Excel 2003 vs 2007: Programación VBA

Sánchez García, Juan Francisco jf.sanchez@upct.es
Bernal García, Juan Jesús juanjesus.bernal@upct.es
Martínez María Dolores, Soledad María soledad.martinez@upct.es
Departamento Métodos Cuantitativos e Informáticos
Universidad Politécnica de Cartagena

RESUMEN

Office 2007 ha supuesto un gran cambio en la interfaz gráfica de todas las aplicaciones que forman parte de esta suite: Access, Excel, PowerPoint, Word. En el presente trabajo se analizan cómo dichos cambios han afectado a la programación VBA en la hoja de cálculo Excel y se profundiza en los nuevos métodos introducidos con esta versión.

ABSTRACT

Office 2007 has been a big challenge in the UI (User Interface) of all applications that are part of this suite: Access, Excel, PowerPoint, Word. This paper analyzes how these changes have affected the VBA programming in Excel and deepened in the new methods introduced in this version.

***Palabras claves*:**
Programación; hoja de cálculo; Visual Basic para Aplicaciones.
***Keywords*:**
Programming; spreadsheet; Visual Basic for Applications
***Clasificación JEL (Journal Economic Literature)*:** C15; C88
***Área temática*:** Informática aplicada a los métodos cuantitativos.

1. INTRODUCCIÓN

La aparición en el año 2007 de la versión 12 del paquete informático Office® de Microsoft introdujo una serie de cambios en Excel, la mayor parte de los cuales son visuales. Así, a modo de resumen y siguiendo a Walkenbach (2007a) se pueden citar las siguientes:

- Nuevo interfaz de usuario.
- Aumento del número de filas por hoja pasando de 65.536 (2^{16}) filas a 1.048.576 (2^{20}).
- Aumento del número de columnas pasando de 256 (2^{8}) columnas a 16.384 (2^{14}).
- Aumento de la capacidad de cálculo: cantidad de memoria, niveles de ordenación, operaciones deshacer, elementos en autofiltros, caracteres por fórmula, argumentos por fórmula, etc.
- Nuevos formatos de ficheros.
- Tablas.
- Estilos y temas.
- Mejores opciones de gráficos.
- Nueva vista de diseño de página.
- Formato condicional mejorado.
- Características de colaboración.
- Comprobación de compatibilidad.
- Tablas dinámicas mejoradas.
- Nuevas funciones.

Uno de los cambios más visibles es la desaparición de la tradicional barra de menús habitual en toda aplicación informática y su sustitución por una "cinta de opciones" (*ribbon*) que, además de ser mucho más visual al estar integrada por iconos en lugar de los tradicionales comandos, supone una reorganización de las opciones existentes en Excel hasta la versión anterior.

2. EXCEL 2007 Y VBA

A priori, los cambios sufridos por Excel 2007 parecen no tener ninguna implicación en la programación de la hoja de cálculo mediante el uso de Visual Basic para Aplicaciones (VBA), como se venía efectuado hasta ahora. Aunque, en líneas generales esta afirmación es cierta, existen dos aspectos que han cambiado y que analizaremos a continuación:

2.1. Formato de ficheros

Desde las primeras versiones de Excel el tipo de archivo utilizado para almacenar las hojas de cálculo ha sido un formato propietario identificado con la extensión XLS (Excel Spreadsheet). Este formato almacena tanto simples hojas de cálculo como hojas de cálculo con módulos en VBA. Junto a este formato se utilizaba la extensión XLA para los complementos (Excel Add-in) que proporcionan nuevas funcionalidades a Excel a través de la programación de menús, herramientas y nuevas funciones y el formato XLT para las plantillas (Excel Template).

Con Excel 2007, Microsoft pasa a usar un nuevo conjunto de formatos de archivos denominados genéricamente "*Office Open XML*", basados en el estándar XML (Extensible Markup Language). Estos archivos, realmente son ficheros ZIP que contienen el documento guardado utilizando el formato XML y pueden ser leídos y escritos por cualquier aplicación, al ser formatos abiertos y por tanto ser públicas su estructura y características. Este formato, en cierta medida, fue la respuesta de Microsoft al formato *OASIS* (*Open Document Format for Office Applications*) utilizado por la suite informática *OpenOffice.org* (*OOo*) y a las exigencias realizadas por diversas administraciones públicas de dejar de usar formatos propietarios.

El cambio de codificación realizado por Microsoft ha conllevado así mismo la creación y separación de diversos formatos:

- XLSX: Libro que no contiene VBA
- XLSM: Libro que contiene VBA
- XLTX: Plantilla que no contiene VBA

- XLTM: Plantilla que contiene VBA
- XLSA: Archivo de complementos
- XLSB: Archivo binario similar al anterior XLS adaptado a las novedades de Office 2007
- XLSK: Archivo de copia de seguridad

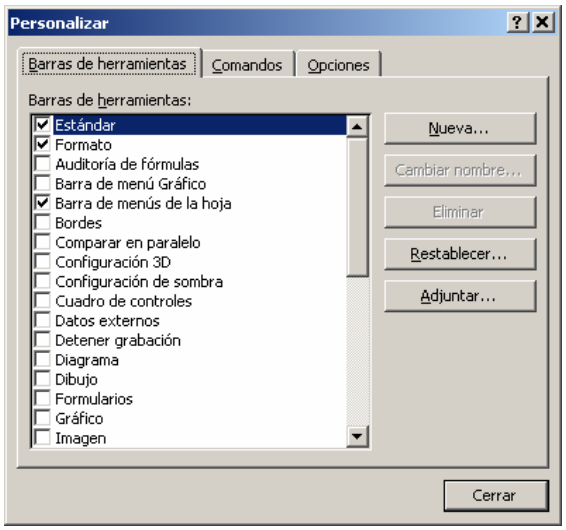
Resulta evidente, que Microsoft ha decidido separar drásticamente los archivos que contienen programación en VBA (genéricamente llamada Macros) de los que no la contienen. Este hecho, posibilita así mismo que sea más sencillo limitar la programación de virus de scripts almacenados en archivos de Excel, ya que ahora el tipo de archivo identificará fácilmente a los archivos potencialmente peligrosos.

2.2. Cinta de opciones

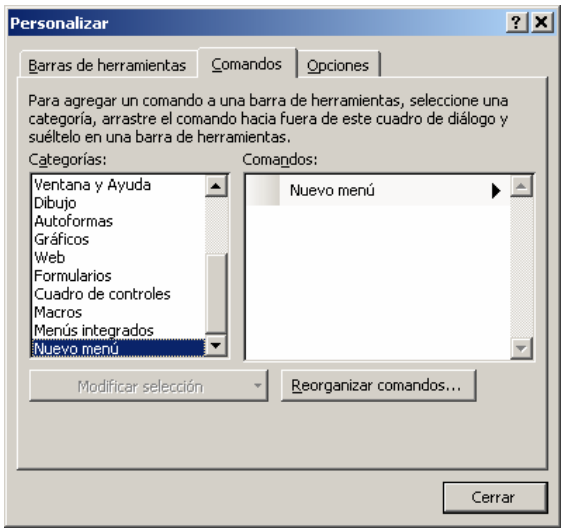
La sustitución de la barra de menús por la cinta de opciones ha provocado que toda la programación que hasta ahora se efectuaba para la creación de barras de menús personalizadas sufra las consecuencias de este cambio de filosofía en las aplicaciones existentes en la suite Office 2007.

La cinta de opciones es un elemento más visual que el anterior sistema de menús y barras de herramientas, y ocupa algo más de espacio del que utilizaban dichas barras. Sin embargo, para aprovechar más el área de trabajo puede ser minimizada utilizando un comando de menú contextual o pulsando la combinación de teclas Ctrl+F1. Junto con la cinta de opciones, para dar cabida a las operaciones más habituales, existe en la misma barra de título de la ventana en la que se ejecuta Excel aparece un nuevo elemento denominado “barra de herramientas de acceso rápido” que puede ser fácilmente personalizado por el usuario.

Hasta la versión 2003 era posible crear una barra de herramientas nueva utilizando el comando “Personalizar” dentro del menú “Herramientas”:



Así, mismo también en ese mismo cuadro de diálogo era posible crear un nuevo menú utilizando para ello la pestaña "Comandos":



Este procedimiento tenía un pequeño defecto y era que la barra de herramientas o menú creado permanecía automáticamente en la aplicación Excel con lo que no estaba asociado a un fichero Excel determinado. De esta forma, si en un libro Excel se incluía una programación VBA no era posible asignarla a los comandos de menú o barras de herramientas salvo que se crearan de forma manual en la instalación de Excel donde se fuera a utilizar. La solución pasaba por que fuese el propio código VBA el que creara el correspondiente menú y lo eliminara al finalizar.

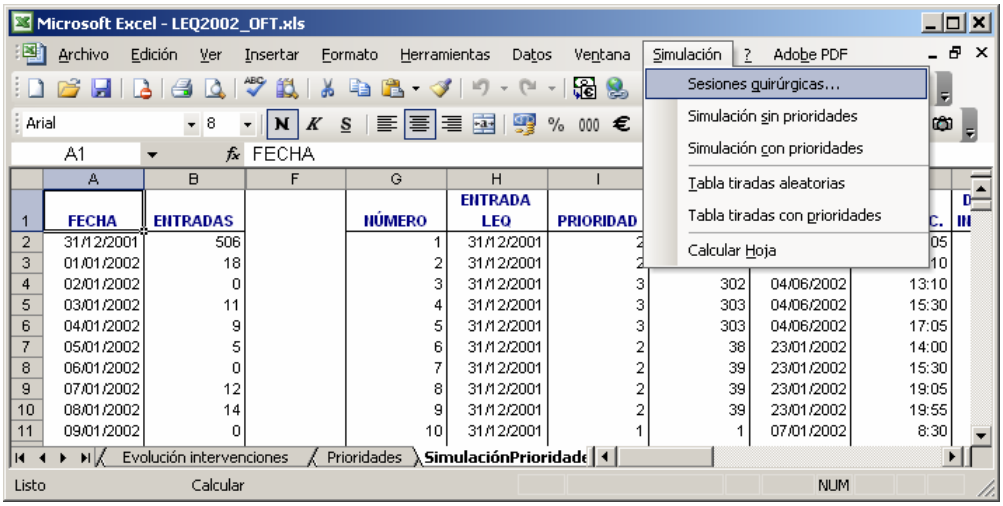
2.3. Caso práctico

Como ejemplo de lo expuesto, a continuación se recoge la creación y eliminación de un menú denominado “Simulación” que incluye 6 comandos llamados “Sesiones quirúrgicas”, “Simulación sin prioridades”, “Simulación con prioridades”, “Tabla tiradas aleatorias” “Tabla tiradas con prioridades” y ”Calcular hoja” (dicha programación forma parte del trabajo empírico recogido en el trabajo de Bernal García, Martínez María Dolores y Sánchez García).

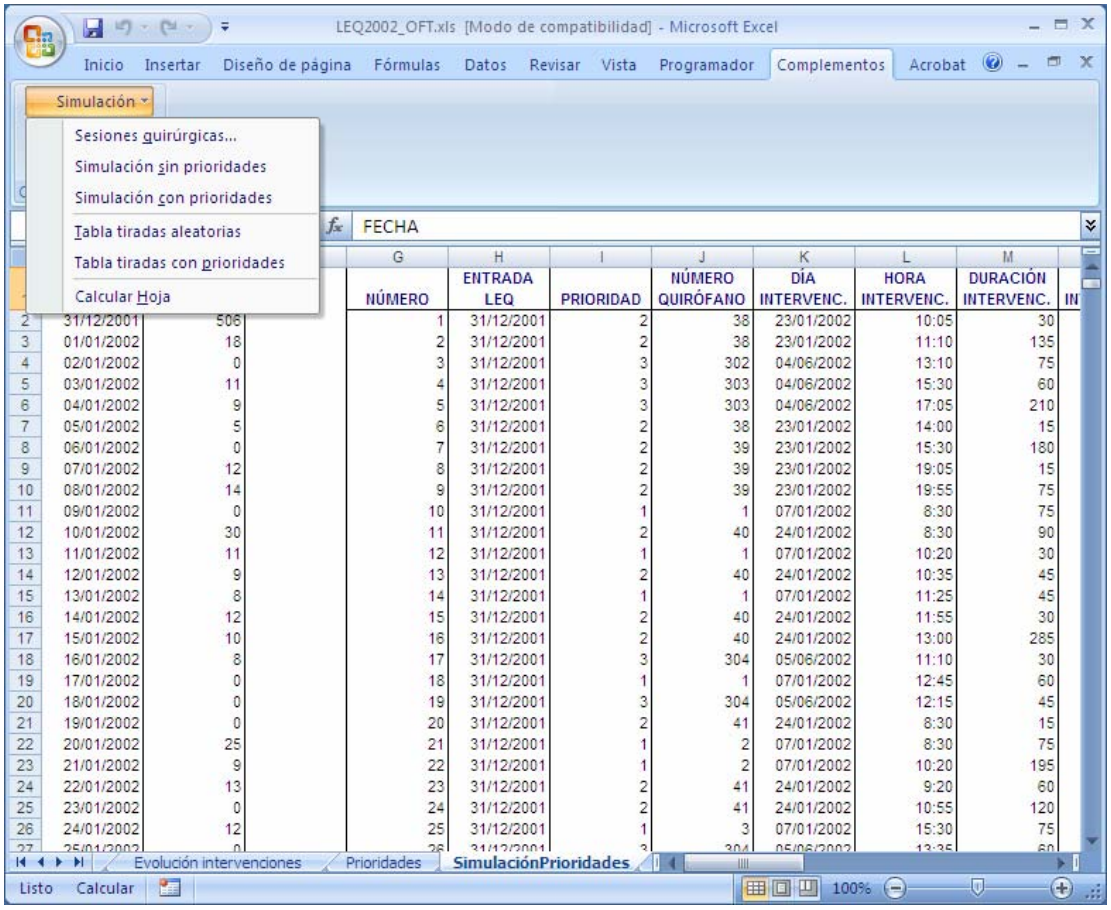
```
Sub AñadirMiMenu()

Dim HelpMenu As CommandBarControl
Dim MiMenu As CommandBarPopup
Dim MenuItem As CommandBarControl
Dim SubmenuItem As CommandBarButton

Set HelpMenu = CommandBars(1).FindControl(ID:=30010)

Set  MiMenu  =  CommandBars(1).Controls.Add(Type:=msoControlPopup,  before:=HelpMenu.Index,
temporary:=True)

MiMenu.Caption = "&Simulación"

Set MenuItem = MiMenu.Controls.Add(Type:=msoControlButton)
With MenuItem
    .Caption = "Sesiones &quirúrgicas..."
    .OnAction = "FormularioSesionesQuirurgicas"
End With

Set MenuItem = MiMenu.Controls.Add(Type:=msoControlButton)
With MenuItem
    .Caption = "Simulación &sin prioridades"
    .OnAction = "CalcularTablas"
End With

Set MenuItem = MiMenu.Controls.Add(Type:=msoControlButton)
With MenuItem
    .Caption = "Simulación &con prioridades"
    .OnAction = "CalcularTablasPrioridades"
End With

Set MenuItem = MiMenu.Controls.Add(Type:=msoControlButton)
With MenuItem
    .Caption = "&Tabla tiradas aleatorias"
    .OnAction = "TablaTiradas"
    .BeginGroup = True
End With

Set MenuItem = MiMenu.Controls.Add(Type:=msoControlButton)
With MenuItem
    .Caption = "Tabla tiradas con &prioridades"
    .OnAction = "TablaTiradasPrioridades"
End With

Set MenuItem = MiMenu.Controls.Add(Type:=msoControlButton)
With MenuItem
    .Caption = "Calcular &Hoja"
    .OnAction = "CalcularHoja"
    .BeginGroup = True
End With

End Sub

Sub QuitarMiMenu()
    On Error Resume Next
    CommandBars(1).Controls("Simulación").Delete
```

```
End Sub
```

En la siguiente pantalla se muestra el efecto de dicha programación VBA en la versión 2003 de Excel:



Sin embargo, cuando abrimos el mismo fichero con Excel 2007 el resultado final es muy diferente, ya que el menú creado pasa a incluirse dentro de la cinta “Complementos” y no es tan intuitiva su localización:



De una forma similar, es posible "recuperar" la antigua barra de menús con toda su funcionalidad, utilizando para ello la técnica descrita en *The Spreadsheet Page* (http://spreadsheetpage.com/index.php/tip/old_style_menus_in_excel_2007) y realizando la correspondiente adaptación al idioma español para asegurar su correcto funcionamiento:

```
Sub MakeOldMenus()
   Dim cb As CommandBar
   Dim cbc As CommandBarControl
   Dim OldMenu As CommandBar

'  Delete it, if it exists
   On Error Resume Next
   Application.CommandBars("Old Menus").Delete
   On Error GoTo 0

'  Create an old-style toolbar
'  Set the last argument to False for a more compact menu
   Set OldMenu = Application.CommandBars.Add("Old Menus", , True)

'  Copy the controls from Excel's "Built-in Menus" shortcut menu
   With CommandBars("Built-in Menus")
     .Controls("&Archivo").Copy OldMenu
     .Controls("&Edición").Copy OldMenu
     .Controls("&Ver").Copy OldMenu
     .Controls("&Insertar").Copy OldMenu
     .Controls("&Formato").Copy OldMenu
     .Controls("&Herramientas").Copy OldMenu
```

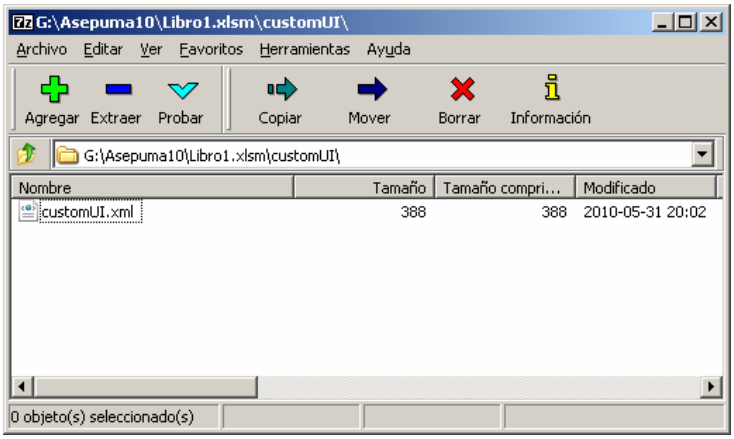
```
     .Controls("&Datos").Copy OldMenu
     .Controls("Ve&ntana").Copy OldMenu
     .Controls("&?").Copy OldMenu
   End With

'   Make it visible. It appears in the Add-Ins tab
   Application.CommandBars("Old Menus").Visible = True
End Sub
```

Lo ideal sería conseguir que se pudieran crear estos menús dentro de la cinta de opciones propia de Excel, al igual que los comandos que incorpora de forma directa Excel 2007, proporcionando una apariencia más profesional. Esta labor que era relativamente sencilla con Excel 2003, y anteriores, se hace bastante más complicada en esta nueva versión.

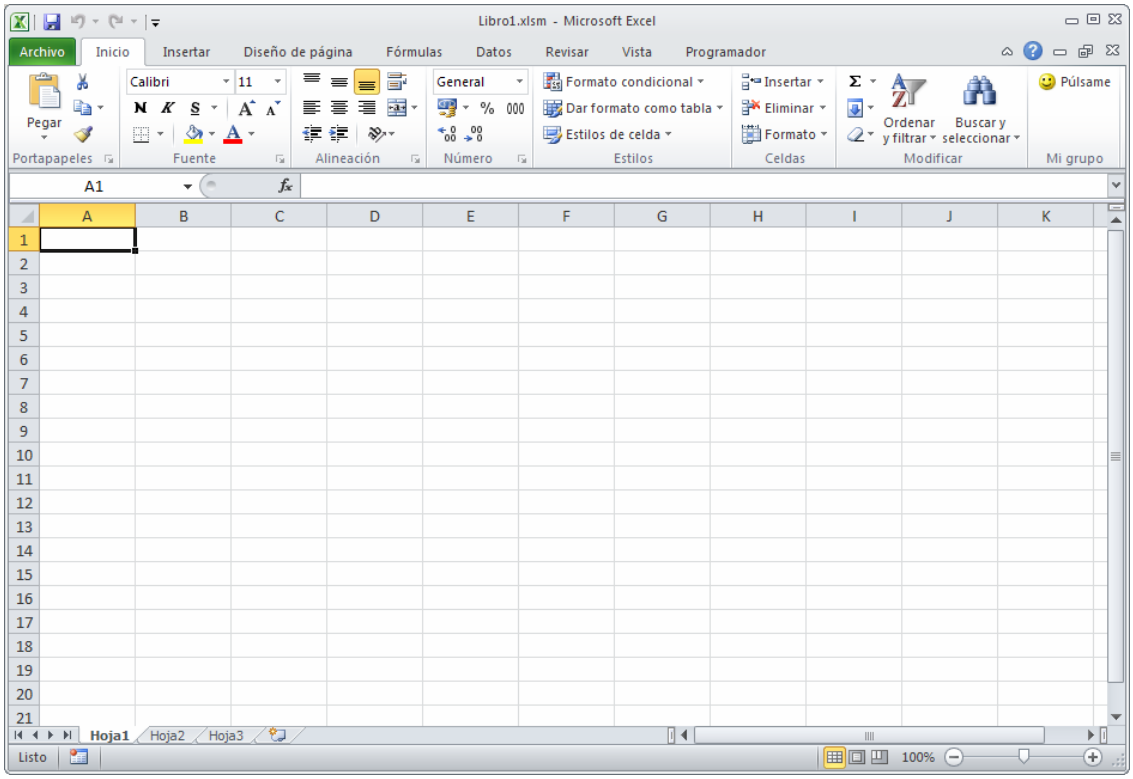
Lo primero que debe quedar claro es que no es posible modificar la cinta de opciones desde VBA (Walkenbach, 2007b). En su lugar, es necesario escribir instrucciones RibbonX e insertar las mismas en el fichero de Excel, cosa que se debe hacer desde fuera de Excel. En este punto, cabe recordar que los nuevos ficheros de Excel son en realidad ficheros ZIP, con lo que es posible agregar ficheros en su interior utilizando programas como WinZip, WinRAR o 7zip. El uso de VBA en 2007 queda limitado a programar las acciones que realizarán las nuevas opciones de la cinta de opciones. Por tanto, lo primero que tenemos que hacer es conocer qué es RibbonX. RibbonX es código XML que describe los controles, dónde se colocan, cuál es su aspecto y qué pasa al activarlos. Por ejemplo, el siguiente código RibbonX añade un nuevo grupo llamado “Mi grupo” al final de la cinta “Inicio” e incluye en él un control llamado “Púlsame” que al pulsarlo llama a la acción “Macro1” que tendrá que ser programada en el editor de VBA.

```
<customUI xmlns="http://schemas.microsoft.com/office/2006/01/customui">
  <ribbon>
    <tabs>

      <tab idMso="TabHome" >
        <group id="customGroup1" label="Mi grupo" insertAfterMso="GroupEditingExcel">

          <button  id="customButton1"  label="Púlsame"  size="normal"  onAction="Macro1"
imageMso="HappyFace" />

        </group>
      </tab>

    </tabs>
  </ribbon>
</customUI>
```

El código anterior tiene que ubicarse en un fichero llamado en *customUI.xml* que se almacena en la carpeta *customUI*, quedando de la siguiente forma al ser abierto con 7zip:



El resultado final al abrir el archivo en Excel 2010 es el siguiente:



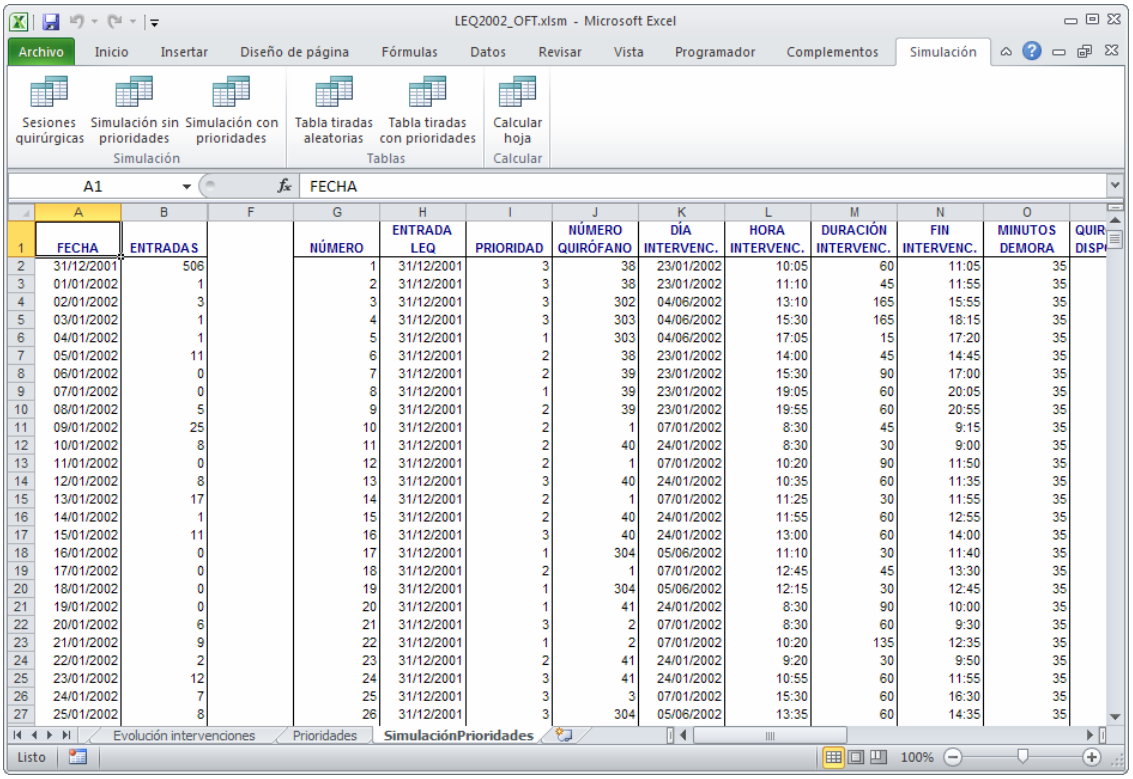
Como ha quedado demostrado, el procedimiento no es sencillo y, por esta razón, han aparecido diversas soluciones para simplificar al menos la inclusión del archivo XML con el código RibbonX dentro del archivo de Excel, evitando tener que crear el archivo de forma manual con un editor y su posterior inclusión con un compresor en la carpeta correspondiente del archivo Excel. Entre las soluciones más utilizadas se encuentra *Custom UI Editor*, software realizado por Microsoft y que puede ser descargado de forma gratuita de la red (http://openxmldeveloper.org/archive/2006/05/26/CustomUIeditor.aspx).

Volviendo a nuestro ejemplo, será necesario por tanto crear una pestaña que podemos denominar "Simulación" y dentro de la cual incluiremos los 6 comandos que hasta ahora teníamos programados como comandos de menú. Ya que inicialmente los teníamos separados en 3 grupos, lo que haremos será crear estos grupos dentro de dicha pestaña.

El código XML con el que podemos conseguir el efecto deseado es el que se recoge a continuación. Para simplificar, se le ha asignado el mismo icono (*QuerySelectQueryType*) a todos los comandos, aunque también se podrían haber dejado sin imagen. Además, para completar mejor la cinta, el tamaño del comando se ha establecido en *large*.

```
<customUI xmlns="http://schemas.microsoft.com/office/2006/01/customui">
  <ribbon>
    <tabs>

      <tab id="CustomTab" label="Simulación">
        <group id="customGroup1" label="Simulación">

          <button id="customButton1" label="Sesiones quirúrgicas" size="large"
onAction="FormularioSesionesQuirurgicas" imageMso="QuerySelectQueryType" />
          <button id="customButton2" label="Simulación sin prioridades" size="large"
onAction="CalcularTablas" imageMso="QuerySelectQueryType" />
          <button id="customButton3" label="Simulación con prioridades" size="large"
onAction="CalcularTablasPrioridades" imageMso="QuerySelectQueryType" />

        </group>

        <group id="customGroup2" label="Tablas">

          <button id="customButton4" label="Tabla tiradas aleatorias" size="large"
onAction="TablaTiradas" imageMso="QuerySelectQueryType" />
          <button id="customButton5" label="Tabla tiradas con prioridades" size="large"
onAction="TablaTiradasPrioridades" imageMso="QuerySelectQueryType" />

        </group>

        <group id="customGroup3" label="Calcular">

          <button id="customButton6" label="Calcular hoja" size="large"
onAction="CalcularHoja" imageMso="QuerySelectQueryType" />
        </group>

      </tab>

    </tabs>
  </ribbon>
</customUI>
```

El resultado final de este código RibbonX es el que a continuación se reproduce.



FECHA	ENTRADAS		NÚMERO	ENTRADA LEQ	PRIORIDAD	NÚMERO QUIRÓFANO	DÍA INTERVENC.	HORA INTERVENC.	DURACIÓN INTERVENC.	FIN INTERVENC.	MINUTOS DEMORA	QUIR DISP
31/12/2001	506		1	31/12/2001	3	38	23/01/2002	10:05	60	11:05	35	
01/01/2002	1		2	31/12/2001	3	38	23/01/2002	11:10	45	11:55	35	
02/01/2002	3		3	31/12/2001	3	302	04/06/2002	13:10	165	15:55	35	
03/01/2002	1		4	31/12/2001	3	303	04/06/2002	15:30	165	18:15	35	
04/01/2002	1		5	31/12/2001	1	303	04/06/2002	17:05	15	17:20	35	
05/01/2002	11		6	31/12/2001	2	38	23/01/2002	14:00	45	14:45	35	
06/01/2002	0		7	31/12/2001	2	39	23/01/2002	15:30	90	17:00	35	
07/01/2002	0		8	31/12/2001	1	39	23/01/2002	19:05	60	20:05	35	
08/01/2002	5		9	31/12/2001	2	39	23/01/2002	19:55	60	20:55	35	
09/01/2002	25		10	31/12/2001	2	1	07/01/2002	8:30	45	9:15	35	
10/01/2002	8		11	31/12/2001	2	40	24/01/2002	8:30	30	9:00	35	
11/01/2002	0		12	31/12/2001	2	1	07/01/2002	10:20	90	11:50	35	
12/01/2002	8		13	31/12/2001	3	40	24/01/2002	10:35	60	11:35	35	
13/01/2002	17		14	31/12/2001	2	1	07/01/2002	11:25	30	11:55	35	
14/01/2002	1		15	31/12/2001	2	40	24/01/2002	11:55	60	12:55	35	
15/01/2002	11		16	31/12/2001	3	40	24/01/2002	13:00	60	14:00	35	
16/01/2002	0		17	31/12/2001	1	304	05/06/2002	11:10	30	11:40	35	
17/01/2002	0		18	31/12/2001	2	1	07/01/2002	12:45	45	13:30	35	
18/01/2002	0		19	31/12/2001	1	304	05/06/2002	12:15	30	12:45	35	
19/01/2002	0		20	31/12/2001	1	41	24/01/2002	8:30	90	10:00	35	
20/01/2002	6		21	31/12/2001	3	2	07/01/2002	8:30	60	9:30	35	
21/01/2002	9		22	31/12/2001	1	2	07/01/2002	10:20	135	12:35	35	
22/01/2002	2		23	31/12/2001	2	41	24/01/2002	9:20	30	9:50	35	
23/01/2002	12		24	31/12/2001	3	41	24/01/2002	10:55	60	11:55	35	
24/01/2002	7		25	31/12/2001	3	3	07/01/2002	15:30	60	16:30	35	
25/01/2002	8		26	31/12/2001	3	304	05/06/2002	13:35	60	14:35	35	

3. CONCLUSIONES

En el presente trabajo hemos abordado cómo afectan los cambios que Microsoft introdujo con Office 2007 a la programación en VBA de la hoja de cálculo, así como a la creación y mantenimiento de barras de herramientas y barras de menús. En anteriores trabajos de los autores quedó evidente la utilidad de la programación en VBA dentro de Excel para la creación y posterior desactivación de menús específicos para cada herramienta desarrollada, haciendo que fuera independiente de la versión de Excel utilizada, hecho este que no es posible aplicarlo de forma directa a Excel 2007.

Si bien, por razones de compatibilidad, la anterior programación realizada pensando en el enfoque de Office 2003 y anteriores sigue siendo válida, es cierto que pierde parte de su funcionalidad y no aprovecha al máximo las nuevas posibilidades que Office 2007 brinda como la cinta de opciones en sustitución de las antiguas barras de menús.

Por último, y a título de resumen, podemos citar los cambios que han supuesto Office 2007 y Office 2010 en cuanto a la creación de menús personalizados y la utilización de VBA:

	Office 2003	**Office 2007**
Archivos de datos	XLS	XLSX
Archivos con macros	XLS	XLSM
Creación de menús	En la aplicación	En el archivo de datos
Forma de creación de menús	Manual y VBA	RibbonX
Automatización de menús	Sí, con VBA	No

4. REFERENCIAS BIBLIOGRÁFICAS

- BERNAL GARCÍA, Juan Jesús; MARTÍNEZ MARÍA DOLORES, Soledad María; SÁNCHEZ GARCÍA, Juan Francisco (2009). "La hoja de cálculo como apoyo a la simulación de los fenómenos de espera con prioridades. Una aplicación a la sanidad", *Recta*, núm. 10, pp. 77-104.
- GETZ, Ken. (2007). *Uso de RibbonX desde Visual Basic*. http://msdn.microsoft.com/es-es/magazine/cc163410.aspx
- WALKENBACH, John (2007a). *Excel® 2007 Bible*. Indianapolis, Indiana: Wiley Publishing, Inc.
- WALKENBACH, John (2007b). *Excel 2007 Power Programming with VBA*. Indianapolis, Indiana: Wiley Publishing, Inc.