VISUAL CHART 6. NOTAS IMPORTANTES PARA DESARROLLADORES

Migración de proyectos de versiones anteriores a Visual Chart 6

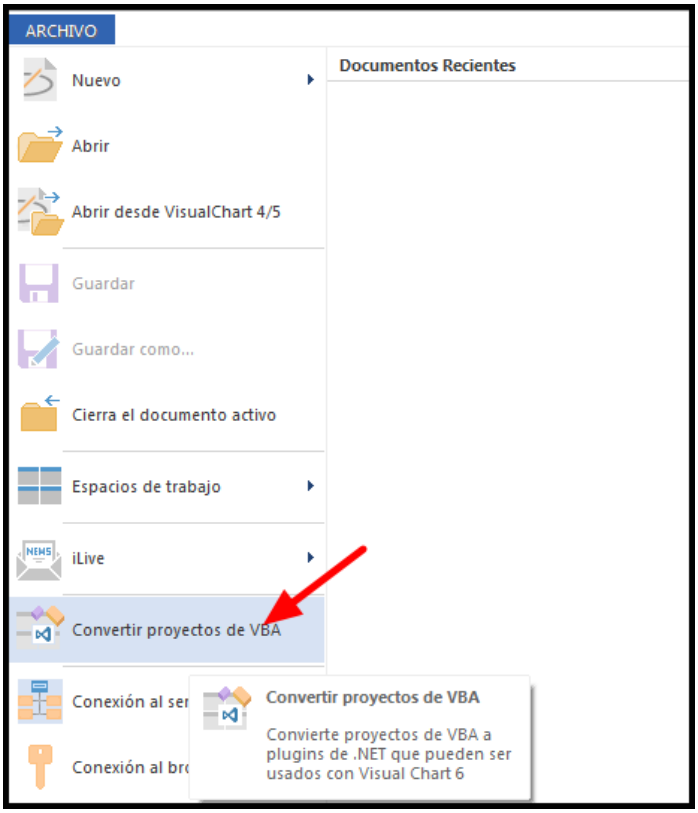
Introducción

El modelo de diseño de estrategias cambia notablemente en Visual Chart 6, dando un salto de calidad al trasladar su desarrollo a los lenguajes de programación .NET. No obstante, esto no debe suponer un impedimento para aquellos usuarios que disponen de sus estrategias en versiones anteriores de Visual Chart, puesto que la nueva versión incorpora una herramienta de conversión automática que facilitará la labor de migración a los desarrolladores.

A continuación, explicaremos cómo realizar dicho proceso.

1. Seleccionar la herramienta de conversión.

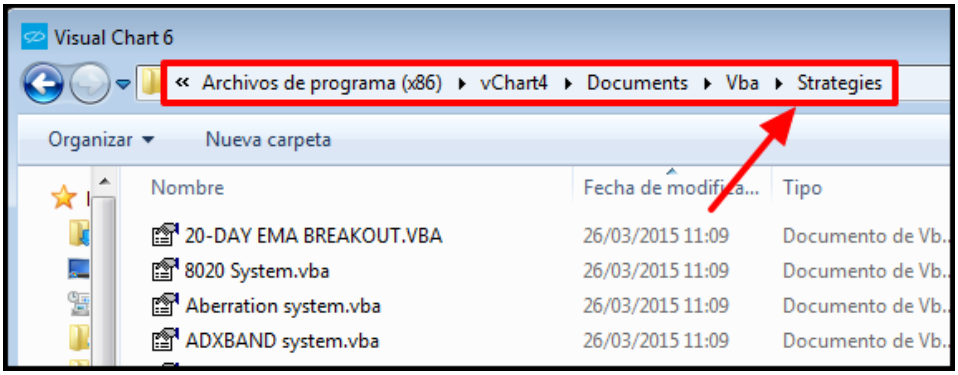
Una vez tenemos abierto Visual Chart 6, accedemos al menú Archivo → Convertir proyectos de VBA.



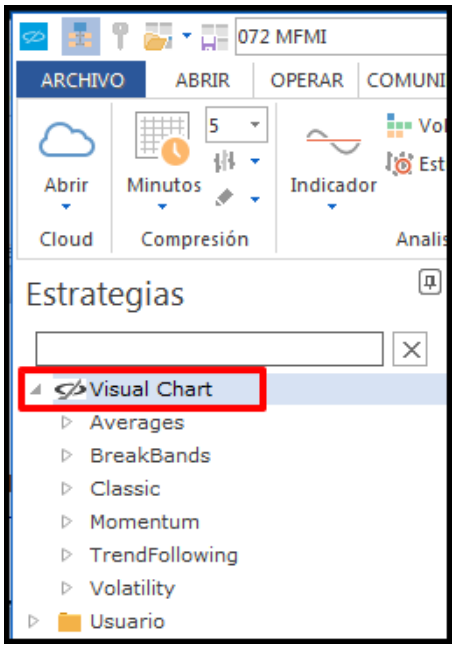
2. Seleccionar el archivo de la estrategia.

Pulsamos el botón y se abrirá una carpeta con acceso a nuestro directorio de estrategias de Visual Chart 5. Buscamos el archivo vba de la estrategia que queremos exportar.

Si disponemos de estrategias de Visual Chart 4 que queramos exportar a la nueva versión, simplemente, buscamos el directorio correspondiente donde se ubican las estrategias de la versión 4:



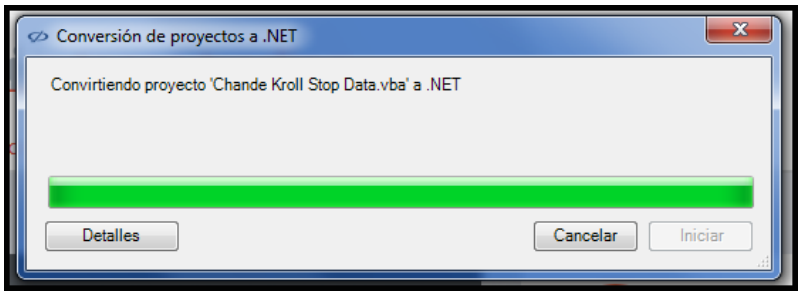
Cabe recordar que sólo debemos convertir aquellas estrategias que sean de nuestra propiedad, puesto que al realizar la conversión, quedarán asociadas a nuestro usuario de forma exclusiva. El resto de estrategias públicas de Visual Chart aparecerán en la lista de estrategias bajo el logo de la compañía.



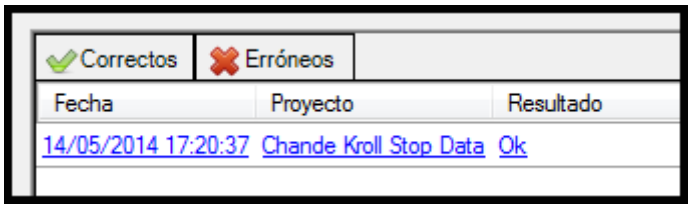
Si el sistema ha sido realizado en PDV, debemos buscar el archivo VBA correspondiente a dicho sistema y seleccionarlo.

3. Iniciar el proceso de conversión.

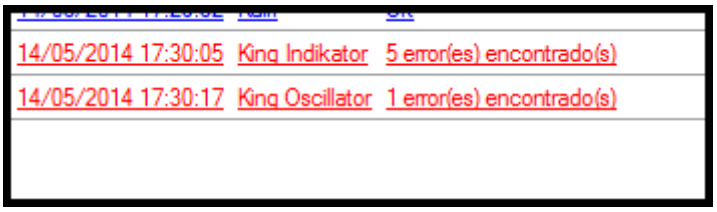
Al pulsar el botón Aceptar, nos aparecerá una ventana como la siguiente:



Una vez el proceso finaliza, aparecerá una nueva ventana indicándonos cómo ha ido el proceso de conversión. Si todo ha ido bien, nos aparecerá el mensaje OK. En tal caso, cerraremos la ventana y comprobaremos que ya tenemos disponible nuestro sistema en Visual Chart 6.

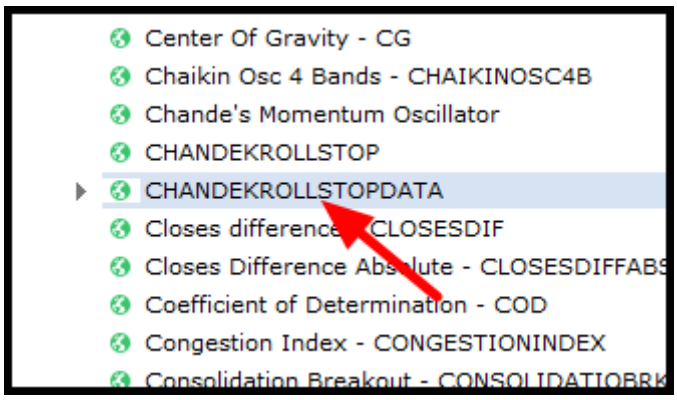


Si se ha producido algún error que el conversor no ha podido solucionar, nos aparecerá un mensaje avisándonos:



Para poder solucionarlos, pincharíamos sobre el nombre del sistema correspondiente. Seguidamente, se abriría el código de programación del mismo en .NET a través del Visual Studio. Si tenemos nociones de programación, podemos averiguar cuál ha sido el error. Si no, nos pondríamos en contacto con el departamento de soporte de Visual Chart.

El cualquier caso, llegados a este punto, cerramos la ventana con los mensajes. Tanto si la estrategia se ha generado correctamente como si no, aparecerá en la lista de estrategias dentro de la carpeta de usuario:



Cambios en el funcionamiento de las estrategias en Visual Chart 6

Introducción

Una de las características más destacadas de Visual Chart 6 consiste en las mejoras añadidas al funcionamiento y cálculo de las estrategias de trading algorítmico. Gracias a ello, los resultados obtenidos durante el *backtesting* se asemejan con mayor fiabilidad a los resultados reales. No sólo eso, sino que además se ha potenciado la operativa de las estrategias dotándolas de mayor robustez.

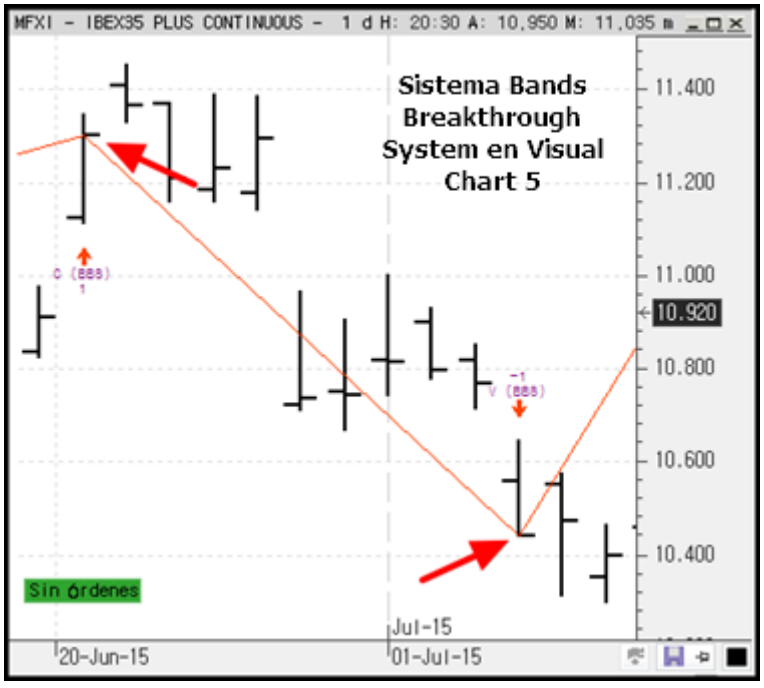
No obstante, estos cambios pueden afectar a los estadísticos arrojados por las estrategias en las diferentes versiones de Visual Chart, cuestión que debe ser tenida en cuenta por los programadores.

A continuación, repasamos las modificaciones añadidas en el proceso de cálculo de las estrategias:

1. Las órdenes *Al Cierre* se sustituyen por órdenes *A Mercado*

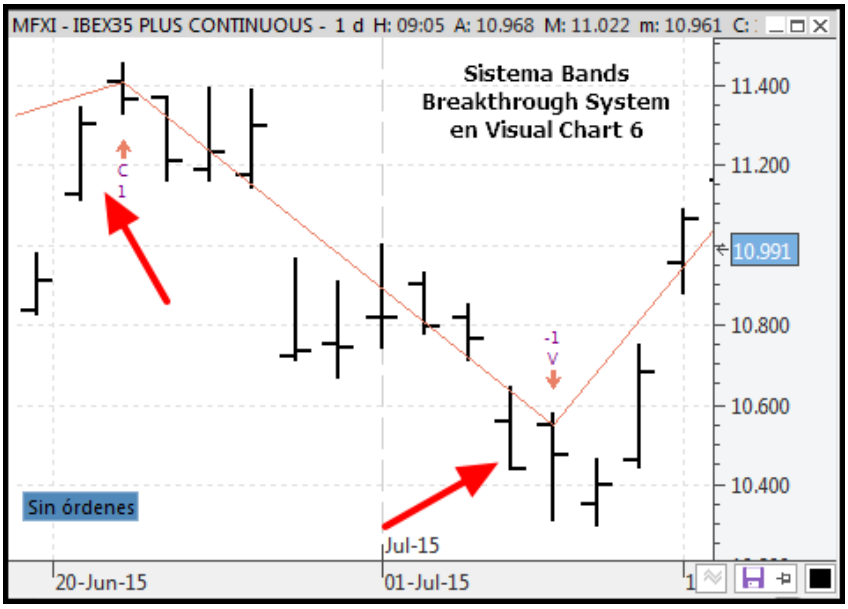
El cambio más relevante respecto a versiones anteriores está asociado al uso de órdenes de tipo *Al Cierre*.

En versiones anteriores, si seleccionábamos este tipo de orden, a la hora de visualizar el histórico de operaciones de la estrategia, observábamos cómo se tomaba como precio de referencia el precio de cierre de la barra sobre en la que se había efectuado dicha orden:

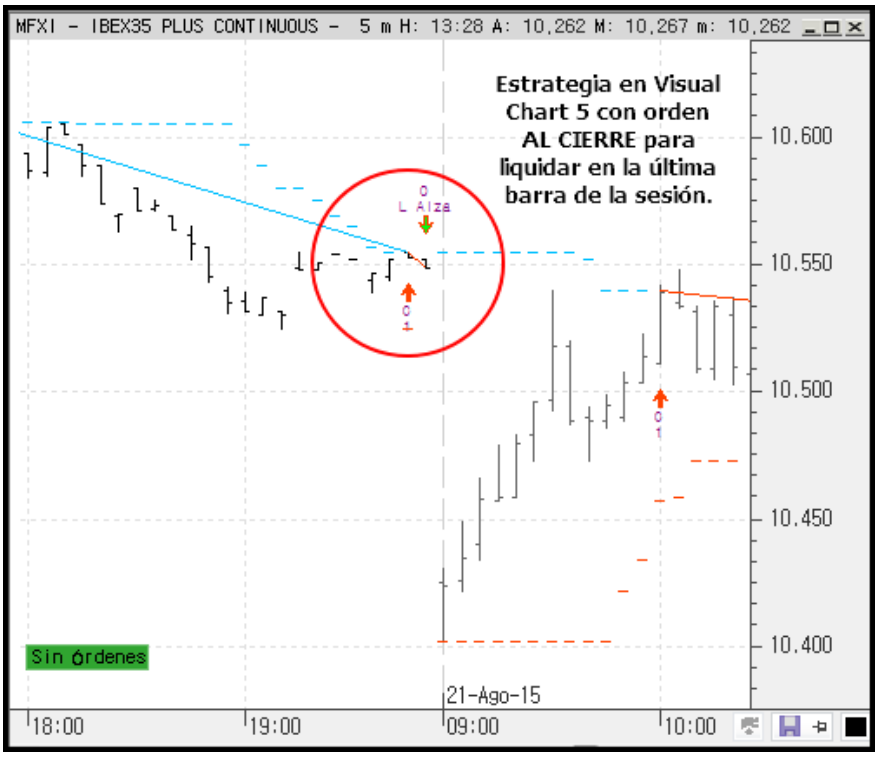


Sin embargo, en tiempo real dicha operación venía a ejecutarse a mercado en lugar de al precio de cierre de la barra finalizada, por lo que suponía una variación apreciable entre los resultados históricos y los dados durante el proceso de operativa real.

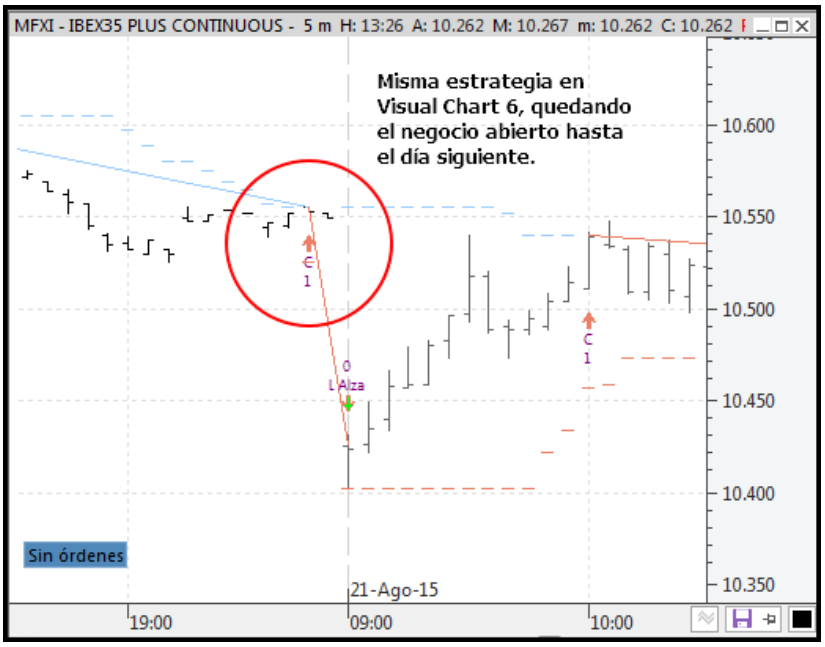
Con vistas a mejorar la fiabilidad de los datos estadísticos, en Visual Chart 6 las operaciones *Al Cierre* pasan a ser operaciones *A Mercado*.



Mención especial tienen las situaciones en las que la estrategia envía una orden *Al Cierre* **en la última barra del día** como medida para liquidar las operaciones al final de sesión, tal y como se muestra en el siguiente ejemplo:



En estos casos, si trasladamos la estrategia a Visual Chart 6 y no la modificamos en nada, obtendremos como resultado que el negocio permanece abierto hasta el inicio de la siguiente sesión:

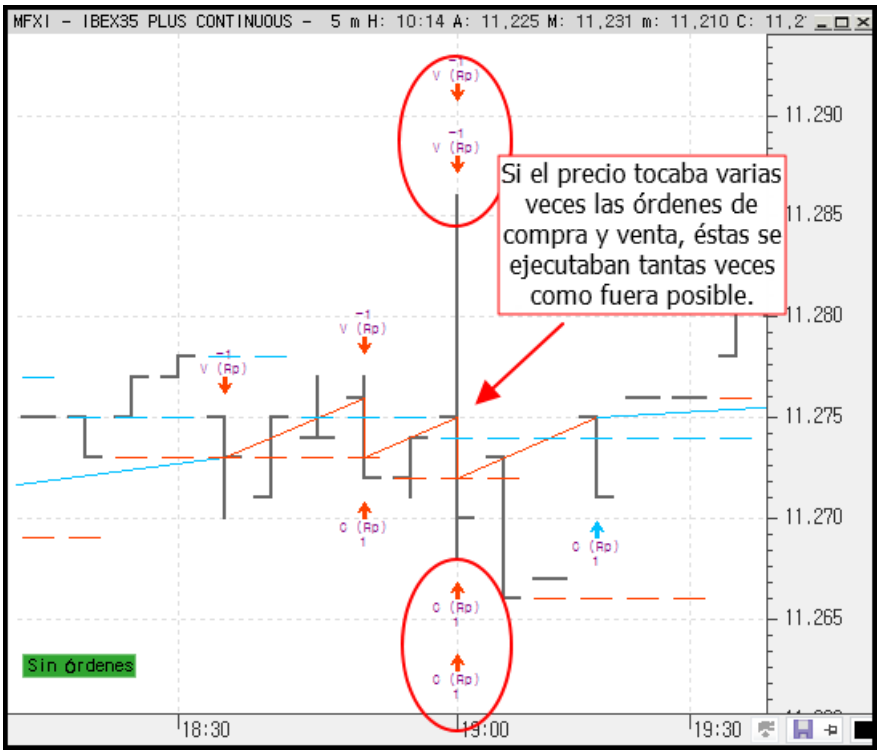


Aconsejamos a nuestros usuarios revisar sus estrategias para evitar que esto ocurra. Pueden aprovechar el nuevo método *ExitPositionEndOfDay*, el cual facilita notablemente la gestión de cierre de posiciones al término de la sesión.

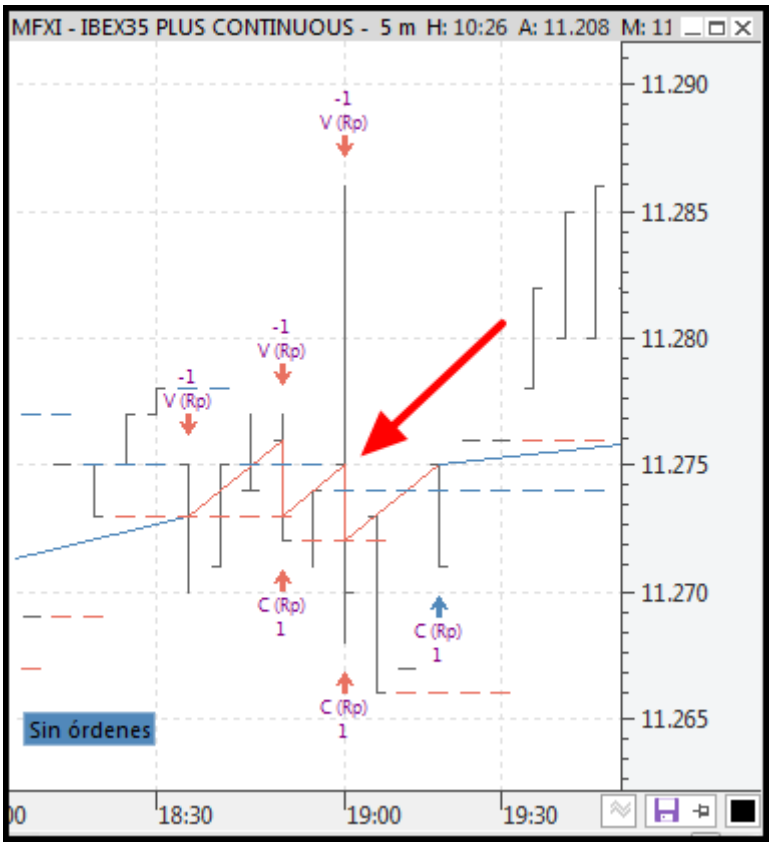
En lo relativo a la programación, las órdenes *AtClose* se siguen manteniendo, a fin de que el cambio no afecte al diseño de las estrategias migradas.

2. Reactivación de las órdenes una vez ejecutadas

Uno de los principales cambios en las estrategias afecta a la reactivación de las órdenes. En versiones anteriores, si activábamos sobre la misma barra dos órdenes opuestas, estas dos órdenes podían generar múltiples operaciones si el precio oscilaba entre ambas:



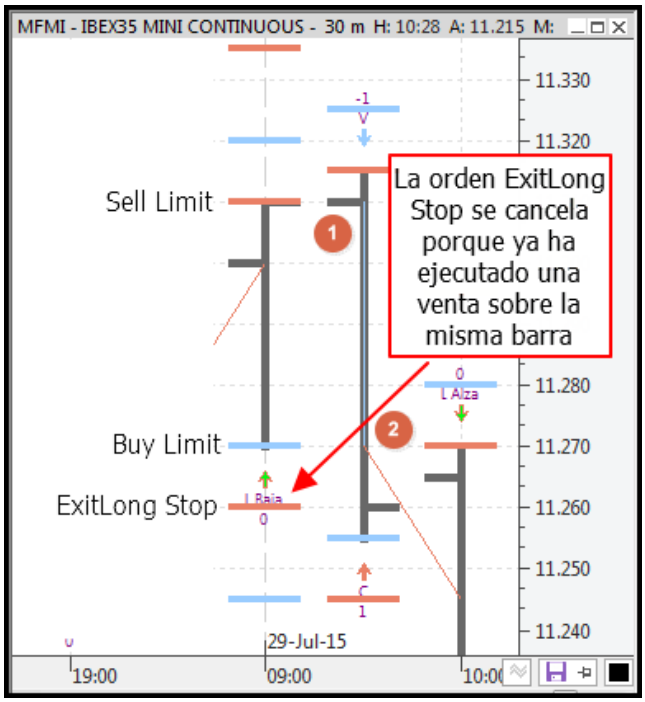
Esta casuística se ha eliminado en la nueva versión:



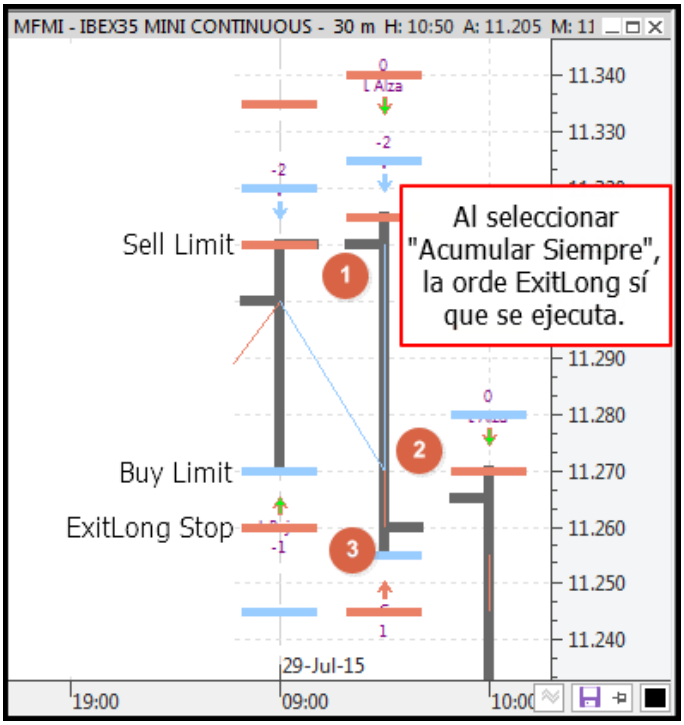
En Visual Chart 6, las órdenes no se reactivan, de modo que cada orden se puede ejecutar una única vez. Si en el momento de la evaluación no puede ejecutarse por el método de entrada, se desprecia.

3. Una entrada y una salida por barra.

En el modo de entrada sin acumular, solo se permite una entrada y una salida sobre la misma barra, independientemente del número de órdenes existentes. De modo que al ejecutarse una orden, se desactivarán todas las órdenes del mismo signo.



En el resto de métodos de entrada, se debe permitir ejecutar todas las órdenes existentes, si bien se respeta la situación indicada en el punto 2.



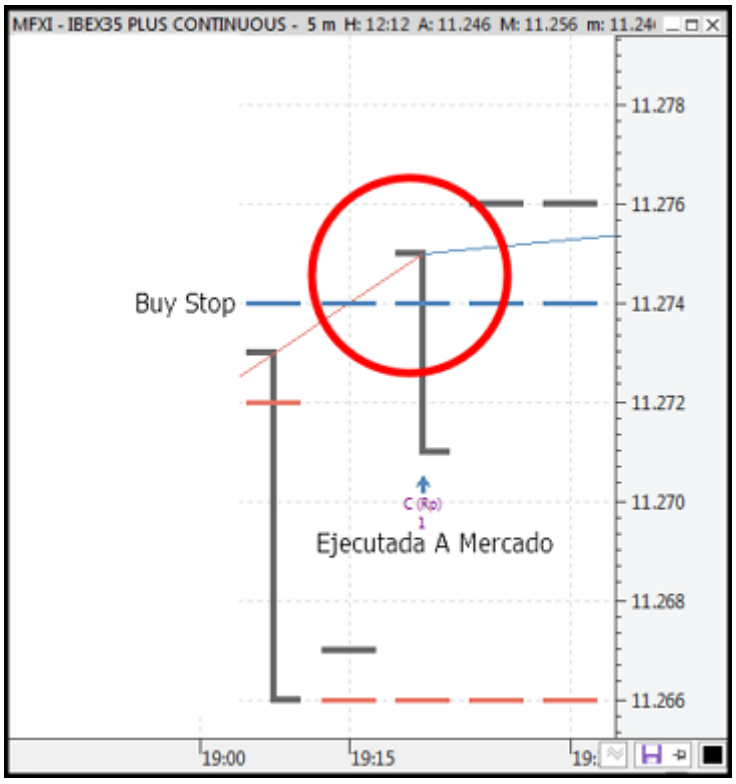
Como vemos en el ejemplo, aunque la estrategia queda fuera de mercado en el punto 3, no vuelve a ejecutar otra entrada a largo puesto que respeta lo indicado en el punto 2 (a diferencia de lo que ocurría en versiones anteriores).

Aclaración relativa a este punto

Pese a indicar que al *ejecutarse una orden, se desactivarán todas las órdenes del mismo signo*, cabe aclarar que esto sólo ocurre si la orden ejecutada es de entrada (órdenes *Sell* y *Buy*). Una orden de liquidación no cancelará al resto de órdenes activas del mismo signo sobre la misma barra, tal y como podemos ver en el siguiente ejemplo:

4. Validación de órdenes limitadas y en stop

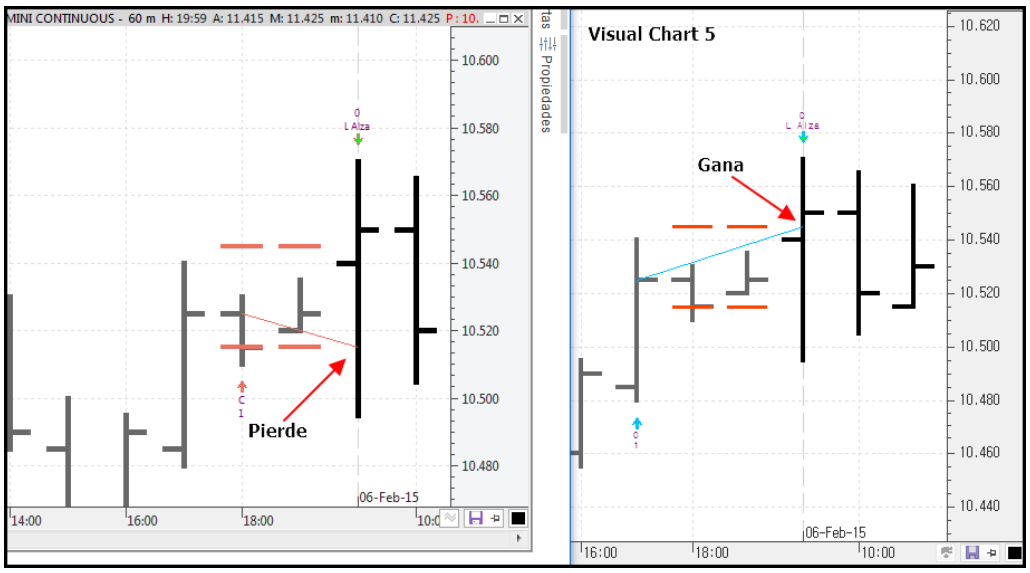
Las estrategias de Visual Chart 6 no van a validar si las órdenes limitadas están cruzadas ni si las órdenes en stop están pasadas de precio. En estos casos, dichas órdenes pasan a convertirse en órdenes a mercado.



5. Cambio en el recorrido interno de las barras de histórico

A la hora de estudiar el *backtesting* de los sistemas, Visual Chart aplica un criterio de recorrido interno de la barra para evaluar cómo se ha movido el precio durante la formación de la misma. Este criterio ha sido modificado, de modo que ahora el recorrido de la barra se ha de hacer siempre asumiendo cuál ha sido el recorrido más corto.

En el siguiente ejemplo, observamos el funcionamiento del sistema Dinamic Zone RSI System sobre MFMI en 60 minutos tanto en Visual Chart 5 como en Visual Chart 6:



En este caso, la nueva versión ejecuta primero el stop puesto que el recorrido más corto sería Apertura → Mínimo→ Máximo → Cierre. En Visual Chart 5 el criterio era distinto, siendo el recorrido Apertura→ Máximo → Mínimo →Cierre, de ahí que ejecute primero el objetivo.

Como ven, esto supondrá ciertos cambios en los resultados de los sistemas, lo cual debe ser tenido en cuenta.

6. Ordenación de las órdenes de stop y limitadas

Se ha cambiado la ordenación de las órdenes de stop y limitadas en el contenedor de órdenes pendientes con el fin de que el criterio por el cual se ejecutan unas antes que otras sea lo más correcto posible. El método de ordenación sigue el siguiente criterio: En la parte baja, deben estar los stop de venta y limitadas de compra, y en la parte alta, los stop de compra y las limitadas de venta, siempre ordenadas de menor a mayor precio:

Buy AtLimit	Sell AtStop	Buy AtLimit	Sell AtStop	Buy AtStop	Sell AtLimit	Buy AtStop
9100	9350	9400	9450	10500	10505	10550

7. Unificación del modo de funcionamiento de las órdenes limitadas.

Las estrategias que utilicen órdenes limitadas van a llevar unificado su modo de funcionamiento tanto en *backtesting* como en *realtime.* Así, en los casos en los que un hueco puede dar un precio diferente al indicado en la orden limitada, se asume que se ejecuta en la apertura de la nueva barra.

8. Unificación en la detección de órdenes pasadas de precio.

Visual Chart pasa a unificar también los modos de detectar si una orden está pasada de precio sin distinguir el modo de ejecución de limitadas (Modo Versión 4 o Modo Versión 5).

Los entornos de programación de Visual Chart 6.

Introducción

En Visual Chart 6 tenemos la posibilidad de trabajar con estrategias de dos formas:

1. **Utilizando estrategias ya creadas.** Bien sea porque forman parte de la lista de estrategias públicos de Visual Chart o bien porque son estrategias compartidos por otros usuarios.
2. **Diseñando nuevas estrategias.** En este caso, se trataría de estrategias que sigan nuestras propias ideas, lo que nos va a permitir conocer sobre qué productos funciona mejor nuestra estrategia.

La segunda opción podemos resolverla a través de dos posibles métodos de programación:

1. A través de la Plataforma Visual (PDV).
2. A través de la programación .NET.

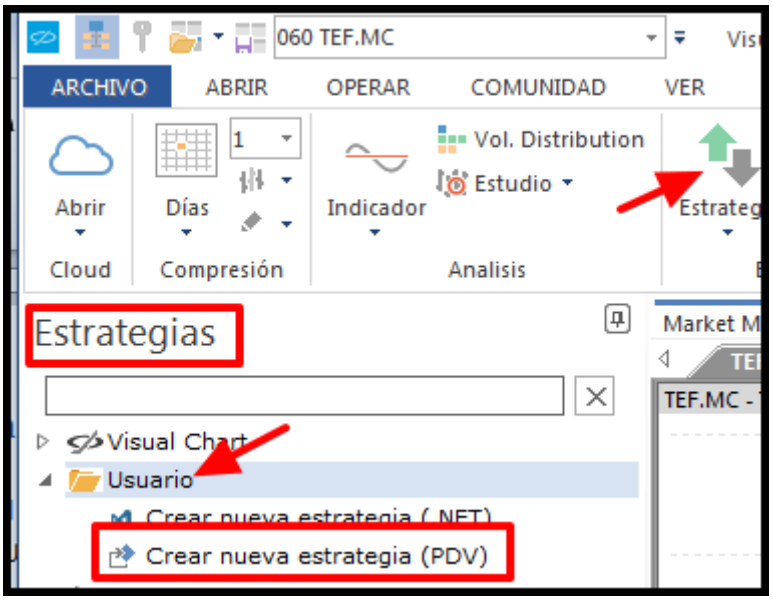
1. La Plataforma de Diseño Visual (PDV)

La Plataforma de Diseño Visual (PDV) **es un asistente que pertenece a Visual Chart.** A través de él podemos diseñar estrategias, indicadores y estudios sin necesidad de tener conceptos de programación.

Al diseñar una estrategia en PDV, una vez ésta queda registrada, Visual Chart genera automáticamente el código correspondiente en VB.NET. Por tanto, la PDV funciona como un

puente entre el usuario y la programación .NET, facilitando la labor de diseño para aquellos usuarios que no dispongan de los suficientes conocimientos.

La creación de un nuevo sistema mediante PDV se lleva a cabo a través del siguiente comando:



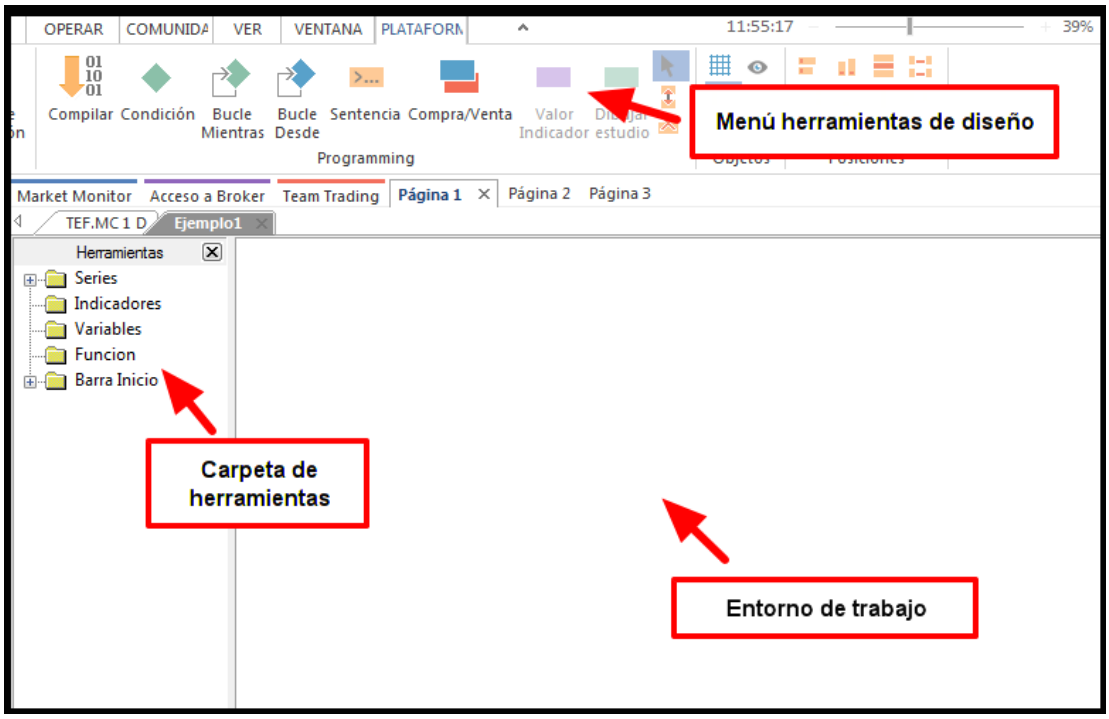
Una vez especificado el nombre del sistema, se abrirá el entorno de programación de la PDV.

Dentro de dicho entorno, definiremos tres zonas como las principales de ésta interfaz:

1. El entorno de trabajo
El entorno de trabajo es el espacio en blanco que aparece al abrirse la plataforma y es donde iremos dibujando el diagrama de flujo que representará a la estrategia.

2. Las carpetas de Herramientas
Estas carpetas contendrán a los distintos elementos que vamos a usar para montar la estrategia. Más adelante veremos cómo añadir elementos a dichas carpetas.

3. El menú de Herramientas de diseño de programación
El menú de Herramientas contiene a las distintas herramientas que usaremos para diseñar el diagrama de flujo.



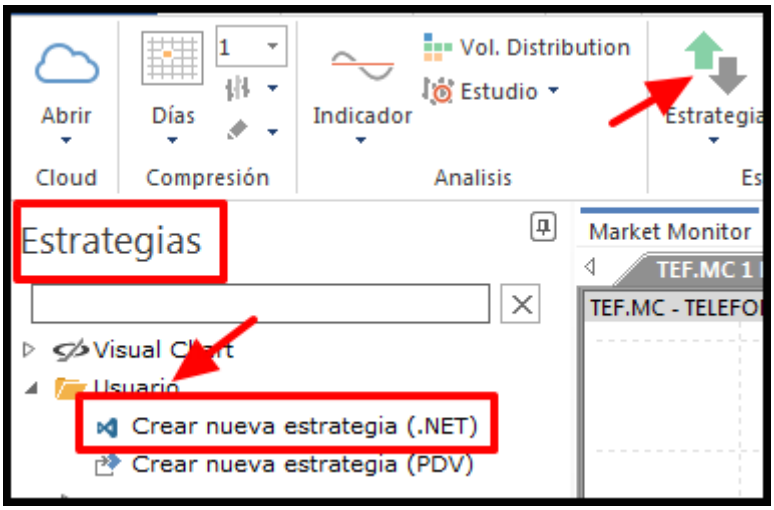
Más adelante entraremos en detalle acerca de cada una de estas zonas.

2. El editor de programación de Visual Studio (.NET)

Si tenemos nociones de programación o necesitamos desarrollar estrategias con cierta complejidad, disponemos de los lenguajes de programación .NET. Teniendo los conocimientos necesarios, esta opción es la más cómoda y versátil, ya que nos permite mayores posibilidades que la otra opción de programación (la Plataforma Visual).

El desarrollo de estrategias en .NET se hace a través del editor Visual Studio o bien Visual Studio Express. Esto dependerá de cómo haya configurado la instalación del programa el usuario.

La creación de un nuevo sistema mediante .NET se lleva a cabo a través del siguiente comando:



Una vez especificado el nombre de la estrategia, se abrirá Visual Studio (o Visual Studio Express).

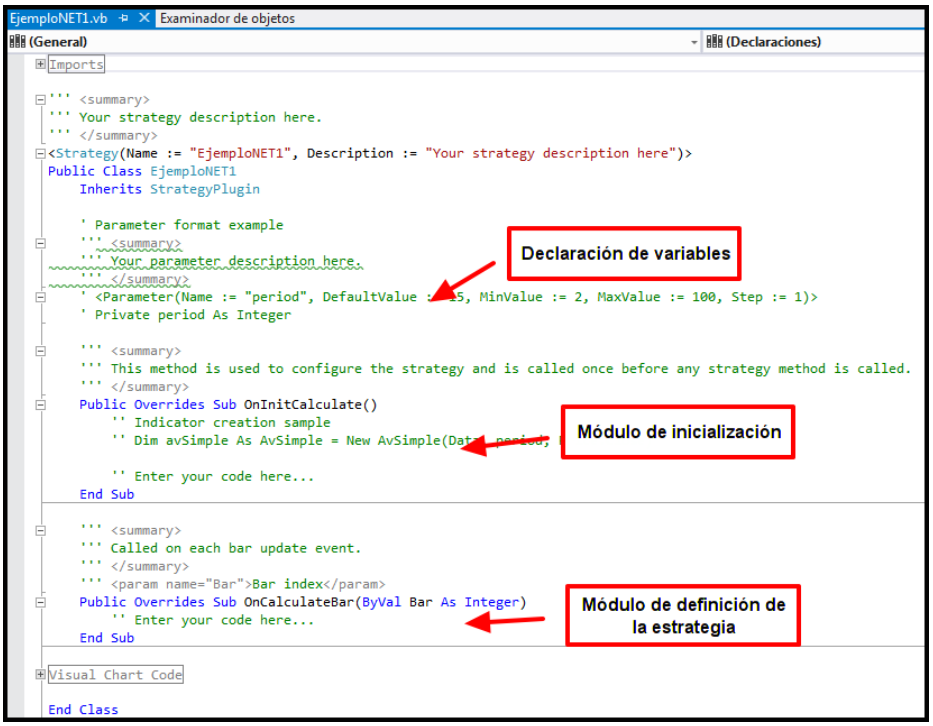
El editor se abrirá habiendo creado un proyecto con el nombre que le hayamos dado al sistema, incluyendo **una estructura pre configurada** sobre la que debemos montar nuestro sistema. Visual Chart genera por defecto esta estructura para facilitar la labor al programador, de manera que sólo tenga que preocuparse de incluir los elementos vinculantes a su estrategia.

Como decimos, nos debemos centrar en las partes sobre las que tenemos que escribir, pudiendo obviar el resto de módulos. A continuación, veamos las características de cada una de dichas partes.

1. Zona Declaración de Variables
Dedicada a la declaración de variables que vamos a usar en la estrategia, diferenciando entre las que son parámetros de la estrategia, y después todas las que se vayan a manejar.

2. Procedimiento OnInitCalculate
Módulo desde donde inicializamos variables, se generan los objetos vinculados a los indicadores, se asignan valores constantes, etc... A este procedimiento recurrirá el programa sólo una vez, antes de comenzar los cálculos sobre las barras del gráfico.

3. Procedimiento OnCalculateBar
Módulo donde se define la estrategia: reglas de entrada y salida, operadores, etc... A este procedimiento recurrirá el programa una vez por barra, y siempre cuando dicha barra ha finalizado.



Anexo 1. Diseño de estrategias enfocadas al Team Trading

Recomendaciones a la hora de definir los parámetros.

Antes de finalizar, queremos dedicar unas líneas a realizar algunas recomendaciones a la hora de definir los parámetros de sus estrategias para aquellos desarrolladores interesados en formar parte de **la comunidad Team Trading**.

1. Definir los parámetros en términos porcentuales

Cuando la estrategia incorpora parámetros tales como filtros, objetivos gananciales o stop de pérdidas, es recomendable definirlos en términos porcentuales, a fin de que a la hora de optimizar nuestra estrategia sobre cualquier mercado pueda adaptarse lo mejor posible a la correspondiente escala de precios.

```
Public Class ruptura_de_bandas
    Inherits StrategyPlugin

    ''' <summary>
    ''' Contratos por negocio.
    ''' </summary>
    <Parameter(Name:="Contratos", DefaultValue:=1, MinValue:=1, MaxValue:=1, Step:=1)>
    Private contracts As Integer

    ''' <summary>
    ''' Stop de pérdidas porcentual.
    ''' </summary>
    <Parameter(Name:="StopPerdidasPct", DefaultValue:=0.1, MinValue:=0.1, MaxValue:=0.7, Step:=0.1)>
    Private stoploss As Double

    ''' <summary>
    ''' Barras cálculo máximo y mínimo..
    ''' </summary>
    <Parameter(Name:="NBarras", DefaultValue:=10, MinValue:=2, MaxValue:=100, Step:=1)>
    Private nbars As Integer
```

2. Parámetros de Hora de Entrada y Hora de Salida en número de barras

Si trabajamos con estrategias *End Of Day* en las cuales aplicamos filtros horarios para determinar los intervalos de operativa, se recomienda que los parámetros que establecen los márgenes horarios vengan especificados por número de barras. Es decir, en lugar de especificar *InitTime* igual a 930 (por ejemplo), pasaríamos a definir este parámetro como *InitBar* igual a 2 (por ejemplo). Gracias a la nueva propiedad *TodayCurrentBar*, controlar si ha alcanzado o no a la barra correspondiente es sumamente sencillo.