USERV Auto Insurance Rule Model in Corticon

Mike Parish Progress Software

Contents

Introduction 3

Vocabulary 4

Database Connectivity 4

Overall Structure of the Decision 6

Preferred Clients 7

1. Preferred Client (Discussion of Operators) 7
2. Elite Client (Discussion of Calculations) 8

Auto Eligibility Rules 8

1. High Theft Probability (Discussion of Lookup Tables) 8
2. Potential Theft Category (Discussion of Ambiguity) 9
3. Potential Occupant Injury Category (Rule Overrides) 12
4. Rules for Auto Eligibility (Discussion of Completeness) 12

The Complete Decision Service 16

The Test Scenarios 16

Sara Klaus Original Preferred Client Rules (V1) 17

Sara Klaus V2 (with the grandfathered rule for preferred client) 18

Mark Houston (with the V2 rules) 18

Mark Houston (Angie's Accidents with V2) 18

Appendix – Additional Rule Sheets 19

Driver Eligibility Rules 19

Driver Training 19

Driver Age 19

Driver Eligibility 20

Driver Record 20

Eligibility Scoring 21

Long Term Client Scoring 21

Eligibility Scoring 22
Scoring Interpretation 23
Pricing Business Rules 23
Auto Base Premium 23
Auto Theft Premium 24
Auto Injury Premium 24
Auto Option Premium 24
Auto Age Premium 25
Auto Airbag Discount 26
Auto Alarm Discount 26
Apply Discount 27
Driver Premiums 27
Market Segment 28
Policy Base Premium 29
Final Results 29

Introduction

USERV Insurance was developed in 2006 as a standard business rules use case that all the vendors could implement so that prospects would have a common example to use when evaluating business rules engines. Back in 2006 this was a prominent feature of the Business Rules Forum. Each vendor would present their solution to the entire conference. The original Corticon solution was implemented using version 4.3. This document presents the solution in the latest version of Corticon (5.4)

The original USERV Use Case Specification is available on the DM Community website.
You may follow this link to get a very detailed description of the business rules, a related fact model, business processes, and test cases

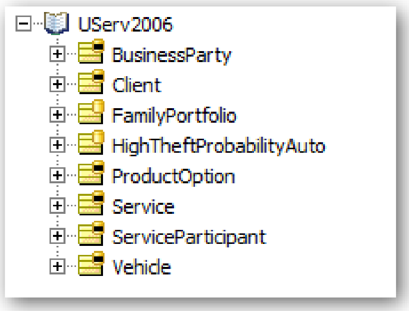
Below is a walk though of the rule sheets comprising the Corticon solution.

Note that this is not intended to be a tutorial on Corticon but rather an explanation of how the USERV use case was modeled with Corticon. Some details will be provided where necessary to understand how Corticon processes rules but it is assumed that readers are already familiar with the concept of business rules engines and in particular the use of decision tables as the primary means to represent business logic.

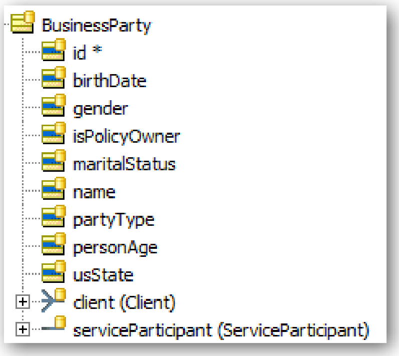
More details regarding Corticon features, functionality and concepts can be found in the online documentation available at
http://documentation.progress.com/output/ua/Corticon

Vocabulary

Every piece of software, whether it’s a traditional programming language or a rule engine, needs to define the data it will work with. In the Corticon world this is referred to as a Vocabulary.
The vocabulary in Corticon contains details about the business objects and their relationships, their attributes, data types and possible values (it can also contain the mapping of business objects to tables and attributes to columns in a database).
The vocabulary was constructed to match the data model in the original USERV specification. It can be automatically imported from a variety of sources.

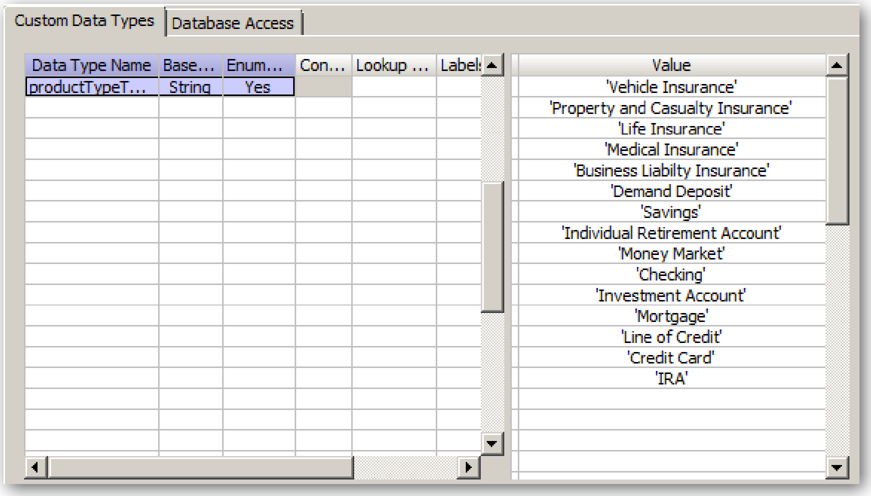


- UServ2006
  - BusinessParty
  - Client
  - FamilyPortfolio
  - HighTheftProbabilityAuto
  - ProductOption
  - Service
  - ServiceParticipant
  - Vehicle



- BusinessParty
  - id *
  - birthDate
  - gender
  - isPolicyOwner
  - maritalStatus
  - name
  - partyType
  - personAge
  - usState
  - client (Client)
  - serviceParticipant (ServiceParticipant)

Custom data types can be defined for attributes that must be restricted to specific values.



Custom Data Types | Database Access

Data Type Name	Base…	Enum…	Con…	Lookup …	Label:
productTypeT…	String	Yes			

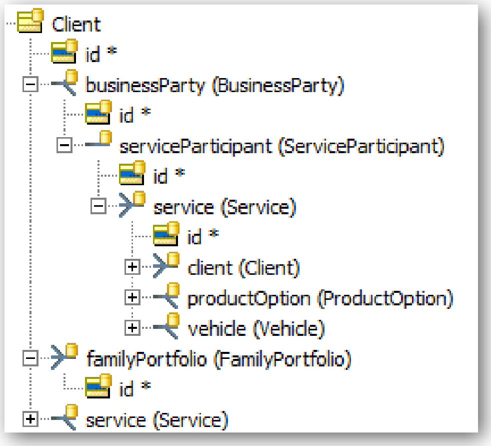
Value
'Vehicle Insurance'
'Property and Casualty Insurance'
'Life Insurance'
'Medical Insurance'
'Business Liabilty Insurance'
'Demand Deposit'
'Savings'
'Individual Retirement Account'
'Money Market'
'Checking'
'Investment Account'
'Mortgage'
'Line of Credit'
'Credit Card'
'IRA'

These values can be typed in or loaded from a database.

Database Connectivity

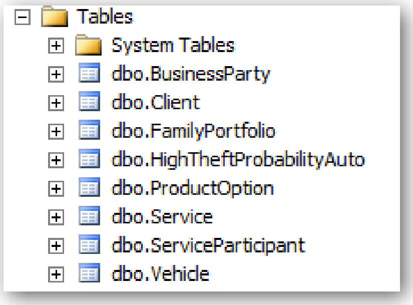
Observe that the icons for the business objects have little disk decorations on them. This is because one of the features of Corticon is the ability to read data directly from a database. This is done automatically and transparently to the rule author. No SQL needs to be written for this.

The diagram below shows the relationships or associations between the objects. Most of the attributes have been hidden in order to keep the diagram small. The asterisk marks the primary key for the database.



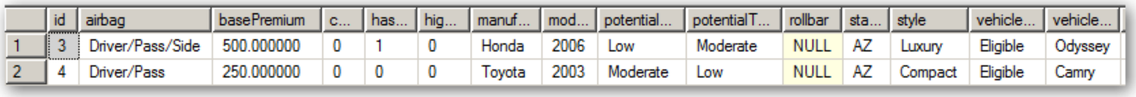
```
Client
  id *
  businessParty (BusinessParty)
    id *
    serviceParticipant (ServiceParticipant)
      id *
      service (Service)
        id *
        client (Client)
        productOption (ProductOption)
        vehicle (Vehicle)
  familyPortfolio (FamilyPortfolio)
    id *
  service (Service)
```

And in SQL



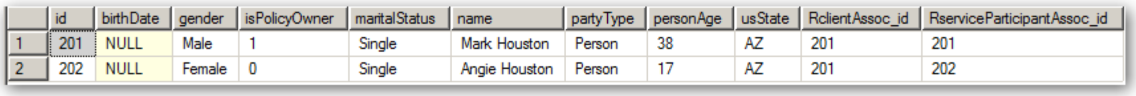
```
Tables
  System Tables
  dbo.BusinessParty
  dbo.Client
  dbo.FamilyPortfolio
  dbo.HighTheftProbabilityAuto
  dbo.ProductOption
  dbo.Service
  dbo.ServiceParticipant
  dbo.Vehicle
```

Querying the Vehicle table might show something like this:



	id	airbag	basePremium	c...	has...	hig...	manuf...	mod...	potential...	potentialT...	rollbar	sta...	style	vehicle...	vehicle...
1	3	Driver/Pass/Side	500.000000	0	1	0	Honda	2006	Low	Moderate	NULL	AZ	Luxury	Eligible	Odyssey
2	4	Driver/Pass	250.000000	0	0	0	Toyota	2003	Moderate	Low	NULL	AZ	Compact	Eligible	Camry

The business Party table (which will contain the drivers) might look like this:

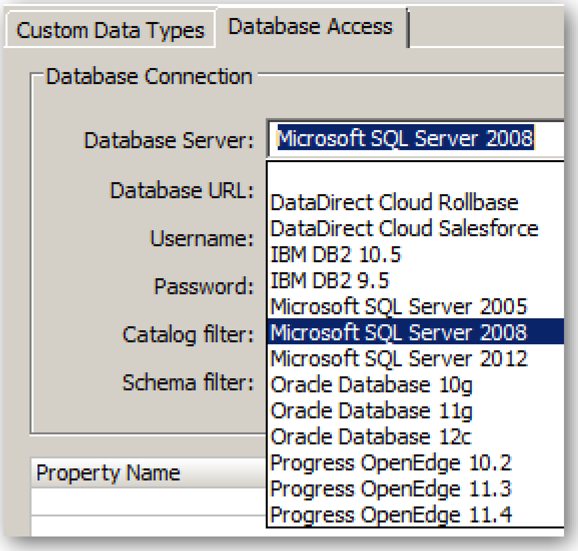


	id	birthDate	gender	isPolicyOwner	maritalStatus	name	partyType	personAge	usState	RclientAssoc_id	RserviceParticipantAssoc_id
1	201	NULL	Male	1	Single	Mark Houston	Person	38	AZ	201	201
2	202	NULL	Female	0	Single	Angie Houston	Person	17	AZ	201	202

The last two columns show how these records are associated with other business objects. Fortunately as a rule author you do not need to be concerned with this technical detail. Corticon will handle that automatically so that if your rules need the vehicles associated with a given policy for a given client, Corticon will automatically create the correct SQL to do the retrieval. Corticon automatically determines the correct SQL JOIN expression from the database schema for related tables.

Corticon can also update the database with the new results after the rules have executed.

Corticon supports a wide variety of relational databases directly:



Database Server: Microsoft SQL Server 2008

DataDirect Cloud Rollbase
DataDirect Cloud Salesforce
IBM DB2 10.5
IBM DB2 9.5
Microsoft SQL Server 2005
Microsoft SQL Server 2008
Microsoft SQL Server 2012
Oracle Database 10g
Oracle Database 11g
Oracle Database 12c
Progress OpenEdge 10.2
Progress OpenEdge 11.3
Progress OpenEdge 11.4

And, in addition, non-relational or proprietary data sources can be accessed using the **Progress Data Direct Cloud** connectors.

These enable simple, fast connections to cloud and on-premise data regardless of the source—**SaaS** apps, big data stores, relational databases, and more—**using a single ODBC, JDBC driver, or OData-based API.**

Note: This feature is available even if you do not use Corticon – it will work with any vendor's rule engine.

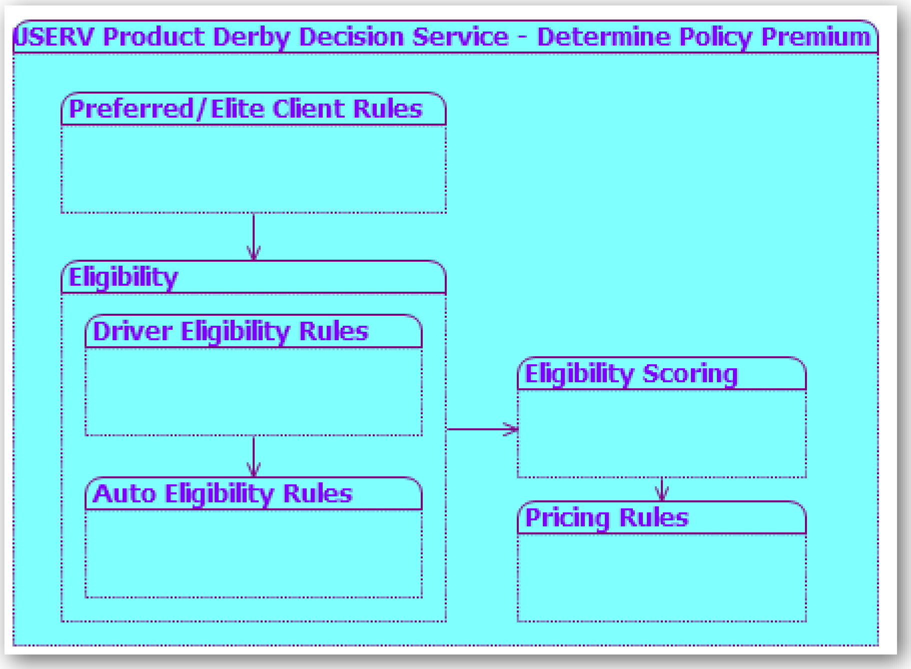
More information on these data sources is available here:
https://www.progress.com/products/datadirect-cloud

Overall Structure of the Decision

The entire decision comprises four main parts:

1. Preferred (and Elite) client rules
2. Eligibility Determination (for Clients, Vehicles and Drivers)
3. Eligibility Scoring (Calculation and Interpretation)
4. Price Determination (Driver, Vehicle, Client premiums and discounts)

The structure of a Corticon decision service is represented by a Rule Flow Diagram.


USERV Product Derby Decision Service - Determine Policy Premium
Preferred/Elite Client Rules
Eligibility
Driver Eligibility Rules
Auto Eligibility Rules
Eligibility Scoring
Pricing Rules

In order to keep the diagram uncluttered, the actual rule sheets that are inside each of the inner sub flows have been omitted. In the subsequent discussion we will show those details.

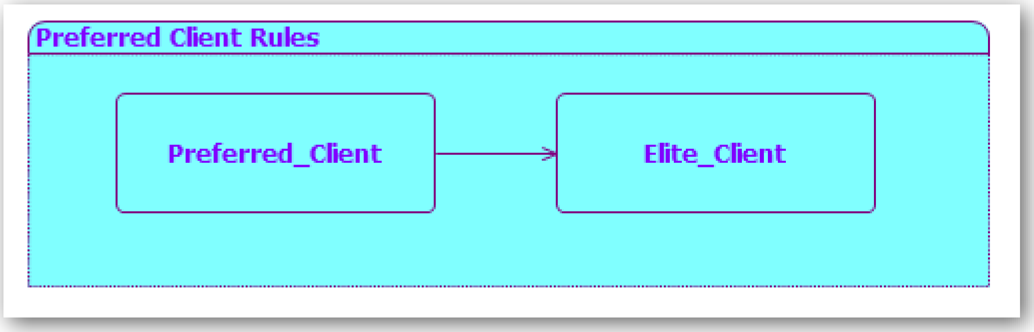
A Corticon rule sheet consists of several parts:

1. **Rule Statements**. These come (copy/paste) directly from the original spec
2. **Decision Table**. Each rule is a column in the table
3. **Scope**. This defines the context for the rules in that rule sheet
4. **Filters**. These define what data should be processed by the rule sheet

We'll look at each of the rule sheets in turn and with each one there will be some relevant discussion of the rule modeling techniques (indicated in parentheses) that might apply. We'll show how Corticon can find the significant number of ambiguities that are in the original specification. Some of the later rule sheets are very similar and so will be presented without discussion in the appendix.

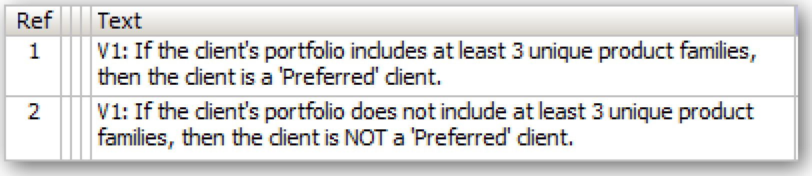
Preferred Clients

There are two rule sheets in this sub flow.



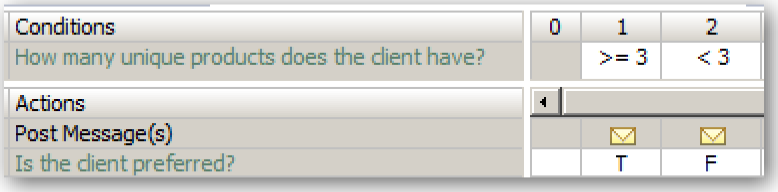
1. Preferred Client (Discussion of Operators)

How it's defined in the Spec



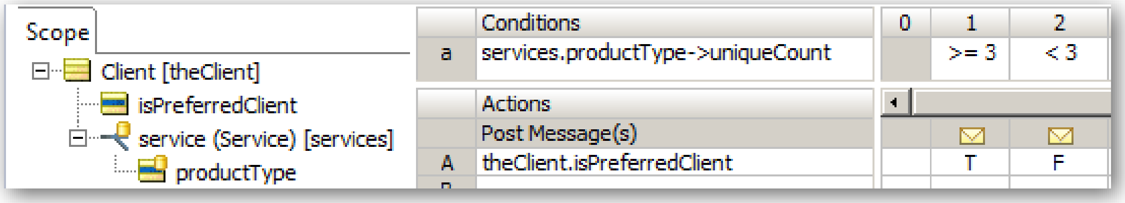
Ref	Text
1	V1: If the client's portfolio includes at least 3 unique product families, then the client is a 'Preferred' client.
2	V1: If the client's portfolio does not include at least 3 unique product families, then the client is NOT a 'Preferred' client.

How it's modeled in Corticon



Conditions	0	1	2
How many unique products does the client have?		>= 3	< 3
Actions			
Post Message(s)		✉	✉
Is the client preferred?		T	F

How it's implemented in Corticon



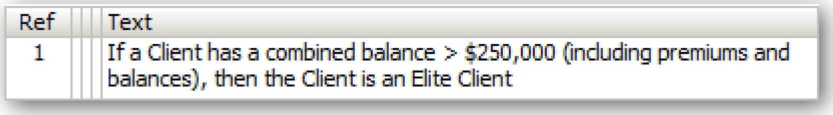
```
Scope
⊟ Client [theClient]
   isPreferredClient
   ⊟ service (Service) [services]
      productType
```

	Conditions	0	1	2
a	services.productType->uniqueCount		>= 3	< 3
	Actions			
	Post Message(s)		✉	✉
A	theClient.isPreferredClient		T	F

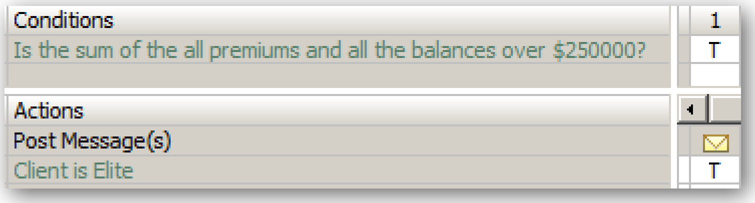
How it Works

The Scope section shows that a **Client** may have one or more **Service** (referred to using the alias **services.** In the condition, **->uniqueCount** is an operator that determines the number of unique values of the attribute **productType** across all the **services** belonging to any given Client. Corticon has a rich set of operators for performing operations on collections (these are similar to the aggregation and other features in the DMN). The built in operators can be supplemented with ones of your own design by writing them in Java.

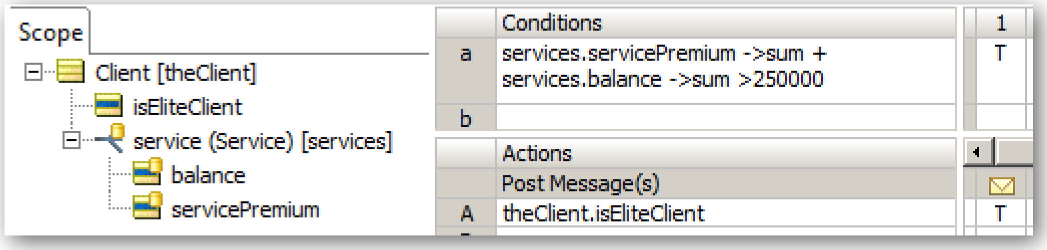
2. Elite Client (Discussion of Calculations)



Ref	Text
1	If a Client has a combined balance > $250,000 (including premiums and balances), then the Client is an Elite Client



Conditions	1
Is the sum of the all premiums and all the balances over $250000?	T
Actions	
Post Message(s)	✉
Client is Elite	T



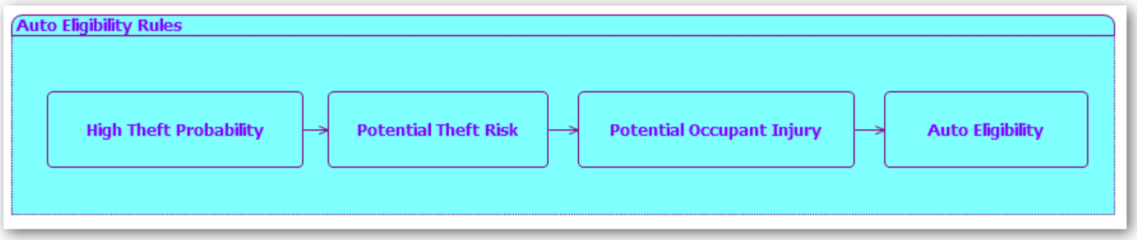
Scope
- Client [theClient]
  - isEliteClient
  - service (Service) [services]
    - balance
    - servicePremium

	Conditions	1
a	services.servicePremium ->sum + services.balance ->sum >250000	T
b		
	Actions	
	Post Message(s)	✉
A	theClient.isEliteClient	T

How it works

The alias "**services**" refers to ALL the services associated with a particular client (a one-to-many association). The **->sum** operator (a built in aggregation operator) adds up all the appropriate **servicePremiums** (and **balances**) for each of the **services**.
The scope section is how Corticon "knows" that clients have many services and that they should all be considered when evaluating the rule.

Auto Eligibility Rules

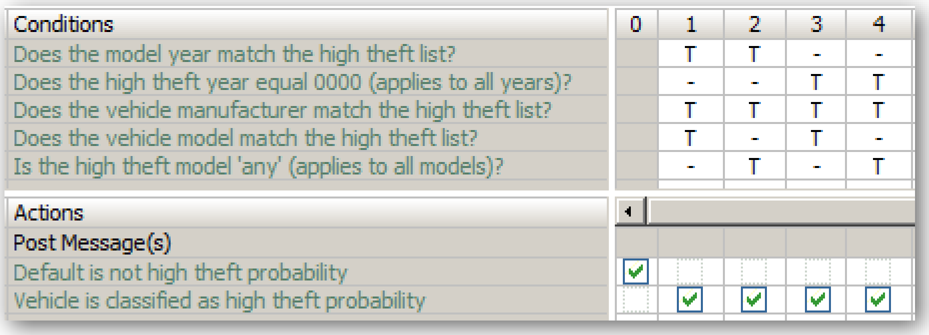


1. High Theft Probability (Discussion of Lookup Tables)

Original Rule Specification

If the vehicle considered for insurance matches the year (including wildcard 0000), make, and model (including wildcard "any") of the High Theft Probability Auto List, then the vehicle is a High Theft Probability risk

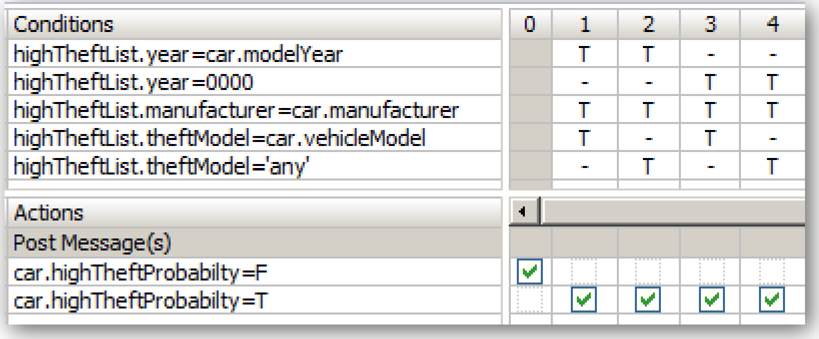
Definition in Corticon (restructured to show the four cases)



Conditions	0	1	2	3	4
Does the model year match the high theft list?		T	T	-	-
Does the high theft year equal 0000 (applies to all years)?		-	-	T	T
Does the vehicle manufacturer match the high theft list?		T	T	T	T
Does the vehicle model match the high theft list?		T	-	T	-
Is the high theft model 'any' (applies to all models)?		-	T	-	T
Actions					
Post Message(s)					
Default is not high theft probability	✓				
Vehicle is classified as high theft probability		✓	✓	✓	✓

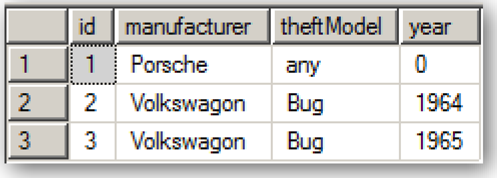
Note: The dash '-' indicates that a particular condition is not applicable in that rule

Implementation



Conditions	0	1	2	3	4
highTheftList.year=car.modelYear		T	T	-	-
highTheftList.year=0000		-	-	T	T
highTheftList.manufacturer=car.manufacturer		T	T	T	T
highTheftList.theftModel=car.vehicleModel		T	-	T	-
highTheftList.theftModel='any'		-	T	-	T
Actions					
Post Message(s)					
car.highTheftProbabilty=F	✓				
car.highTheftProbabilty=T		✓	✓	✓	✓

The alias **highTheftList** is a lookup table read from a database
Example of the table:

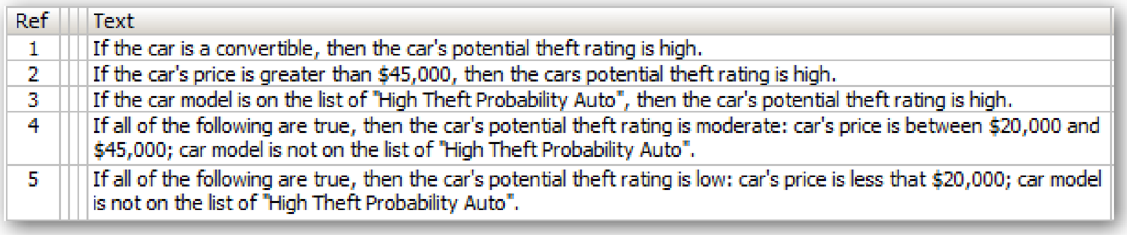


	id	manufacturer	theftModel	year
1	1	Porsche	any	0
2	2	Volkswagon	Bug	1964
3	3	Volkswagon	Bug	1965

Corticon can load the table when it's required and then the rules will locate any record that meets any of the four cases.
As an alternative to storing the high theft vehicles in a database table we could simply have made them into entries in a rule sheet. This may be faster than doing a database lookup.

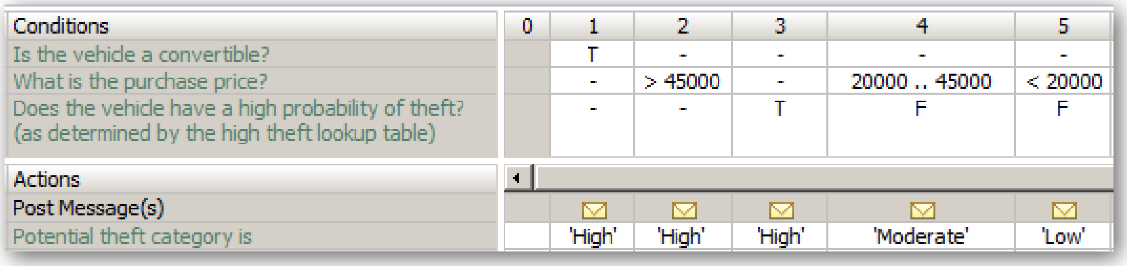
2. Potential Theft Category (Discussion of Ambiguity)

Here are the rules for determining the Potential Theft Category exactly as specified in the original documentation. But are they correct?



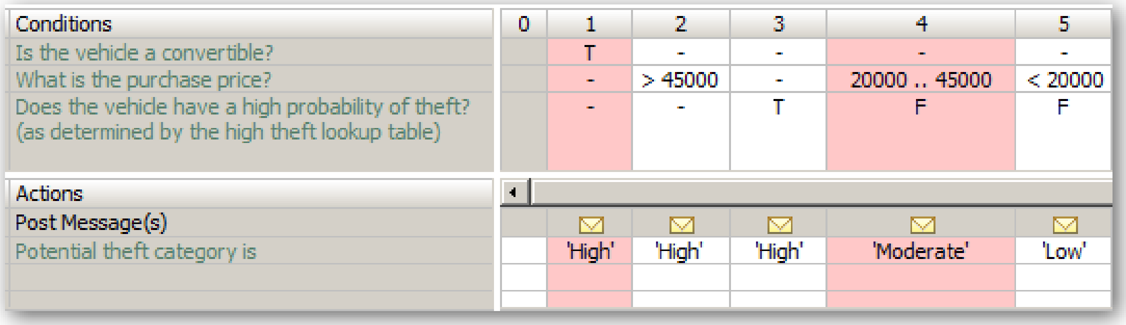
Ref	Text
1	If the car is a convertible, then the car's potential theft rating is high.
2	If the car's price is greater than $45,000, then the cars potential theft rating is high.
3	If the car model is on the list of "High Theft Probability Auto", then the car's potential theft rating is high.
4	If all of the following are true, then the car's potential theft rating is moderate: car's price is between $20,000 and $45,000; car model is not on the list of "High Theft Probability Auto".
5	If all of the following are true, then the car's potential theft rating is low: car's price is less that $20,000; car model is not on the list of "High Theft Probability Auto".

Here are the same rules modeled in Corticon:



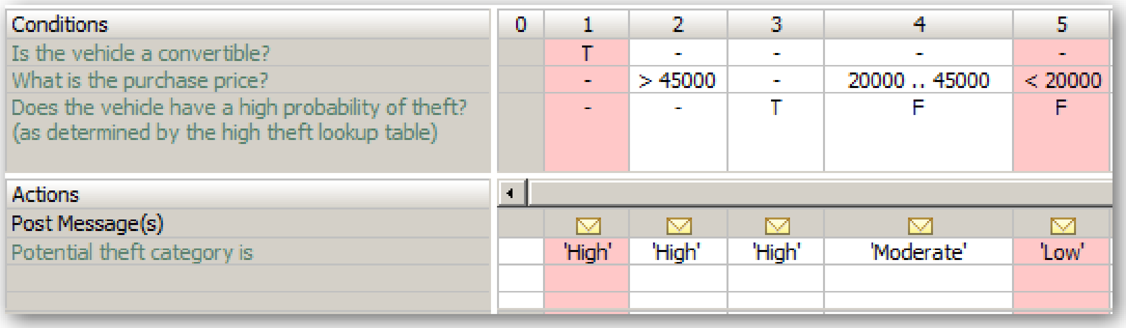
Conditions	0	1	2	3	4	5
Is the vehicle a convertible?		T	-	-	-	-
What is the purchase price?		-	> 45000	-	20000 .. 45000	< 20000
Does the vehicle have a high probability of theft? (as determined by the high theft lookup table)		-	-	T	F	F
Actions						
Post Message(s)		✉	✉	✉	✉	✉
Potential theft category is		'High'	'High'	'High'	'Moderate'	'Low'

The ***Corticon Conflict Checker*** tells us that there are conflicts (the pink columns):



Conditions	0	1	2	3	4	5
Is the vehicle a convertible?		T	-	-	-	-
What is the purchase price?		-	> 45000	-	20000 .. 45000	< 20000
Does the vehicle have a high probability of theft? (as determined by the high theft lookup table)		-	-	T	F	F
Actions						
Post Message(s)		✉	✉	✉	✉	✉
Potential theft category is		'High'	'High'	'High'	'Moderate'	'Low'

How should we rate a convertible that costs $30,000 that is not on the high theft list?
Rule 1 says "High" but rule 4 says "Moderate".
Corticon also detects a second conflict:

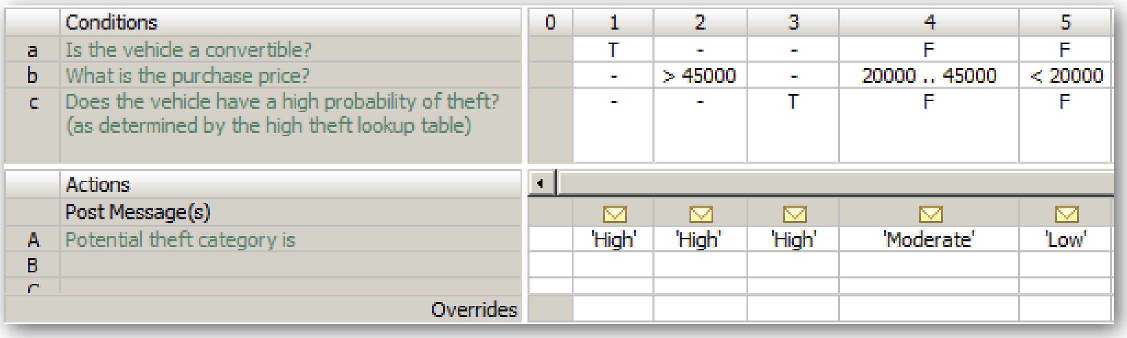


Conditions	0	1	2	3	4	5
Is the vehicle a convertible?		T	-	-	-	-
What is the purchase price?		-	> 45000	-	20000 .. 45000	< 20000
Does the vehicle have a high probability of theft? (as determined by the high theft lookup table)		-	-	T	F	F
Actions						
Post Message(s)		✉	✉	✉	✉	✉
Potential theft category is		'High'	'High'	'High'	'Moderate'	'Low'

Let's assume in both these cases "convertible" wins and we want the rating to be "High".
In general there are two ways to resolve conflicts:

1. Make the rules more precise
2. Add an override at the bottom of the rule column.

The preferred method is #1. In this case rules 4 and 5 can be modified to explicitly exclude Convertibles:

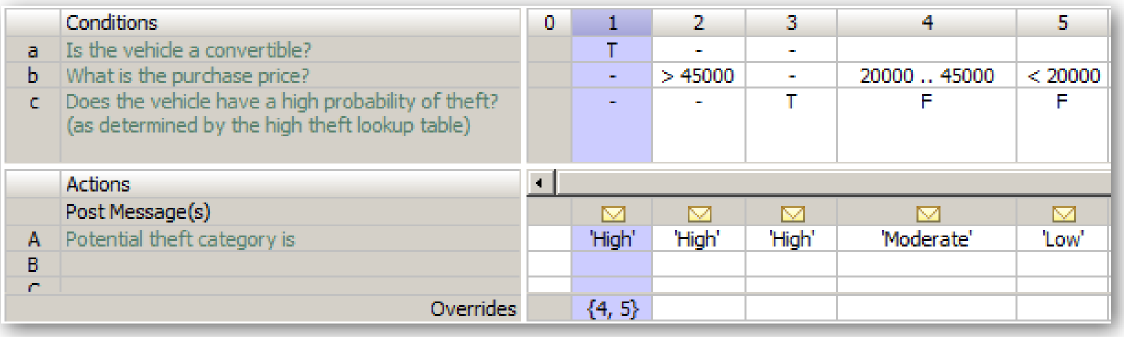


	Conditions	0	1	2	3	4	5
a	Is the vehicle a convertible?		T	-	-	F	F
b	What is the purchase price?		-	> 45000	-	20000 .. 45000	< 20000
c	Does the vehicle have a high probability of theft? (as determined by the high theft lookup table)		-	-	T	F	F

	Actions	0	1	2	3	4	5
	Post Message(s)		✉	✉	✉	✉	✉
A	Potential theft category is		'High'	'High'	'High'	'Moderate'	'Low'
B							
C							
	Overrides						

Rules 2 and 3 also overlap rule 1, they do not conflict because they conclude the same "High" rating. Corticon will execute both rules.

An alternative method is to explicitly provide an override:

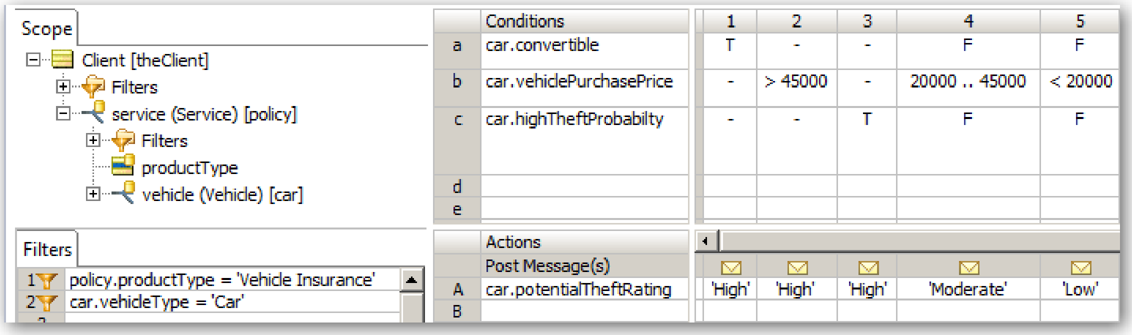


	Conditions	0	1	2	3	4	5
a	Is the vehicle a convertible?		T	-	-		
b	What is the purchase price?		-	> 45000	-	20000 .. 45000	< 20000
c	Does the vehicle have a high probability of theft? (as determined by the high theft lookup table)		-	-	T	F	F

	Actions	0	1	2	3	4	5
	Post Message(s)		✉	✉	✉	✉	✉
A	Potential theft category is		'High'	'High'	'High'	'Moderate'	'Low'
B							
C							
	Overrides		{4, 5}				

Effectively an override is really just another business rule. So if it's important to understand what the override is really saying it's usually better to represent it as an actual rule entry (case #1). If there are a lot of overrides it can be hard to understand what's going on.

How it's implemented



Scope

- Client [theClient]
  - Filters
  - service (Service) [policy]
    - Filters
    - productType
    - vehicle (Vehicle) [car]

Filters

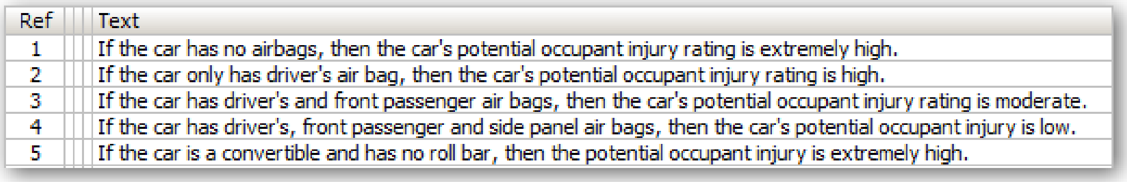
1 policy.productType = 'Vehicle Insurance'
2 car.vehicleType = 'Car'

	Conditions	1	2	3	4	5
a	car.convertible	T	-	-	F	F
b	car.vehiclePurchasePrice	-	> 45000	-	20000 .. 45000	< 20000
c	car.highTheftProbabilty	-	-	T	F	F
d						
e						

	Actions	1	2	3	4	5
	Post Message(s)	✉	✉	✉	✉	✉
A	car.potentialTheftRating	'High'	'High'	'High'	'Moderate'	'Low'
B						

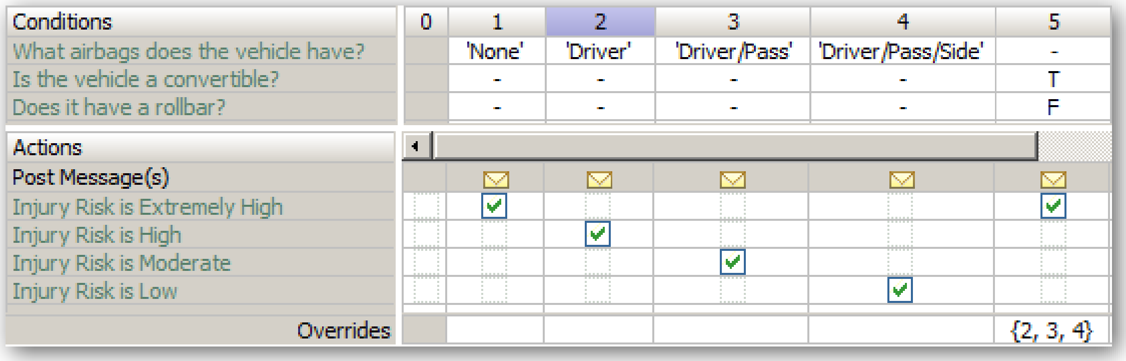
Notice the use of the filters to ensure that the rules only get applied to Vehicle Insurance policies and vehicles that are cars. Boats are presumably handled differently. The alias **car** is used to make it very clear that the rule applies to cars.

3. Potential Occupant Injury Category (Rule Overrides)



Ref	Text
1	If the car has no airbags, then the car's potential occupant injury rating is extremely high.
2	If the car only has driver's air bag, then the car's potential occupant injury rating is high.
3	If the car has driver's and front passenger air bags, then the car's potential occupant injury rating is moderate.
4	If the car has driver's, front passenger and side panel air bags, then the car's potential occupant injury is low.
5	If the car is a convertible and has no roll bar, then the potential occupant injury is extremely high.

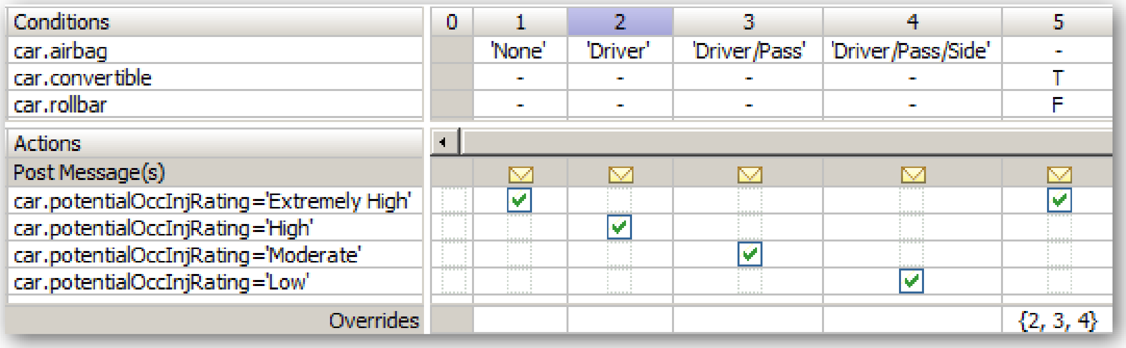
These rules also contain ambiguities:



Conditions	0	1	2	3	4	5
What airbags does the vehicle have?		'None'	'Driver'	'Driver/Pass'	'Driver/Pass/Side'	-
Is the vehicle a convertible?		-	-	-	-	T
Does it have a rollbar?		-	-	-	-	F
Actions						
Post Message(s)		✉	✉	✉	✉	✉
Injury Risk is Extremely High		✓				✓
Injury Risk is High			✓			
Injury Risk is Moderate				✓		
Injury Risk is Low					✓	
Overrides						{2, 3, 4}

By making rule 5 override rules 2, 3 and 4 we can resolve this ambiguity.

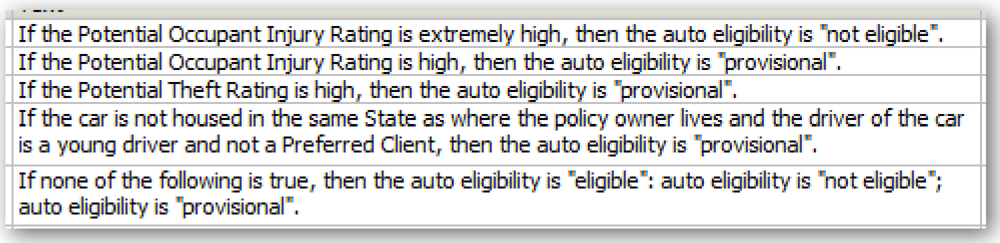
Implementation



Conditions	0	1	2	3	4	5
car.airbag		'None'	'Driver'	'Driver/Pass'	'Driver/Pass/Side'	-
car.convertible		-	-	-	-	T
car.rollbar		-	-	-	-	F
Actions						
Post Message(s)		✉	✉	✉	✉	✉
car.potentialOccInjRating='Extremely High'		✓				✓
car.potentialOccInjRating='High'			✓			
car.potentialOccInjRating='Moderate'				✓		
car.potentialOccInjRating='Low'					✓	
Overrides						{2, 3, 4}

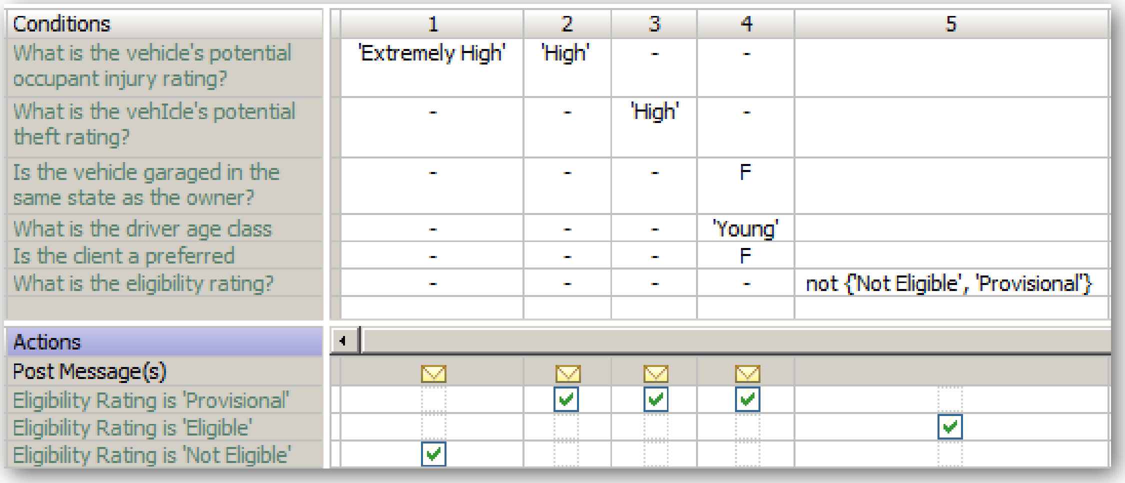
4. Rules for Auto Eligibility (Discussion of Completeness)

Here are the original rules for auto eligibility. Do you see any problems with these rules?



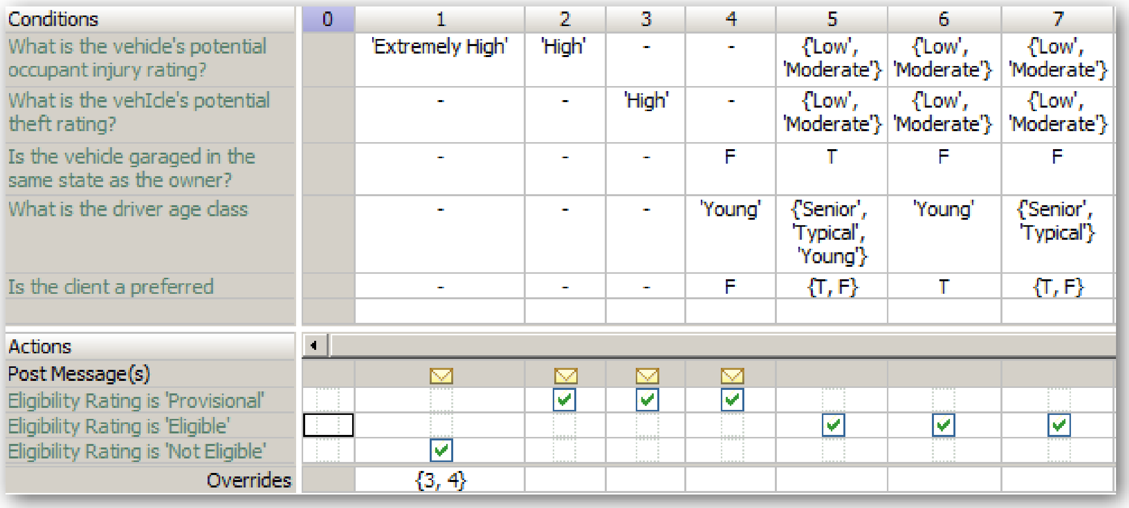
If the Potential Occupant Injury Rating is extremely high, then the auto eligibility is "not eligible".
If the Potential Occupant Injury Rating is high, then the auto eligibility is "provisional".
If the Potential Theft Rating is high, then the auto eligibility is "provisional".
If the car is not housed in the same State as where the policy owner lives and the driver of the car is a young driver and not a Preferred Client, then the auto eligibility is "provisional".
If none of the following is true, then the auto eligibility is "eligible": auto eligibility is "not eligible"; auto eligibility is "provisional".

Since it's hard to see the flaws in the English statements, let's first model this in Corticon exactly as stated:



Conditions	1	2	3	4	5
What is the vehicle's potential occupant injury rating?	'Extremely High'	'High'	-	-	
What is the vehIcle's potential theft rating?	-	-	'High'	-	
Is the vehicle garaged in the same state as the owner?	-	-	-	F	
What is the driver age class	-	-	-	'Young'	
Is the client a preferred	-	-	-	F	
What is the eligibility rating?	-	-	-	-	not {'Not Eligible', 'Provisional'}
Actions					
Post Message(s)	✉	✉	✉	✉	
Eligibility Rating is 'Provisional'		✓	✓	✓	
Eligibility Rating is 'Eligible'					✓
Eligibility Rating is 'Not Eligible'	✓				

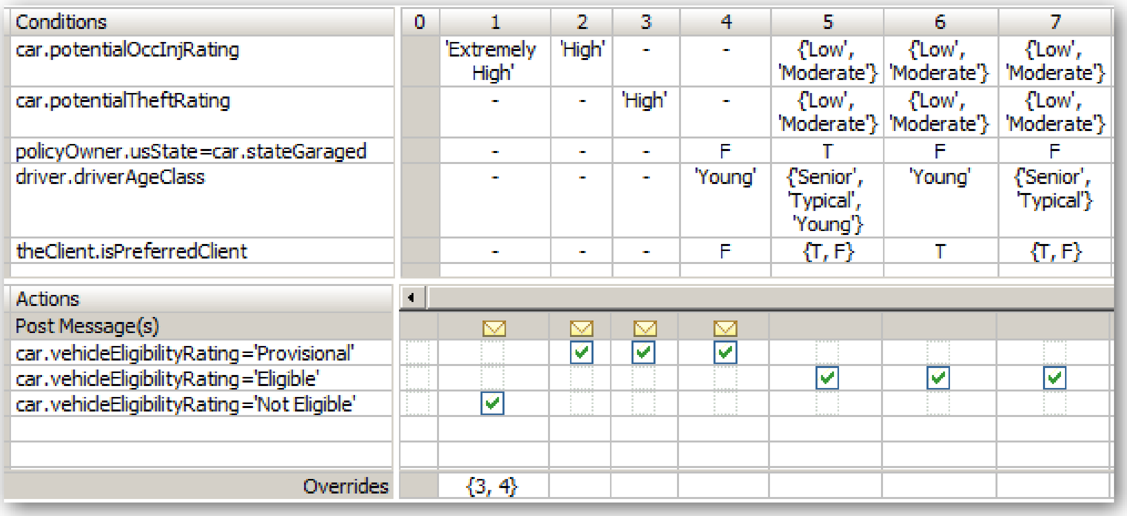
First the ambiguity checker will tell us that rule 1 is in conflict with rules 3 and 4 and 5
Secondly Corticon will tell us there is a logical loop since rule 5 references a value that is set in its own action. While Corticon can handle a self-referencing rule it's not the clearest way of expressing the rule.
A better way might be to enumerate the specific rules that assign "Eligible".
A quick way to set this up is to use the completeness checker to automatically add the missing conditions (5, 6, and 7). Corticon can't decide what action is required – it's not that smart! – you have to add that)



Conditions	0	1	2	3	4	5	6	7
What is the vehicle's potential occupant injury rating?		'Extremely High'	'High'	-	-	{'Low', 'Moderate'}	{'Low', 'Moderate'}	{'Low', 'Moderate'}
What is the vehIcle's potential theft rating?		-	-	'High'	-	{'Low', 'Moderate'}	{'Low', 'Moderate'}	{'Low', 'Moderate'}
Is the vehicle garaged in the same state as the owner?		-	-	-	F	T	F	F
What is the driver age class		-	-	-	'Young'	{'Senior', 'Typical', 'Young'}	'Young'	{'Senior', 'Typical'}
Is the client a preferred		-	-	-	F	{T, F}	T	{T, F}
Actions								
Post Message(s)		✉	✉	✉	✉			
Eligibility Rating is 'Provisional'			✓	✓	✓			
Eligibility Rating is 'Eligible'						✓	✓	✓
Eligibility Rating is 'Not Eligible'		✓						
Overrides		{3, 4}						

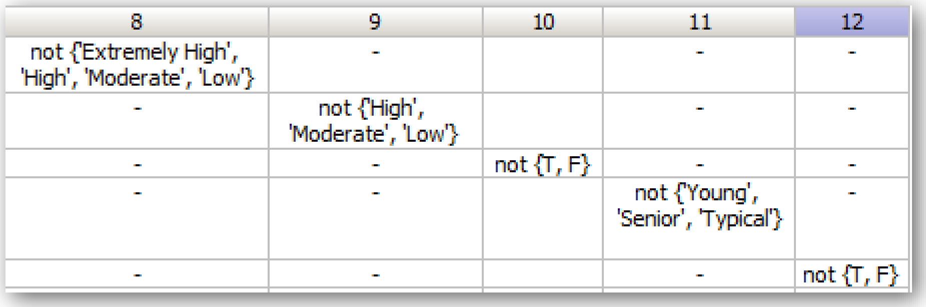
Note: In this rule sheet we have chosen to resolve the conflicts by using overrides. Sometimes it might actually be simpler than trying to adjust the rules to avoid ambiguity.

How it's implemented



Conditions	0	1	2	3	4	5	6	7
car.potentialOccInjRating		'Extremely High'	'High'	-	-	{'Low', 'Moderate'}	{'Low', 'Moderate'}	{'Low', 'Moderate'}
car.potentialTheftRating		-	-	'High'	-	{'Low', 'Moderate'}	{'Low', 'Moderate'}	{'Low', 'Moderate'}
policyOwner.usState=car.stateGaraged		-	-	-	F	T	F	F
driver.driverAgeClass		-	-	-	'Young'	{'Senior', 'Typical', 'Young'}	'Young'	{'Senior', 'Typical'}
theClient.isPreferredClient		-	-	-	F	{T, F}	T	{T, F}
Actions								
Post Message(s)		✉	✉	✉	✉			
car.vehicleEligibilityRating='Provisional'			✓	✓	✓			
car.vehicleEligibilityRating='Eligible'						✓	✓	✓
car.vehicleEligibilityRating='Not Eligible'		✓						
Overrides		{3, 4}						

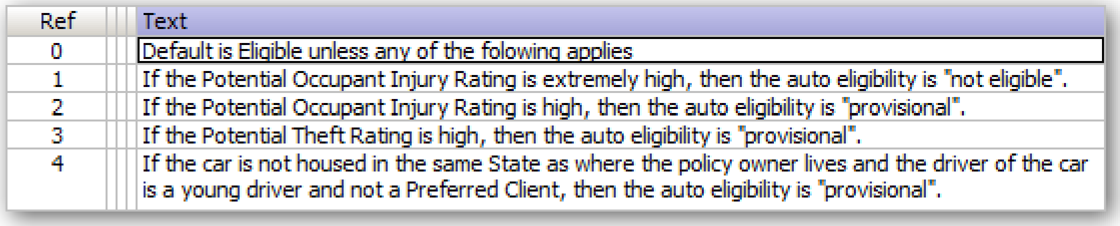
And even this is not the final solution. If we run the completeness checker, Corticon will want to know what action it should take for these conditions:



8	9	10	11	12
not {'Extremely High', 'High', 'Moderate', 'Low'}	-		-	-
-	not {'High', 'Moderate', 'Low'}		-	-
-	-	not {T, F}	-	-
-	-		not {'Young', 'Senior', 'Typical'}	-
-	-		-	not {T, F}

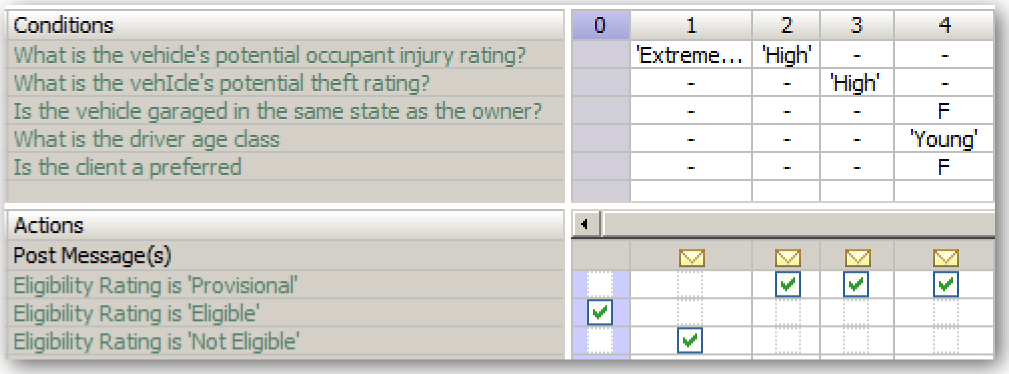
You might argue that these conditions should not arise. But as a general best practice it's safer to accommodate all possible outcomes. Sooner or later they will occur either as the result of bad data passed in from a client application or a rule modification that misspelled a word. Adding these error catching rules can make the rule sheet a bit bigger so an alternative is to have separate rule sheets that do the data validation. This technique can be useful because rule sheets can be reused in different decisions so you only need to develop the data validation logic once rather than repeating it in each rule sheet where an attribute is tested.

Another way to handle the "Eligible" case is to say a vehicle is considered eligible by default unless excluded by one of the four rules.



Ref	Text
0	Default is Eligible unless any of the folowing applies
1	If the Potential Occupant Injury Rating is extremely high, then the auto eligibility is "not eligible".
2	If the Potential Occupant Injury Rating is high, then the auto eligibility is "provisional".
3	If the Potential Theft Rating is high, then the auto eligibility is "provisional".
4	If the car is not housed in the same State as where the policy owner lives and the driver of the car is a young driver and not a Preferred Client, then the auto eligibility is "provisional".

This can be done conveniently by putting the action in column zero (which does not require any conditions). And then you don't need rules 5, 6 and 7.

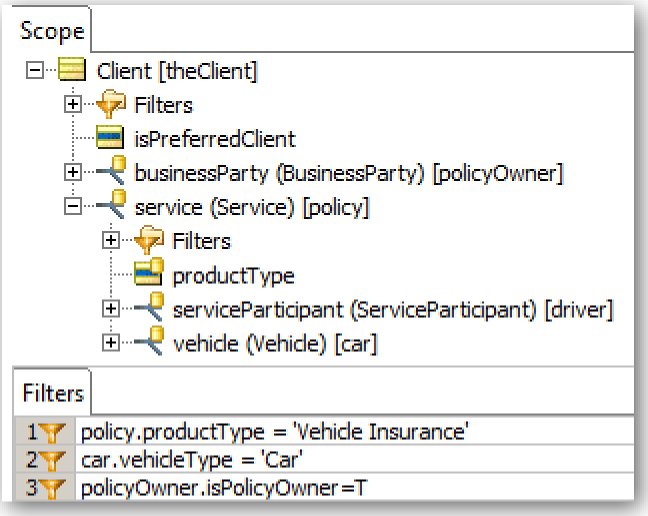


Conditions	0	1	2	3	4
What is the vehicle's potential occupant injury rating?		'Extreme...	'High'	-	-
What is the vehIcle's potential theft rating?		-	-	'High'	-
Is the vehicle garaged in the same state as the owner?		-	-	-	F
What is the driver age class		-	-	-	'Young'
Is the client a preferred		-	-	-	F
Actions					
Post Message(s)		✉	✉	✉	✉
Eligibility Rating is 'Provisional'			✓	✓	✓
Eligibility Rating is 'Eligible'	✓				
Eligibility Rating is 'Not Eligible'		✓			

Note: Setting "Eligible" explicitly is a better practice if you need to know the precise reason for assigning "Eligible". If you don't care about the reason then the default method is acceptable. This pattern is most often seen with validation – you care about the specific reason when the data is invalid. But if it's valid you usually don't need to know the all the reasons why.

Scope, Aliases and Filters

This rule sheet also has an accompanying **SCOPE** and **FILTER** section that defines the local meaning of terms such as "**car**" (which is just one type of **Vehicle**), "**driver**" (a **ServiceParticipant**), "**policy**" (a type of **Service**) and "**policyOwner**" (a type of **BusinessParty**).



```
Scope
⊟ Client [theClient]
   ⊞ Filters
     isPreferredClient
   ⊞ businessParty (BusinessParty) [policyOwner]
   ⊟ service (Service) [policy]
      ⊞ Filters
        productType
      ⊞ serviceParticipant (ServiceParticipant) [driver]
      ⊞ vehicle (Vehicle) [car]
```

Filters

1	policy.productType = 'Vehicle Insurance'
2	car.vehicleType = 'Car'
3	policyOwner.isPolicyOwner=T

We could have used those original terms but defining aliases enables us to make rules that are easier to read.

The filters ensure that we only select the "**BusinessParty**" that is identified as the **policyOwner**. Similarly the alias "**car**" is a **Vehicle** filtered to only those of type=**'car'**

In effect defining a filter is like adding that condition across every rule in the rule sheet.

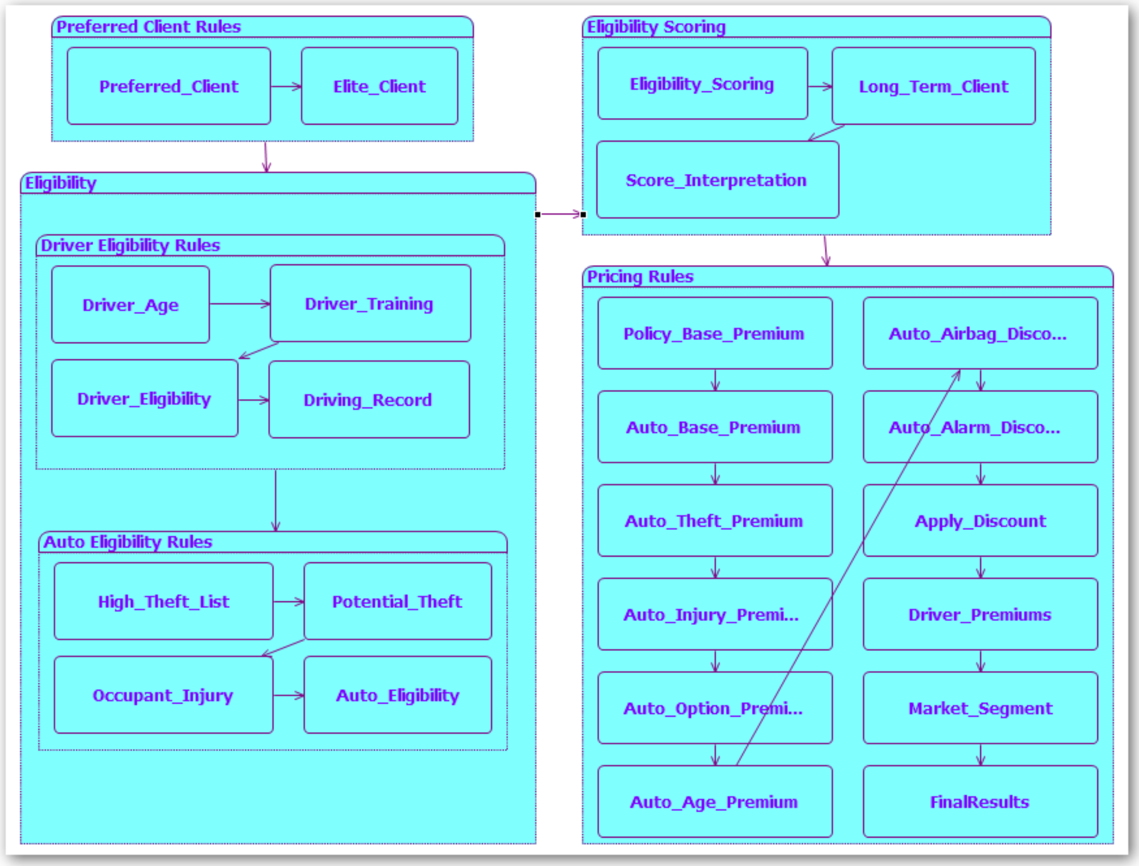
If necessary you can use complex expressions in the filters (such as ->sum, ->max, ->min, ->avg, ->exists, ->forAll, ->substring, ->contains, ->yearsBetween, ->equalsIgnoreCase etc.)

You can also use the filter section to "join" two entities that do not have an explicit association defined (just like you might do in SQL with database tables). Then the rules will process them as a matched set.

The remainder of the rule sheets are described in the Appendix.

The Complete Decision Service

When all the individual rule sheets are connected up using the rule flow diagram we have a Decision Service that can be published and executed.



The Test Scenarios

Let's look at the scenarios (in the interest of space just V1 and V2 are presented.)
Note also that the results presented here are based on the 2006 dates specified in the scenario – not the current date.
If you use 2015 then your results may be different. (The drivers and vehicles will be older).
Corticon supports effective dating so it can be run for any date, past or future.

>> Original Preferred Client Rules (V1)

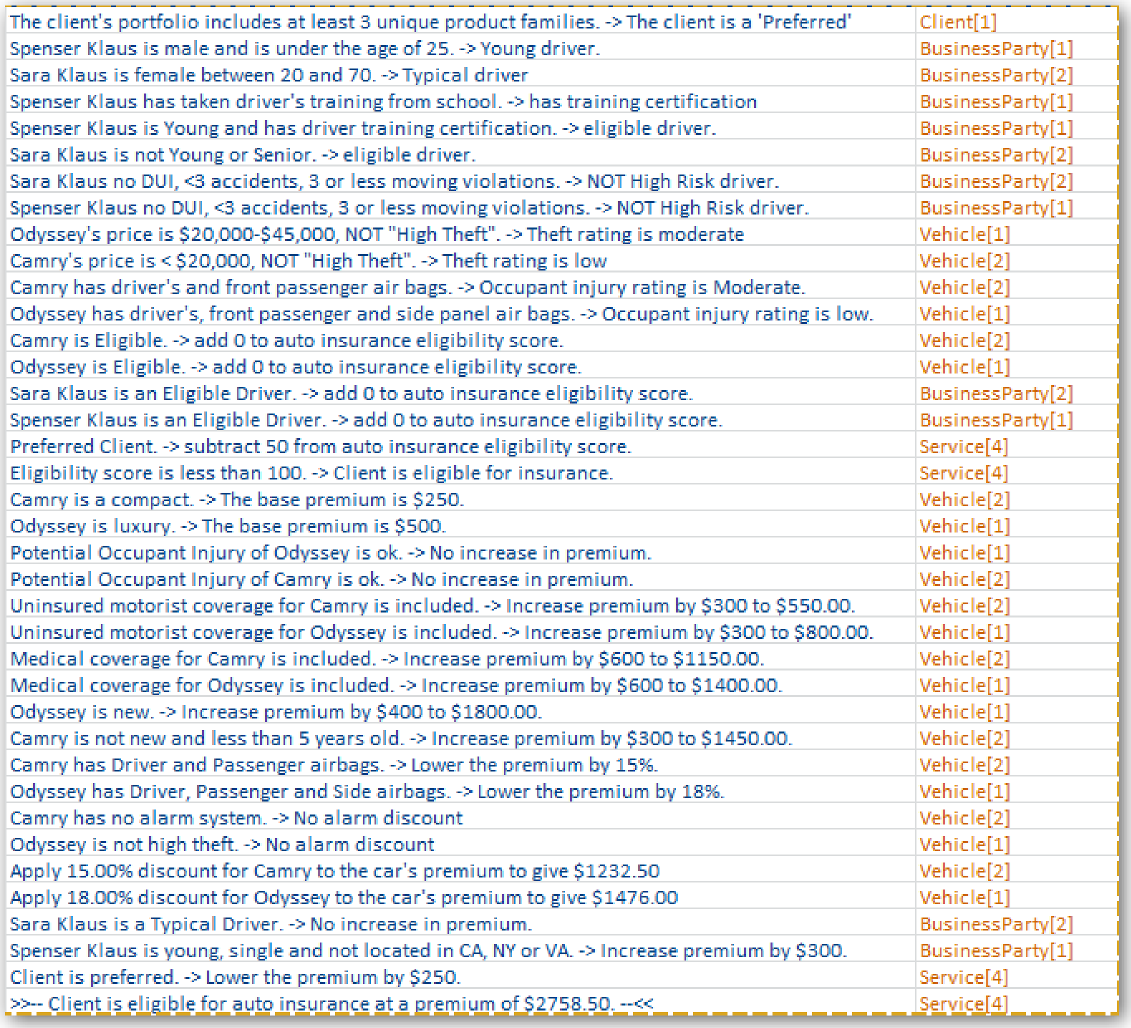
Sara	4/2/2006	Preferred (has at least 3 products) $2758.50

>> Preferred Client Rules Change on 4/1/2006 (V2)

Sara on	7/8/2006	Still Preferred (Grandfathered in): $2758.50
Mark on	7/8/2006	Not Preferred under the new rules so pays more: $3008.50
Mark on	7/8/2006	Angie moves to CA and has 3 accidents: $4878.50

Sara Klaus Original Preferred Client Rules (V1)

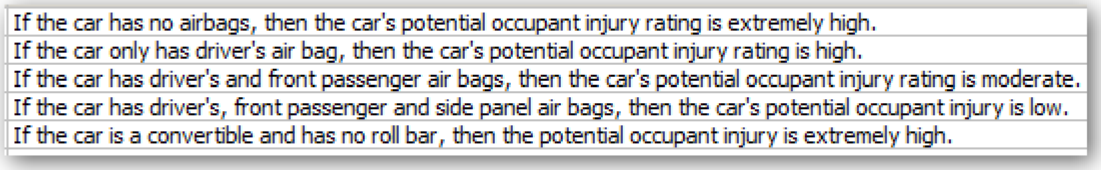
This complete audit trail was generated from the rule statements



The client's portfolio includes at least 3 unique product families. -> The client is a 'Preferred'	Client[1]
Spenser Klaus is male and is under the age of 25. -> Young driver.	BusinessParty[1]
Sara Klaus is female between 20 and 70. -> Typical driver	BusinessParty[2]
Spenser Klaus has taken driver's training from school. -> has training certification	BusinessParty[1]
Spenser Klaus is Young and has driver training certification. -> eligible driver.	BusinessParty[1]
Sara Klaus is not Young or Senior. -> eligible driver.	BusinessParty[2]
Sara Klaus no DUI, <3 accidents, 3 or less moving violations. -> NOT High Risk driver.	BusinessParty[2]
Spenser Klaus no DUI, <3 accidents, 3 or less moving violations. -> NOT High Risk driver.	BusinessParty[1]
Odyssey's price is $20,000-$45,000, NOT "High Theft". -> Theft rating is moderate	Vehicle[1]
Camry's price is < $20,000, NOT "High Theft". -> Theft rating is low	Vehicle[2]
Camry has driver's and front passenger air bags. -> Occupant injury rating is Moderate.	Vehicle[2]
Odyssey has driver's, front passenger and side panel air bags. -> Occupant injury rating is low.	Vehicle[1]
Camry is Eligible. -> add 0 to auto insurance eligibility score.	Vehicle[2]
Odyssey is Eligible. -> add 0 to auto insurance eligibility score.	Vehicle[1]
Sara Klaus is an Eligible Driver. -> add 0 to auto insurance eligibility score.	BusinessParty[2]
Spenser Klaus is an Eligible Driver. -> add 0 to auto insurance eligibility score.	BusinessParty[1]
Preferred Client. -> subtract 50 from auto insurance eligibility score.	Service[4]
Eligibility score is less than 100. -> Client is eligible for insurance.	Service[4]
Camry is a compact. -> The base premium is $250.	Vehicle[2]
Odyssey is luxury. -> The base premium is $500.	Vehicle[1]
Potential Occupant Injury of Odyssey is ok. -> No increase in premium.	Vehicle[1]
Potential Occupant Injury of Camry is ok. -> No increase in premium.	Vehicle[2]
Uninsured motorist coverage for Camry is included. -> Increase premium by $300 to $550.00.	Vehicle[2]
Uninsured motorist coverage for Odyssey is included. -> Increase premium by $300 to $800.00.	Vehicle[1]
Medical coverage for Camry is included. -> Increase premium by $600 to $1150.00.	Vehicle[2]
Medical coverage for Odyssey is included. -> Increase premium by $600 to $1400.00.	Vehicle[1]
Odyssey is new. -> Increase premium by $400 to $1800.00.	Vehicle[1]
Camry is not new and less than 5 years old. -> Increase premium by $300 to $1450.00.	Vehicle[2]
Camry has Driver and Passenger airbags. -> Lower the premium by 15%.	Vehicle[2]
Odyssey has Driver, Passenger and Side airbags. -> Lower the premium by 18%.	Vehicle[1]
Camry has no alarm system. -> No alarm discount	Vehicle[2]
Odyssey is not high theft. -> No alarm discount	Vehicle[1]
Apply 15.00% discount for Camry to the car's premium to give $1232.50	Vehicle[2]
Apply 18.00% discount for Odyssey to the car's premium to give $1476.00	Vehicle[1]
Sara Klaus is a Typical Driver. -> No increase in premium.	BusinessParty[2]
Spenser Klaus is young, single and not located in CA, NY or VA. -> Increase premium by $300.	BusinessParty[1]
Client is preferred. -> Lower the premium by $250.	Service[4]
>>-- Client is eligible for auto insurance at a premium of $2758.50. --<<	Service[4]

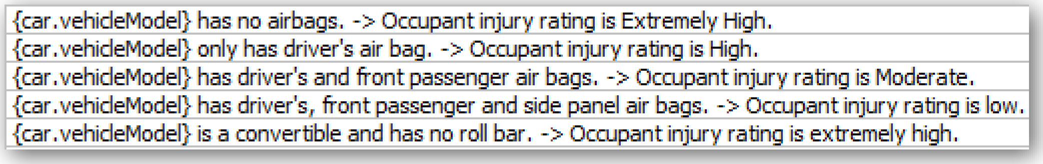
In fact both forms can be maintained in the rule statement section of the rule sheet. E.g:

Original rule statements (IF... THEN format)



If the car has no airbags, then the car's potential occupant injury rating is extremely high.
If the car only has driver's air bag, then the car's potential occupant injury rating is high.
If the car has driver's and front passenger air bags, then the car's potential occupant injury rating is moderate.
If the car has driver's, front passenger and side panel air bags, then the car's potential occupant injury is low.
If the car is a convertible and has no roll bar, then the potential occupant injury is extremely high.

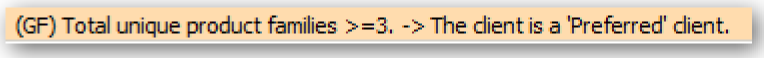
Customized version (FACT...CONCLUSION format)



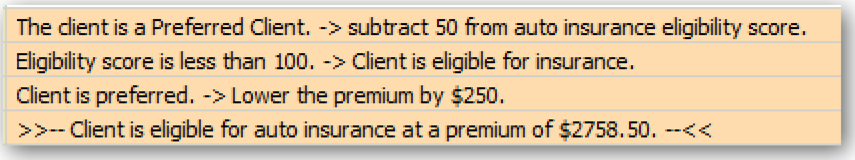
{car.vehicleModel} has no airbags. -> Occupant injury rating is Extremely High.
{car.vehicleModel} only has driver's air bag. -> Occupant injury rating is High.
{car.vehicleModel} has driver's and front passenger air bags. -> Occupant injury rating is Moderate.
{car.vehicleModel} has driver's, front passenger and side panel air bags. -> Occupant injury rating is low.
{car.vehicleModel} is a convertible and has no roll bar. -> Occupant injury rating is extremely high.

Sara Klaus V2 (with the grandfathered rule for preferred client)

When Sara is processed after the new Preferred Client rules are implemented we see the Grandfathered rule executed. Everything else is the same.



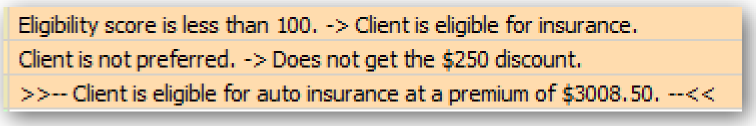
```
(GF) Total unique product families >=3. -> The client is a 'Preferred' client.
```



```
The client is a Preferred Client. -> subtract 50 from auto insurance eligibility score.
Eligibility score is less than 100. -> Client is eligible for insurance.
Client is preferred. -> Lower the premium by $250.
>>-- Client is eligible for auto insurance at a premium of $2758.50. --<<
```

Mark Houston (with the V2 rules)

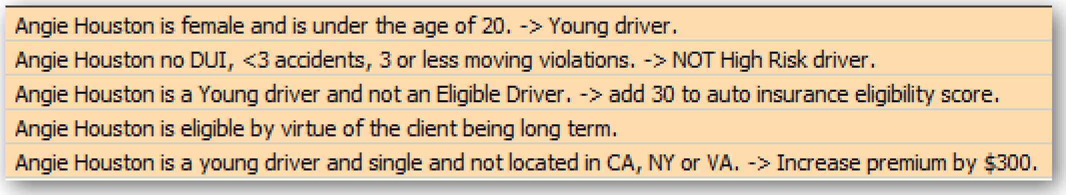
Mark applies for insurance after the new Preferred Client rules go into effect. So even though he has more than three products he still doesn't qualify for Preferred because his average number of products is less than 3. This means his premium is $250 higher than Sara's. Everything else is the same



```
Eligibility score is less than 100. -> Client is eligible for insurance.
Client is not preferred. -> Does not get the $250 discount.
>>-- Client is eligible for auto insurance at a premium of $3008.50. --<<
```

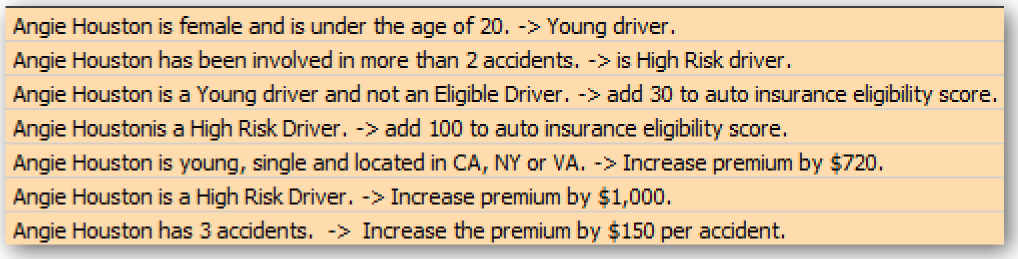
Mark Houston (Angie's Accidents with V2)

Before moving to CA this is how the rule model rates Angie



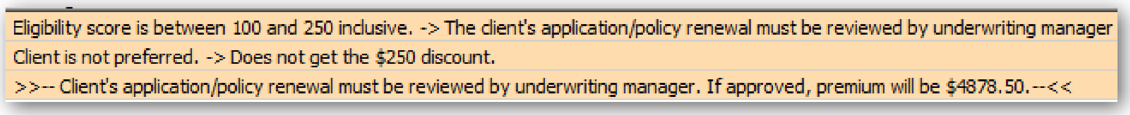
```
Angie Houston is female and is under the age of 20. -> Young driver.
Angie Houston no DUI, <3 accidents, 3 or less moving violations. -> NOT High Risk driver.
Angie Houston is a Young driver and not an Eligible Driver. -> add 30 to auto insurance eligibility score.
Angie Houston is eligible by virtue of the client being long term.
Angie Houston is a young driver and single and not located in CA, NY or VA. -> Increase premium by $300.
```

When she moves to CA and has three accidents Angie gets rated like this:



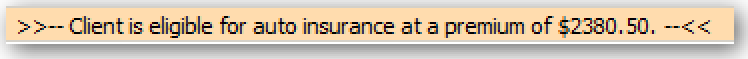
```
Angie Houston is female and is under the age of 20. -> Young driver.
Angie Houston has been involved in more than 2 accidents. -> is High Risk driver.
Angie Houston is a Young driver and not an Eligible Driver. -> add 30 to auto insurance eligibility score.
Angie Houstonis a High Risk Driver. -> add 100 to auto insurance eligibility score.
Angie Houston is young, single and located in CA, NY or VA. -> Increase premium by $720.
Angie Houston is a High Risk Driver. -> Increase premium by $1,000.
Angie Houston has 3 accidents.  -> Increase the premium by $150 per accident.
```

This affects both the eligibility score and the premium.
And the policy gets these messages:



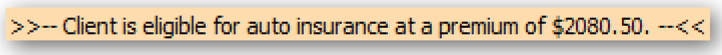
```
Eligibility score is between 100 and 250 inclusive. -> The client's application/policy renewal must be reviewed by underwriting manager
Client is not preferred. -> Does not get the $250 discount.
>>-- Client's application/policy renewal must be reviewed by underwriting manager. If approved, premium will be $4878.50.--<<
```

Incidentally if we run Sara in 2014 the cars will be older (but keep Spenser at 17) we get



```
>>-- Client is eligible for auto insurance at a premium of $2380.50. --<<
```

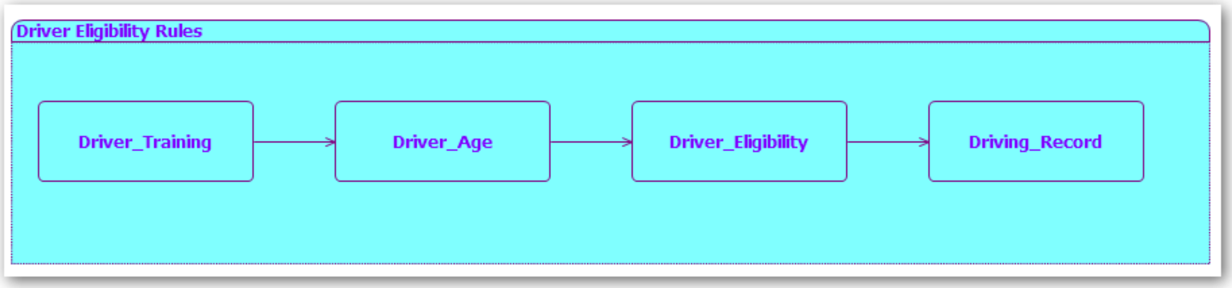
Adjusting ages (Spenser is really 27 and no longer Young) gives



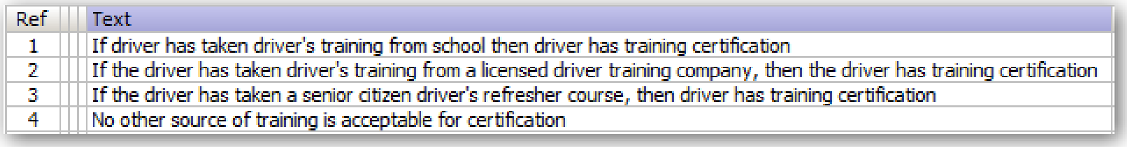
```
>>-- Client is eligible for auto insurance at a premium of $2080.50. --<<
```

Appendix – Additional Rule Sheets

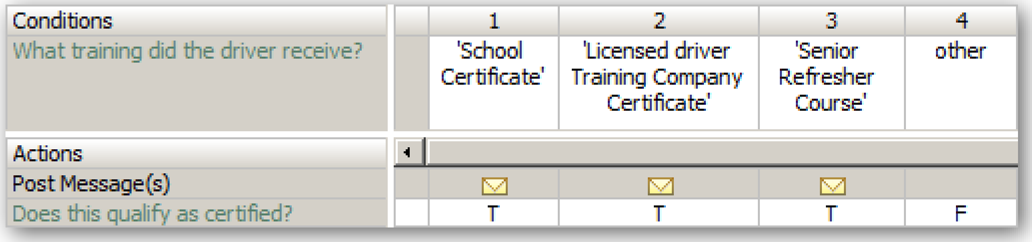
Driver Eligibility Rules



Driver Training

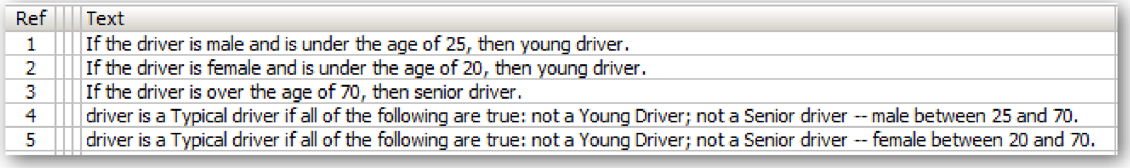


Ref	Text
1	If driver has taken driver's training from school then driver has training certification
2	If the driver has taken driver's training from a licensed driver training company, then the driver has training certification
3	If the driver has taken a senior citizen driver's refresher course, then driver has training certification
4	No other source of training is acceptable for certification

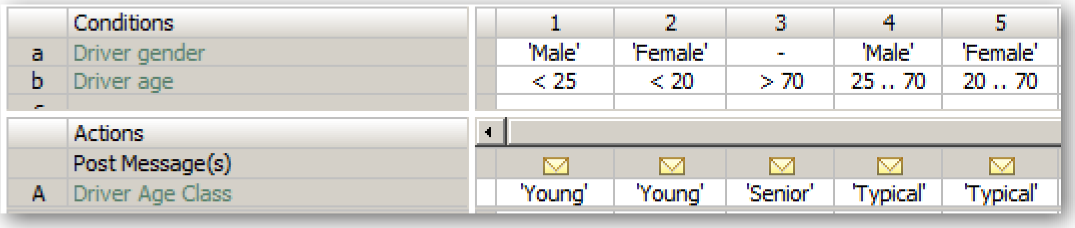


Conditions	1	2	3	4
What training did the driver receive?	'School Certificate'	'Licensed driver Training Company Certificate'	'Senior Refresher Course'	other
Actions				
Post Message(s)	✉	✉	✉	
Does this qualify as certified?	T	T	T	F

Driver Age



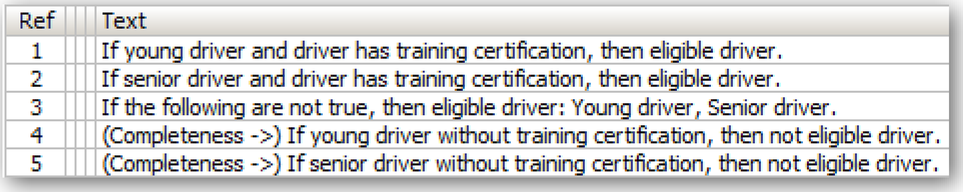
Ref	Text
1	If the driver is male and is under the age of 25, then young driver.
2	If the driver is female and is under the age of 20, then young driver.
3	If the driver is over the age of 70, then senior driver.
4	driver is a Typical driver if all of the following are true: not a Young Driver; not a Senior driver -- male between 25 and 70.
5	driver is a Typical driver if all of the following are true: not a Young Driver; not a Senior driver -- female between 20 and 70.



	Conditions	1	2	3	4	5
a	Driver gender	'Male'	'Female'	-	'Male'	'Female'
b	Driver age	< 25	< 20	> 70	25 .. 70	20 .. 70
	Actions					
	Post Message(s)	✉	✉	✉	✉	✉
A	Driver Age Class	'Young'	'Young'	'Senior'	'Typical'	'Typical'

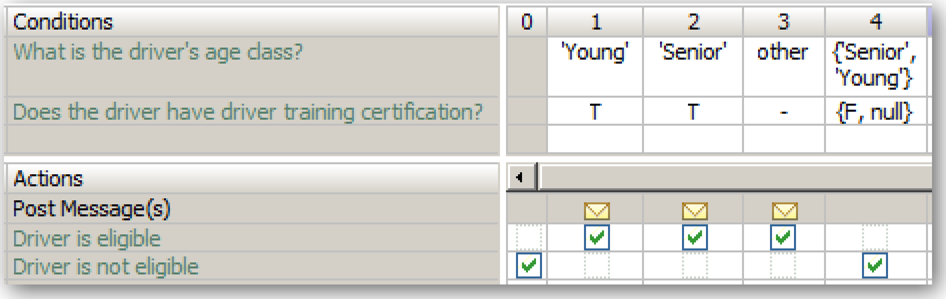
This rule sheet also introduces the idea of a “Typical” driver, one that is neither Young nor Senior.

Driver Eligibility



Ref	Text
1	If young driver and driver has training certification, then eligible driver.
2	If senior driver and driver has training certification, then eligible driver.
3	If the following are not true, then eligible driver: Young driver, Senior driver.
4	(Completeness ->) If young driver without training certification, then not eligible driver.
5	(Completeness ->) If senior driver without training certification, then not eligible driver.

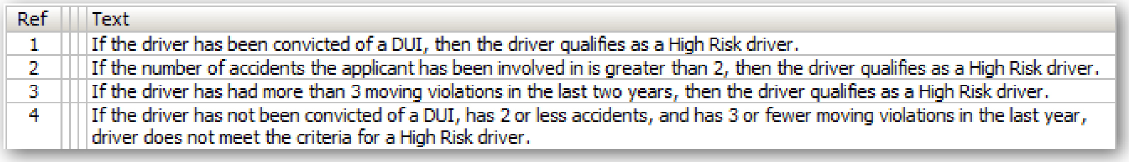
4 and 5 can be merged automatically using the Corticon rule compression feature



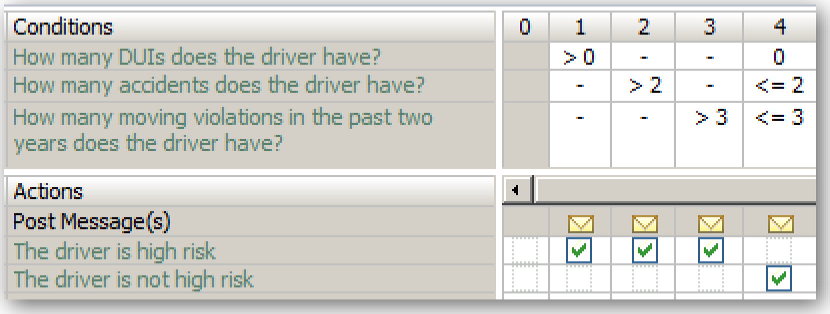
Conditions	0	1	2	3	4
What is the driver's age class?		'Young'	'Senior'	other	{'Senior', 'Young'}
Does the driver have driver training certification?		T	T	-	{F, null}
Actions					
Post Message(s)		✉	✉	✉	
Driver is eligible		✓	✓	✓	
Driver is not eligible	✓				✓

Null is used in case the input data for training certification is missing.

Driver Record

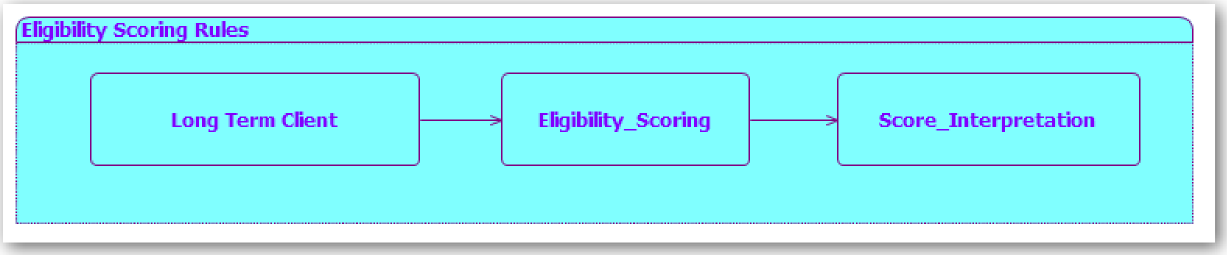


Ref	Text
1	If the driver has been convicted of a DUI, then the driver qualifies as a High Risk driver.
2	If the number of accidents the applicant has been involved in is greater than 2, then the driver qualifies as a High Risk driver.
3	If the driver has had more than 3 moving violations in the last two years, then the driver qualifies as a High Risk driver.
4	If the driver has not been convicted of a DUI, has 2 or less accidents, and has 3 or fewer moving violations in the last year, driver does not meet the criteria for a High Risk driver.

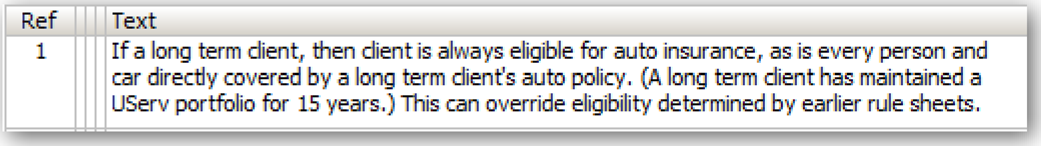


Conditions	0	1	2	3	4
How many DUIs does the driver have?		> 0	-	-	0
How many accidents does the driver have?		-	> 2	-	<= 2
How many moving violations in the past two years does the driver have?		-	-	> 3	<= 3
Actions					
Post Message(s)		✉	✉	✉	✉
The driver is high risk		✓	✓	✓	
The driver is not high risk					✓

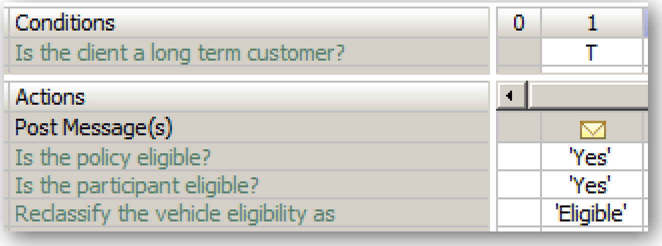
Eligibility Scoring



Long Term Client Scoring



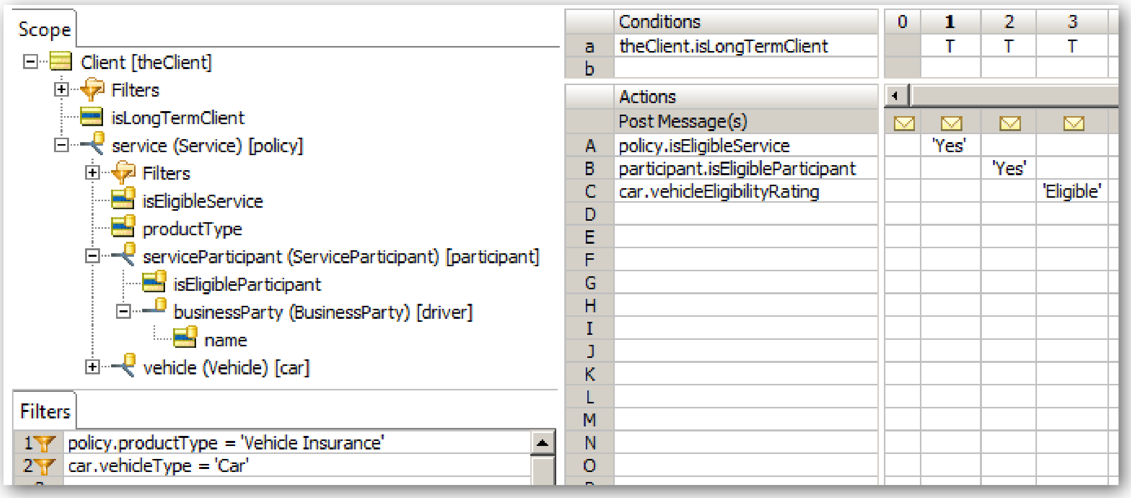
Ref	Text
1	If a long term client, then client is always eligible for auto insurance, as is every person and car directly covered by a long term client's auto policy. (A long term client has maintained a UServ portfolio for 15 years.) This can override eligibility determined by earlier rule sheets.



Conditions	0	1
Is the client a long term customer?		T

Actions		
Post Message(s)		✉
Is the policy eligible?		'Yes'
Is the participant eligible?		'Yes'
Reclassify the vehicle eligibility as		'Eligible'

How it's Implemented



```
Scope
⊟ Client [theClient]
    ⊞ Filters
      isLongTermClient
    ⊟ service (Service) [policy]
        ⊞ Filters
          isEligibleService
          productType
        ⊟ serviceParticipant (ServiceParticipant) [participant]
              isEligibleParticipant
            ⊟ businessParty (BusinessParty) [driver]
                  name
        ⊞ vehicle (Vehicle) [car]

Filters
1  policy.productType = 'Vehicle Insurance'
2  car.vehicleType = 'Car'
```

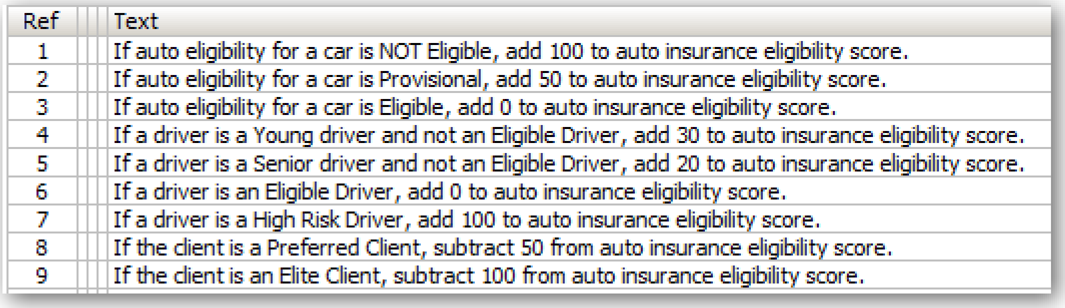
	Conditions	0	1	2	3
a	theClient.isLongTermClient		T	T	T
b					

	Actions				
	Post Message(s)	✉	✉	✉	✉
A	policy.isEligibleService		'Yes'		
B	participant.isEligibleParticipant			'Yes'	
C	car.vehicleEligibilityRating				'Eligible'
D					
E					
F					
G					
H					
I					
J					
K					
L					
M					
N					
O					

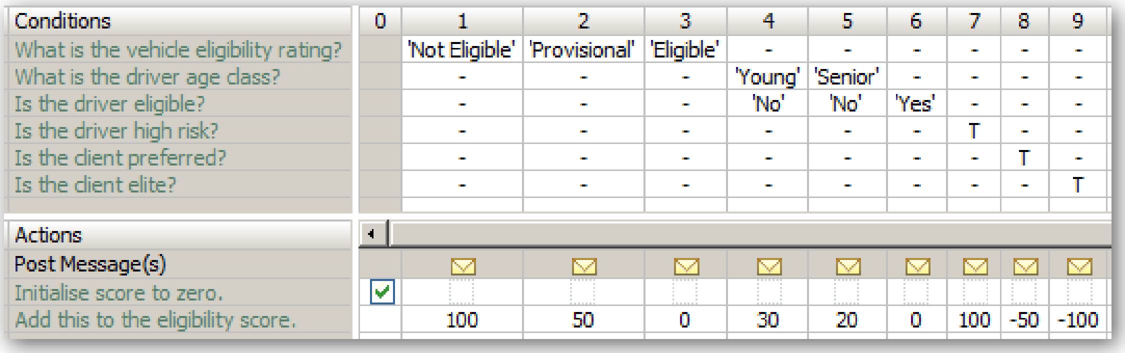
How it Works

Clients can have one or more **policy** (an alias for Service). A **policy** can have one or more **participant** and one or more **car**. Unlike programming languages which would require the programmer to create various loops to iterate over the repeating objects, Corticon is smart enough to know how to interpret the actions of the rules. It knows to apply the actions to ALL policies for this client. And for every policy it applies the actions to ALL the participants and vehicles. Furthermore, Corticon knows (from the filter expression) that we are only interested in the policies that are for Vehicle Insurance and only vehicles that are cars. So those three very simple actions could end up doing a lot of work if there are many policies with many drivers and vehicles. This might be the case if the Client was a taxi company.

Eligibility Scoring

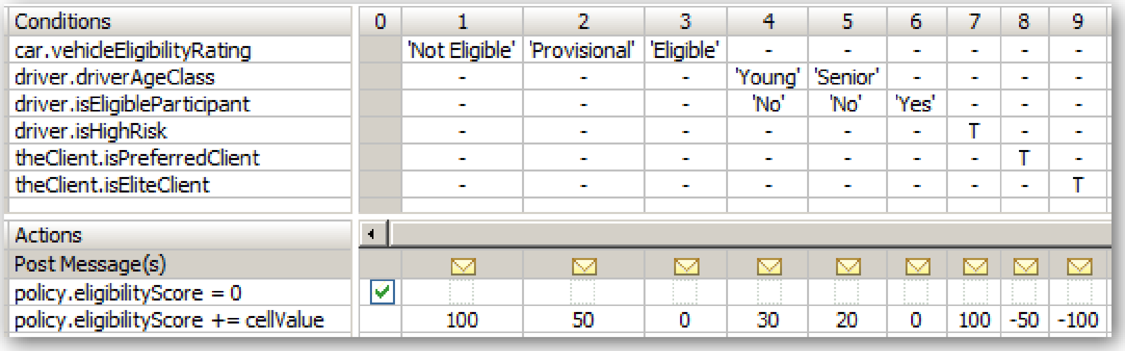


Ref	Text
1	If auto eligibility for a car is NOT Eligible, add 100 to auto insurance eligibility score.
2	If auto eligibility for a car is Provisional, add 50 to auto insurance eligibility score.
3	If auto eligibility for a car is Eligible, add 0 to auto insurance eligibility score.
4	If a driver is a Young driver and not an Eligible Driver, add 30 to auto insurance eligibility score.
5	If a driver is a Senior driver and not an Eligible Driver, add 20 to auto insurance eligibility score.
6	If a driver is an Eligible Driver, add 0 to auto insurance eligibility score.
7	If a driver is a High Risk Driver, add 100 to auto insurance eligibility score.
8	If the client is a Preferred Client, subtract 50 from auto insurance eligibility score.
9	If the client is an Elite Client, subtract 100 from auto insurance eligibility score.



Conditions	0	1	2	3	4	5	6	7	8	9
What is the vehicle eligibility rating?		'Not Eligible'	'Provisional'	'Eligible'	-	-	-	-	-	-
What is the driver age class?		-	-	-	'Young'	'Senior'	-	-	-	-
Is the driver eligible?		-	-	-	'No'	'No'	'Yes'	-	-	-
Is the driver high risk?		-	-	-	-	-	-	T	-	-
Is the client preferred?		-	-	-	-	-	-	-	T	-
Is the client elite?		-	-	-	-	-	-	-	-	T
Actions										
Post Message(s)		✉	✉	✉	✉	✉	✉	✉	✉	✉
Initialise score to zero.	✓									
Add this to the eligibility score.		100	50	0	30	20	0	100	-50	-100

Implementation



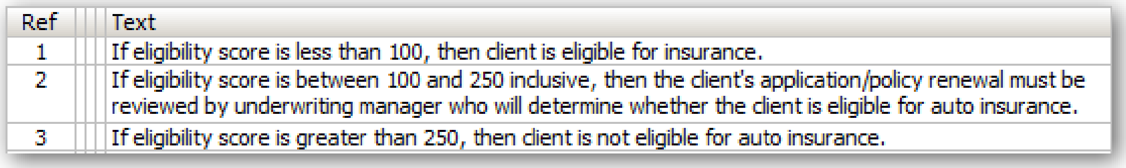
Conditions	0	1	2	3	4	5	6	7	8	9
car.vehicleEligibilityRating		'Not Eligible'	'Provisional'	'Eligible'	-	-	-	-	-	-
driver.driverAgeClass		-	-	-	'Young'	'Senior'	-	-	-	-
driver.isEligibleParticipant		-	-	-	'No'	'No'	'Yes'	-	-	-
driver.isHighRisk		-	-	-	-	-	-	T	-	-
theClient.isPreferredClient		-	-	-	-	-	-	-	T	-
theClient.isEliteClient		-	-	-	-	-	-	-	-	T
Actions										
Post Message(s)		✉	✉	✉	✉	✉	✉	✉	✉	✉
policy.eligibilityScore = 0	✓									
policy.eligibilityScore += cellValue		100	50	0	30	20	0	100	-50	-100

The keyword **cellValue** allows a value from the rule column to be used in the action.

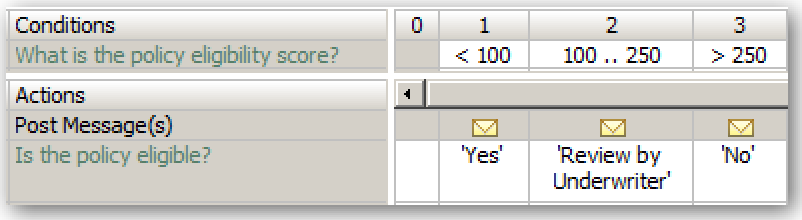
Observations/Thoughts

This sheet combines a number of unrelated attributes. E.g. vehicle eligibility score is independent of driver eligibility score and client score (any or all may get added). Should these be split on to separate sheets? i.e. rules 1,2,3 are mutually exclusive and 4,5,6 are also mutually exclusive. Should the eligibility scores be kept separate (for vehicle, and driver and client) and totaled at the end? If we did this we'd have visibility of the contribution of each factor and this could be displayed in the bill (or audit report).

Scoring Interpretation

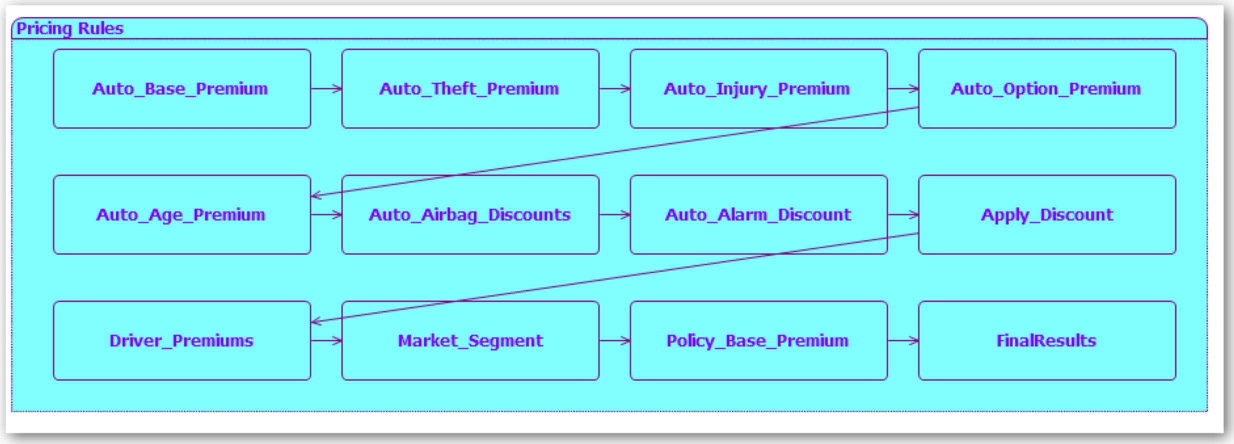


Ref	Text
1	If eligibility score is less than 100, then client is eligible for insurance.
2	If eligibility score is between 100 and 250 inclusive, then the client's application/policy renewal must be reviewed by underwriting manager who will determine whether the client is eligible for auto insurance.
3	If eligibility score is greater than 250, then client is not eligible for auto insurance.

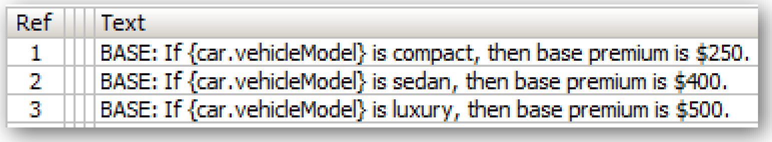


Conditions	0	1	2	3
What is the policy eligibility score?		< 100	100 .. 250	> 250
Actions				
Post Message(s)		✉	✉	✉
Is the policy eligible?		'Yes'	'Review by Underwriter'	'No'

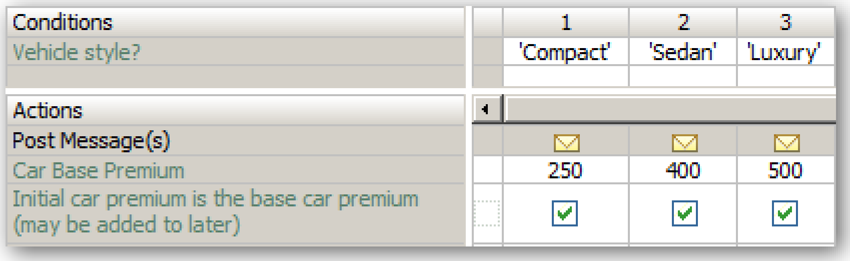
Pricing Business Rules



Auto Base Premium

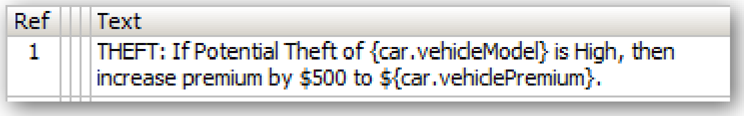


Ref	Text
1	BASE: If {car.vehicleModel} is compact, then base premium is $250.
2	BASE: If {car.vehicleModel} is sedan, then base premium is $400.
3	BASE: If {car.vehicleModel} is luxury, then base premium is $500.

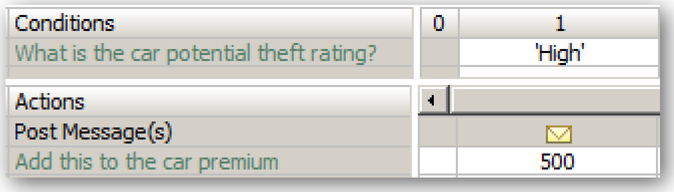


Conditions		1	2	3
Vehicle style?		'Compact'	'Sedan'	'Luxury'
Actions				
Post Message(s)		✉	✉	✉
Car Base Premium		250	400	500
Initial car premium is the base car premium (may be added to later)		✓	✓	✓

Auto Theft Premium

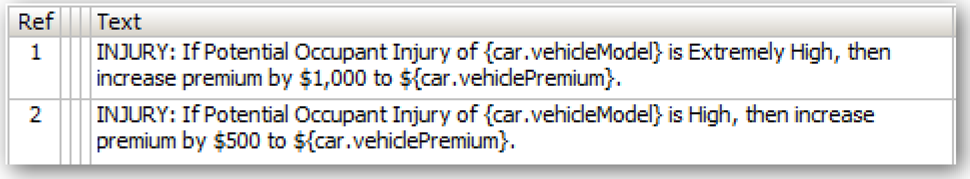


Ref	Text
1	THEFT: If Potential Theft of {car.vehicleModel} is High, then increase premium by $500 to ${car.vehiclePremium}.

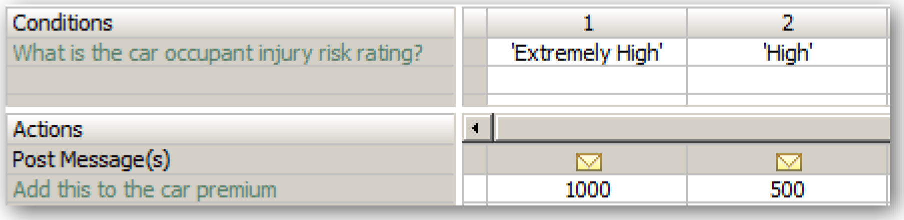


Conditions	0	1
What is the car potential theft rating?		'High'
Actions		
Post Message(s)		✉
Add this to the car premium		500

Auto Injury Premium

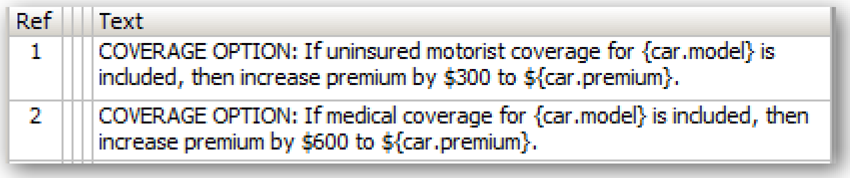


Ref	Text
1	INJURY: If Potential Occupant Injury of {car.vehicleModel} is Extremely High, then increase premium by $1,000 to ${car.vehiclePremium}.
2	INJURY: If Potential Occupant Injury of {car.vehicleModel} is High, then increase premium by $500 to ${car.vehiclePremium}.

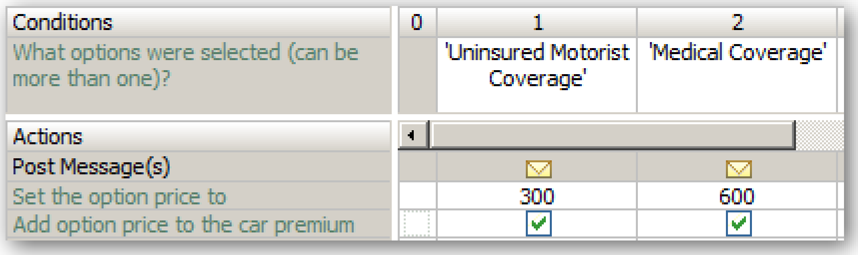


Conditions		1	2
What is the car occupant injury risk rating?		'Extremely High'	'High'
Actions			
Post Message(s)		✉	✉
Add this to the car premium		1000	500

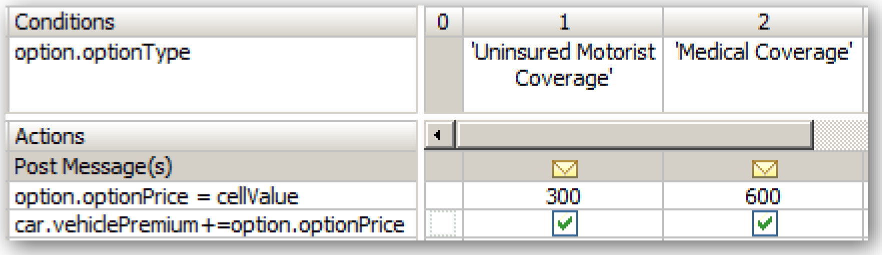
Auto Option Premium



Ref	Text
1	COVERAGE OPTION: If uninsured motorist coverage for {car.model} is included, then increase premium by $300 to ${car.premium}.
2	COVERAGE OPTION: If medical coverage for {car.model} is included, then increase premium by $600 to ${car.premium}.

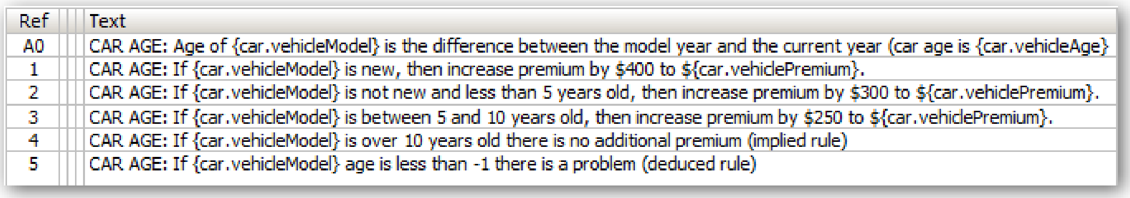


Conditions	0	1	2
What options were selected (can be more than one)?		'Uninsured Motorist Coverage'	'Medical Coverage'
Actions			
Post Message(s)		✉	✉
Set the option price to		300	600
Add option price to the car premium	☐	✓	✓

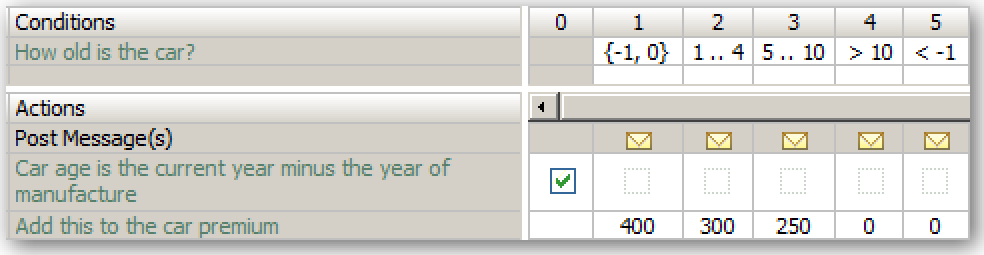


Conditions	0	1	2
option.optionType		'Uninsured Motorist Coverage'	'Medical Coverage'
Actions			
Post Message(s)		✉	✉
option.optionPrice = cellValue		300	600
car.vehiclePremium+=option.optionPrice	☐	✓	✓

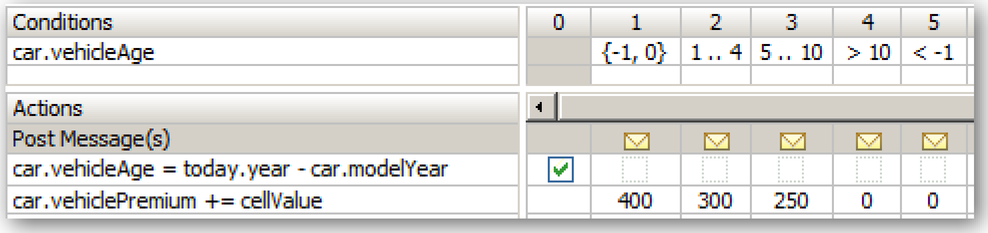
Auto Age Premium



Ref	Text
A0	CAR AGE: Age of {car.vehicleModel} is the difference between the model year and the current year (car age is {car.vehicleAge}
1	CAR AGE: If {car.vehicleModel} is new, then increase premium by $400 to ${car.vehiclePremium}.
2	CAR AGE: If {car.vehicleModel} is not new and less than 5 years old, then increase premium by $300 to ${car.vehiclePremium}.
3	CAR AGE: If {car.vehicleModel} is between 5 and 10 years old, then increase premium by $250 to ${car.vehiclePremium}.
4	CAR AGE: If {car.vehicleModel} is over 10 years old there is no additional premium (implied rule)
5	CAR AGE: If {car.vehicleModel} age is less than -1 there is a problem (deduced rule)

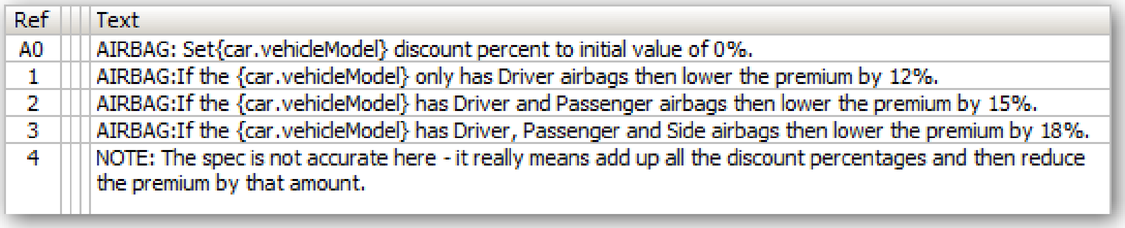


Conditions	0	1	2	3	4	5
How old is the car?		{-1, 0}	1 .. 4	5 .. 10	> 10	< -1
Actions						
Post Message(s)		✉	✉	✉	✉	✉
Car age is the current year minus the year of manufacture	✓	☐	☐	☐	☐	☐
Add this to the car premium		400	300	250	0	0

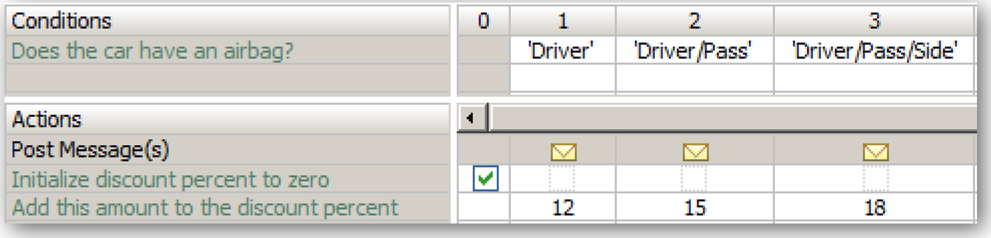


Conditions	0	1	2	3	4	5
car.vehicleAge		{-1, 0}	1 .. 4	5 .. 10	> 10	< -1
Actions						
Post Message(s)		✉	✉	✉	✉	✉
car.vehicleAge = today.year - car.modelYear	✓	☐	☐	☐	☐	☐
car.vehiclePremium += cellValue		400	300	250	0	0

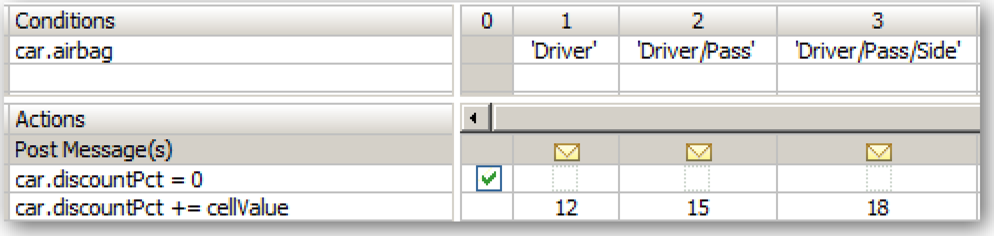
Auto Airbag Discount



Ref	Text
A0	AIRBAG: Set{car.vehicleModel} discount percent to initial value of 0%.
1	AIRBAG:If the {car.vehicleModel} only has Driver airbags then lower the premium by 12%.
2	AIRBAG:If the {car.vehicleModel} has Driver and Passenger airbags then lower the premium by 15%.
3	AIRBAG:If the {car.vehicleModel} has Driver, Passenger and Side airbags then lower the premium by 18%.
4	NOTE: The spec is not accurate here - it really means add up all the discount percentages and then reduce the premium by that amount.

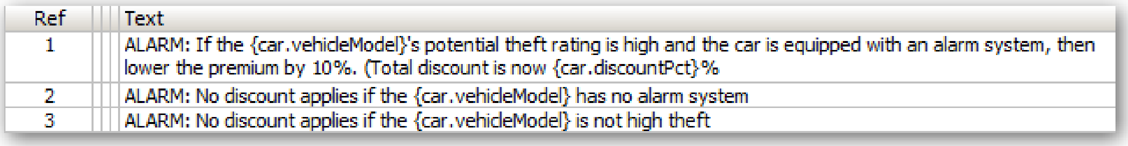


Conditions	0	1	2	3
Does the car have an airbag?		'Driver'	'Driver/Pass'	'Driver/Pass/Side'
Actions				
Post Message(s)		✉	✉	✉
Initialize discount percent to zero	✓	☐	☐	☐
Add this amount to the discount percent		12	15	18

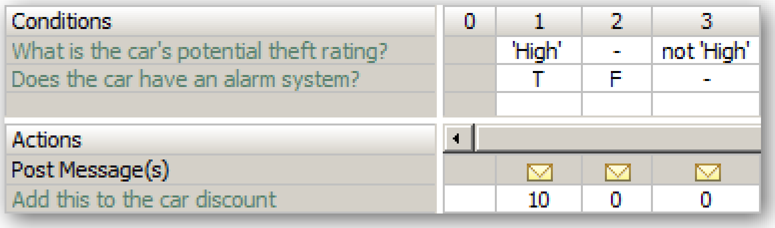


Conditions	0	1	2	3
car.airbag		'Driver'	'Driver/Pass'	'Driver/Pass/Side'
Actions				
Post Message(s)		✉	✉	✉
car.discountPct = 0	✓	☐	☐	☐
car.discountPct += cellValue		12	15	18

Auto Alarm Discount



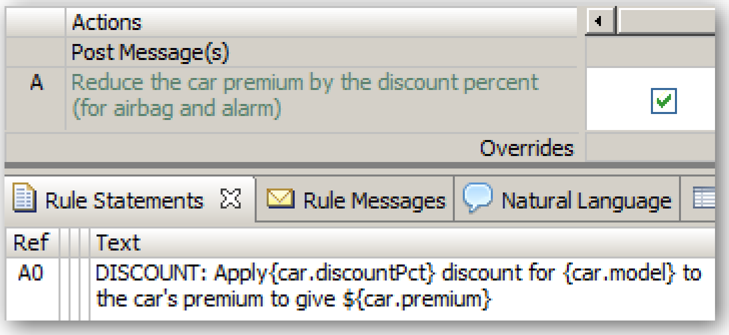
Ref	Text
1	ALARM: If the {car.vehicleModel}'s potential theft rating is high and the car is equipped with an alarm system, then lower the premium by 10%. (Total discount is now {car.discountPct}%
2	ALARM: No discount applies if the {car.vehicleModel} has no alarm system
3	ALARM: No discount applies if the {car.vehicleModel} is not high theft



Conditions	0	1	2	3
What is the car's potential theft rating?		'High'	-	not 'High'
Does the car have an alarm system?		T	F	-
Actions				
Post Message(s)		✉	✉	✉
Add this to the car discount		10	0	0

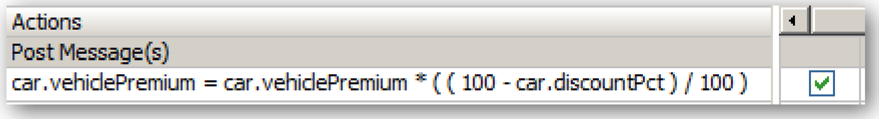
Apply Discount

The action is in column zero since there are no conditions



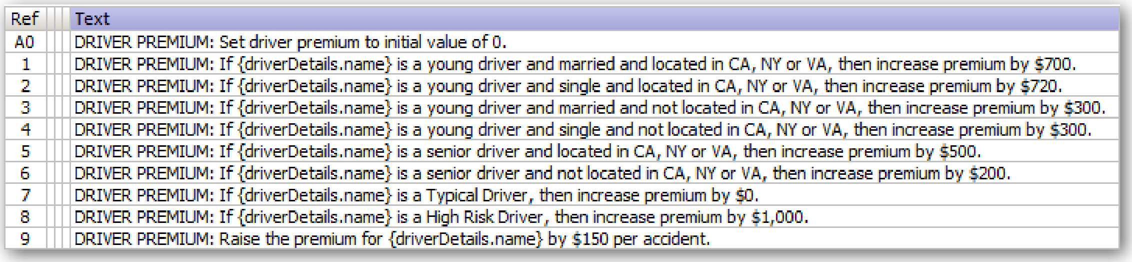
	Actions	
	Post Message(s)	
A	Reduce the car premium by the discount percent (for airbag and alarm)	✓
	Overrides	

Ref	Text
A0	DISCOUNT: Apply{car.discountPct} discount for {car.model} to the car's premium to give ${car.premium}

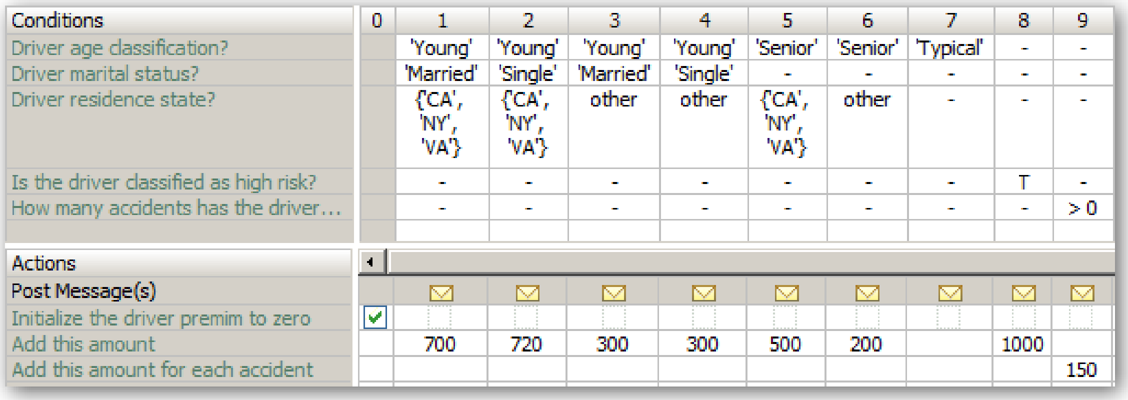


Actions	
Post Message(s)	
car.vehiclePremium = car.vehiclePremium * ((100 - car.discountPct) / 100)	✓

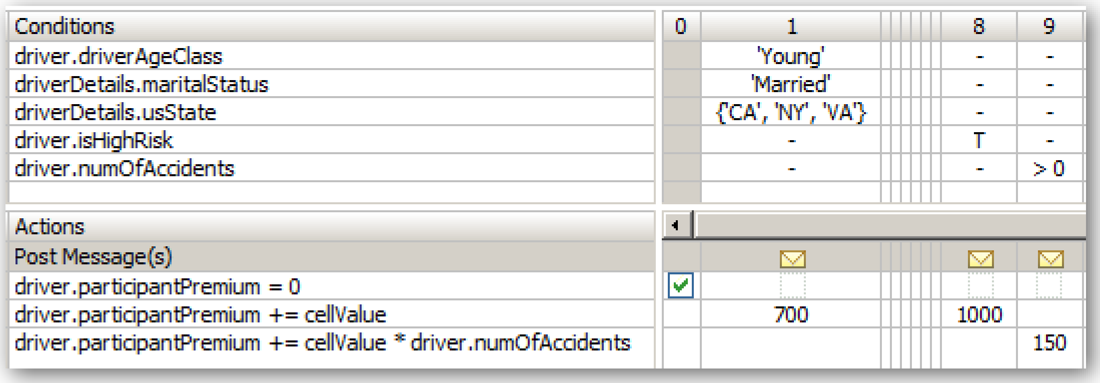
Driver Premiums



Ref	Text
A0	DRIVER PREMIUM: Set driver premium to initial value of 0.
1	DRIVER PREMIUM: If {driverDetails.name} is a young driver and married and located in CA, NY or VA, then increase premium by $700.
2	DRIVER PREMIUM: If {driverDetails.name} is a young driver and single and located in CA, NY or VA, then increase premium by $720.
3	DRIVER PREMIUM: If {driverDetails.name} is a young driver and married and not located in CA, NY or VA, then increase premium by $300.
4	DRIVER PREMIUM: If {driverDetails.name} is a young driver and single and not located in CA, NY or VA, then increase premium by $300.
5	DRIVER PREMIUM: If {driverDetails.name} is a senior driver and located in CA, NY or VA, then increase premium by $500.
6	DRIVER PREMIUM: If {driverDetails.name} is a senior driver and not located in CA, NY or VA, then increase premium by $200.
7	DRIVER PREMIUM: If {driverDetails.name} is a Typical Driver, then increase premium by $0.
8	DRIVER PREMIUM: If {driverDetails.name} is a High Risk Driver, then increase premium by $1,000.
9	DRIVER PREMIUM: Raise the premium for {driverDetails.name} by $150 per accident.

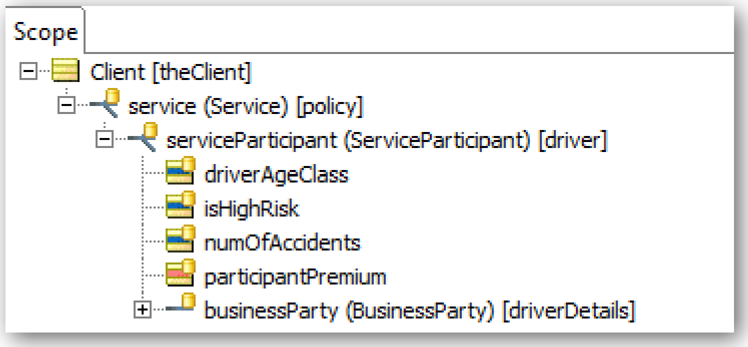


Conditions	0	1	2	3	4	5	6	7	8	9
Driver age classification?		'Young'	'Young'	'Young'	'Young'	'Senior'	'Senior'	'Typical'	-	-
Driver marital status?		'Married'	'Single'	'Married'	'Single'	-	-	-	-	-
Driver residence state?		{'CA', 'NY', 'VA'}	{'CA', 'NY', 'VA'}	other	other	{'CA', 'NY', 'VA'}	other	-	-	-
Is the driver classified as high risk?		-	-	-	-	-	-	-	T	-
How many accidents has the driver…		-	-	-	-	-	-	-	-	> 0
Actions										
Post Message(s)		✉	✉	✉	✉	✉	✉	✉	✉	✉
Initialize the driver premim to zero	✓									
Add this amount		700	720	300	300	500	200		1000	
Add this amount for each accident										150



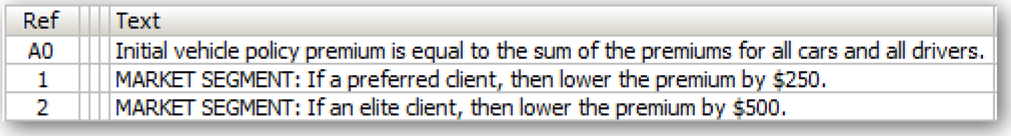
Conditions	0	1	8	9
driver.driverAgeClass		'Young'	-	-
driverDetails.maritalStatus		'Married'	-	-
driverDetails.usState		{'CA', 'NY', 'VA'}	-	-
driver.isHighRisk		-	T	-
driver.numOfAccidents		-	-	> 0
Actions				
Post Message(s)		✉	✉	✉
driver.participantPremium = 0	✓			
driver.participantPremium += cellValue		700	1000	
driver.participantPremium += cellValue * driver.numOfAccidents				150

Using this Scope:

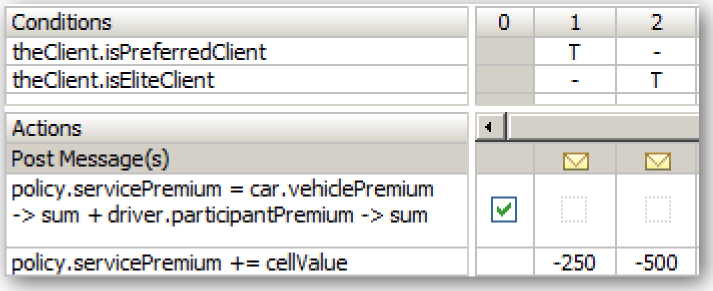


```
Scope
⊟ Client [theClient]
   ⊟ service (Service) [policy]
      ⊟ serviceParticipant (ServiceParticipant) [driver]
           driverAgeClass
           isHighRisk
           numOfAccidents
           participantPremium
         ⊞ businessParty (BusinessParty) [driverDetails]
```

Market Segment



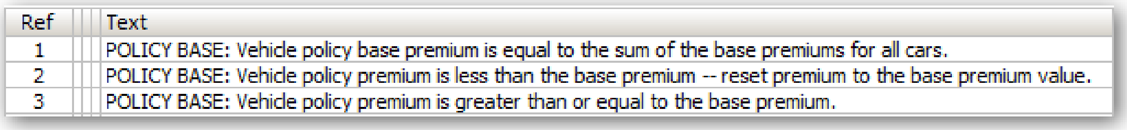
Ref	Text
A0	Initial vehicle policy premium is equal to the sum of the premiums for all cars and all drivers.
1	MARKET SEGMENT: If a preferred client, then lower the premium by $250.
2	MARKET SEGMENT: If an elite client, then lower the premium by $500.



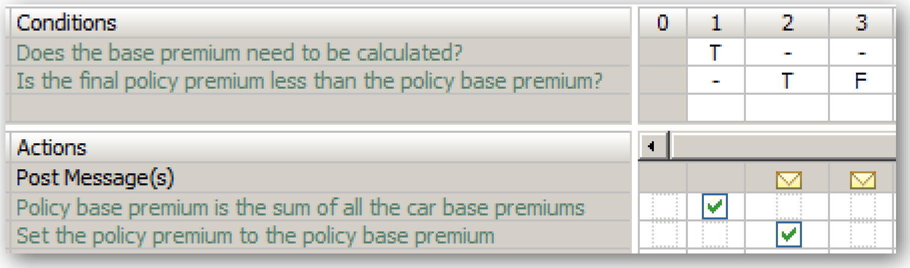
Conditions	0	1	2
theClient.isPreferredClient		T	-
theClient.isEliteClient		-	T
Actions			
Post Message(s)		✉	✉
policy.servicePremium = car.vehiclePremium -> sum + driver.participantPremium -> sum	✓		
policy.servicePremium += cellValue		-250	-500

Column zero is a non-conditional rule and gets executed first to calculate the sum of the vehicle premiums and driver premiums. Then rules 1 and 2 make deductions from that total premium.

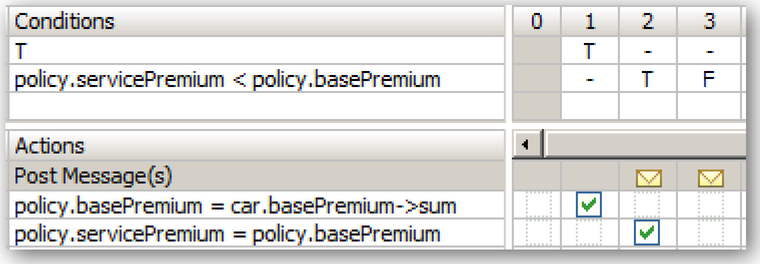
Policy Base Premium



Ref	Text
1	POLICY BASE: Vehicle policy base premium is equal to the sum of the base premiums for all cars.
2	POLICY BASE: Vehicle policy premium is less than the base premium -- reset premium to the base premium value.
3	POLICY BASE: Vehicle policy premium is greater than or equal to the base premium.



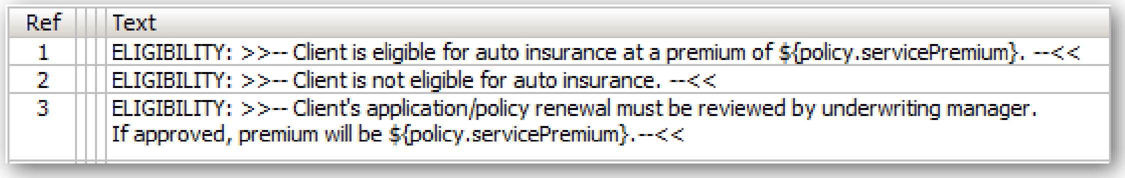
Conditions	0	1	2	3
Does the base premium need to be calculated?		T	-	-
Is the final policy premium less than the policy base premium?		-	T	F
Actions				
Post Message(s)			✉	✉
Policy base premium is the sum of all the car base premiums		✓		
Set the policy premium to the policy base premium			✓	



Conditions	0	1	2	3
T		T	-	-
policy.servicePremium < policy.basePremium		-	T	F
Actions				
Post Message(s)			✉	✉
policy.basePremium = car.basePremium->sum		✓		
policy.servicePremium = policy.basePremium			✓	

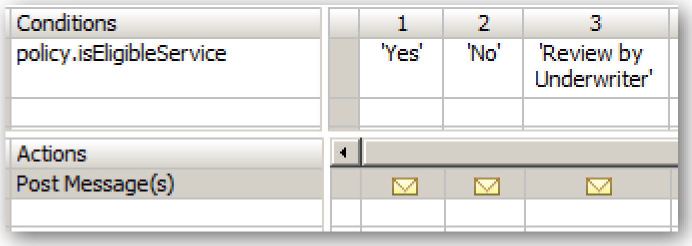
The alias "car" refers to ALL of the vehicles on a specific policy belonging to a particular client

Final Results



Ref	Text
1	ELIGIBILITY: >>-- Client is eligible for auto insurance at a premium of ${policy.servicePremium}. --<<
2	ELIGIBILITY: >>-- Client is not eligible for auto insurance. --<<
3	ELIGIBILITY: >>-- Client's application/policy renewal must be reviewed by underwriting manager. If approved, premium will be ${policy.servicePremium}.--<<

This rule sheet simply displays the final outcome of the decision:



Conditions		1	2	3
policy.isEligibleService		'Yes'	'No'	'Review by Underwriter'
Actions				
Post Message(s)		✉	✉	✉

The only actions are to post the final messages