OPENRULES®

Open Source Business Decision Management System

Release 6.2.1

Decision Modeling Tutorial

"Determine Patient Therapy"

Part 1

OpenRules, Inc.

www.openrules.com

June-2011

Table of Contents

INTRODUCTION

In this document we provide a detailed description of how to create your own executable decision with OpenRules®. We describe a simple medical use case to explain how to present a business decision with business facts and related decision tables in Excel.

Let's assume that two people will work together to create and test the decision:

- BA - Business Analyst who is familiar with OpenRules but has no IT background
- ME - Medical Expert who is a pure subject matter expert with no OpenRules and decision modeling experience.

While in real life the actual implementation can be done by one person or by more than two different people, the roles of a business analyst and a subject matter expert will frequently follow the pattern of interactions shown below.

BUSINESS CASE

ME. You asked me to prepare a simple use case for when a patient visits a doctor. I tried to describe rules used by a doctor to make a decision about the required therapy for an encounter diagnosis. Here is a simplified scenario when the diagnosis is Acute Sinusitis.

Medication Rules:

If Patient is 18 years old or older, then a therapy choice is Amoxicillin.
If Patient is younger than 18, a therapy choice is Cefuroxime.
If patient Penicillin allergic, therapy of choice is Levofloxacin.
Check if patient on active medication. Coumadin and Levofloxacin can result in reduced effectiveness of Coumadin. Produce the proper warning.

Dosing Rules:
For patients between 15 and 60, the dose is 500mg every 24 hours for 14 days.
If Patient's creatinine level > 1.4, commence creatinine clearance (CCr) calculation according to the formula:

$$\text{CCr, in mL/min} = \frac{(140 - \text{age}) \times \text{lean body weight [kg]}}{\text{PCr [mg/dL]} \times 72}$$

If patient's creatinine clearance < 50 ml/min, then the dose is 250mg every 24 hours for 14 days.
Dosing also depends on renal function, immune state, or liver function but the proper rules will be added later.

BA. Thank you, this is a pretty good description of your clinical guidelines. Today we will try to create the proper decision and to execute it using several simple test cases. Let's start with a definition of a decision we want to make.

STARTING WITH DECISION

BA. What is the objective of our future decision? It should determine the fact "Patient Therapy", which in our case can be Amoxicillin or Cefuroxime or Levofloxacin. I suggest calling our decision "DeterminePatientTherapy".

ME. Sounds good to me. But I specified two different types of rules: one for Medication and

another for Dosing…

BA. It means that our high-level decision "DeterminePatientTherapy" consists at least of two sub-decisions:

- Define Medication (e.g. Levofloxacin)
- Define Dosing (e.g. 250mg every 24 hours for 14 days)

Two types of rules in your description specify how to make these two decisions for different patients.

ME. I got it – you start not with rules but with decisions.

BA. Yes, OpenRules recommends to always start with decisions – they call it a Top-Down approach. We may modify rules that lead us to the decisions later on but the decisions themselves will stay the same. Now we need to define rules (or better to say "decision tables") that actually implement these decisions. So, let's do it together.

ME. Do I need to know how to do programming?

BA. Not at all, I have very limited programming expertise myself. However, I've already used OpenRules to develop several simple decision myself. You will see that it is quite intuitive and the best thing about it is that we do not need any fancy tools beyond regular Excel and Windows Explorer.

I already have OpenRules® installed in the folder "c:/`openrules.decisions`" on my hard drive. First, let's use Windows Explorer to create a new folder called "`DecisionPatientTherapy`" inside "`openrules.decisions`". We will call it the "project folder" and it will serve as a repository for all our files.

Now I will go to Windows Explorer and create a subfolder "`rules`" inside our project folder: we will call it "rules repository" because we plan to keep all Excel files related to our rules-based project inside this folder. For simplicity, I will use only one main Excel file. So, I am opening Microsoft Excel and will save an empty Excel workbook under the name "`DecisionPatientTherapy.xls`" inside the folder "`rules`".

ME. OK, so far there is nothing new – I use Excel all the time and I love it.

BA. Good. Now I will rename the first worksheet "Decision" and will create a new table that describes the structure of our high-level decision. Here it is:

Decision DeterminePatientTherapy	
Decisions	**Execute Decision Tables**
Define Medication	:= DefineMedication()
Define Dosing	:= DefineDosing()

The first line defines the name of our decision (`DeterminePatientTherapy`) that follows the keyword "Decision". OpenRules has many predefined tables and they all are recognized by their keywords like "Decision", "DecisionTable", "Data", etc. This decision table has two columns "Decisions" and "Execute DecisionTables". The first column contains the names of all our sub-decisions - here we can use any combinations of words as decision names. The second column is a little bit trickier: it contains exact names of our future decision tables that implement these sub-decisions. The names cannot contain spaces or special characters (except for "underscore") and they should always be preceded by ":=", which indicates the decision tables will actually be executed by the OpenRules engine.

ME. I got this part, but I noticed that you merged all cells that comprise the very first line with the decision name. Why?

BA. Good for you - I missed this part when I built my first decision with OpenRules and got a very strange error when I tried to run my first decision. This merge is necessary to indicate to the OpenRules engine what the actual width of the decision table is. This table has only two columns but other tables such as decision tables may have any number of columns.

Note that you also should have empty cells above the first line and all empty cells below the decision table – they indicate the beginning and the end of the table. These requirements are common for all OpenRules tables.

ME. OK, I will remember to surround OpenRules tables with empty cells. Do I have to use the same colors as you do for all my tables?

BA. Of course not – you can use any color you prefer, but they told me that the majority of OpenRules users prefer a black background and a white foreground for all title rows. It has become a de-facto standard, so let's stick with it. Now, it is time to define our decision tables.

Defining Decision Tables

BA. Let's first represent medication rules that we specified above as a decision table "DefineMedication". I will add another worksheet to our file "DecisionPatientTherapy.xls", and will call it "Medication Rules". Let me start with the title row:

DecisionTable DefineMedication

Here I am using a keyword "DecisionTable" and after a space I put the name

"DefineMedication" – exactly what we called this decision table inside our table "Decision". What will be inside this decision table? Usually a decision table contains multiple conditions connected by a logical "AND" operator and a single conclusion column. What are the conditions and the conclusion in your medication rules?

ME. Our first 3 medication rules are:

1) If Patient is 18 years old or older, then the therapy choice is Amoxicillin
2) If Patient is younger than 18, then the therapy choice is Cefuroxime.
3) If Patient is allergic to Penicillin, then the therapy choice is Levofloxacin.

They all end up with a conclusion that defines a recommended medication based on a patient's age and possible allergies.

BA. Right. Our rule conditions and conclusion deal with so called "decision variables". For example, the common conclusion for all three above rules specifies the fact

"Medication is < Amoxicillin or Cefuroxime or Levofloxacin>"

ME. Why did you use the term "Recommended Medication" instead of "Therapy Choice"?

BA. You can use any words to define your decision variable. We just called our decision table "DefineMedication", so I wanted to be consistent with the names. So, our conditions deals with two decision variables "Patient Age" and "Patient Allergies":

- "Patient Age <is, is more, or is less than> <Age>"
- "Patient Allergies include <Recommended Medication>"

With OpenRules we may represent the decision table as follows:

DecisionTable DefineMedication					
Condition		Condition		Conclusion	
Patient Age		**Patient Allergies**		**Recommended Medication**	
>=	18	Do not include	Penicillin	Is	Amoxicillin
<	18	Do not include	Penicillin	Is	Cefuroxime
		Include	Penicillin	Is	Levofloxacin

Each condition here consists of two columns: one for an operator (like ">=" or "Do not include") and another for a value (like "18" or "Penicillin"). For example, the first rule should be read as: "IF Patient Age is more than or equal to 18 AND Patient Allergies do

not include Penicillin, THEN Recommended Medication is Amoxicillin". Remember that all conditions inside a decision table are connected by the operator "AND" and never by "OR".

ME. OK, I understand that you are trying to assign Amoxicillin or Cefuroxime based on the patient's age only when the patient's allergies do not include Penicillin. But you left the Age cells empty for the third rule. Probably this represents the fact that when a patient is allergic to Penicillin we recommend Levofloxacin independently of the patient's age. Correct?

BA. Absolutely! Empty cells mean that the proper condition is always satisfied. Note that we also have to make sure that our rule family covers all possible combinations of conditions and they are all mutually exclusive.

ME. I guess it would not be as simple when we have more conflicts between allergies and recommended medications.

BA. You are right again, but this is your business logic and sometimes it would be hard to simplify it. So, we should be more proactive defining our decision tables always thinking about future changes. For example, in this example the second condition uses operators "Do not include" and "Include" instead of the operator "Is". These operators allow us to list inside the conditions several allergies separated by commas. Similarly, when necessary you may use the operator "Are" instead of "Is" in the conclusion to specify several values.

ME. If my therapy choice also depends on a patient's weight I can probably simply add one more condition for the variable "Patient Weight".

BA. You got it. How about the "Encounter Diagnosis"?
ME. Let's add it as the very first condition of this decision table.

BA. Here it goes:

DecisionTable DefineMedication							
Condition		Condition		Condition		Conclusion	
Encounter Diagnosis		**Patient Age**		**Patient Allergies**		**Recommended Medication**	
Is	Acute Sinusitis	>=	18	Do not include	Penicillin	Is	Amoxicillin
Is	Acute Sinusitis	<	18	Do not include	Penicillin	Is	Cefuroxime
Is	Acute Sinusitis			Include	Penicillin	Is	Levofloxacin

It should work.

ME. Now we have to add more rules when a patient is taking Coumadin that may interact with Levofloxacin.

BA. I'd suggest worrying about these interaction rules later on probably by creating another decision table. In general, we should never overcomplicate decision tables: it is better to create two different simple decision tables than to have a complex and difficult to understand single decision table. Let's first complete dosing rules, so we can test our decision.

ME. OK. As I wrote, for dosing we will consider only the patient's age and creatinine clearance but as I also wrote, I want to make sure that later on I can add conditions that deal with liver function, immune state, etc.

BA. No problem. In this case, your conditions will use two decision variables "Patient Age" and "Patient Creatinine Clearance", and your conclusion will define the variable "Recommended Dose".

ME. Let me add the proper decision table myself. I am adding another spreadsheet called "Dosing Rules" by copying/pasting your table, and replacing "DefineMedication" with "DefineDosing".

BA. Good, "DefineDosing" is exactly what we called this decision table in the decision "DeterminePatientTherapy".

ME. I will leave the first two conditions "Encounter Diagnosis" and "Patient Age" without change, and I will simply replace the condition "Patient Allergies" with "Patient Creatinine Level". The conclusion "Recommended Medication" now will be called "Recommended Dose".

Wait a minute! How could I tell that a Patient Age is between 15 and 60? Should I add another column and use two different operators for "less" and "more"?

BA. You certainly can do that, but it would be more compact just to use the operator "Within" with a value cell specified as the interval "15 – 60". By default, the bounds 15 and 60 are included but you always may define your interval explicitly like "[15;60]".

ME. That's nice. But I am afraid we have to add one more condition "Patient Creatinine Clearance" to be used when "Patient Creatinine Level" is more than 1.4.

BA. You are right – we frequently understand our rules better when we are actually trying to represent them in a decision table format.

ME. No problem, I've just added another condition. Here is my decision table:

DecisionTable DefineDosing									
Condition		Condition		Condition		Condition		Conclusion	
Encounter Diagnosis		**Patient Age**		**Patient Creatinine Level**		**Patient Creatinine Clearance**		**Recommended Dose**	
Is	Acute Sinusitis	Within	15 - 60					Is	500mg every 24 hours for 14 days
Is	Acute Sinusitis			>	1.4	<	50	Is	250mg every 24 hours for 14 days

BA. It seems like a correctly organized decision table and it is almost identical to your plain English description. I would not worry about the Creatinine Clearance calculation formula at this point (assuming it is somehow calculated before this decision table will be executed). What really worry me are the multiple situations not covered by these rules. For instance, what if a patient is older than 60 and has a Creatinine Level less than 1.4 or Creatinine Clearance larger than 50?

ME. Do you mean that I have to add rules that cover all possible combinations of Patient Age, Creatinine Level, and Creatinine Clearance?

BA. Yes. Otherwise, your dose will remain undefined. But you may add these rules later on, while for now you may simply add one more rule for all "otherwise" situations, like this:

DecisionTable DefineDosing									
Condition		Condition		Condition		Condition		Conclusion	
Encounter Diagnosis		**Patient Age**		**Patient Creatinine Level**		**Patient Creatinine Clearance**		**Recommended Dose**	
Is	Acute Sinusitis	Within	15 - 60					Is	500mg every 24 hours for 14 days
Is	Acute Sinusitis			>	1.4	<	50	Is	250mg every 24 hours for 14 days
Is	Acute Sinusitis							Is	Not defined yet

ME. I will certainly add more rules later on. How far are we from running our decision tables and seeing some results?

BA. We have the major decision elements in place but we still have to define the concepts we used inside our two decision tables and create a few test cases to test our decision.

DEFINING BUSINESS GLOSSARY

BA. Our two decision tables deal with the following decision variables:

- Encounter Diagnosis
- Patient Age
- Patient Allergies
- Recommended Medication
- Patient Creatinine Level
- Patient Creatinine Clearance
- Recommended Dose.

I simply listed the titles of all columns from both decision tables. Now we need to associate these decision variables with business concepts and their attributes. We should ask ourselves a question: "Which business concepts do all of these decision variables belong to?" I'd say here we are dealing with only two business concepts:

- Patient
- Doctor Visit

Patient has Age, Allergies, Creatinine Level and Clearance, i.e. those variables can be attributed to a patient. The concept "Patient" is simply a placeholder for these attributes and their current values.

ME. And I guess you want to use the concept "Doctor Visit" as a placeholder for other variables that are defined during a visit like Recommended Medication and Dose.

BA. Yes, and Encounter Diagnosis too, as well as the date of the visit and probably other attributes that we currently do not use. OpenRules provides a special table called "glossary" that defines these relationships and serves as a bridge to actual business objects used by our IT people. I am putting this table in a separate spreadsheet "Glossary":

Glossary glossary		
Variable	**Business Concept**	**Attribute**
Encounter Diagnosis	Doctor Visit	encounterDiagnosis
Recommended Medication		recommendedMedication
Recommended Dose		recommendedDose
Patient Age	Patient	age
Patient Allergies		allergies
Patient Creatinine Level		creatinineLevel
Patient Creatinine Clearance		creatinineClearance

The first column "Variable Name" contains names of our facts exactly as they were defined

in decision tables. The second column contains our newly introduced business concepts – note that I merged rows to indicate which fact belongs to which concept. And finally, the third row has the "technical" names of our facts. We may name these attributes by simply omitting spaces in our fact names. I was told that it is better to follow a Java naming convention meaning the first letter should be lowercase and words inside an attribute names should start with capital letters. These names will be used only by IT when they integrate our decision with their actual information system – they actually may even change them.

Now we are ready to define our test cases.

DEFINING TEST DATA

BA. To test our decision, we will define test data using OpenRules Datatype and Data tables that correspond to our business concepts. First, we define two Datatypes "Patient" and "DoctorVisit" (without a space!):

Datatype Patient	
String	name
int	age
double	creatinineLevel
double	creatinineClearance
String[]	allergies

Datatype DoctorVisit	
Date	date
String	encounterDiagnosis
String	recommendedMedication
String	recommendedDose

An important point here is that these data types should use exactly the same attributes that we used in the Glossary above. They also can have additional attributes, e.g. "date", "name". Note that the attribute "allergies" was defined as an array of strings using "String[]".

ME. You are getting a little bit technical but I am still with you.

BA. Thank you, be patient – a few more technical details and we will run our decision. Now I will create data tables with concrete test-instances of the types Patient and DoctorVisit. Let's start with tests for Patient:

Data Patient patients				
name	age	allergies	creatinineLevel	creatinineClearance
Name	**Age**	**Allergies**	**Creatinine Level**	**Creatinine Clearance**
John Smith	58	Penicillin Streptomycin	2.00	44.42
Mary Smith	65		1.80	48.03

Inside a Data table the second row usually contains the names of attributes as they are defined in the proper Datatype table, and the third row contains descriptions of these attributes (similarly to our decision variable names in decision tables). For simplicity, I put some test data for the Creatinine Clearance – I will show you later how to use your formula. Note how I specified an array of "allergies" in the table "patients": I created sub-rows for John Smith's "Allergies" column and merged rows in his other colums. Hope it looks intuitive to you.

ME. Yes, it does.

BA. And here are two tests of the type DoctorVisit:

Data DoctorVisit visits			
date	encounterDiagnosis	recommendedMedication	recommendedDose
Date	**Encounter Diagnosis**	**Recommended Medication**	**Recommended Dose**
2/15/2011	Acute Sinusitis	?	?
2/25/2011	Acute Sinusitis	?	?

I defined a diagnosis but left recommended medication and dose unknown.

ME. Should I always use question marks?

BA. No, you may even omit these two columns: recommended medication and dose should be defined by our decision.

And finally, I need to create a table of the predefined type "DecisionObject" to map business concepts in the Glossary with our test data:

DecisionObject decisionObjects	
Business Concept	**Business Object**
Patient	:= patients[0]
Doctor Visit	:= visits[0]

The second column "Business Object" specifies which data instances should be used for testing. Here we will use the first patient (defined as := patients[0] because arrays start with

an index 0) and the first visit (defined as := visits[0]).

EXECUTING DECISION

BA. Now we are ready to execute the defined decision against the above tests. I will run our decision "DeterminePatientTherapy" by double-clicking on the provided batch file "run.bat". Here are the execution results:

```
Define Medication
Conclusion: Recommended Medication Is Levofloxacin
Define Dosing
Conclusion: Recommended Dose Is 500mg every 24 hours for 14 days
```

Do they look good to you?

ME. Let me see. Our first patient John Smith is 58 and he is allergic to Penicillin. So, the first two rules from the decision table "DefineMedication" cannot be applied but the third rule should recommend `Levofloxacin`. That's good. And now let me look at the dosing rules. John's age falls in to the interval "15-60" so the recommended dose `500mg every 24 hours for 14 days` is also correct.

BA. Congratulations! Do you want to try another patient?

ME. Of course, and probably I'd add more tests.

BA. You may simply modify your table "DecisionObject".

ME. OK, now I will select patients[1] in this table

DecisionObject decisionObjects	
Business Concept	**Business Object**
Patient	:= patients[1]
Doctor Visit	:= visits[0]

Can I run our decision myself?

BA. Of course. Simply use Windows explorer to double-click on the file "run.bat".

ME. Here are my new results:

```
Define Medication
Conclusion: Recommended Medication Is Amoxicillin
Define Dosing
Conclusion: Recommended Dose Is 500mg every 24 hours for 14 days
```

Here is patient Mary Smith. She is 19 with no allergies. So, the very first medication rule correctly recommends `Amoxicillin`. Why is the recommended dose `500mg every 24 hours for 14 days`? Because she still falls in the age category "15-60".

BA. Very good.

ME. Let me go to the data file and change her age from 19 to 65. Here are new results:

```
Define Medication
Conclusion: Recommended Medication Is Amoxicillin
Define Dosing
Conclusion: Recommended Dose Is 250mg every 24 hours for 14 days
```

Only dosing was changed… Of course, 65 is still >= 18 and the first medication rule again recommended `Amoxicillin`. And her creatinine level of 1.8 is more than 1.4 while the creatinine clearance 48.75 is less than 50. So, the second dosing rules produced a different dose `250mg every 24 hours for 14 days`.

BA. I believe that now you are ready to expand our decision yourself. You may add more test cases to your Data table and more rules to your decision tables to provide more complete clinical guidelines.

ME. I certainly can do it.

BA. We still have two more topics to cover:
- Add a formula for creatinine clearance calculation
- Add rules that deal with drug interaction.

Adding Calculation Formulas

BA. Let's add your formula

$$\text{CCr, in mL/min} = \frac{(140 - \text{age}) \times \text{lean body weight [kg]}}{\text{PCr [mg/dL]} \times 72}$$

to our executable decision. First of all, a patient's creatinine clearance CCr depends on the person's age, weight, and creatinine level PCr. So, we have to add weight to our Glossary and test data. Could you modify the proper tables?

ME. Here they are:

Glossary glossary		
Variable	**Business Concept**	**Attribute**
Encounter Diagnosis	Doctor Visit	encounterDiagnosis
Recommended Medication		recommendedMedication
Recommended Dose		recommendedDose
Patient Age	Patient	age
Patient Weight		weight
Patient Allergies		allergies
Patient Creatinine Level		creatinineLevel
Patient Creatinine Clearance		creatinineClearance

Datatype Patient	
String	name
int	age
double	creatinineLevel
double	creatinineClearance
String[]	allergies
double	weight

Data Patient patients					
name	age	allergies	creatinineLevel	creatinineClearance	weight
Name	**Age**	**Allergies**	**Creatinine Level**	**Creatinine Clearance**	**Weight**
John Smith	58	Penicillin Streptomycin	2.00	44.42	78
Mary Smith	65		1.80	48.03	83

But I do not understand where I should add this formula.

BA. We may create another decision table (say "CalculateCreatinineClearance") that defines fact "Patient Creatinine Clearance". OpenRules allows us to write any formulas within decision table cells after "::=". I'd suggest the following implementation of this decision table:

DecisionTable CalculateCreatinineClearance	
Conclusion	
Patient Creatinine Clearance	
Is	::= (140 - getInt("Patient Age")) * getReal("Patient Weight") / (getReal("Patient Creatinine Level") * 72)

As you can see, this decision table has no conditions. I am using the predefined methods getInt(name) and getReal(name) to get values for the fact by their names. Hopefully, the way I wrote this formula is intuitive enough for you too. By the way, I was told that this is a valid Java syntax. So, you and I now may boast that we did some "Java programming" – just kidding!

ME. OK. Can I run the updated decision?

BA. Not yet. We need to tell our decision that this new decision table should be executed. So, I will add one more sub-decision to our decision table:

Decision DeterminePatientTherapy	
Decisions	**Execute Decision Tables**
Define Medication	:= DefineMedication()
Define Creatinine Clearance	:= CalculateCreatinineClearance()
Define Dosing	:= DefineDosing()

It is important that Patient Creatinine Clearance should be calculated before we apply dosing rules that use its value. Now you can run the decision again.

ME. And here are the results:

```
Define Medication
Conclusion: Recommended Medication Is Amoxicillin
Define Creatinine Clearance
Conclusion: Patient Creatinine Clearance Is 48.03240740740741
Define Dosing
Conclusion: Recommended Dose Is 250mg every 24 hours for 14 days
```

There are no changes in the recommended medication and dosing. But now it also calculated `Patient Creatinine Clearance` as 48.03; which is not too different from the 48.75 we had in our test data. Both values are less than 50 and as such satisfy the conditions of the second dosing rule.

BA. Very good. Do you want to add your drug interaction rules yourself?

ADDING DRUG INTERACTION RULES

ME. Here is what I wrote about it:

> Check if patient on active medication. Coumadin and Levofloxacin can result in reduced effectiveness of Coumadin. Produce the proper warning.

So, I suggest that we add one more decision table "WarnAboutDrugInteraction" that should be executed at the very end and generate a warning about drug conflicts.

BA. Good. Just one small suggestion. Along with "condition" and "Conclusion" columns OpenRules allows us to use columns of the predefined type "Message". "Message" columns do not have any operators, do not deal with the glossary, but simply produce messages.

ME. OK, here is my new decision table:

DecisionTable WarnAboutDrugInteraction		
Condition	Condition	Message
Recommended Medication	**Patient Active Medication**	**Warning**
Levofloxacin	Coumadin	Coumadin and Levofloxacin can result in reduced effectiveness of Coumadin.

After double-clicking on "run.bat" I receive… The same results? Of course, because the recommended `Amoxicillin` does not conflict with `Coumadin`.

BA. Not only because of this. Your rule family was never executed – you were too quick and forgot to add it to the table "Decision".

ME. OK, it is easy to fix. Here is the changed table "Decision":

Decision DeterminePatientTherapy	
Decisions	**Execute Decision Tables**
Define Medication	:= DefineMedication()
Define Creatinine Clearance	:= CalculateCreatinineClearance()
Define Dosing	:= DefineDosing()
Check Drug Interaction	:= WarnAboutDrugInteraction()

I will use the first test-patient John Smith again because he was given a recommendation to use Levofloxacin. But I need to show that he already takes Coumadin. So, I am adding an active medication to our test data:

Datatype Patient	
String	name
int	age
double	creatinineLevel
double	creatinineClearance
String[]	allergies
double	weight
String	activeMedication

Data Patient patients						
name	age	allergies	creatinineLevel	creatinineClearance	weight	activeMedication
Name	**Age**	**Allergies**	**Creatinine Level**	**Creatinine Clearance**	**Weight**	**Active Medication**
John Smith	58	Penicillin Streptomycin	2.00	44.42	78	Coumadin
Mary Smith	65		1.80	48.03	83	

Now, if I run the decision again I will receive… What is this???

```
Define Medication
Conclusion: Recommended Medication Is Levofloxacin
Define Creatinine Clearance
Conclusion: Patient Creatinine Clearance Is 44.416666666666664
Define Dosing
Conclusion: Recommended Dose Is 500mg every 24 hours for 14 days
Check Drug Interaction
ERROR in Glossary: cannot find fact <Patient Active Medication>
org.apache.commons.lang.exception.NestableRuntimeException:
java.lang.reflect.InvocationTargetException at
org.openl.util.RuntimeExceptionWrapper.wrap(RuntimeExceptionWrapper.java:
22)
```

BA. Of course, you forgot to add a new fact "Recommended Active Medication" to your glossary. So, the OpenRules engine correctly describes the error:

```
ERROR in Glossary: cannot find fact <Patient Active Medication>
```

ME. That's nice of him. Let me correct the error:

Glossary glossary		
Variable	**Business Concept**	**Attribute**
Encounter Diagnosis	Doctor Visit	encounterDiagnosis
Recommended Medication		recommendedMedication
Recommended Dose		recommendedDose
Patient Age	Patient	age
Patient Weight		weight
Patient Allergies		allergies
Patient Creatinine Level		creatinineLevel
Patient Creatinine Clearance		creatinineClearance
Patient Active Medication		activeMedication

Can I run the decision now?

BA. Be my guest.

ME. This looks much better:

```
Define Medication
Conclusion: Recommended Medication Is Levofloxacin
Define Creatinine Clearance
Conclusion: Patient Creatinine Clearance Is 44.416666666666664
Define Dosing
Conclusion: Recommended Dose Is 500mg every 24 hours for 14 days
Check Drug Interaction
Coumadin and Levofloxacin can result in reduced effectiveness of
Coumadin. from WarnAboutDrugInteraction
```

BA. These are the results we both expected to receive. In the future we would improve this decision further. I plan to contact our IT staff and work with them on the integration of this decision with their actual Java application. Meanwhile, I will make sure that you can share project "DecisionPatientTherapy" with your colleagues by clicking here.

ME. It will be great. Let's call it a day. Thank you. I really enjoyed our team work!

BA. Me too. Now we can officially say that our first OpenRules session is complete. It was my pleasure to work with you today as well!

Next time I will explain you how to get Patient information from a database – see http://openrules.com/pdf/Tutorial.DecisionPatientTherapyDB.pdf.