Using kbmMW with ReportBuilder

for kbmMW Commercial v. 1.01+ and v. 2.00+
rev 3

ReportBuilder from Digital Metaphors is a widely used and popular report creation software made for use in Delphi and C++Builder.

Reports can be made in two ways – predefined reports, and end user defined reports.

Predefined reports

They are very easy to get going with a kbmMW client, since all whats needed are some TDataset compatible components from where ReportBuilder can fetch its data from.

For this purpose, one would on a kbmMW based client use the TkbmMWClientQuery or TkbmMWClientStoredProc to deliver data from the application server in the TDataset format needed by RB.

It would be no different from using a TQuery.

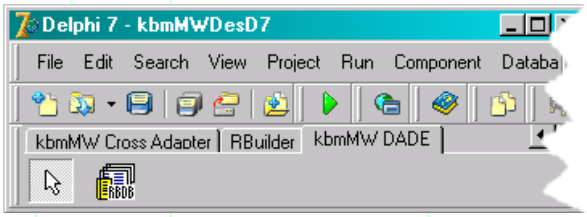
End user defined reports

For end users to be able to define their own reports, they need to get access to more meta information from the application server, like database names, table names etc. For this to work, Digital Metaphors developed an API which they call DADE. The DADE API allows 3rdparty developers to add support for end user defined reports for many different types of databases.

kbmMW Commercial v. 1.01 now also support the DADE API.

Make sure either to select the correct ReportBuilder version when installing kbmMW, or to open the kbmMWConfig.inc file and modify the settings in the ReportBuilder section to match the version of ReportBuilder you have. The DADE API have changed slightly between the versions of RB.

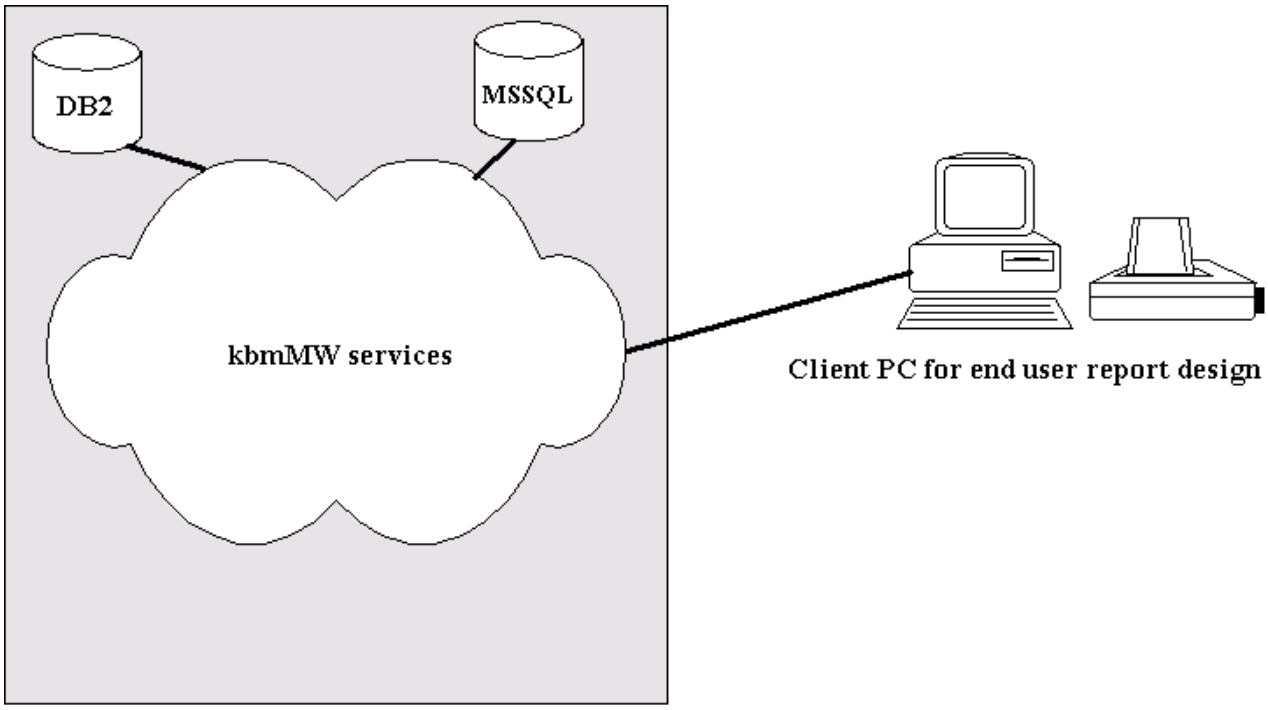
After successfully installing kbmMW support for RB DADE, you will find a new tab in the component palette named kbmMW DADE. In it there will be one component, namely the TdakbmMWDatabase component (with the RBDB symbol on it).



The purpose of it is to define ‘databases’ which the end user can use for designing the report.

Since a kbmMW application server is not a database in itself, and since it very easily could be utilizing several external databases for its work, there have to be some way to associate the RB end user GUI with a virtual database. This happens in corporation between the TdakbmMWDatabase component and the published queries on the query service which is pointed to.

Lets look at an overview of a typical kbmMW application server.



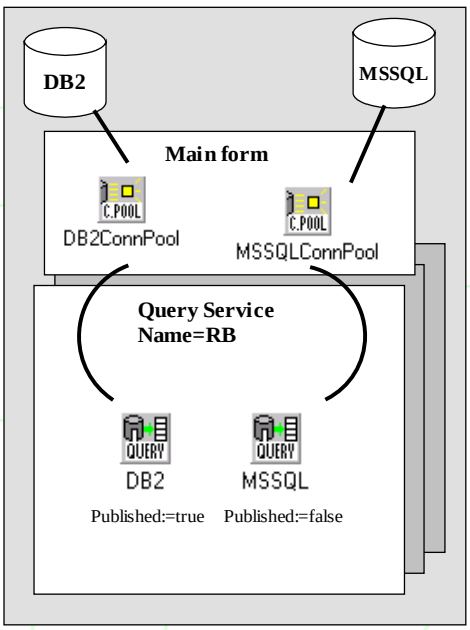
Typical kbmMW application server setup

As you can see, there are in this case actually two databases behind the application server. Should the client be allowed to do reporting on both? This are usually design questions a developer must ask one self. And often it ends up in that the client should be limited.

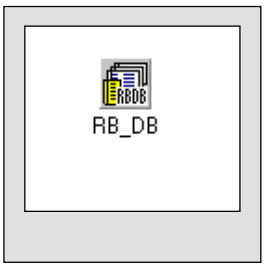
Thus simply showing the DB2 and the MSSQL database as the two databases for the end user client would not work.

Further one of the beauties of kbmMW is that the client never need to know what database is behind the application server. Actually there may not even be a real database… kbmMW could obtain its information from an email system or some internal calculations or some other place, and present it as a result from a client query.

This is illustrated in the next figure.



Typical kbmMW application server setup

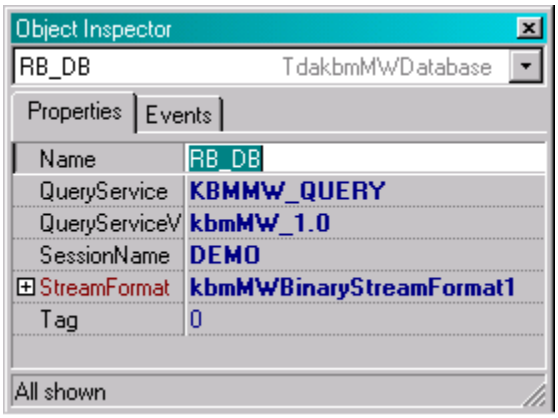


Client

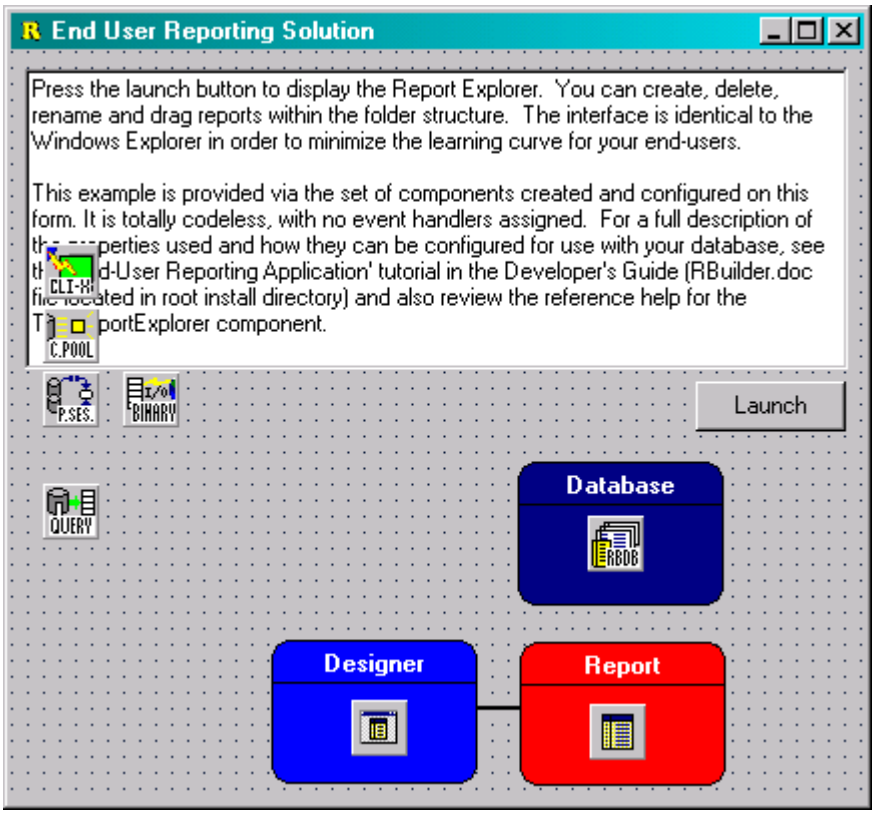
Thus in this case we have one query service (RB) which contains two query components, one which is published, pointing to the DB2 database (via a pooled session and a connection pool component), and another unpublished one pointing to the MSSQL database. The query components which will be used by ReportBuilder must have the AllowClientQuery property set to true.

At the same time we can see on the client side that a TadkbmMWDatabase component has been added.

The TadkbmMWDatabase component has several properties which must be filled out.



QueryService/QueryServiceVersion should contain the name and version of the query service on the application server which will be contacted to get information for the end user report.
SessionName contains a name of a pooled session component on the client side. Its used by the kbmMW DADE to find out which client connection pool will be used to get a connection to the application server.
Finally the StreamFormat component must point to a dataset streamformat matching the one used on the server side.
Using the End User Report demo's from Digital Metaphors, we can now see how things start to work together.



When the Launch button is pressed, the designer opens up and the user decides to setup database settings, the kbmMW DADE will automatically search for all TdakbmMWDatabase components in the project. For each of them, the kbmMW DADE will connect to the application server according to the SessionName, QueryService and QueryServiceVersion settings, and ask for the names of published queries. The combined name of the TdakbmMWDatabase component and the name of the published query constitutes a virtual database name.
Thus in our sample, one virtual database would list: RB_DB_DB2 (combination of RB_DB and DB2).
Via this one, the designer can now obtain table meta information from the backend database (if it supports that) and the user can start to design a new report.

As you can see, its possible to have more than one TdakbmMWDatabase on the client side.
That allows for the designer to obtain data from several query services on the same application server (by choosing another QueryService/QueryServiceVersion setting), or from different application servers (by choosing another SessionName).

This concludes the kbmMW DADE whitepaper.

Kim Madsen
Components4Developers