# Case Study: Mortgage Recommender

## Utilizing the New Decision Model & Notation (DMN) Standard for Decision Representation, BPMN 2.0 for Process Representation, and OpenRules® BDMS for Implementation

**GIL RONEN | REVVISION CONSULTING**

**COPYRIGHT JUNE 2014**

## Synopsis

This paper takes an end-to-end approach to the craft of designing and implementing a business decision using state-of-the-art business logic standards. Using the complex-enough problem of Mortgage Loan Recommendation as the story board, the business case is represented as a process using the BPMN 2.0 standard, utilizes the new OMG Decision Model & Notation (DMN) standard for representing decision logic, and implements the decision with OpenRules® BDMS, a popular business rules product that utilizes MS Excel for defining and executing business rules and decisions. This paper may also be considered as an introduction to DMN.

# Contents

# Figures

## Standards: BPMN 2.0 & DMN Beta1

Decision Model and Notation (DMN) is a new standard of the Object Management Group (OMG) that provides constructs for decision modeling. DMN can be viewed as complementary to other forms of business logic documentation such as the Business Process Model and Notation (BPMN) standard for process modeling. Particular care has been taken in crafting DMN so that it is approachable and its work product modifiable not only by IT, and primarily by business users.

DMN specifies how to collect all the necessary artifacts to describe a decision in an encapsulated manner. This includes but is not limited to the interrelationships between decisions, decision requirements, decision logic representation (e.g. Decision Table) and semantics, decision knowledge sources, etc. One of the main goals of DMN is to "Facilitate and encourage communication of decision models between people [and] applications."

To review the distinctions and interrelationships between the process and decision standards, one can look at the OMG descriptions of the two. OMG describes BPMN as follows:

"A standard Business Process Model and Notation (BPMN) will provide businesses with the capability of understanding their internal business procedures in a graphical notation and will give organizations the ability to communicate these procedures in a standard manner. Furthermore, the graphical notation will facilitate the understanding of the performance collaborations and business transactions between the organizations. This will ensure that businesses will understand themselves and participants in their business and will enable organizations to adjust to new internal and B2B business circumstances quickly."

OMG describes DMN as:

"The purpose of DMN is to provide the constructs that are needed to model decisions, so that organizational decision-making can be readily depicted in diagrams, accurately defined by business analysts, and (optionally) automated.

DMN will provide constructs spanning both decision requirements and decision logic modeling. For decision requirements modeling, it defines the concept of a Decision Requirements Graph (DRG) comprising a set of elements and their connection rules, and a corresponding notation: the Decision Requirements Diagram (DRD). For decision logic modeling it provides a language called FEEL for defining and assembling decision tables, calculations, if/then/else logic, simple data structures, and externally defined logic from Java and PMML into executable expressions with formally defined semantics. It also provides a notation for decision logic allowing

components of the decision logic level to be drawn graphically and associated with elements of a Decision Requirements Diagram (DRD)."

BPMN includes a BusinessRulesTask element that invokes a decision component. DMN adds notation for documenting the decision logic encapsulated in the BusinessRulesTask. This decision logic may take the form of Decision Tables, Decision Trees, etc. It is important to note that BPMN further defines a GlobalBusinessRuleTask that allows for the reuse of the encapsulated decision logic and indeed it is a strategic goal to allow for an enterprise-level view of decision logic. In particular, care should be given to how decisions relate one to another across the enterprise in their structure, taxonomies, and governance. This broader view of decisions may suggest methods for cross validation as well as identify potential for automation.

## Implementation Tool

It is all well and good to plan and diagram but one doesn't really know if an approach will work until the rubber hits the road. Therefore, to take this case study from end-to-end this paper will also go ahead and implement the business case. There is nothing more comforting than *a working executable system*. On the other hand, getting bogged down in details such as database servers will not serve the purpose of exploring decision representation so the tool selection fell on an open source business rules tool, OpenRules BDMS that is geared towards business people and is designed to work very closely with spreadsheets. As stated on their site: "OpenRules® is a general purpose Business Rules and Decisions Management System available as an Open Source product. It allows subject matter experts and software developers to create, test, execute, and maintain enterprise-class decision support applications."

## Business Case: Objective-based Mortgage Loan Recommendation

Mortgage lenders typically offer multiple products (such as fixed rate, adjustable rate, military veteran programs, construction programs). They engage clients through various channels: mortgage brokers (Broker Channel), online (Consumer Direct), through the client's retail bank (client walks into their bank and asks about mortgage options), as part of larger processes (such as within a Wealth Management component), etc. There are many lenders and many products and the various touch-points with the end-client may not be familiar with the intricacies of each lender's products. Therefore, over the past 15 years, there has been a push to

automate mortgage loan recommendations. Implementations run the gamut from simple lookups to more elaborate analysis based on client objectives such as will be described herein.

For example, a client may request a 30 year fixed mortgage loan with an objective of getting the lowest possible monthly payment. A set of recommendations using this scenario is dependent on many factors, such as the products available with a particular lender. As importantly, the recommendations depend on economic factors such as the yield curve that determines rates as a function of loan term at a particular point in the economic cycle. Specifically, the advantageousness of a loan product is also a function of its relative price to other loan products at a particular time. Back to our example, a recommendation may include a shorter term adjustable rate mortgage (e.g. 5/1 ARM) carrying a lower rate or a combination of 1st and 2nd mortgages (two loans taken out concurrently) that avoids the need to pay mortgage insurance.

In its most general form then, the process includes eliciting the client objectives, identifying individualized loan options, performing calculations based on rate (such as total monthly payment) and, finally, ranking of the options based on the client objective and displaying them back to the client.

## Example Request: Client Objective and Preferences

For the implementation examples going forward the client objective and preferences shown in Figure 1 will be used[1].

| Request Objective | Requested Program | Requested Product Category | Requested Product Term | Requested LTV | Requested Loan Amount |
|---|---|---|---|---|---|
| Lowest Monthly Payment | Non-Prime | Fixed | 30 | 87.50 | $350,000 |

Figure 1: Client Objective and Preferences

The business interpretation of the above table is as follows:

*The client is looking for the mortgage loan option with the lowest monthly payment possible, has stated that their credit is Non-prime, and believe that their best option is a 30-year fixed mortgage with a loan amount of $350,000. The LTV (loan-to-value ratio) of 87.5% is calculated based on the requested loan amount and the client's stated valuation of the house. The actual house dollar amount value is not used beyond the LTV calculation.*

[1] Figure 1 shows an OpenRules table of the predefined type "Data" created in MS Excel.

## Example Response: Ranked Mortgage Loan Options

The mortgage loan recommendation decision is different in its outcome than binary decisions (such as Eligible, Ineligible) but the representation of the decision logic still follows along a similar path, the output simply being a set of options instead of a single value. This example therefore expands the scope of what is appropriate for DMN representation.

Figure 2 shows potential recommendations derived from the scenario depicted in Figure 1. The objective of *lowest monthly payment* implies that the loan options need to be ranked and sorted by their total monthly payment; the best options are returned first. In this case, five relevant alternatives are identified and constructed. For each, the various components of the monthly payment are determined or calculated and the set is ranked and sorted by the total monthly payment.

| Program | Category | Term | Liens | LTV 1st | LTV 2nd | Amount | Mortgage Insurance | Tax & Insurance | Principle & Interest | Total Monthly Payment | Benefits |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Prime | Fixed | 30 | First | 87.50% | --- | $350,000 | $175.00 | $437.50 | $2,017.01 | $2,629.51 | Corresponding Prime product carries a lower rate; Requires qualifying by more stringent criteria. |
| Non-prime | Fixed | 30 | First | 85.00% | --- | $340,000 | $170.00 | $425.00 | $2,123.38 | $2,718.38 | Price advantage at 85% over higher LTV; Specific to a particular lender. |
| Non-prime | ARM | 3/1 | First | 87.50% | --- | $350,000 | $175.00 | $437.50 | $2,126.64 | $2,739.14 | ARM products typically offer lower rates initially; 3/1 is fixed for first 3 years. |
| Non-prime | Fixed | 30 | First+ Second | 80.00% | 10.00% | $360,000 | $0.00 | $450.00 | $2,364.94 | $2,814.94 | Avoids Mortgage Insurance; Scenario includes 2 loans closing simultaneously. |
| Non-prime | Fixed | 40 | First | 87.50% | --- | $350,000 | $175.00 | $437.50 | $2,238.85 | $2,851.35 | A longer term reduces the monthly payment. |
| Non-prime | Fixed | 30 | First | 87.50% | --- | $350,000 | $175.00 | $437.50 | $2,243.40 | $2,855.90 | Requested scenario. |

**Figure 2: Ranked Mortgage Loan Options**

The alternatives differ from the client's preferred loan (stated in Figure 1) in the following attributes:

1. Credit Rating – Prime instead of Non-prime (the client stated "Fair" credit but until a credit report is actually analyzed their input can only be viewed as an approximation),

2. Loan Amount – $340,000 instead of $350,000 (taking advantage of pricing ladder information),
3. Product – ARM 3/1 instead of Fixed 30 (adjustable rate instead of fixed rate),
4. Lien composition – two loans closing simultaneously instead of one, and,
5. Term – 40 year term instead of 30.

In this example, the scenario provided by the client is the least advantageous given their objective of lowest monthly payment – it has the highest total monthly payment!

## Decision Representation

The business case described above can be viewed in one of three discrete perspectives:

(1) **Business Process**
(2) **Decision Requirements**
(3) **Decision Logic**

The Business Process representation uses an existing standard (BPMN 2.0[2]) and is not part of DMN, but when looking at the overall context of a decision, the processes that invoke and provide input to the decision are part and parcel of the context and therefore bundled herein with the DMN representation as an exercise in best practices.

## Business Process

The process of generating mortgage loan recommendations can be described by a sequence of tasks that include decisions (business rule tasks) and calculations (general automated tasks). Figure 3 uses the BPMN 2.0 standard notation to describe this process.

[2] BPMN being just one such existing standard for process representation.

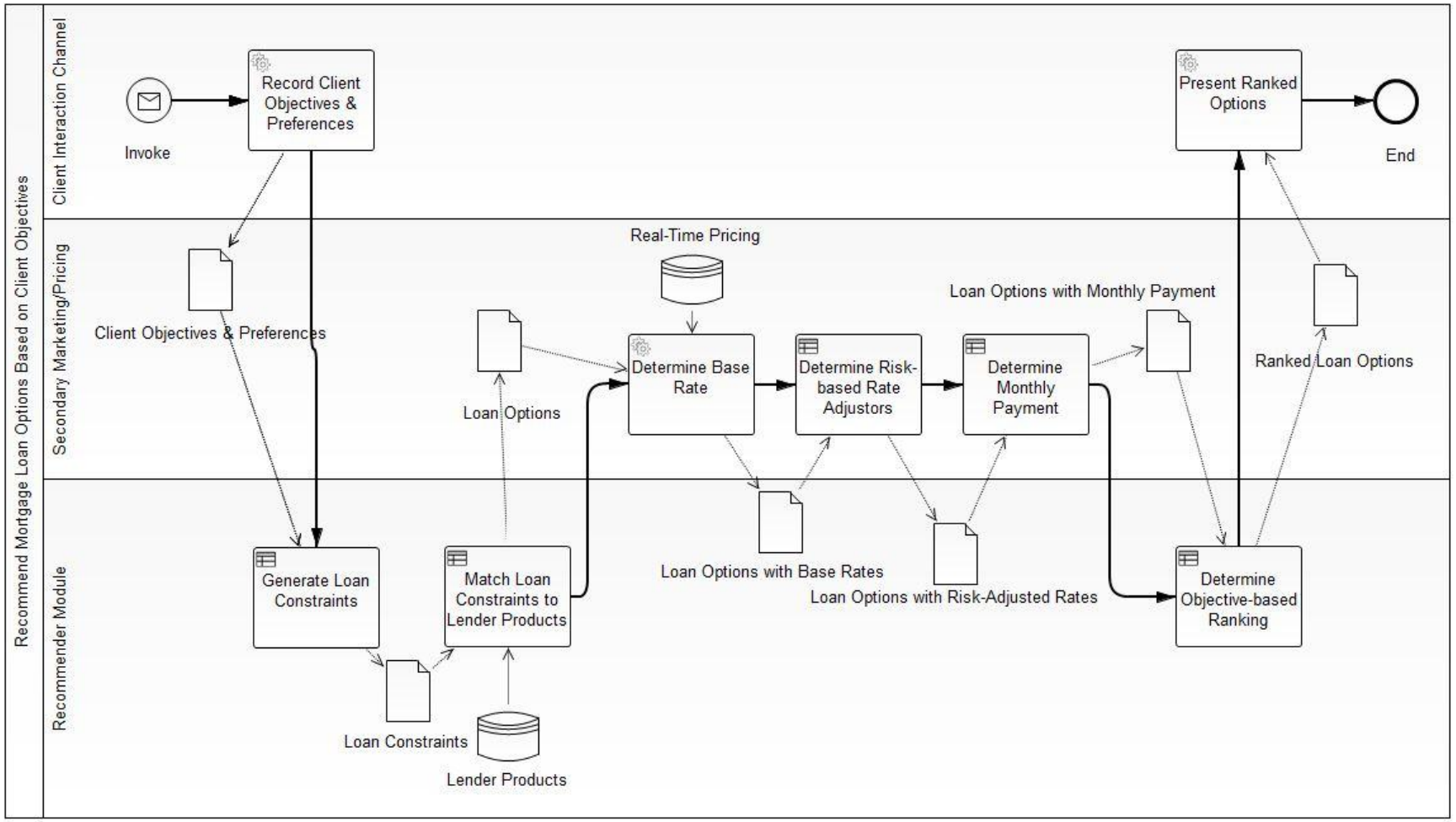


Figure 3: BPMN 2.0 Process Modell

1. **Record Client Objectives:** Regardless of the channel used to interact with the client, the process requires as input information on the client's preferences and objectives. For example, are they looking to build equity? Do they require the lowest monthly payment in order to qualify? Are they looking for a lower rate in a refinance scenario? Will they stay in the property for 3 years or 15?

2. **Generate Loan Constraints:** Identify possible scenarios based on client objectives and other client information. The more information provided the better the recommendations. For example, a credit report would help determine eligibility of certain products. However, credit information may come at different levels of coarseness and accuracy and at different points in the process so the rules need to take into account missing data, fuzzy data and client-stated data. Constraints can include any loan attribute such as Program, Product Category, Loan Amortization Term, Lien combinations, Loan Amount, etc.
3. **Match Loan Constraints to Lender Products:** Based on the identified general scenarios (lists of constraints on the loan attributes), specific products offered by different lenders are identified. Each set of constraints is matched to one or more products.
4. **Determine Base Rate:** Rate sheets are dynamic and typically generated at least once a day by the Secondary Markets department. The rate sheet first identifies a base rate. Base rates are often modeled as database lookup.
5. **Determine Risk-based Rate Adjustors:** Adjustors to the rate that are based on various attributes of the loan (such as credit rating or loan-to-value ratio) are added to the base rate. Adjustors are often modeled as rules.
6. **Determine Monthly Payment:** For each option that is a combination of a product and adjusted rate a monthly payment is calculated. The complexity includes handling of a 1st and 2nd lien concurrently, payment of upfront points, etc. The rate is coupled with other factors such as taxes and mortgage insurance to arrive at a total monthly payment.
7. **Determine Objective-based Ranking:** Different objectives require a different sort order for the set of loan options developed during the process.
8. **Present Ranked Options:** Display the final set of ranked options to the client invoking the process.

## Decision Requirements

Now that the process context of the decision has been represented it is time to look at the decision through two additional perspectives:

(1) The decision as an encapsulated element with its own domain, influencers and description, and,

(2) As part of a larger context of how decisions are handled generally in the organization – a decision lifecycle independent of process and/or code lifecycles (or at least carrying its own nuances).

## DMN Decision Requirements Graph

The DMN Decision Requirements Graph (DRG) models a domain of decision-making in its entirety showing the dependencies between decisions that it encompasses. The elements modeled are the key decisions, their data inputs, encapsulated business logic (termed Business Knowledge Models) and the authorities for said logic (termed Knowledge Sources). The interaction between any two elements is called a requirement (supported are Information Requirement, Knowledge Requirement, and Authority Requirement).

Figure 4 below shows the full domain for the objective-based mortgage loan recommendation business case.

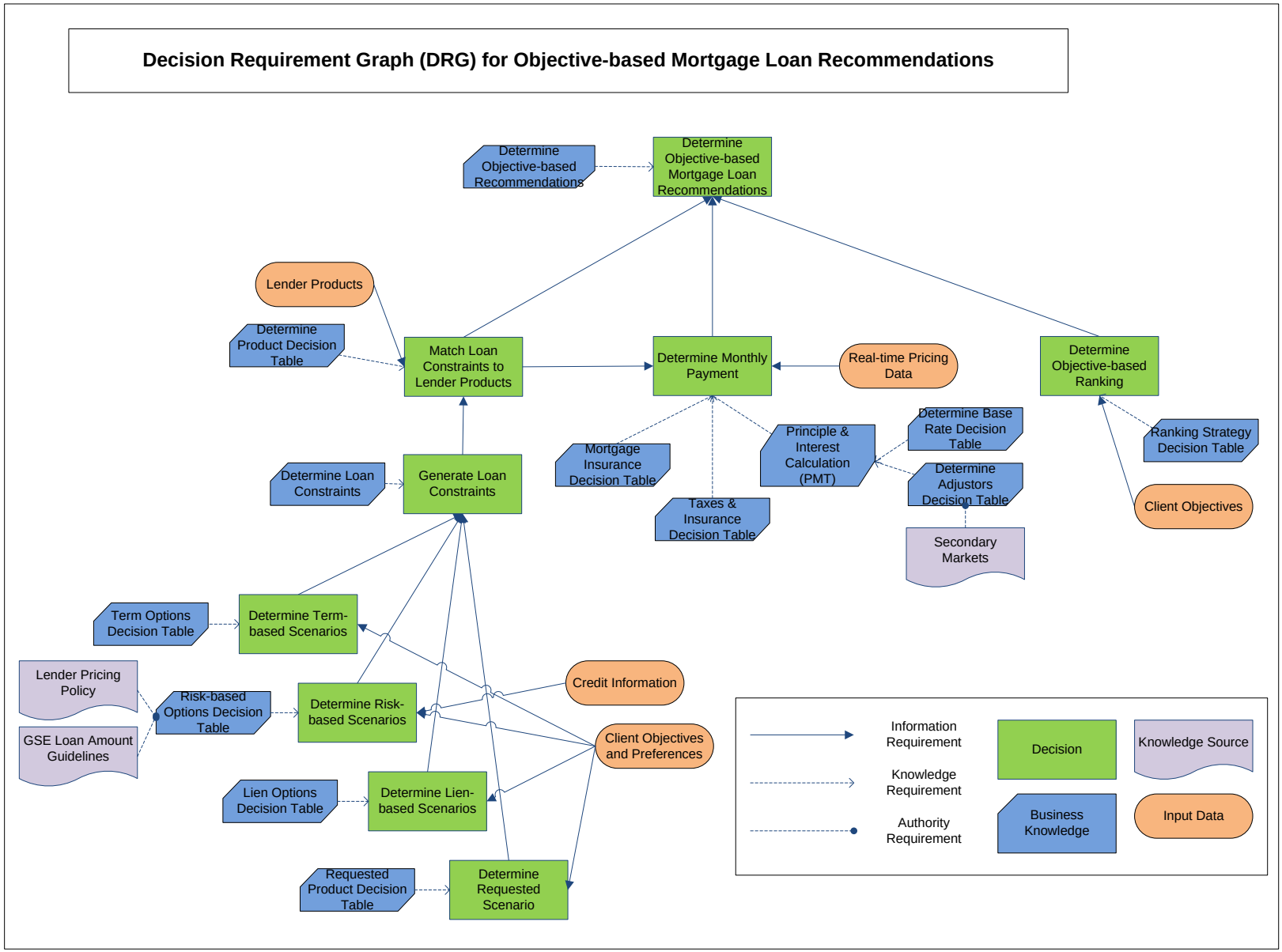


Figure 4: Decision Requirement Graph (DRG) for the Domain of Determine Objective-Based Mortgage Loan Recommendations

In this case, there is a root decision (Determine Objective-based Mortgage Loan Recommendations) and it shows three decisions that it is dependent on: Match Loan Constraints to Lender Products, Determine Monthly Payment, and Determine Objective-based Ranking. Match Loan Constraints to Lender Products is dependent on the decision Generate Loan Constraints, which in turn, is dependent on four other decisions. Each of these four decisions has a Knowledge Requirement (in this case described in the language of decision tables) and at least one Information Requirement (input data).

Since a DRG covers an entire domain it can quickly become unwieldy, however, in this scenario several simplifications have been used and therefore the entire DRG is easily accessible. For example, a robust recommendation system will also utilize credit report data and analysis as well as an eligibility determination for each loan option. Handling of intricacies of different input channels (such as different pricing) are also part of the bigger picture that may add complexity. Adding these elements would blow-out the DRG.

"For any significant domain of decision-making a [Decision Requirements Diagram] DRD representing the complete DRG may be a large and complex diagram. Implementations MAY provide facilities for displaying DRDs which are partial or filtered views of the DRG, e.g., by hiding categories of elements, or hiding or collapsing areas of the network. DMN does not specify how such views should be notated, but whenever information is hidden implementations SHOULD provide a clear visual indication that this is the case."

If a graphical representation for Decision Requirements becomes unmanageable, one may switch to an equivalent tabular representation, e.g. using the tables of the type "Decision" provided by OpenRules.

## DMN Decision Requirements Diagram

Figure 5 below shows the filtered notation employed in a high-level Decision Requirements Diagram (DRD) highlighting only the main decisions. Only the decision Determine Objective-based Mortgage Loan Recommendations is specified fully as all its requirements are presented. It uses three other decisions as input and its business logic is described in the Business Knowledge Model Determine Objective-based Recommendations.

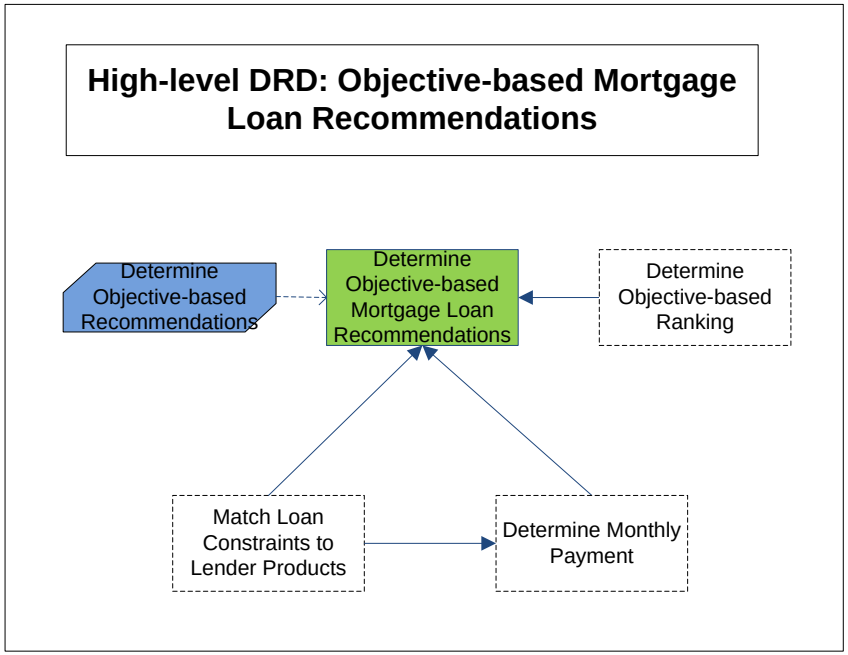


Figure 5: High-level DRD for the Determine Objective-based Mortgage Loan Recommendations Decision

From this high-level DRD, the "Generate Loan Constraints" sub-decision is selected for further detail. Figure 6 below is a DRD that shows a view of the DRG that filters all the decisions except for fully specifying the Generate Loan Constraints decision.

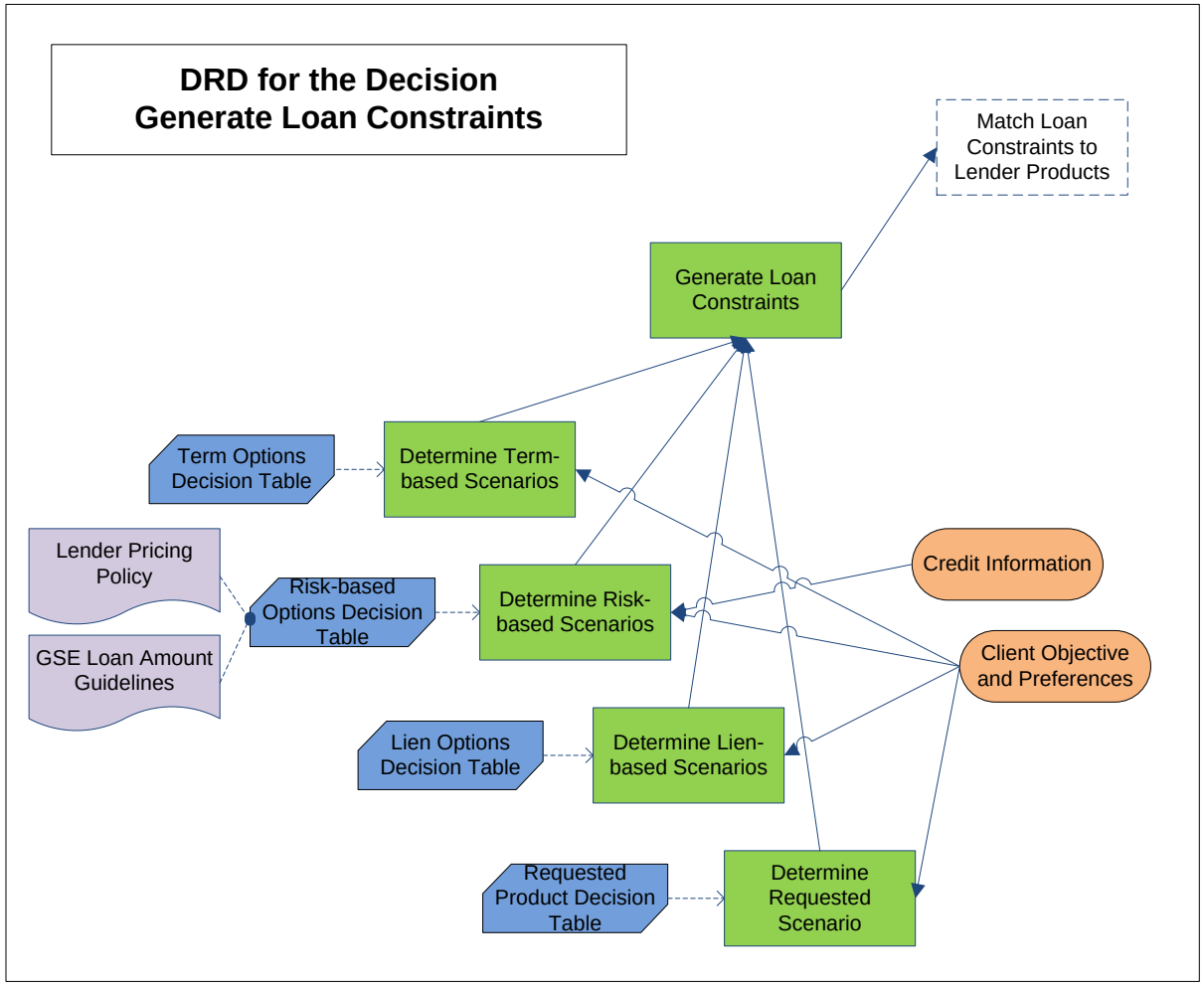


Figure 6: DRD for the Generate Loan Constraints Decision

The Generate Loan Constraints decision is dependent on four decisions:

- **Determine Term-based Scenarios:** This decision generates options that are based on the amortization term of the loan. For example, if the requested term is 30 years then for an objective of Lowest Monthly Payment, offer a 15-year fixed loan.

- **Determine Risk-based Scenarios:** This decision generates options that have to do with pricing strategies; if the client's requested scenario is close to a break point in pricing then an option that is below that point is offered back. For example, loan amounts greater than the Fannie Mae limits incur a rate hike. Keeping the loan below the limit may be attractive to the client due to the better pricing if the difference from the requested loan amount can be bridged by other means.
- **Determine Lien-based Scenarios:** This decision looks at options that include a 2nd loan (termed a 2nd lien) that will close simultaneously with the 1st-lien. This strategy is often used to avoid Mortgage Insurance.
- **Determine Requested Scenario:** The requested scenario is always tacked on to the set of generated options in case it happens to be the best option for the client's objective. After all the options are generated they are ranked based on the objective. The options are presented to the client with their respective ranking.

## Decision Logic

Now that the processes have been mapped out and the decisions represented in their appropriate context, it is time to provide the actual decision logic in its most appropriate representation. This may be a decision tree, decision table, function, algorithm, etc.

"A business knowledge model may contain any decision logic which is capable of being represented as a function. This will allow the import of many existing decision logic modeling standards (e.g. for business rules and analytic models) into DMN. An important format of business knowledge, specifically supported in DMN, is the Decision Table. "

### DMN Decision Tables: Term Options Decision Table

The Term Options decision feeds into the Generate Loan Constraints decision and generates a subset of the recommended mortgage loan options. The objective of its Business Knowledge Model, represented by the decision table shown below, is to generate loan options based on the term of the loan. Multiple options (or none) may be generated using this table and as per the process flow they are later matched to specific lender products and ranked before being presented back to the client.

Decision Tables in DMN are used to represent Business Knowledge Models tied to the decisions (sub-decisions) represented in the DRG and DRDs. For example, the Determine Term-based Scenarios sub-decision from the DRD in Figure 6 uses the decision table Term Options Decision Table that may be represented as shown in Figure 7.

| Term Options Decision Table | | | | NI Collect | | |
|---|---|---|---|---|---|---|
| **#** | **Request Objective** | **Requested Product Category** | **Requested Product Term** | **Recommended Product Category** | **Recommended Product Term** | **Recommended Liens** |
| | **LOWEST MONTHLY PAYMENT, EQUITY BUILDER, LOWER RATE** | **FIXED, ARM** | ***see products list*** | **FIXED, ARM** | ***see products list*** | **FIRST, FIRST+SECOND** |
| 1 | LOWEST MONTHLY PAYMENT | FIXED | | ARM | 3/1 | FIRST |
| 2 | | FIXED | 15 | FIXED | 20 | FIRST |
| 3 | | FIXED | 20 | FIXED | 30 | FIRST |
| 4 | | FIXED | 30 | FIXED | 40 | FIRST |
| 5 | | ARM | 7/1 | ARM | 3/1 | FIRST |
| 6 | | ARM | 5/1 | ARM | 3/1 | FIRST |
| 7 | | ARM | 3/1 | ARM | 1/1 | FIRST |
| 8 | EQUITY BUILDER | FIXED | 30 | FIXED | 20 | FIRST |
| 9 | | FIXED | 20 | FIXED | 15 | FIRST |
| 10 | | FIXED | 15 | FIXED | 10 | FIRST |
| 11 | | ARM | | FIXED | 20 | FIRST |
| 12 | LOWER RATE | FIXED | 30 | FIXED | 20 | FIRST |
| 13 | | FIXED | 20 | FIXED | 15 | FIRST |
| 14 | | FIXED | | ARM | 5/1 | FIRST |
| 15 | | ARM | Not 1/1 | ARM | 1/1 | FIRST |

**Figure 7: Decision Table Representation of the Term Options Decision Table Business Knowledge Model**

An example business rule represented by row #4 in the decision table above is: If the client objective is to get a loan option with the lowest possible monthly payment and they've specified a preference for a 30-year fixed term loan then add a recommended loan option that is a 40-year fixed term loan as a 1st lien (only one loan product). The rationale for this business rule is that a loan amortized over a longer period will have a lower monthly payment given the same rate[3]. In reality, this table may contain more action-columns that specify more recommended loan attributes than those shown here. This can be seen in the actual implementation representation discussed in subsequent sections.

[3] We'll see later that the rates may not be the same and therefore the monthly payment for both options needs to be calculated before ranking the various options.

DMN standardizes some general aspects of decision table modeling concepts such as “Hit Policy” and “Aggregation”. The most popular types of Hit Policy are “Single Hit” and “Multiple Hit”. For a single hit decision table the execution stops when the first rule (row) is satisfied. For multiple hit decision tables all satisfied rules will be executed allowing rule overrides and various kinds of aggregation. For example our decision table in Figure 7 should be defined as “Multiple Hit” as we want any satisfied rule to add a new recommended loan option. As a possible representation, DMN Beta1 offers the designation “NI Collect” in the topmost right cell of the decision table, where “N” denotes a type of Multiple Hit policy and “Collect” specifies that a set of rows is returned[4].

The next standardization is the provision of enumerated values in the column headings: a list of possible values for literals such as “FIXED” or “ARM” for the Recommended Product Category column.

Decision Tables in DMN Beta1 are a type of “Boxed Expression” which is tied to a Business Knowledge Model through a “Boxed Invocation”. The decision Determine Term-based Scenarios uses the boxed expression in Figure 8 to invoke the Business Knowledge Model represented in the decision table shown in Figure 7. This is somewhat cumbersome in practice but does provide a necessary container for the bindings between the columns in a decision table and the data, calculations, and algorithms that drive them[5].

As explained in the DMN Beta1 specification: “We define a graphical notation for decision logic called boxed expressions. This notation serves to decompose the decision logic model into small pieces that can be associated with DRG artifacts. The DRD plus the boxed expressions form a complete, mostly graphical language that completely specifies Decision Models… Boxed expressions are defined recursively, *i.e.* boxed expressions can contain other boxed expressions. The top-level boxed expression corresponds to the decision logic of a single DRG artifact… An invocation is a container for the parameter bindings that provide the context for the evaluation of the body of a business knowledge model.”

[4] More completely “NI Collect” denotes not only that more than a single row may be correct (“Multiple Hit”), but also that there is no particular order for the set of rows that are applicable (“No Order”), the table doesn’t include all possible combinations of input data and therefore may not match on any rows (“Incomplete”) and whatever rows are applicable are returned as a set (“Collect”) for further processing elsewhere in the process. For a full set of designations see the DMN Beta1 specification.

[5] As the DMN specification evolves this notion of Boxed Invocation may change significantly but the information it currently houses will likely be preserved in some form or other.

| Determine Term-based Scenarios | |
|---|---|
| Term Options Decision Table | |
| Request Objective | ApplicantObjective.objective |
| Requested Product Category | Preference.productCategory |
| Requested product Term | Preference.productTerm |

Figure 8: Boxed Invocation for Term Options Decision Table

The invocation includes the name of the decision, the name of the decision table and the instructions for how to map the decision inputs into the columns of the decision table. As an example of the most basic mapping, the Requested Product Term column takes its input from the productTerm attribute of the client Preferences input.

## DMN Cumulative Decision Tables: Determine Adjustors Decision Table

Skipping towards the end of the loan recommendation business process, each loan option, before it can be ranked by objective, requires a total monthly payment determination. This is deemed a determination more than a calculation because it includes both straight calculations and sets of business rules. Part of the determination includes business logic for adjusting a base rate for a mortgage product by adding to the rate according to various risk criteria. Example criteria are modeled in the decision table represented in Figure 9.

| Determine Adjustors Decision Table | | | | NI SUM |
|---|---|---|---|---|
| **#** | **Recommended Program** | **Recommended Liens** | **Recommended 1st Lien LTV** | **Adjustor** |
| **#** | **PRIME, NON-PRIME** | **FIRST, FIRST+SECOND** | **[0…100]** | |
| 1 | | | | 0 |
| 2 | | FIRST | [85.01..100] | 0.0025 |
| 3 | NON-PRIME | | | 0.01 |

Figure 9: Determine Adjustors Decision Table

As can be seen in the DRG (Figure 4) The Determine Monthly Payment decision is complex and includes the invocation of multiple Business Knowledge Models represented both as decision tables and calculations. As part of the determination of the monthly payment, the base rate of the mortgage loan product needs to be adjusted for risk. This is accomplished by invoking the rules in the

Determine Adjustors Decision Table. For example, rule #2 in the table above states that for a 1st lien with a loan-to-value ratio that is greater than 85% and lower than 100% the base rate needs to be adjusted by ¼ of a percent. Rule #1 implies a default Adjustor of 0.

This decision table is of type "NI SUM" which translates into a multi-hit table that does not include all combinations of its inputs and whose results need to be aggregated (summed up); Adjustor is an cumulative attribute of the loan option that is added to the base rate and used in the calculation of the loan's Principle and Interest.

Note that the base rate needs to be adjusted for every loan option generated and so the Business Knowledge Model needs to be invoked once for each generated option. This is not represented in the invocation boxed expression for this decision table. The iterative nature of the Determine Monthly Payment decision itself (the Determine Adjustors Decision Table being just a small part of) will be indicated in the Business Knowledge Model for the decision Determine Objective-based Mortgage Loan Recommendations[6].

## DMN Value Expressions: Determine Monthly Payment Decision

While this case study focuses on business rules and their representation as decision tables, DMN provides facilities for other representations. For example, the composition of a total monthly mortgage payment consists of several components: a Principle & Interest amount, a Taxes & Insurance amount, and, a Mortgage Insurance amount if applicable. This composite element and its underlying calculations do not necessitate representation in decision tables. DMN handles these elements of the Business Knowledge Model using Value Expressions that are defined as follows:

"In DMN 1.0, the elements of decision logic modeled as value expressions are literal expressions, decision tables and invocations:

- A **literal expression** represents decision logic as text that describes how an output value is derived from its input values. The expression language may, but need not, be formal or executable: examples of literal expressions include a plain English description of the logic of a decision, a first order logic theory, a Java computer program and a PMML document. DMN 1.0 specifies an expression language: **FEEL** (see section 9), and a basic subset of FEEL (see section 8) that is the default language for literal expressions in DMN decision tables (section 7).

[6] Traditionally, a loop of this nature would be represented in the process flow diagram that includes this task but with the additional DMN designations indicating that the returned product from a decision table can be a set of elements this would become cumbersome to manage.

- A **decision table** is a tabular representation of decision logic, based on a discretization of the possible values of the inputs of a decision, and organized into rules that map discretized input values onto discrete output values (see section 7).
- An **invocation** may be used as an alternative to FEEL, to model how a decision invokes decision logic that is represented by a Business Knowledge Model."

## Decision Logic Implementation in OpenRules® BDMS

To this point, descriptions for the process logic (as represented in BPMN 2.0) and decision logic (as represented in DMN Beta1) have been provided and they are tool-independent. However, tools implement standards in different ways and especially at the code-generation and execution level. OpenRules, chosen as per the criteria described above, is no exception. From this point onwards the paper will focus on the translation from the DMN Decision Logic level to the OpenRules representation (also MS Excel-based) and show the viability of the solution by executing the decision and reviewing the results.

Please take a minute to review the two figures at the top, Figure 1: Client Objective and Preferences and Figure 2: Ranked Mortgage Loan Options, that show the inputs and the outputs for the Determine Objective-based Mortgage Loan Recommendations decision.

*Author's caveat: While the organization of this paper seems to support the myth that the development lifecycle flows from one higher level set of diagrams, to a lower level set and finally to implementation, in reality, as is usually the case, the process was decidedly iterative. However, reading a description of an iterative process may seem confusing and therefore the paper is organized as it is with implementation as the final piece.*

### Term Options Decision Table

Figure 10 is the OpenRules decision table equivalent[7] to the DMN decision table of Figure 7 that implements the Business Knowledge Model for the Determine Term-based Scenarios decision. The "DecisionTableMultiHit" designation is the equivalent of "NI Collect" in the DMN representation.

[7] Only the rows for the *Lowest Monthly Payment* objective are provided. The full set of MS Excel spreadsheets and code that were developed for this proof-of-concept will be made available online.

| DecisionTableMultiHit TermOptionGenerationRules | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| # | If | If | If | ActionAny | Then | Then | Then | Then | Then | Then | Then |
| **#** | **Request Objective** | **Requested Product Category** | **Requested Product Term** | **Add New Recommendation** | **Recommended Program** | **Recommended Product Category** | **Recommended Product Term** | **Recommended Liens** | **Recommended 1st Lien LTV** | **Recommended 2nd Lien LTV** | **Recommended Loan Amount** |
| 1 | Lowest Monthly Payment | Fixed | | :=add(decision) | := ${Requested Program} | ARM | 3/1 | First | ::= $R{Requested LTV} | 0 | ::= $R{Requested Loan Amount} |
| 2 | | Fixed | 15 | :=add(decision) | := ${Requested Program} | Fixed | 20 | First | ::= $R{Requested LTV} | 0 | ::= $R{Requested Loan Amount} |
| 3 | | Fixed | 20 | :=add(decision) | := ${Requested Program} | Fixed | 30 | First | ::= $R{Requested LTV} | 0 | ::= $R{Requested Loan Amount} |
| 4 | | Fixed | 30 | :=add(decision) | := ${Requested Program} | Fixed | 40 | First | ::= $R{Requested LTV} | 0 | ::= $R{Requested Loan Amount} |
| 5 | | ARM | 7/1 | :=add(decision) | := ${Requested Program} | ARM | 3/1 | First | ::= $R{Requested LTV} | 0 | ::= $R{Requested Loan Amount} |
| 6 | | ARM | 5/1 | :=add(decision) | := ${Requested Program} | ARM | 3/1 | First | ::= $R{Requested LTV} | 0 | ::= $R{Requested Loan Amount} |
| 7 | | ARM | 3/1 | :=add(decision) | := ${Requested Program} | ARM | 1/1 | First | ::= $R{Requested LTV} | 0 | ::= $R{Requested Loan Amount} |

**Figure 10: OpenRules Decision Table Implementing the DMN Term Options Decision Table**

In OpenRules the decision table behavior is governed by a template; in this case, the template dictates that the result columns are executed from left to right. The "Add New Recommendation" column is added so that if a rule executes then a new option is explicitly created by the Java code implemented in the custom add method applied to a decision object. The code snippet that is executed is:

```
Method void add(Decision decision)

LoanOptions loanOptions = decision.get("LoanOptions");
LoanOption option = loanOptions.addLoanOption();
decision.useBusinessObject("LoanOption",option);
```

Four additional columns are added to explicitly implement the default instruction that if a cell is empty its value is taken from the initial client preferences. Given the client preferences, business rule #6 can be read as: If the objective is Lowest Monthly Payment and the client preference is a 5/1 adjustable rate mortgage (ARM) then add an option with the Program of Non-prime, a 3/1 ARM 1st Lien, with no 2nd lien, a requested LTV of 87.5% and a Loan Amount of $350,000.

With a little more tinkering it is also possible to hide these columns in the definition of the decision table template, however, showing them here gives a bit more of the flavor of possible customizations versus out-of-the-box behavior.

## Generate Scenarios

The decision Determine Term-based Scenarios is one of four decisions that feed into the Generate Loan Constraints decision. To execute these four decisions together in OpenRules they are simply placed in a table as shown in Figure 11.

| Decision GenerateLoanConstraints | |
|---|---|
| **Decisions** | **Execute** |
| Term Option Generation Decision | TermOptionGenerationRules |
| Price Option Generation Decision | PriceOptionGenerationRules |
| Lien Option Generation Decision | LienOptionGenerationRules |
| Requested Product Generation Decision | RequestedProductGenerationRules |

**Figure 11: The Open Rules Representation for the Generate Loan Constraints Decision**

The output of the Generate Loan Constraints decision is the union of all the scenarios generated in the individual decisions that feed into it. Each scenario describes constraints that are then used to match to a specific lender's product. For example, the scenario could be for an adjustable rate mortgage with an adjustment after 5 years, requiring Prime credit and for a loan-to-value ratio of 90%. This could match to a product provided by the lender CityLender called "Prime ARM 5/1 Hi-LTV"[8].

## Determine Adjustors Decision Table

Figure 12 describes the OpenRules implementation of rules that cumulatively add risk-based rate adjustors to the base rate. It is equivalent to the business rules of Figure 9 represented using the DMN standard. The DMN version specifies the table as "SUM"

[8] For brevity, this case study does not describe a multi-lender product set. Also, to keep the length of the article manageable, the product definitions are not included. A full set of files for the OpenRules implementation will be provided online.

which automatically adds the values of each row that is triggered and so the result of the table invocation is the desired cumulative amount. In OpenRules, same is accomplished by initializing the total adjustor amount to 0 in the first row (note that there are no values for any of the "If" columns; this guarantees that this row will be triggered) and using "+=" as the operator for all other rows. Any row that is triggered adds its value to the total.

| DecisionTableMultiHit DetermineRateAdjustorRules | | | | |
|---|---|---|---|---|
| If | If | If | Conclusion | |
| **Recommended Program** | **Recommended Liens** | **Recommended 1st Lien LTV** | **Adjustor** | |
| | | | = | 0 |
| | First | [85.01..100] | += | 0.0025 |
| Non-Prime | | | += | 0.01 |

Figure 12: OpenRules Decision Table Implementing the DMN Determine Adjustors Decision Table

For example, given the client preferences stated above (Non-prime credit and an LTV of 87.5% on the 1st lien), all 3 rows will be triggered adding 0.0125 to the base rate. If the base rate is 5% (or 0.05) then the rate plus adjustors is 6.25% (or 0.0625).

## Determine Monthly Payment Decision

| Decision DetermineMonthlyPayment | | |
|---|---|---|
| ActionPrint | ActionExecute | Message |
| **Decisions** | **Execute** | **Description** |
| Define PMT | DefinePMT | Calculate Principle and Interest based on rate, term and loan amount. |
| Define Taxes And Insurance | DefineTaxesAndInsurance | Calculate Taxes and Insurance as 1.5% of the loan amount. |
| Define Mortgage Insurance | DefineMortagageInsurance | Determine Mortgage Insurance based on the 1st-lien loan-to-value ratio. |
| Calculate Monthly Payment | CalculateMonthlyPayment | Calculate the total Monthly Payment as the sum of all of the above. |

Figure 13: Determine Monthly Payment

Determining the total Monthly Payment includes a few calculations and some straightforward decision tables.

The top-level component sequentially executes four steps:

1. Define PMT: This calculates the Principle and Interest on the loan given the base mortgage rate, the adjustors to the rate, the number of months the loan is amortized over and the loan amount.

| **DecisionTable DefinePMT** |
|---|
| Action |
| **PMT** |
| ::= PMT($R{Base Rate} + $R{Adjustor}, $I{Months}, $R{Recommended Loan Amount}) |

| **Method double PMT(double rate, int months, double amount)** |
|---|
| double r = rate/12;<br>double t = months;<br>double pmt = (amount*r)/(1 - Math.pow(1+r,-t));<br>return pmt; |

"$R" denotes a Real value.
"$I" denotes an Integer value.

2. Define Taxes and Insurance:

| **DecisionTable DefineTaxesAndInsurance** |
|---|
| Action |
| **Taxes and Insurance** |
| ::= $R{Recommended Loan Amount} * 0.015/12 |

3. Define Mortgage Insurance:

Implements additional mortgage insurance if the 1st-lien loan-to-value ratio (LTV) is greater than 80%.

| DecisionTable DefineMortagageInsurance | |
|---|---|
| If | Action |
| Recommended 1st Lien LTV | **Mortgage Insurance** |
| >80 | ::= $R{Recommended Loan Amount} * 0.0005 |
| | 0 |

4. Define Monthly Payment:

Simply adds up the previous three values.

| DecisionTable DefineMonthlyPayment |
|---|
| Action |
| **Monthly Payment** |
| ::= $R{PMT} + $R{Taxes and Insurance} + $R{Mortgage Insurance} |

## Determine Objective-based Mortgage Loan Recommendations Decision

At this point we have gone through examples of all the building blocks for a working application in OpenRules implementing the DMN representation of the mortgage recommendation decision logic. Not all the material is presented in this paper for brevity and clarity. One quirk that presents itself at the topmost level needs to be highlighted: While at face value the table representing the Determine Objective-based Mortgage Loan Recommendation decision seems to be similar to other tables in that it executes several decisions sequentially, there remains the issue of invoking business logic on a *set* of results.

| Decision Main | | | |
|---|---|---|---|
| Condition | | ActionPrint | ActionExecute |
| **Number of Loan Options** | | **Decisions** | **Execute** |
| | | Generate Loan Constraints and Match them to Lender Products | GenerateLoanConstraintsAndMatchProducts |
| > | 0 | Determine Monthly Payments for Generated Loan Options | DetermineMonthlyPayments |
| > | 1 | Determine Objective-based Ranking | RankLoanOptions[9] |

Figure 14: The Topmost Decision Determine Objective-based Mortgage Loan Recommendations

As shown in the DRG in Figure 4, the topmost decision is dependent on 3 other decisions[10]. The OpenRules tabular representation implies not only the dependency shown in Figure 4 but also the order of execution derived from the process flow shown in Figure 3. Drilling down to an additional level of the decision Generate Loan Constraints and Match Products, see Figure 15, shows that it in turn is dependent on 2 decisions: Generate Loan Constraints and Match to Lender Products.

| Decision GenerateLoanConstraintsAndMatchProducts | | | |
|---|---|---|---|
| Condition | | ActionPrint | ActionExecute |
| **Number of Loan Options** | | **Decisions** | **Execute** |
| | | Generation Loan Constraints (initial loan options) | GenerateLoanConstraints |
| > | 0 | For each loan constraint, Match It to Lender Products | MatchToLenderProducts |

Figure 15: Decision Generate Loan Constraints and Match to Lender Products

[9] The decision of determining the rank for each loan option is not fully implemented for the proof-of-concept. The default rank order implemented is by ascending total monthly payment which is appropriate for the *lowest monthly payment* objective used in the test data. That is, the loan option with the lowest monthly payment is presented first.

[10] This dependency is not temporal. That is, in a DMN representation, decision dependencies imply a required input. Without that input, a decision simply cannot execute. A temporal flow on the other hand, can be ascertained from the process representation in Figure 3.

Already discussed is that Generate Loan Constraints returns a set of loan scenarios. For each scenario there needs to be a matched product and a monthly payment before the loan options can be ranked. That is, Match Lender Products and Determine Monthly Payment need to be invoked individually for each loan option. OpenRules allows us to use loops (and any other valid Java constructs) inside MS Excel tables of the type "Method" as shown below where the decision DetermineBaseProductRules is invoked for each loan option. DetermineBaseProductRules will utilize the loan constraints derived above by GenerateLoanConstraints and match them to specific lender products[11]:

| Method void MatchToLenderProducts(Decision decision) |
| --- |
| ArrayList loanOptions = $O{LoanOptions}.getLoanOptions();<br>for(int i=0; i< loanOptions.size(); i++) {<br>  LoanOption rec = loanOptions.get(i);<br>  decision.useBusinessObject("LoanOption", rec);<br>  decision.log("\n"+rec);<br>  DetermineBaseProductRules(decision);<br>} |

[11] The relationship between constraints and products is that of one-to-many. That is, there may be multiple products from a single lender that match a loan constraint, or, there may be products provided by multiple lenders that match these constraints. This implementation, however, limits this relationship to that of one-to-one for the sake of simplicity.

In the OpenRules implementation, the act of executing the top level decision cascades until all the components are satisfied. The final results for the input values shown in Figure 1 (client objective and preferences) are:

**Execution Report of Wed Jun 11 12:50:43 EDT 2014**

| Request | ID | Program | Category | Term | Liens | LTV 1st | LTV 2nd | Amount | Mortgage Insurance | Tax & Insurance | Base Rate | Adjustors | P&I | Total Monthly Payment |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 31 | Prime | Fixed | 30 | First | 87.5 | 0.0 | 350000.0 | 175.0 | 437.5 | 5.385 | 0.25 | 2,017.01 | 2,629.51 |
| 1 | 21 | Non-Prime | Fixed | 30 | First | 85.0 | 0.0 | 340000.0 | 170.0 | 425.0 | 5.385 | 1.0 | 2,123.38 | 2,718.38 |
| 1 | 01 | Non-Prime | ARM | 3/1 | First | 87.5 | 0.0 | 350000.0 | 175.0 | 437.5 | 4.875 | 1.25 | 2,126.64 | 2,739.14 |
| 1 | 41 | Non-Prime | Fixed | 30 | First+Second | 80.0 | 10.0 | 360000.0 | 0.0 | 450.0 | 5.875 | 1.0 | 2,364.94 | 2,814.94 |
| 1 | 11 | Non-Prime | Fixed | 40 | First | 87.5 | 0.0 | 350000.0 | 175.0 | 437.5 | 6.0 | 1.25 | 2,238.85 | 2,851.35 |
| 1 | 51 | Non-Prime | Fixed | 30 | First | 87.5 | 0.0 | 350000.0 | 175.0 | 437.5 | 5.385 | 1.25 | 2,243.40 | 2,855.90 |

The benefits of each loan option have also automatically generated in the HTML format:

| Request | ID | Benefits |
|---|---|---|
| 01 | 31 | Corresponding Prime product carries a lower rate; Requires qualifying by more stringent criteria. |
| 01 | 21 | Price advantage at 85% over higher LTV; Specific to a particular lender. |
| 01 | 01 | ARM products typically offer lower rates initially; 3/1 is fixed for first 3 years. |
| 01 | 41 | Avoids Mortgage Insurance; Scenario includes 2 loans closing simultaneously. |
| 01 | 11 | A longer term reduces the monthly payment. |
| 01 | 51 | Requested product |

In this particular case, the requested product is the least advantageous to the client based on their objective and preferences. Five other, better, options are generated and presented in order of increasing monthly payment.

## Recap and Conclusions

Using the complex-enough problem of Mortgage Loan Recommendation as a backdrop, the BPMN 2.0 standard was used to model the business process, the new OMG Decision Model & Notation Beta1 standard was used to represent the decision logic and OpenRules® was utilized to implement a proof-of-concept using verifiable data.

As input, a client's stated preference for a mortgage solution as well as their objective were used in conjunction with lender product and pricing data and a set of mortgage loan configuration rules, to produce a ranked set of mortgage loan options and their corresponding benefits. The rankings were influenced by the client objective.

A chain of decisions was executed to arrive at the final outcome.

In closing, using Decision Tables for simplicity and compactness in representing decision logic is not a new concept but the formalization of a standard for representing a complete package of material in support of decision tables in particular (and other forms of decision logic in general) is a welcome addition. Thus, DMN's ability to bring together context of decision table invocation, authority levels for decision management maintenance and governance, and interdependencies between decisions can bridge the gap between business process representations and low-level IT data models and code. Encapsulation of decision logic brings with it the benefits of modularity and flexibility as-well-as transparency throughout the enterprise from business people to IT staff.

## About the Author

Gil Ronen is a technologist with a track record building award-winning, published, and patented automated decision-making systems. Gil's focus is on strategy, methodology, modeling and management at the nexus of business rules (BRM), business process (BPM), algorithms and scoring models.

Specialties: Automated Reasoning (BRM, BPM, algorithms, models for underwriting, pricing, deal-structuring, counter-offers, user-modeling); Knowledge Architecture (elicitation, modeling and life-cycle; BPMN, DMN, UML); Delivery management (P&L management and execution, engagement management, client partner relationships, Six Sigma delivery); Product and technical development (hands-on experience developing knowledge management tools); Talent acquisition, mentoring, and retention.

Contact:
Email: GilRonen@RevVisionConsulting.com
Phone: 917.345.7476
Linked In: www.linkedin.com/in/gilronen/

## Acknowledgements

Thanks to OpenRules CTO Jacob Feldman for his help with the OpenRules implementation.

Thanks to Nick Broom of Visionalysis for his whitepaper The Decision Model and Notation (DMN) Standard – A Worked Example (http://tinyurl.com/mfekt95) that was helpful in the preparation of this case study.

## References

The OpenRules implementation of the Mortgage Recommender decision model can be found on the website:
www.DMCommunity.org
Object Management Group, 2014. Decision Model and Notation. [Online] Available at:
http://www.omg.org/spec/DMN/1.0/Beta1/PDF/

## Trademarks, Copyrights

OpenRules® - http://openrules.com/
MS Excel® is a registered trademark of the Microsoft Corporation.
All other products or company names mentioned are used for identification purposes only, and may be trademarks of their respective owners.

## Disclaimer of Warranty

The author is not a member of the DMN committee or affiliated in any way, and has not been financially compensated by any party for any of the content found herein. The entire risk as to the quality and performance of software developed using this specification is borne by you. This disclaimer of warranty constitutes an essential part of the license granted to you to use this specification.

WHILE THIS PUBLICATION IS BELIEVED TO BE ACCURATE, IT IS PROVIDED "AS IS" AND MAY CONTAIN ERRORS OR MISPRINTS. GIL RONEN, REVVISION CONSULTING AND THE COMPANIES LISTED ABOVE MAKE NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS PUBLICATION, INCLUDING BUT NOT LIMITED TO ANY WARRANTY OF TITLE OR OWNERSHIP, IMPLIED WARRANTY OF MERCHANTABILITY OR WARRANTY OF FITNESS FOR A PARTICULAR PURPOSE OR USE. IN NO EVENT SHALL GIL RONEN, REVVISION CONSULTING OR ANY OF THE COMPANIES LISTED ABOVE BE LIABLE FOR ERRORS CONTAINED HEREIN OR FOR DIRECT, INDIRECT, INCIDENTAL, SPECIAL, CONSEQUENTIAL, RELIANCE OR COVER DAMAGES, INCLUDING LOSS OF PROFITS, REVENUE, DATA OR USE, INCURRED BY ANY USER OR ANY THIRD PARTY IN CONNECTION WITH THE FURNISHING, PERFORMANCE, OR USE OF THIS MATERIAL, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.