Object lesson

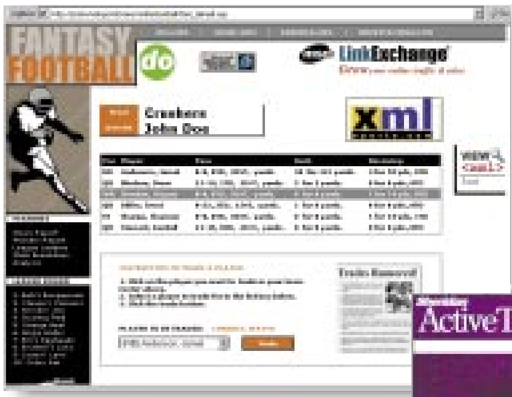
Tim Anderson looks at object control. And, setting up a mailslot server.

The holy grail of distributed objects is seamless interaction across the boundaries of geography, operating system and programming language — and, to some extent, this is now possible. For instance, Java-to-Java communication via Remote Method Invocation (RMI) works well cross-platform, while CORBA object brokers let objects communicate across language boundaries. Microsoft's COM and DCOM is used almost exclusively on Windows, but that in itself is a substantial coverage considering the ubiquity of the operating system.

Just because you can do something, though, doesn't make it easy. In the real world, getting objects to communicate across system boundaries is fraught with difficulty. For example, while the idea of instantiating a remote object and calling its methods sounds good, you can easily run into problems with timing. If the calling process is waiting for a response from an object over a wide area network, with all that implies in terms of frailty of linkage, then systems can easily get sluggish or fail. There are answers such as using asynchronous communication via message queues, but the point remains that remote object invocation is non-trivial.

Microsoft is known to be a keen promoter of COM and DCOM but is now singing a somewhat different tune. At the Business Applications conference in London, earlier this year, the presenters were touting XML as the simple solution to all kinds of data interchange problems. XML is short for eXtensible Markup Language, and its key difference from HTML is that you can define a data structure, or other kinds of document structure, within the document. That enables applications to exchange data through XML, sent over the wire via HTTP, just like a web page.

You can also exploit its relationship with HTML [Fig 1]; for instance, parsing XML for display in HTML over the web. Its great advantage is simplicity. All that the applications need to understand is how to generate and read XML data.



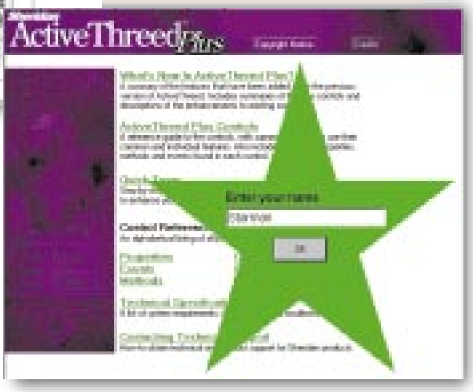
◀Fig 1 XML Sports, a demo application that shows XML and HTML used together to create a data-driven browser interface

Although Microsoft is promoting XML, it is unlikely to be hijacked as a Microsoft technology. XML is in the process of being standardised by the W3C, the same committee which oversees HTML, and Sun is also working on XML support in Java. There are advantages to Microsoft though, partly on an 'anything but Java' basis, and partly because Internet Explorer 5.0 is the only browser with serious XML support.

With many possible applications, expect to hear a lot more about XML in the coming months. Take a look at the links in the *PCW Contacts* box (p252) to explore some of the available resources.

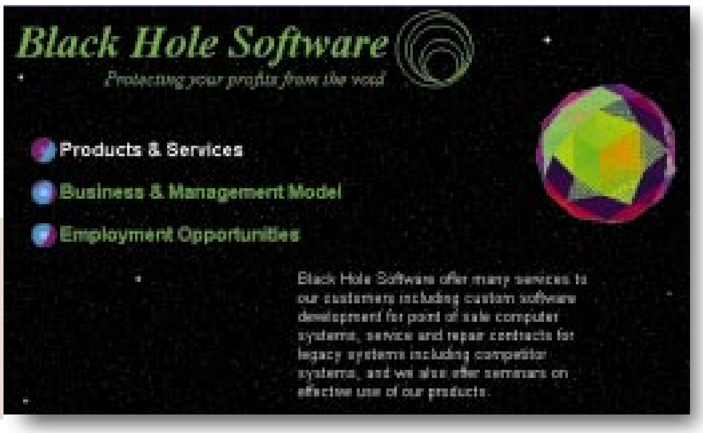
■ Active Threed Plus

VB veterans will remember THREED.VBX, a natty item that gave controls a three-dimensional look in the days when such things were cool. To sell the new Active Threed package, Sheridan needs to offer more than shadow effects and in fact the name is now misleading. The bundle contains 11 controls which handle animation, transition effects, non-rectangular buttons and more. The most spectacular is the splash control [Fig 2] which allows forms to take on the shape of a picture. After years of gazing at rectangular windows, the splash effect has real impact. It's not only for splash screens, though — you can use it for a whole application if you want, although users will grumble if they don't have a title bar to grab on to.



▼Fig 2 The spectacular splash control lets you create non-rectangular windows

The next new control is SSResizer, which is an elastic control: you place it on a form and any controls on the form


Black Hole Software
Protecting your profits from the void
Products & Services
Business & Management Model
Employment Opportunities

Black Hole Software offer many services to our customers including custom software development for point of sale computer systems, service and repair contracts for legacy systems including competitor systems, and we also offer seminars on effective use of our products.

▶Fig 3 This presentation screen, created with Active Threed, is full of animation. The globe spins and the stars twinkle in the sky

[FIG 6] A MailSlot server

```
Public hMailslot As Long
Private Sub Form_Load()

hMailslot = CreateMailslot("\\.✓
\mailslot\MyMSlot", 255, MAILSLOT✓
_WAIT_FOREVER, 0)
If hMailslot = INVALID_HANDLE_✓
VALUE Then
frmMailSlot.Caption = "Failed to✓
 create mailslot"
Else
frmMailSlot.Caption = "Mailslot✓
 is active"
End If
End Sub

Private Sub ReadSlot()
Dim iResult As Long
Dim sMessage As String
Dim iNextMessageSize As Long
Dim iNumMessages As Long
Dim iBytesRead As Long

Do
iResult = GetMailslotInfo✓
(hMailslot, 255, iNextMessageSize,✓
 iNumMessages, 0)

If iResult = 0 Then
frmMailSlot.lbStatus.Caption =✓
 "GetMailSlotInfo failed"
Exit Sub
End If

If iNextMessageSize = MAILSLOT_✓
NO_MESSAGE Then
frmMailSlot.lbStatus.Caption =✓
 "No message is waiting"
Exit Sub
End If

If iNumMessages <> 0 Then
sMessage = Space$(256) 'allocate✓
 buffer
iResult = ReadFile(hMailslot,✓
 ByVal sMessage, iNextMessageSize,✓
 iBytesRead, 0)

  If iResult = 0 Then
  frmMailSlot.lbStatus.Caption =✓
 "ReadFile failed"
  Exit Sub
  End If

lstMessages.AddItem (sMessage)
End If Loop Until iNumMessages =✓
 0 End Sub
```

(Listing contd. above, right)

[FIG 6 CONTD]

```
Private Sub Form_Unload(Cancel✓
 As Integer)

If hMailslot <> INVALID_✓
HANDLE_VALUE Then
CloseHandle (hMailslot)
End If
End Sub

Private Sub Timer1_Timer()
ReadSlot
End Sub
```

(*Key:* ✓ *code string continues*)

automatically stretch when the user resizes it. There are other elastic controls but an unusual feature of this one is that the font optionally resizes with the controls. I'm not sure that this is desirable though, and would also argue that a Java-style layout manager in VB would be more useful for truly resize-friendly applications. It isn't that you can't achieve the results in VB, just that it takes a lot of careful work to get it right.

SSScroll lets you have a scrolling client area on a form so you can fit in lots of controls. It works at design-time and at run-time, and works well except that scrolling forms are hardly user-friendly. A better idea is SSTransition, offering 37 transition effects like BlindsVertical, Dissolve, or CheckerBoardDown. This is normally the field of presentation graphics but could be an attractive feature in certain kinds of application, like a demonstration or tutorial.

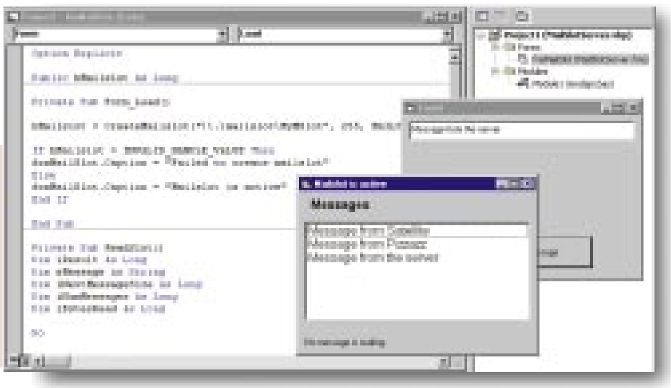
Other controls are enhanced. The SSCommand and SSRibbon button controls can now be non-rectangular. The Splitter control has Save and Restore methods for easy persistence of user customisation. Animation effects are improved in all the controls that can use them, so for example you can vary the speed while it is running, or jump to a specific frame. If you need some interesting graphic effects in your Visual Basic application, then this package could well be the answer. But even though Sheridan is still strongly VB-focused there is no guarantee that controls will work in other environments such as Delphi or Visual FoxPro. This is a missed opportunity, given the wide support for ActiveX controls. There are some notes about Visual C++ and Internet Explorer, so there is hope if those are your targets.

A question of mailslots

Reader Vincent Wong asks: 'Can you tell me how to use mailslots in VB (5.0)? I have succeeded in creating one using the API CreateMailSlot. I have also created a file in the mailslot using the API CreateFile, but I can't get WriteFile to create a message in the mailslot.'

A mailslot is so named because it receives messages but does not post them. In operation it works like a virtual directory. A server application creates a mailslot so clients can put messages into it using file-handling functions. The clients can be on the same machine or elsewhere on the network. The WinPopup utility that comes with some versions of Windows uses a mailslot. It's an easy way to send messages around a network, say for logging purposes. You can have more than one server using the same mailslot name but running on different machines. Then, you can use WriteFile with a wildcard character so that the message is broadcast to all machines on the domain [Fig 4].

Here's a simple example: The first task is to set up a mailslot server. This is an application which creates a mailslot and can read messages from it:



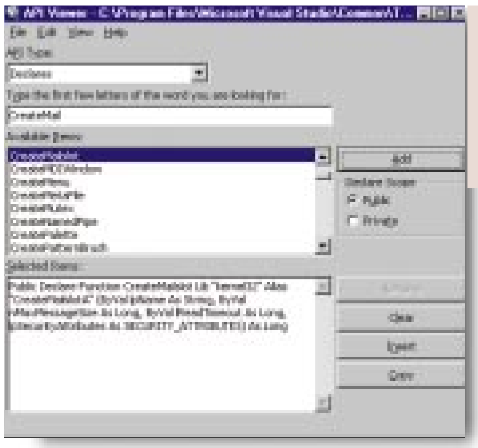
▸FIG 4 FUN WITH MAILSLOTS: MESSAGES ARRIVING FROM ALL OVER THE NETWORK

[FIG 7]

The mailslot client

```
Private Sub Command1_Click()
Dim iResult As Long
Dim sMessage As String
Dim hFile As Long
Dim iBytesWritten As Long

hFile = CreateFile("\\*\mailslot\MyMSlot",✓
 GENERIC_WRITE, FILE_SHARE_READ, 0,✓
 OPEN_EXISTING, FILE_ATTRIBUTE_NORMAL, 0)

If hFile = INVALID_HANDLE_VALUE Then
MsgBox "Could not open mailbox"
Exit Sub
End If

sMessage = txtMessage.Text
iResult = WriteFile(hFile, ByVal sMessage,✓
 Len(sMessage) + 1, iBytesWritten, 0)

If iResult = 0 Then
MsgBox "Error writing to file"
End If

iResult = CloseHandle(hFile)
End Sub
```

(*Key:* ✓ *code string continues*)



Public Declare Function CreateMailslot Lib "kernel32" Alias "CreateMailslotA" (ByVal lpName As String, ByVal nMaxMessageSize As Long, ByVal lReadTimeout As Long, lpSecurityAttributes As SECURITY_ATTRIBUTES) As Long

◀Fig 5 **The API Viewer is useful, but do not trust it too far**

☛ **Start** a new Visual Basic project and use the API viewer [Fig 5] to declare the following functions:

```
CreateMailSlot
GetMailSlotInfo
CreateFile
ReadFile
WriteFile
CloseHandle
```

☛ **You will also need** several constants:

```
MAILSLOT_NO_MESSAGE
MAILSLOT_WAIT_FOREVER
INVALID_HANDLE_VALUE
FILE_SHARE_READ
FILE_SHARE_WRITE
FILE_ATTRIBUTE_NORMAL
GENERIC_WRITE
OPEN_EXISTING
```

If you put these declares in a module, you can use it in both client and server projects. Note that I haven't bothered to include the type of declarations for SECURITY_ATTRIBUTES or OVERLAPPED, although these are parameters for CreateMailSlot and ReadFile respectively — there is no need because the parameters can be null. To persuade Visual Basic to accept a zero parameter though, you need to change the declaration from say, 'lpOverlapped as OVERLAPPED' to 'lpOverlapped as Long'.

☛ **Now add** a list box (lstMessages), a label (lbStatus) and a Timer to the form (frmMailSlot). The listbox will display messages received by the mailslot, while the label displays a status message. Fig 6 (p251) shows the code behind the form for the mailslot server.

The second parameter of CreateMailSlot is MaxMessageSize, and I've used it here to limit the size of a message to 255 bytes. In any case, there's a limit of 400 bytes for messages broadcast around a network. The server works by creating the mailslot when the form loads. A timer calls the ReadSlot procedure at whatever interval you choose. This procedure searches the mailslot with GetMailSlotInfo and reads any messages.

The example client application has a text box and a button. Clicking the button sends the contents of the text box to the mailslot. Fig 7 shows the code. It is just a matter of creating the virtual file, writing to it, and then closing it.

In both these examples, as always when working with the Windows API, you need to be careful about whether parameters are passed by value or by reference. The confusing aspect is that ByVal has a special meaning when used with Visual Basic strings. If you pass a VB string ByVal, then VB makes a null-terminated copy of the string and passes the DLL a pointer to that copy. When the function returns, this string is copied back to the VB variable. It's confusing, as it has the effect of a parameter passed by reference, not by value. You cannot rely on the API viewer to get this right for you; an example being the WriteFile function where you will have to add ByVal for the second parameter yourself. Perhaps this is why it didn't work for Vincent Wong.

■ Mailslot problems

You'd think that an established API function like this would be free of bugs by now. Unfortunately not. The Windows 95 or 98 API only recognises 8.3 names for mailslots, while NT accepts long names. The result is that if you do as I did and create a mailslot called MyMailSlot, you will find it works fine between NT machines, or between Windows 98 machines, but not across a mixed network. The reason is that Windows 95/98 mangles the name to fit in 8.3. The answer is to use a short name.

It seems likely that this bug will never be fixed, so why not document it in the API reference for CreateMailSlot, instead of tucking it away in a KnowledgeBase article where developers will only find it after looking everywhere for bugs in their own code?

The second problem isn't really a bug. If you have several network protocols running say NETBEUI, TCP/IP and IPX, then the mailslot message will be broadcast on all the protocols and you're likely to get each message three times. Microsoft claims this is by design, and suggests that you have the server look for duplicates and delete them. The other option is to unclutter your network, by just using TCP/IP.

PCW CONTACTS

Tim Anderson welcomes your Visual Programming comments and queries. Contact him at visual@pcw.co.uk or via the PCW editorial office (address p10).

Information on XML: *see www.w3.org and search for XML; and see www.microsoft.com/xml for an online tutorial and samples; and www.xml.com for an independent evaluation of the state of play.*

Active Threed Plus *is £146.88 (£125 ex VAT) from Contemporary Software 01344 873434, www.contemporary.co.uk.*