Mo.net Quotations Service
Introducing Modelling-as-a-Query

September 2022
Revision 4

Purpose

The purpose of this paper is to outline the potential benefits of integrating Mo.net Quotations Service directly with a RDBMS platform, such as Microsoft SQL Server, using the innovative modelling-as-a-query ("MaaQ") concept.

Background

One of the many advantages of the Mo.net Financial Modelling Platform over traditional / legacy modelling systems is its service-oriented architecture ("SOA"). This mature architectural paradigm allows enterprise systems to be developed using a set of reusable, loosely coupled components, rather than the tightly-coupled bespoke and potentially brittle elements typically found within the financial modelling arena.

Database Integration

While the integration of financial modelling platforms with databases is nothing new, these integration points are usually limited to reading source data, accessing assumptions & parameters, or writing results. Such functionality has been available within the Mo.net Model Development Studio product for many years and many customers already make extensive use of this capability.

"Modelling-as-a-Query"

What sets Mo.net apart from the competition, by leveraging the underlying service-oriented architecture of the platform, is the ability to consume models or calculations developed in Mo.net directly from within the database environment. This means that simply by executing an appropriately designed query or stored procedure, data from one or more SQL Server tables or views can be passed to a model / calculation encapsulated as a service and the result inserted back into the database, all without leaving the database environment.

While this functionality might initially be most interesting for members of the IT community, it does introduce the potential to development of some differentiating business solutions with very tangible benefits. Some of these are outlined later in this paper.

Platform Components

The following table summarises each component of the candidate solution outlined within this document:

Component	Purpose / Role
Mo.net Model Development Studio	Integrated development environment for financial model / calculation design, development, testing and analysis.
Mo.net Quotations Service Package	Bundle of model artefacts, including model binaries and static tables / parameters, and dependent Mo.net libraries.
Mo.net Enterprise Service Manager	Conduit between the model development and operational environment. Responsible for staging and publishing of model / calculation service packages, and management of all Mo.net Enterprise Service components. Also provides service activity monitor functionality and the ability to download specific calculation requests back into the development environment for analysis / debugging.
Mo.net Quotations Service	Component of Mo.net Enterprise Services responsible for hosting one or many Mo.net Quotations Service Packages for consumption by a range of client applications / services. Provides real time / synchronous processing of requests. End points for RESTful and SOAP / WCF standards.
Microsoft SQL Server	Enterprise-scale RDBMS platform
End User Applications	Any application used by the user community that uses Microsoft SQL Server as its primary data source / target.

Table 1 - Platform Components

The interaction between each of these components is summarised in the figure below.

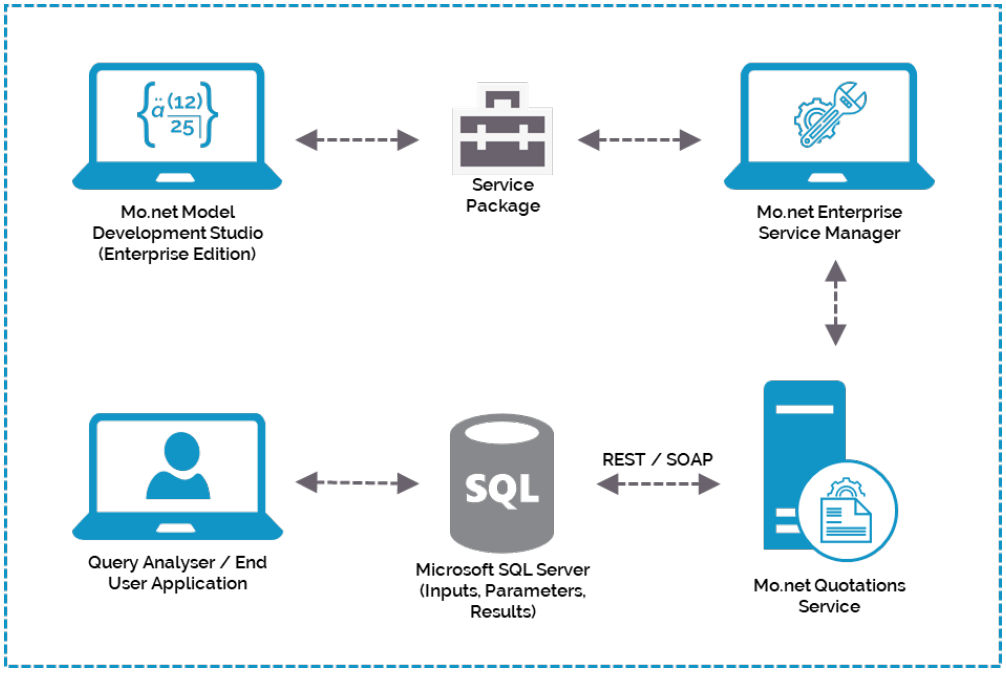

Figure 1 - Component Architecture

Basic Use Case

Consider a SQL Server table with input (source) data for a model, in this case the LinkedFunds sample project that comes with Mo.net Model Development Studio:

PolNo	Sex1	AgeEnt1	FundType	SumAssured	PeriodIF	Term	PremFreq	PremInit_Equity	PremInit_FI	PremInit_Cash	PremInit_Prop
1	0	35	Equity	100000	0	20	4	480	360	240	120
2	0	56	FI	100000	0	20	4	120	360	240	120
3	0	48	Cash	100000	0	20	4	0	360	240	120

Figure 2 - Input (Source) Data

Let's assume that the user wants to calculate the UnitFund value at a specific time point for each record using the LinkedFunds model and insert the result into a new column called UnitFund in the above table. Ordinarily this might involve creating a task within Mo.net Model Development Studio to read each record from the above table, calculate the UnitFund value, and then insert the result into a results table which is then merged with the source table. Alternatively, a more experienced developer might write some group projection code to create the results table from scratch.

While these approaches are perfectly acceptable for users familiar with Mo.net, the alternative approach allows the calculated result to be inserted into the source table directly from within the database environment.

PolNo	Sex1	AgeEnt1	FundType	SumAssured	PeriodIF	Term	PremFreq	PremInit_Equity	PremInit_FI	PremInit_Cash	PremInit_Prop	UnitFund
1	0	35	Equity	100000	0	20	4	480	360	240	120	611899.7
2	0	56	FI	100000	0	20	4	120	360	240	120	611774.5
3	0	48	Cash	100000	0	20	4	0	360	240	120	611861.7

Figure 3 - Input Data with Calculated Unit Fund

How it Works

The key element of the solution is a custom connector for SQL Server which enables communication with the Mo.net Quotations Service, which hosts one or more models, as described in the conceptual architecture above. The connector is installed directly in the database that holds the data to be passed to the model.

Once the connector is configured, it is simply a case of making an appropriately formed request to the Mo.net Quotations Service using standard T-SQL. Such requests can be made interactively through SQL Server Query Analyser, or embedded into stored procedures, functions cursors, or even database triggers. This provides the potential for autonomous data-driven and unattended modelling.

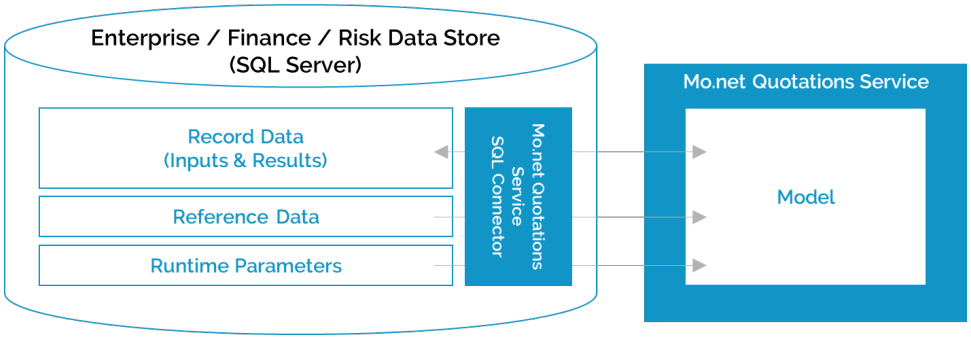

Figure 4 - How it Works

Potential Use Cases and Benefits

While modelling-as-a-query is an entirely new concept, there are numerous applications of the functionality with very tangible business benefits. A couple of potential uses are outlined below.

New Business Value vs Original Premium

In this use case the "actuarial" value (reserve) of each line of new business is calculated as soon as the policy data appears in the appropriate data store. There is no need to export data from the policy administration or new business data store; the calculations are performed from within the database.

This value (and potentially expected cashflows) can then be stored for comparison against the original premium to derive downstream metrics associated with profitability and pricing assumptions.

Continuous Valuation / Total Customer Value

In this use case, each policy is valued on a daily or less frequent basis based on current market conditions & business assumptions.

The results from each policy are then stored and combined with any other policies held by the same party allowing the calculation & trending of total customer value.

This allows the specific value of each policy / customer to be tracked on a continuous basis, and the results presented on-demand in any standard business intelligence tool.

Contact Us

For more information regarding Modelling-as-a-Query, the Mo.net Quotations Service, or any other aspect of the Mo.net Financial Modelling Platform, please contact us:

Software Alliance Limited
30 Stamford Street, London, SE1 9LQ
Tel: +44 (0) 20 3964 2755
www.softwarealliance.net