

# Rule Modeling Case Study

## Generic

## Diabetic Monitoring

Mike Parish

## Contents

Table of Figures.....	3
Rule Modeling Case Study .....	4
Step 1 Review and clarify the rule statements .....	4
Step 2 Determine the Data Model and Vocabulary.....	4
Step 3 Determine the Decisions and Sub-Decisions .....	4
Step 4 Identify the rule sheets .....	5
Step 5 Copy the relevant rule statements into the appropriate rule sheet .....	6
Step 6 Model the Formal Rule Expression. ....	6
Step 7 Verify the Rules.....	6
Step 8 Add Natural Language Statements .....	9
Step 9 Deploy the Decision .....	9
Step 10 Checkin The Project .....	14
Step 11 Connect the Rules to an Application .....	18
Main Patient List .....	18
Detail for one Patient.....	18
Mobile Deployment .....	19

## Table of Figures

Figure 1 Rule Flow Diagram .....	5
Figure 2 Project Folder Structure and Rule Sheets .....	5
Figure 3 Rule Statements .....	6
Figure 4 Formal Rule Expressions .....	6
Figure 5 Testing the Rules As Specified.....	7
Figure 6 Ambiguity checker finds conflicting rules .....	7
Figure 7 Rule Sheet with ambiguity resolved. ....	8
Figure 8 Completeness Checker finds missing rule.....	8
Figure 9 Complete and Consistent Rule Sheet.....	8
Figure 10 Adding Natural Language Statements .....	9
Figure 11 Publishing from within Studio.....	9
Figure 12 Selecting the Server to Publish to .....	10
Figure 13 Choosing the Rule Flow.....	10
Figure 14 Viewing the Decision in the Web Console .....	11
Figure 15 Overview of the Decision Properties .....	11
Figure 16 Configuring the Decision Service .....	12
Figure 17 Select the Deployed Decision Service for Testing in Studio.....	12
Figure 18 The Decision Service URL .....	13
Figure 19 Test Results .....	13
Figure 20 Execution Statistics .....	13
Figure 21 Project Structure with Change Decorations .....	14
Figure 22 Checking In a Project.....	15
Figure 23 Documenting the Changes .....	15
Figure 24 The Complete History of Changes.....	16
Figure 25 Details of a Specific Change .....	16
Figure 26 Comparing Assets.....	16
Figure 27 Comparing Changes .....	17
Figure 28 The Change Structure Diagram .....	17
Figure 29 Comparing Changes .....	17

## Rule Modeling Case Study

### Step 1 Review and clarify the rule statements

When a patient comes into the emergency department (E.D), the patient information gets captured.

Today we have an ETL that executes every 4 hours and get this captured information into a SQL Server database. A program then executes that scores the risk factor of these patients based on certain parameters like blood glucose levels etc. The patient is given a risk score from 1 to 10. Where 1 being patient with least amount of risk and 10 being the high risk.

A care navigator is assigned to a patient. Care navigators can have many patients under them. Using an UI the care navigator can create lists of patients based on their risk factors. A patient can be added or removed from a list based on their risk score.

For patients who are kept on high risk list, a workflow needs to be executed. The workflow steps could be:

1. Call the patient
2. Make appointment for the patient
3. Order medications and other diabetic testing supplies for the patient

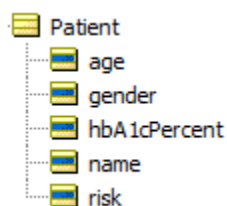
Let's assume for now that the actual rules are as follows:

*R1: If age > 80 then diabetes risk is High*

*R2: If HbA1C less than 7% then diabetes risk is Low*

### Step 2 Determine the Data Model and Vocabulary

A very simple vocabulary that will support the rules is as follows:



### Step 3 Determine the Decisions and Sub-Decisions

At the moment we only have one rule sheet. To construct the decision service we drag the sheet on to the rule flow diagram and specify some additional parameters such as the version and effective dates.

## Complete Diabetic Rules

The screenshot shows a software window with a menu bar at the top containing 'Rule Messages', 'Natural Language', 'Properties', and 'History'. Below the menu bar, the 'Rule Vocabulary' is set to 'file:/C:/Users/mparish/CorticonDecisionModels/workspace/NorthShore LIJ/SkydiverD', with a 'Browse...' button to its right. The 'Work Document Entity' is set to 'Patient' in a dropdown menu. Below this, there are input fields for 'Major Version' (2), 'Minor Version' (1), and 'Version Label' (Diabetes V2.1). The 'Effective Date' is set to '12/13/2013' with a dropdown arrow, followed by a 'Time' section with four input fields (0, 0, 0, AM) and a 'Clear' button. The 'Expiration Date' is set to '/' with a dropdown arrow, followed by another 'Time' section with four input fields (0, 0, 0, AM) and a 'Clear' button. At the bottom, the 'Total Number of Rules' is set to '3' in a text box.

Rule Vocabulary: file:/C:/Users/mparish/CorticonDecisionModels/workspace/NorthShore LIJ/SkydiverD

Work Document Entity: Patient

Major Version: 2

Minor Version: 1

Version Label: Diabetes V2.1

Effective Date: 12/13/2013  Time : 0   0   0   AM

Expiration Date: / /  Time : 0   0   0   AM

Total Number of Rules: 3

Figure 1 Rule Flow Diagram

## Step 4 Identify the rule sheets

The project folder might look something like this:

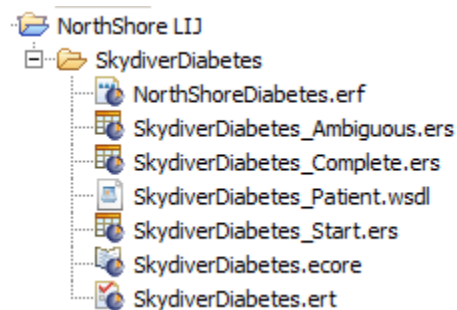


Figure 2 Project Folder Structure and Rule Sheets

## Step 5 Copy the relevant rule statements into the appropriate rule sheet

This is really two rules that define whether a patient is high or low risk

Rule Statements			✕	✉ Rule Messages	💬 Natural Language
Ref		Text			
1		R1: If age > 80 then diabetes risk is High			
2		R2: If HbA1C less than 7% then diabetes risk is Low			

Figure 3 Rule Statements

## Step 6 Model the Formal Rule Expression.

Conditions		1	2
a	Patient.age	> 80	-
b	Patient.hbA1cPercent	-	< 7
c			
Actions			
Post Message(s)		✉	✉
A	Patient.risk	'high'	'low'
R			
Overrides			

Rule Statements			✕	✉ Rule Messages	💬 Natural Language	📄 Pr
Ref		Text				
1		R1: If age > 80 then diabetes risk is High				
2		R2: If HbA1C less than 7% then diabetes risk is Low				

Figure 4 Formal Rule Expressions

## Step 7 Verify the Rules

We could try to verify the rules using test data:

Input	Output	Expected
<div> <div>Patient [1] {Tom age=85 HbA1c=9.5}</div> <div>age [85]</div> <div>hbA1cPercent [9.500000]</div> <div>name [Tom]</div> </div> <div> <div>Patient [2] {Dick age=25 HbA1c=5.5}</div> <div>age [25]</div> <div>hbA1cPercent [5.500000]</div> <div>name [Dick]</div> </div> <div> <div>Patient [3] {Harry age=85 HbA1c=5.5}</div> <div>age [85]</div> <div>hbA1cPercent [5.500000]</div> <div>name [Harry]</div> </div> <div> <div>Patient [4] {Jenny age=25 HbA1c=9.5}</div> <div>age [25]</div> <div>hbA1cPercent [9.500000]</div> <div>name [Jenny]</div> </div>	<div> <div>Patient [1] {Tom age=85 HbA1c=9.5}</div> <div>age [85]</div> <div>hbA1cPercent [9.500000]</div> <div>name [Tom]</div> <div>risk [high]</div> </div> <div> <div>Patient [2] {Dick age=25 HbA1c=5.5}</div> <div>age [25]</div> <div>hbA1cPercent [5.500000]</div> <div>name [Dick]</div> <div>risk [low]</div> </div> <div> <div>Patient [3] {Harry age=85 HbA1c=5.5}</div> <div>age [85]</div> <div>hbA1cPercent [5.500000]</div> <div>name [Harry]</div> <div>risk [low]</div> </div> <div> <div>Patient [4] {Jenny age=25 HbA1c=9.5}</div> <div>age [25]</div> <div>hbA1cPercent [9.500000]</div> <div>name [Jenny]</div> <div>risk [low]</div> </div>	<div> <div>Patient [1] {Tom age=85 HbA1c=9.5% HIGH risk because age &gt; 80}</div> <div>age [85]</div> <div>hbA1cPercent [9.500000]</div> <div>name [Tom]</div> <div>risk [high]</div> </div> <div> <div>Patient [2] {Dick age=25 HbA1c=5.5% LOW risk because low HbA1C}</div> <div>age [25]</div> <div>hbA1cPercent [5.500000]</div> <div>name [Dick]</div> <div>risk [low]</div> </div> <div> <div>Patient [3] {Harry age=85 HbA1c=5.5% HIGH risk because age &gt; 80}</div> <div>age [85]</div> <div>hbA1cPercent [5.500000]</div> <div>name [Harry]</div> <div>risk [high]</div> </div> <div> <div>Patient [4] {Jenny age=25 HbA1c=9.5% HIGH risk because HbA1c elevated}</div> <div>age [25]</div> <div>hbA1cPercent [9.500000]</div> <div>name [Jenny]</div> <div>risk [high]</div> </div>

Severity	Message	Entity
Info	R1: If age > 80 then diabetes risk is High	Patient[3]
Info	R1: If age > 80 then diabetes risk is High	Patient[1]
Info	R2: If HbA1C less than 7% then diabetes risk is Low	Patient[3]
Info	R2: If HbA1C less than 7% then diabetes risk is Low	Patient[2]

Figure 5 Testing the Rules As Specified

We can see that not all the results are what we expected.

Harry was classified as 'low' risk when he should have been 'high' (because of his age)

Jenny was not classified at all.

If we go back into Corticon Studio and run the ambiguity checker we will discover the following:

Conditions	1	2	3	4
a Patient.age	> 80	-		
b Patient.hbA1cPercent	-	< 7		
Actions				
Post Message(s)				
A Patient.risk	'high'	'low'		
Overrides				

Ref	Text
1	R1: If age > 80 then diabetes risk is High
2	R2: If HbA1C less than 7% then diabetes risk is Low

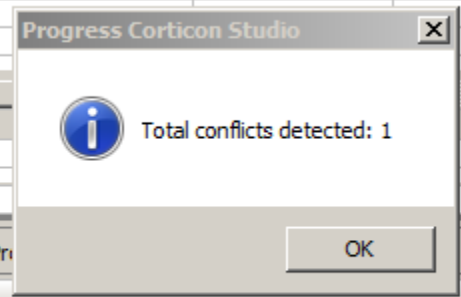


Figure 6 Ambiguity checker finds conflicting rules


We can resolve the ambiguity by making rule 2 more specific. Probably what it should have said was:

R2: If HbA1C less than 7% then diabetes risk is Low (for people 80 or under)

The modified rule sheet now looks like this:

Conditions		1	2
a	Patient.age	> 80	<= 80
b	Patient.hbA1cPercent	-	< 7
Actions			
Post Message(s)			
A	Patient.risk	'high'	'low'
Overrides			

**Progress Corticon Studio**

 No conflicts detected

OK


Ref	Text
1	R1: If age > 80 then diabetes risk is High
2	R2: If HbA1C less than 7% then diabetes risk is Low (for people 80 or under)

Figure 7 Rule Sheet with ambiguity resolved.

If we run the completeness checker we get:

Conditions		3
a	Patient.age	<= 80
b	Patient.hbA1cPercent	>= 7
Actions		
Post Message(s)		
A	Patient.risk	
Overrides		

**Progress Corticon Studio**

 Completeness check has added 1 missing scenarios which have been automatically compressed in to 1 non-overlapping columns.

OK

Ref	Text
1	R1: If age > 80 then diabetes risk is High
2	R2: If HbA1C less than 7% then diabetes risk is Low (for people 80 or under)
3	R3: if age is 80 or less and HbA1C is not over 7 then diabetes risk is low.

Figure 8 Completeness Checker finds missing rule

This was the missing case that resulted in Jenny not getting classified.

We can add an appropriate rule statement for this:

*R3: if age is 80 or less and HbA1C is not over 7 then diabetes risk is low.*

So after fixing this we end up with:

Conditions		1	2	3
a	Patient.age	> 80	<= 80	<= 80
b	Patient.hbA1cPercent	-	< 7	>= 7
Actions				
Post Message(s)				
A	Patient.risk	'high'	'low'	'low'
Overrides				

Ref	Text
1	R1: If age > 80 then diabetes risk is High
2	R2: If HbA1C less than 7% then diabetes risk is Low (for people 80 or under)
3	R3: if age is 80 or less and HbA1C is not over 7 then diabetes risk is low.

Figure 9 Complete and Consistent Rule Sheet

Which is both complete and consistent.



## Step 8 Add Natural Language Statements

These help make the rules easier to read for those people who are not regular users of Corticon Studio.

Conditions	1	2	3
What is the Patient's age?	> 80	<= 80	<= 80
What is the Patient's HbA1C level?	-	< 7	>= 7
Actions			
Post Message(s)	✉	✉	✉
What is the Patient risk?	'high'	'low'	'low'

Figure 10 Adding Natural Language Statements

## Step 9 Deploy the Decision

Decisions may be deployed in two ways:

1. As web services
2. As in-process execution

In this case study we'll use Web Services Deployment

When the Corticon Execution Environment (Server) is installed (under Tomcat, Websphere, Weblogic etc) you can publish decision services in a number of ways

1. Publish option in Studio
2. Create a CDD using the deployment console
3. Use the Web Console to deploy a decision

For this case study we'll use the Studio publish option.

A decision service in Corticon is defined by a Rule Flow.

To deploy a rule flow (Decision Service) you can use the publish option:

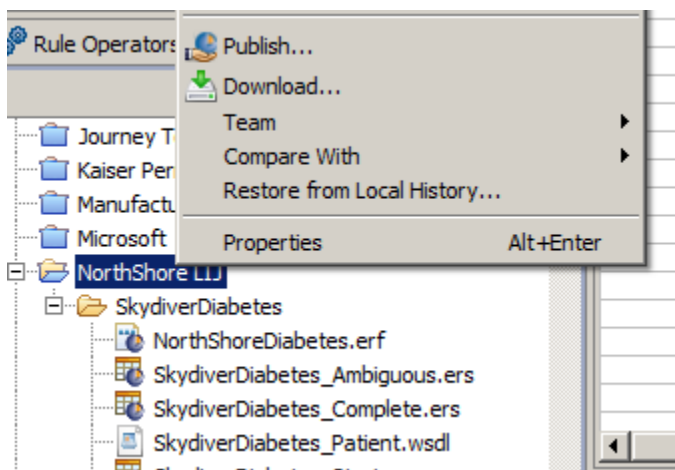


Figure 11 Publishing from within Studio

The first step is to select the server that you want to deploy to:

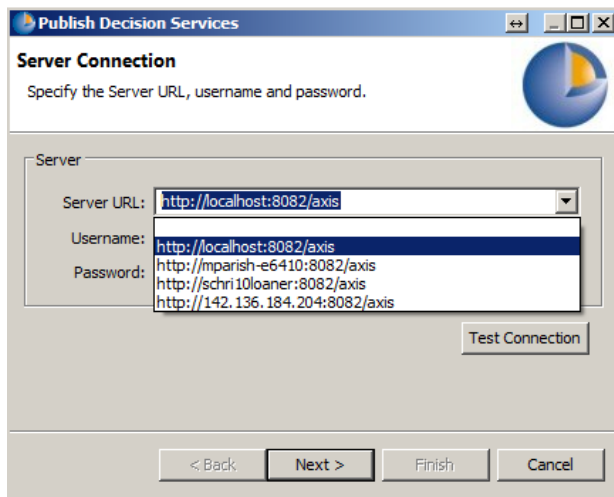


Figure 12 Selecting the Server to Publish to

We'll use localhost. This brings up a list of the decisions that are available for deployment:

Select the one you want. You can rename it if you like.

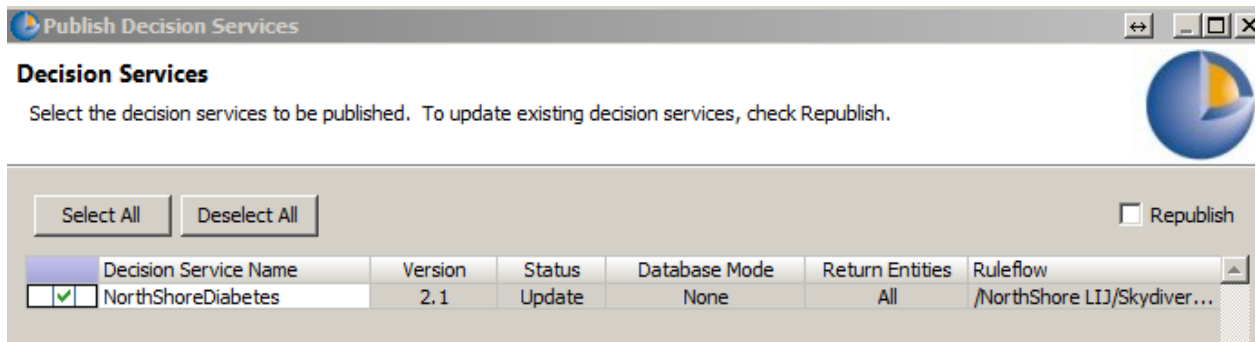


Figure 13 Choosing the Rule Flow

You should then see the decision listed in the console browser window:

Service Name	Version	Live	Effective Expires	Deployed from CDD	Dynamic Reload	Executions	Avg Time (ms)	Clear Stats
 <span style="float: right;">The Power of Decisions™</span>								
NorthShoreDiabetes	1.1			No/Remove	No	28	1	Clear
NorthShoreDiabetes	2.1		Dec 13, 2013	No/Remove	No	14	0	Clear
PIR79942 Retirement Eligibility Decision	1.0		Dec 1, 2013 Dec 31, 2013	No/Remove	No	0	0	Clear
ProcessOrder	1.10			Yes	No	0	0	Clear

Figure 14 Viewing the Decision in the Web Console

Notice that the execution count and average execution time columns have some values (28 for version 1.1 and 14 for version 2.1)

You can also see some other decisions on the server.

You can inspect the properties of the decision by clicking on its name:

Overview	Service Configuration	Rules Report	Test Execution
<b>General Settings</b>			
Deployed As Test Decision Service:	No		
Deployed EDS file:	./C31386958053746/NorthShoreDiabetes_v2_1.eds		
Ruleflow URL:	./U11386958051227/NorthShoreDiabetes.erf ( View )		
Rulesheet URLs:	Start ( View ) Ambiguous ( View ) Complete ( View )		
XML Message Style:	Auto-detect		
Dynamic Reload:	No		
Loaded from CDD file:	No		
Deployment Timestamp:	Dec 13, 2013 7:09:49 AM		
Ruleflow Timestamp:	Dec 13, 2013 10:07:31 AM		
EDS Timestamp:	Dec 13, 2013 10:07:33 AM		
Total Number of Rules Deployed:	3		
Effective Date Start:	Dec 13, 2013 12:00:00 AM		
Effective Date End:			
Total Execution Count:	14		
Average Execution Time (ms):	0		

Figure 15 Overview of the Decision Properties

You can use this screen to configure the decision service:

Overview **Service Configuration** Rules Report Test Execution

Current Rule Asset URL: ./C31386958053746/NorthShoreDiabetes\_v2\_1.eds

Rule Asset URL:  Browse...

Minimum Pool Size:

Maximum Pool Size:

XML Message Style: Auto-detect ▼

Database Access Mode: None ▼

Database Access Entities Returned Mode: All Entities ▼

Current Properties File Location:

Database Access Properties File:  Browse...

Update

**Monitored Attributes** **Analysis Buckets**

☒ Patient.age

☒ Patient.hbA1cPercent

☒ Patient.risk

Figure 16 Configuring the Decision Service

You can see we have indicated three attributes to be monitored by the Corticon Server.

Now we are ready to execute this decision.

In a production environment we might write a java program that makes a SOAP call to the rule engine, or we might use a Business Process Engine that can invoke a web service as one of its tasks.

Before we get to that point however, we can test the web service call from within Studio.

This is done in the tester.

To do this we just select the Corticon Server URL (rather than a rule flow or rule sheet inside Studio):

Remote Servers

http://localhost:8082/axis

http://mparish-e6410:8082/axis

Update List

Remote Decision Services

Mortgage

NorthShoreDiabetes

Figure 17 Select the Deployed Decision Service for Testing in Studio

The test subject will change accordingly:

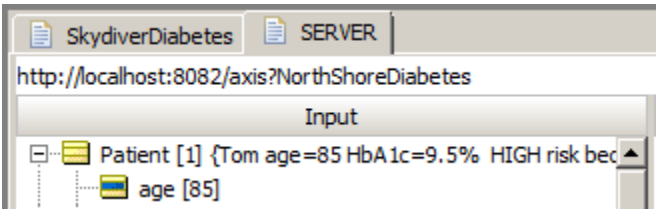


Figure 18 The Decision Service URL

Incidentally this is the URL you would use to call the decision service from other applications (such as Rollbase)

Now we can run the test (say 10 times)

We should get some results back like this:

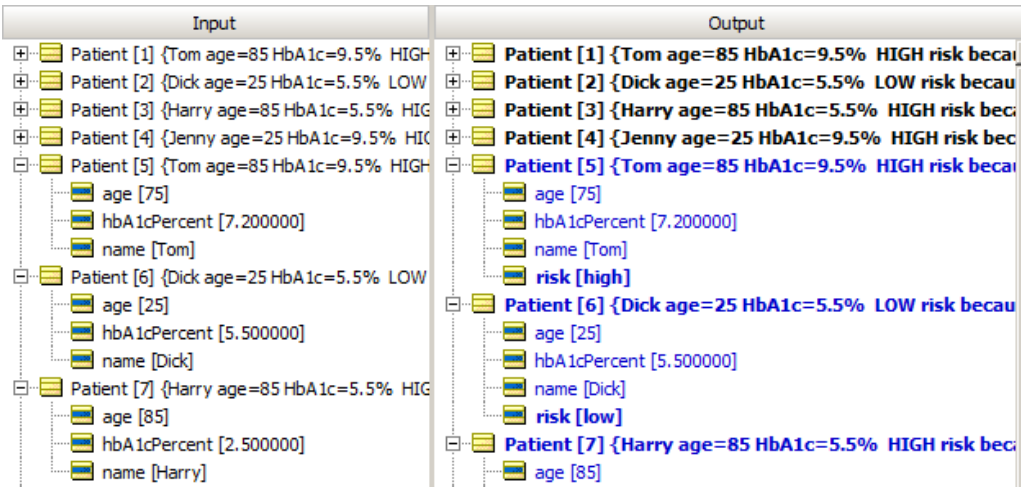


Figure 19 Test Results

Now if we go back to the Web Console we'll see that the execution count has been updated:

NorthShoreDiabetes	1.1			No/Remove	No	28	1	Clear
NorthShoreDiabetes	2.1		Dec 13, 2013	No/Remove	No	45	0	Clear

Figure 20 Execution Statistics

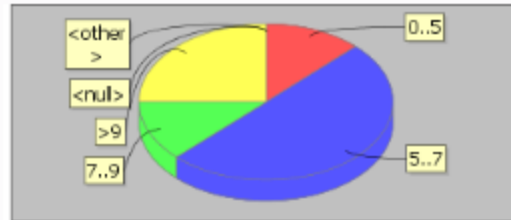
The distribution chart shows how the results were broken down.

If you click on the execution count you will see the monitored attribute charts:

#### Patient.hbA1cPercent

Value	Quantity	Percentage
0..5	4	12.5
5..7	16	50
7..9	4	12.5
>9	8	25
<other>	0	0.00
<null>	0	0.00

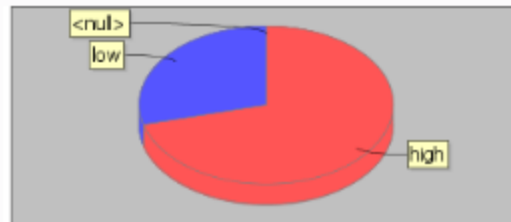
**Total Changes 32**



#### Patient.risk

Value	Quantity	Percentage
high	112	70.89
low	46	29.11
<null>	0	0.00

**Total Changes 158**



## Step 10 Checkin The Project

At this point we might want to check in the project to our repository for safe keeping.

Because Corticon's rule assets are all simple files (XML) any content management system can be used for controlling the assets.

In this case study we'll use the TEAM plugin from Roundtable Tugboat (A Progress partner) to manage our assets.

This project (not the Diabetes Monitoring project) has already been configured to use TEAM:

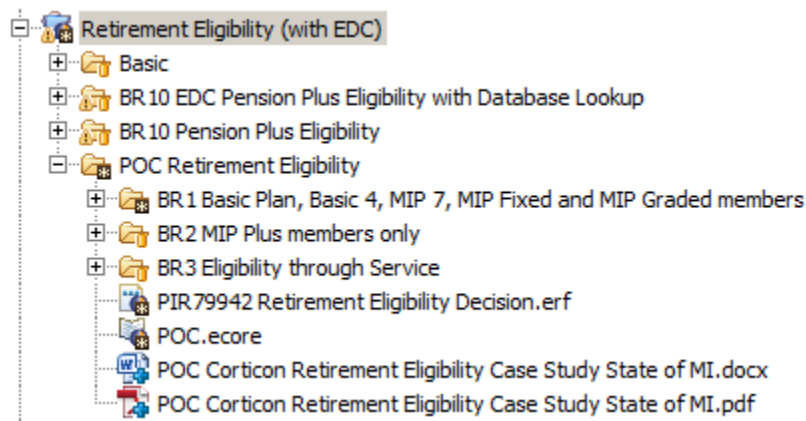


Figure 21 Project Structure with Change Decorations

You can see some of the assets are flagged with \* (they have been modified since the last checkin) and some of them are flagged with + (they are new). The rest haven't changed.

The TEAM checkin is performed here:

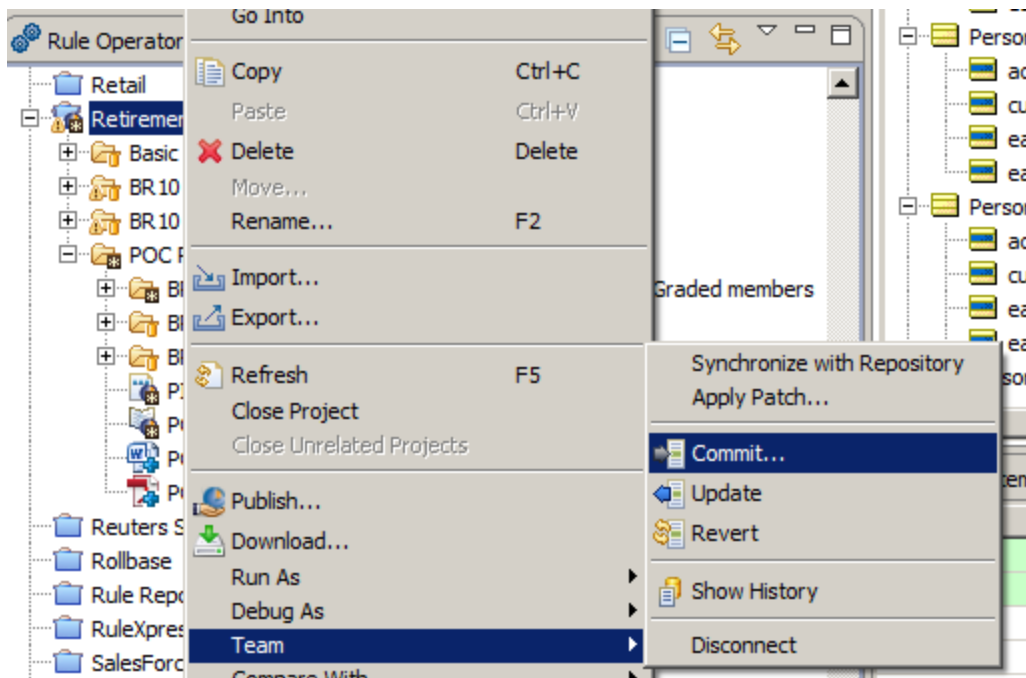


Figure 22 Checking In a Project

We can record some suitable commentary about the changes we are checking in:

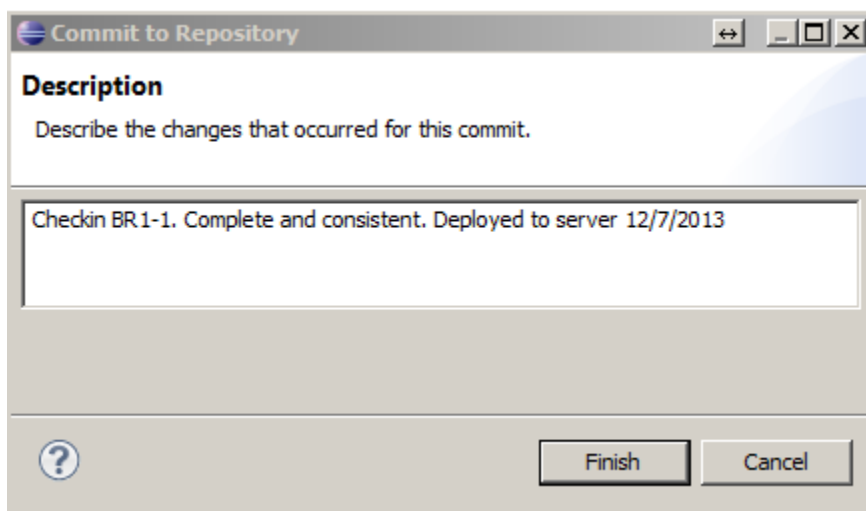


Figure 23 Documenting the Changes

If we go to the history tab we can see the history of changes made to this project (and who made the changes):

Retirement Eligibility (with EDC)			
Change	Date	Author	Description
<b>81</b>	<b>Dec 7, 2013 7:37:16 PM</b>	<b>mparish</b>	<b>Checkin BR1-1. Complete and consisten. Deployed to Server 12/7/2013</b>
80	Dec 6, 2013 5:59:26 PM	mparish	Rule flow modeled. Rule sheets created and rule statements added. Sheet BR1-1...
79	Dec 5, 2013 9:30:43 AM	mparish	Modified docs
78	Dec 5, 2013 9:27:10 AM	mparish	Rules specification
77	Dec 5, 2013 9:26:26 AM	mparish	New Project folder Structure for Retirement Eligibility rules POC for HP and State ...
76	Dec 4, 2013 1:31:46 PM	mparish	

Figure 24 The Complete History of Changes

If you select a specific change you can see all of the assets that we modified as part of that change:

Retirement Eligibility (with EDC)			
Change	Date	Author	Description
<b>81</b>	<b>Dec 7, 2013 7:37:16 PM</b>	<b>mparish</b>	<b>Checkin BR1-1. Complete and consisten. Deployed to Server 12/7/2013</b>
80	Dec 6, 2013 5:59:26 PM	mparish	Rule flow modeled. Rule sheets created and rule statements added. Sheet BR1-1...
79	Dec 5, 2013 9:30:43 AM	mparish	Modified docs
78	Dec 5, 2013 9:27:10 AM	mparish	Rules specification
77	Dec 5, 2013 9:26:26 AM	mparish	New Project folder Structure for Retirement Eligibility rules POC for HP and State ...
76	Dec 4, 2013 1:31:46 PM	mparish	

Name	Action	Comment
As Specified BR1-1 Active ...	Add	
POC Corticon Retirement E...	Add	
POC Corticon Retirement E...	Add	
~\$C Corticon Retirement E...	Add	
As Specified BR1-1 Active ...	Update	
BR 1 Test Cases.ert	Update	
BR 1-1 Active at time of ret...	Update	
PIR79942 Retirement Eligi...	Update	
POC.ecore	Update	
POC Rules State of MI.docx	Delete	

Figure 25 Details of a Specific Change

If you select two assets you can compare them:

As Specified BR1-1 Active ...	Update	
BR 1 Test Cases.ert	Update	
BR 1-1 Active at time of ret...	Update	
PIR79942 Retirement Eligi...	Update	
POC.ecore	Update	

Open...	
Compare With	Each Other
Synchronize To	

Figure 26 Comparing Assets

You can also compare two changes:



Change	Date	Author	Description
81	Dec 7, 2013 7:37:16 PM	mparish	Checkin BR1-1. Complete and consisten. Deployed to Server 12/7/2013
80	Dec 6, 2013 5:		Rule flow modeled. Rule sheets created and rule statements added. Sheet BR1-1...
79	Dec 5, 2013 9:		Modified docs
78	Dec 5, 2013 9:		Rules specification
77	Dec 5, 2013 9:		New Project folder Structure for Retirement Eligibility rules POC for HP and State ...
76	Dec 4, 2013 1:		

Name	Compare With	Each Other	Checkin BR1-1. Complete and consisten. Deployed to Ser
As Specified BR 1-1 Acti	Replace	Local	

Figure 27 Comparing Changes

This will bring up a structure compare diagram:

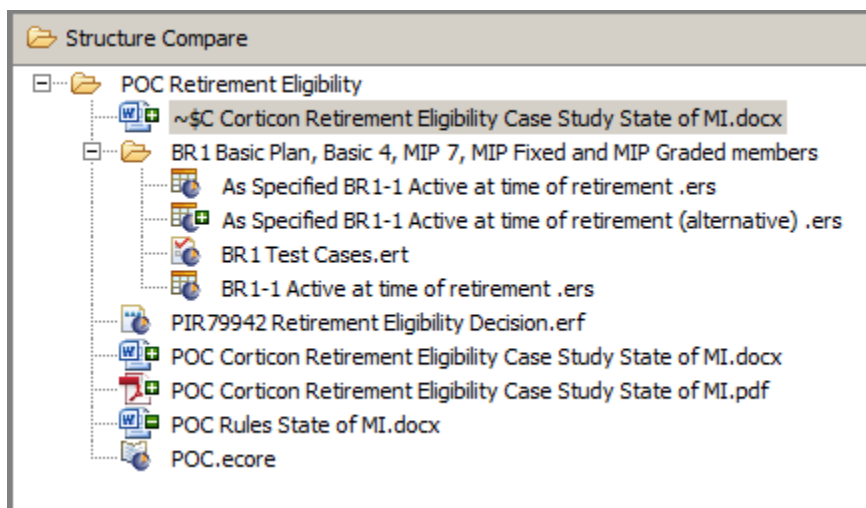


Figure 28 The Change Structure Diagram

Selecting an item will bring up a detailed compare screen:

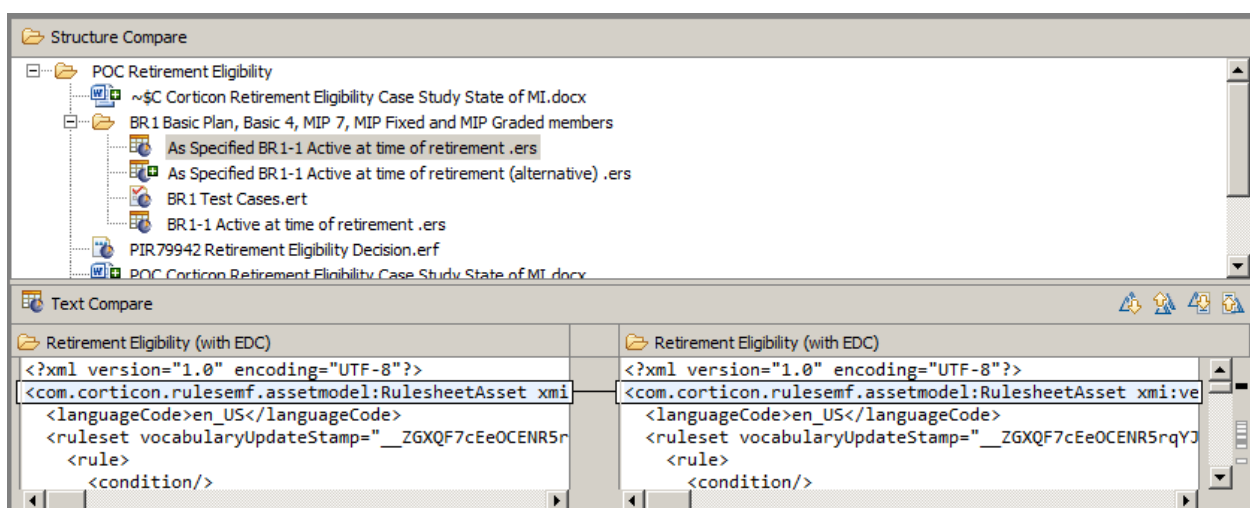


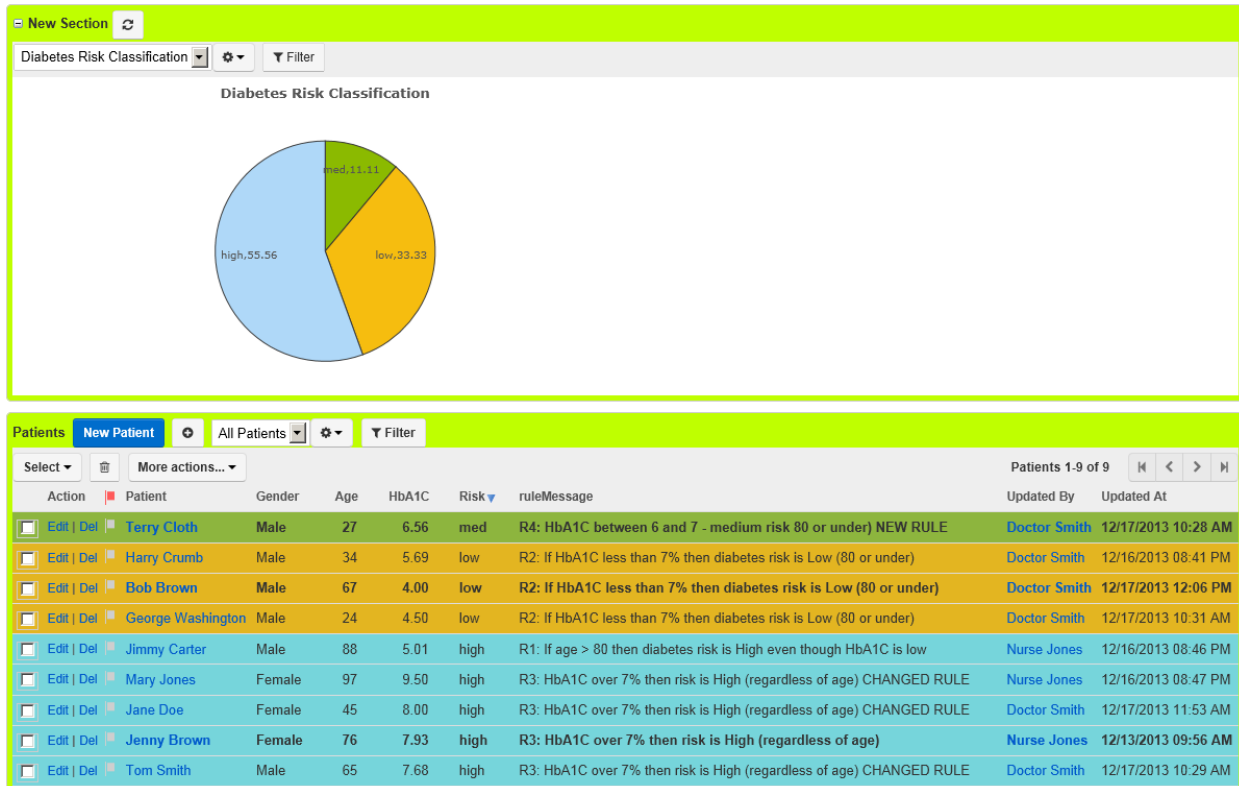
Figure 29 Comparing Changes

## Step 11 Connect the Rules to an Application

Eventually the rules will be used by a real application. This might be a batch processing job that sends many records to the rules for evaluation or it might be a web based application that handles a single record at a time.

For this case study we'll examine the use of Rollbase, one of Progress's Pacific software products that enables the rapid development and deployment of web based applications.

### Main Patient List



### Detail for one Patient

Patient: Jimmy Carter

[Back to List](#)

[Edit](#)

[Delete](#)

[More actions...](#)

[Previous](#)

[Next](#)

Patient Info

System Info

Contact Information

First Name	Jimmy	Title	President
Middle Name	Billy Bob	Mobile Phone	
Last Name	Carter	Email Address	
Phone	(909)123-4567	Contact Owner	
Fax		vCard File	<a href="#">Synchronize</a>
Age	88	HbA1C level	5.01
risk score	9	risk classification	high
ruleMessage	R1: If age > 80 then diabetes risk is High even though HbA1C is low	gender	Male

Workflow Information

The risk classification is determined by the decision service we looked at earlier.

Whenever a change is made to the patient information, the decision service is invoked and the display is updated with any new information.

## Mobile Deployment

This same app can be deployed automatically on any mobile platform.

For example the iPhone:

