequence>
```

Valores posibles del campo Accion
C:Controlar
A:Controlar y Aplicar

Valores posibles del campo Destino

F:Oferta
P:Operacion
A:Oferta y Operación

Valores posibles del campo CompraOVenta
C:Compra
V:Venta

TDatosValorizacionIn

```
<xs:sequence>
 <xs:element name="Rueda" type="xs:string"/>
 <xs:element name="Especie" type="xs:string"/>
 <xs:element name="Cantidad" type="xs:double"/>
 <xs:element name="PrecioOTasa" type="xs:double"/>
 <xs:element name="EnteLiq" type="xs:string"/>
 <xs:element name="Fecha" type="xs:string"/>(*F)
 <xs:element name="PlazoVuelta" type="xs:int"/>
 <xs:element name="VigenciaTasa" type="xs:int"/>
 <xs:element name="IndCorreccion" type="xs:int"/>
 <xs:element name="FechaFija" type="xs:string"/>(*F)
 <xs:element name="BaseAnual2" type="xs:double"/>
 <xs:element name="TipoDias2" type="xs:string"/>
 <xs:element name="DatosTNE" type="ns1:TDatosTNE"/>
 <xs:element name="CantGarantia" type="xs:double"/>
 <xs:element name="ValorIndice" type="xs:double"/>
 <xs:element name="DiasAyer" type="xs:int"/>
 <xs:element name="Opciones" type="xs:int"/>
</xs:sequence>
```

Valores posibles del campo TiposDias2
1:Habiles
2:Corridos
3:Segun Tipo Nego Contado
4:Comerciales
5:Segun Tipo Nego Plazo
6:Metodo Americano
7:Metodo Europeo
8:Metodo Argentino
9:Corridos Sin Bisiestos
A:Comer Metodo Europeo XLS

Valores posibles del campo Opciones
Suma de los valores que se deseen incluir
0:Ninguno
1:NoLiqFuturo
2:SinPrecision
4:ChequearObtenerPrecioGarantia
8:CalcTasasEquivalentes

TDatosTNE

```
<xs:sequence>
 <xs:element name="FechaEmision" type="xs:string"/>(*F)
 <xs:element name="FechaVencimiento" type="xs:string"/>(*F)
 <xs:element name="PeriodicidadPago" type="xs:string"/>
 <xs:element name="ModalidadPago" type="xs:string"/>
 <xs:element name="PeriodicidadReinversion" type="xs:string"/>
 <xs:element name="ModalidadReinversion" type="xs:string"/>
 <xs:element name="TasaFacial" type="xs:double"/>
 <xs:element name="Premio" type="xs:double"/>
 <xs:element name="CodigoDeEmisor" type="xs:string"/>
</xs:sequence>
```

TDatosValorizacionOut

```
<xs:sequence>
 <xs:element name="Residual" type="xs:double"/>
 <xs:element name="FechaLiq" type="xs:string"/>(*F)
 <xs:element name="FechaVto" type="xs:string"/>(*F)
 <xs:element name="Interes" type="xs:double"/>
 <xs:element name="Plazo" type="xs:int"/>
 <xs:element name="PlazoCalculado" type="xs:int"/>
 <xs:element name="Dias" type="xs:int"/>
 <xs:element name="PlazoVuelta" type="xs:int"/>
 <xs:element name="FechaLiqPlazoVuelta" type="xs:string"/>(*F)
 <xs:element name="Precio" type="xs:double"/>
 <xs:element name="Tasa" type="xs:double"/>
 <xs:element name="Total" type="xs:double"/>
 <xs:element name="TotalDolares" type="xs:double"/>
 <xs:element name="TotalEmision" type="xs:double"/>
 <xs:element name="PU" type="xs:double"/>
 <xs:element name="DiasAlVtoTitulo" type="xs:int"/>
 <xs:element name="TasaFacialTitulo" type="xs:double"/>
 <xs:element name="HuboFlujosIntermedios" type="xs:string"/>
 <xs:element name="TasaNeta" type="xs:double"/>
</xs:sequence>
```

TDatosAltaTrading

```
<xs:sequence>
 <xs:element name="Especie" type="xs:string"/>
 <xs:element name="PlazoVuelta" type="xs:int"/>
 <xs:element name="Destinatario" type="xs:string"/>
 <xs:element name="Cantidad" type="xs:double"/>
 <xs:element name="Precio" type="xs:double"/>
 <xs:element name="TipoOpOr" type="xs:string"/>
 <xs:element name="TipoOpDe" type="xs:string"/>
 <xs:element name="EnteLiq" type="xs:string"/>
 <xs:element name="TipoLiq" type="xs:string"/>
 <xs:element name="Rueda" type="xs:string"/>
 <xs:element name="ComitenteOrigen" type="xs:string"/>
 <xs:element name="ComitenteContra" type="xs:string"/>
 <xs:element name="IdAdicionalContra" type="xs:int"/>
```

```
      <xs:element name="IdAdicional2Contra" type="xs:int"/>
      <xs:element name="VigenciaTasa" type="xs:int"/>
      <xs:element name="IndCorreccion" type="xs:int"/>
      <xs:element name="PrecioIdaSelic" type="xs:double"/>
      <xs:element name="Interes" type="xs:double"/>
      <xs:element name="Total" type="xs:double"/>
      <xs:element name="TotalDolar" type="xs:double"/>
      <xs:element name="PlazoCobertura" type="xs:int"/>
      <xs:element name="PorcAfecSettle" type="xs:double"/>
      <xs:element name="PorcAfecPrice" type="xs:double"/>
      <xs:element name="OrdenTipo" type="xs:string"/>
      <xs:element name="OrdenNro" type="xs:double"/>
      <xs:element name="Texto" type="xs:string"/>
      <xs:element name="TipoTrading" type="xs:string"/>
      <xs:element name="DatosTNE" type="ns1:TDatosTNE"/>
      <xs:element name="CuentaCliente" type="xs:string"/>
      <xs:element name="NomContraInt" type="xs:string"/>
      <xs:element name="NomCustodioInt" type="xs:string"/>
      <xs:element name="FechaConcertacion" type="xs:string"/>(*F)
      <xs:element name="HoraConcertacion" type="xs:string"/>(*H)
      <xs:element name="ValorIndice" type="xs:double"/>
      <xs:element name="EspecieGarantia" type="xs:string"/>
      <xs:element name="CantidadGarantia" type="xs:double"/>
      <xs:element name="PrecioGarantia" type="xs:double"/>
      <xs:element name="TasaEquiGarantia" type="xs:double"/>
      <xs:element name="PorcentajeGarantia" type="xs:double"/>
      <xs:element name="ValorizacionGarantia" type="xs:double"/>
      <xs:element name="ValorizacionFutura" type="xs:double"/>
      <xs:element name="ValorizacionDolarFutura" type="xs:double"/>
      <xs:element name="CodDemora" type="xs:string"/>
      <xs:element name="MotDemora" type="xs:string"/>
      <xs:element name="CompraVenta" type="xs:string"/>
    </xs:sequence>
```

TDatosEstadoTrading

```
    <xs:sequence>
      <xs:element name="Id" type="xs:string"/>
      <xs:element name="Estado" type="xs:string"/>
      <xs:element name="SecHost" type="xs:string"/>
      <xs:element name="CodRtaHost" type="xs:string"/>
      <xs:element name="MsgRtaHost" type="xs:string"/>
      <xs:element name="SecHostPata2" type="xs:string"/>
      <xs:element name="RelacionOperaciones" type="xs:int"/>
    </xs:sequence>
```

Valores posibles para el campo Estado

N:Nueva, esperando respuesta

R:Rechazada

Si CodRtaHost=N, el rechazo se efectuo en el Host.

Si CodRtaHost=3, el rechazo lo efectuo la contraparte.

V:Vencida

S:Aceptada

Valores posibles para el campo Destinatario
000000: Si el trading es contra un tercero
MAAAOO: Mercad+Agente+Operador destinatario del trading

Valores posibles para el campo TipoTrading
M: Trading masivo
Vacio: Trading standard

Valores posibles para el campo CompraVenta
C: Compra
V: Venta

SecHostPata2: si la operación tiene 2 patas, esta es la secuencia de la otra pata, caso contrario el valor es 0
RelacionOperaciones: si la operación tiene 2 patas, este valor es la referencia que relaciona ambas patas, caso contrario el valor es 0

(*F) Campos del tipo Fecha: se deben informar con el siguiente formato AAAAMMDD
(*H) Campos del tipo Hora: se deben informar con el siguiente formato HHMMSS

Configuración

Se utiliza un archivo de configuración del tipo Ini el cual debe llamarse wsSistemaInternoSiopel.ini y se debe encontrar en la misma carpeta donde se ubica la aplicación (wsSistemaInternoSiopel.dll). A continuación se enumeran los posibles valores de configuración.

```
[connection]
;Ip: ip donde se encuentra el MC activo al que se conectará el WS
Ip=127.0.0.1
;Port: Puerto del MC donde espera conexiones de los Sistemas Internos
Port=2011

[general]
;MAO: Mercado, Agente y Operador con el cual el WS se identificará ante el MC
MAO=E93799
;ModuloApp: es el código de aplicación que se utilizará en el login al Host
ModuloApp=
;ResponseTimeOut: Es el tiempo máximo de espera por una respuesta a un requerimiento desde el WS al MC. Se expresa en segundos.
ResponseTimeOut=5
;SessionTimeOut: Es el tiempo máximo de espera entre un requerimiento y otro al WS desde el cliente que lo consuma, pasado ese tiempo el WS da por finalizada la sesión. Se expresa en segundos.
SessionTimeOut=10
;MaxActiveSessions: Es el numero máximo de clientes distintos que el WS puede aceptar
MaxActiveSessions=1
```

Ejemplos de uso

Login

Este ejemplo invoca al método login y como clave envía el valor Password para que el Host Siopel la valide. Si la respuesta al comando es 0 en el campo RespMessage vendrá el ID de Sesion q se utilizará para los restantes métodos del WS.

…

```
var
  Rta:TWSResponse
begin
  inherited;
  Rta:=IwsSistemaInternoSiopel.Login('Password');
  if(Rta.Code=0)then begin
    Session:=Rta.RespMessage;
    ShowMessage('Proceso exitoso');
  end else ShowMessage('El proceso fallo. Causa: '+Inttostr(Rta.Code)+' - '+ Rta.RespMessage);

…
```

Logout

Envia un logout sobre una sesión ya identificada con éxito previamente

```
…
var
  Rta:TWSResponse;
begin
  Rta:=IwsSistemaInternoSiopelInt.Logout(Session);
  if(Rta.Code=0)then begin
    Session:='';
    ShowMessage('Proceso exitoso');
  end else ShowMessage('El proceso fallo. Causa: '+Inttostr(Rta.Code)+' - '+ Rta.RespMessage);

…
```

StillAlive

```
…
Var
  Rta:TWSResponse;
begin
  Rta:=IwsSistemaInternoSiopelInt.StillAlive(Session);
  if(Rta.Code=0)then
    ShowMessage('Proceso exitoso')
  else ShowMessage('El proceso fallo. Causa: '+Inttostr(Rta.Code)+' - '+ Rta.RespMessage);
```

TradingNuevo

```
…
var
  Rta: TWSResponse
  DatosTrading:TDatosAltaTrading;
  Id:WideString;
begin
  DatosTrading:=TDatosAltaTrading.Create;
  try
    GetDatosTrading(DatosTrading);//esta rutina devuelve la estructura DatosTrading con los datos necesarios para ingresar un Trading nuevo
    Rta:= IwsSistemaInternoSiopel.TradingNuevo(Sesion,DatosTrading,Id);
    if(Rta.Code=0)then
      ShowMessage('Proceso exitoso')
```

```
  else ShowMessage('El proceso fallo. Causa: '+Inttostr(Rta.Code)+' - '+ Rta.RespMessage);
 finally
  FreeAndNil(DatosTrading);
 end;
```

TradingConsultarEstado

El siguiente ejemplo devuelve los datos de un Alta de un Trading identificado con el valor **18050110101012345**. Este valor fue devuelto en el parametro ID del método **TradingNuevo**

```
…
var
 Rta: TWSResponse
 DatosEstado:TDatosEstadoTrading;
begin
 DatosEstado:=TDatosEstadoTrading.Create;
 try
  Rta:=GetIwsSistemaInternoSiopel.TradingConsultarEstado(Sesion, '18050110101012345', DatosEstado);
  if(Rta.Code=0)then
   ShowMessage('Proceso exitoso')
  else ShowMessage('El proceso fallo. Causa: '+Inttostr(Rta.Code)+' - '+ Rta.RespMessage);
 finally
  FreeAndNil(DatosEstado);
 end;
```