

# MIDAS 3 Connection Brokering

by Bob Swart

*In early April (in London) and May (in Edinburgh) I gave a MasterClass about Delphi 5 and MIDAS 3. One of the more advanced topics of this MasterClass focused on Connection Brokering, the ability to allow a MIDAS Client to connect to a MIDAS Server from a list of potentially available MIDAS Servers.*

We need a simple MIDAS Server and MIDAS Client to demonstrate this feature, so let's get started with the easy stuff first, and then continue with the last steps to enter connection brokering.

## Simple MIDAS 3 Server

Start a new application in Delphi, call it MidasServer and make sure you can identify the main form (for example by adding a label with a large font that says "MIDAS Server"). Now, we need to do File | New and add a remote data module from the Multitier tab of the Object Repository. The CoClass Name of the Remote Data Module should be DrBob42 (or something else that you like), but leave the other options at their default settings.

Now, drop a TTable on the remote data module, and make sure it points to one of your favourite tables (such as BIOLIFE from DBDEMOS, but anything is fine here). Finally, drop a TDataSetProvider (from the Midas tab) on the remote data module, set its DataSet property to the Table you just configured. Now, save your work and run the MIDAS Server once to make sure it can be executed.

We will be using the TSocketConnection component in this example, so make sure the sccktsrvr.exe is running (which can be found in Delphi's BIN directory, and can also be executed as an NT service).

## Simple MIDAS 3 Client

A MIDAS 3 Client is even easier: start a new application and drop a TSocketConnection component (from the Midas tab) on your main form. Inside this component, you must first specify the address of the (remote or local) machine you want to connect to. I have had some problems with localhost, so usually specify the real IP-address like 192.168.92.201 in my case. Now, assuming the Borland Socket Server is running, you can open the ServerName combobox to select the MIDAS Server we made in the previous section. At this time, you can set the Connected property to True and see the MIDAS Server appear (if not then there is a problem somewhere :).

Now we need to drop some additional components to use the connection. In the simplest case, you need to drop a ClientDataSet component (also from the Midas tab) and a DataSource and DBGrid to show the data.

The RemoteServer property of the ClientDataSet must be set to your SocketConnection component. Which leaves only the ProviderName property to be given a value (in our case, there will be only one choice: the DataSetProvider1 component. At this time, you can Open the ClientDataSet (set Active to True) and connect the DataSource and DBGrid in order to see the contents (which are in fact provider by the MIDAS 3 Server).

## Connection Brokering

Using our MIDAS Server and Client, we can demonstrate the feature of Connection Brokering. Using this feature, a client can connect to a choice of servers. So you need at least two machines (connected by a network) to make it work. When using the SocketConnection component, as we do, both machines must also run the Borland Socket Server application (which will automatically start the MIDAS 3 on their machines when needed).

First of all, we must make sure that the current connection isn't already active. So set the Active property of the ClientDataSet to False, and the Connected property of the SocketConnection component as well. The MIDAS 3 Server should no longer be running now. To make sure that you still see your data when you run the client, you should write one line of code (inside the Form's OnCreate event handler):

```
procedure TForm2.FormCreate(Sender: TObject);
begin
    ClientDataSet1.Open
end;
```

This will activate the ClientDataSet which will force the SocketConnection component to start a connection and fire up the MIDAS Server.

Next, we need to use a special component found on the Midas tab: the TSimpleObjectBroker component. We need to add this component to our MIDAS Client application (since it's the Client who will have a choice of connecting to one of a list of MIDAS Servers). The SimpleObjectBroker component has two important properties: LoadBalanced and Servers. To start with the latter: the Servers property contains a list of MIDAS Servers. In this case, I'm talking about MIDAS Server machines, since the GUID of the MIDAS Server application itself is already known: the SimpleObjectBroker can only broker between multiple machines, but all must be running (or capable of running) the same MIDAS Server application. If you click on the ellipsis next to the Servers property, you get the Servers Collection Editor. In here, you can add your servers (again, only the properties of the machines, not of the actual MIDAS applications). In our specific example, we'll assume that apart from my local 192.168.92.201 machine, there is also a machine with IP-address 192.168.92.245 which can also run the same MIDAS 3 Server application. In that case, I can add two ServerItems in the dialog. For the first ServerItem, I need to set the ComputerName to 192.168.92.201 and for the second one I need to set the ComputerName to 192.168.92.245. I do not need to change the Enabled property (which is set to True, so we know the servers are available by default). I only need to change the Port property if I have also specified inside the Borland Socket Server that we're not using Port 211 but some other

port for the communication (remember that our MIDAS 3 Client will first connect and talk to the Borland Socket Server, which will then reroute the (data) requests to the MIDAS 3 Server on that machine.

After you've made a list of MIDAS 3 Servers (which of course is only useful if you have more than one entry), you must not forget to set the most important property of the SimpleObjectBroker: the LoadBalanced property (I'm afraid I forgot to set it to True in London, until someone reminded me about it). If you allow this property to be set to False (the default value), then the SimpleObjectBroker will simply return the first server from the list. No load balancing, but every client will be connected to the first server instead. If, on the other hand, you explicitly request LoadBalanced to be True, then the SimpleObjectBroker will use an algorithm to select the MIDAS server machine for this particular client (note that for each client, the SimpleObjectBroker is invoked, but there is no communication possible between clients, so the SimpleObjectBroker must use an algorithm that doesn't yield the same result in all cases. In fact, Borland's implementation of the SimpleObjectBroker makes sure of that, since the algorithm is based on a powerful... random generator).

After we've configured the SimpleObjectBroker, we must make sure that the SocketConnection actually uses the SimpleObjectBroker to find the MIDAS Server, by pointing its ObjectBroker property to the SimpleObjectBroker component. That way, whenever a connection must be made, the SimpleObjectBroker is invoked and will return the information for a particular MIDAS server machine (on which the MIDAS server application must be installed and ready to run, of course).

## Testing...

To test this, we have to install the MIDAS 3 server applications on every MIDAS server machine we want to use in the connection brokering example. To install a MIDAS 3 Server on a machine, you need the application itself, the possible database software you need (in our case the BDE as well as the local BDE tables), the MIDAS.DLL (either inside the Windows/System32 directory or in the same directory as your MIDAS Server application), and - since we're using the SocketConnection component - the scktsrvr.exe Borland Socket Server. Make sure the MIDAS 3 Server application has at least run once on each server machine so it's registered properly (and you know it works).

Then, simply start a few MIDAS 3 Clients and see which machines they use to connect. On my particular machines, about half of the time the client will automatically connect to 192.168.92.201 and the other half of the time to the other machine at 192.168.92.45. Indeed random. But the important thing is that when one server machine isn't available for whatever reason, then the SimpleObjectBroker will make sure that the MIDAS Client can always connect to another MIDAS Server.

## Customisation

If you want to write your own ObjectBroker component with an even more sophisticated algorithm (for example looking at the actual load of the individual machines), you should take a look at the TCustomObjectBroker component which is the base class for the MIDAS Object Brokers in Delphi.

For more information on MIDAS - or DataSnap as the new name appears to be in Delphi 6 - please stay tuned to my website where I'll be publishing more papers and source code examples in the near future.