Building Domain-Specific Decision Models

Jacob Feldman, PhD
OpenRules, Inc., CTO
www.openrules.com

San Jose, Oct 13-15, 2014

Outline

- Transformation from Domain-specific programming languages (DSL) to **business-oriented Domain-Specific Decision Models**
  - Motivation
  - General Approach
- Real-world examples of domain-specific models
  - Decision Modeling in Constraint Programming
  - Decision Modeling in Geospatial Domain
  - Decision Modeling for Dynamic Web Interaction

Domain-Specific Decision Models

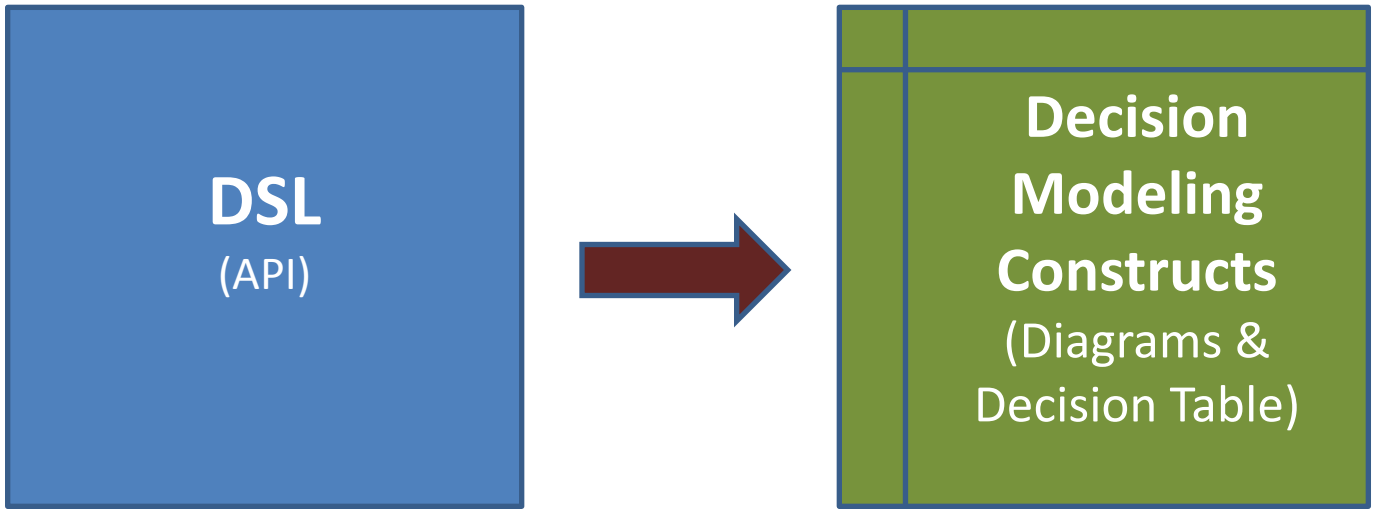
- **Motivation**
  - There are many solid Java (C++,…) libraries that support a Domain-Specific Languages (DSL) with their own Java API, e.g. 350+ JSRs
  - DSLs provide specialized concepts and methods for a particular problem domain
  - DSLs are oriented to software developers
- However, we cannot offer such APIs to business users: they need some kind of a "**Business DSL**"!

General Purpose vs Domain-Specific

- **General Purpose BRMS** is similar to General Purpose Language (e.g. Java, C++):
  - **Forces its users to define all concepts, relationships, and methods from scratch**
  - Real Value of Java is in surrounding libraries not in language;
  - What is an analogy for BRMS?
- **Domain-Specific Decision Model** is similar to DSL:
  - **Provide specialized concepts, relationships, and methods for a particular problem domain**
  - A subject matter expert feels "at home"
  - Real Value in modeling facilities not in an engine

Domain-Specific Decision Models

- **Objective**
  - Given: a domain-specific language with an API
  - Goal: create a domain-specific decision modeling "language" oriented to business users



Moving From Java to Business Decision Modeling

- Building business-oriented decision modeling facilities on top of domain-specific Java APIs
- Hiding object-orientation, complex embedding, and pure programming constructs
- No new classes but rather a business glossary connected to existing object
- Transforming long chains of if-then-else or switch-statements to easy understood and maintained decision tables

General Purpose Decision Modeling

- OpenRules provides a powerful **templatization mechanism** that allowed us to implement generic decision modeling approaches such as DMN and TDM
- The standard templates are provided in Excel and allow a business user to create concrete Excel tables of the type Decision, DecisionTable, DecisionTableMultiHit, Glossary, and more
- No coding is required – users just construct tables using predefined column types

Business-oriented Decision Table and its Technical Template

DecisionTable DefineSalutation

Condition		Condition		Condition		Conclusion	
Gender		**Marital Status**		**Date of Birth**		**Salutation**	
Is	Male					Is	Mr.
Is	Female	Is	Married	<	January 1, 2007	Is	Mrs.
Is	Female	Is	Single			Is	Ms.
				>=	January 1, 2007	Is	Little

Rules String DecisionTableTemplate(Decision decision)

[Condition]		[If]	[Conclusion]		[Action]	[Then]	[Ac
decision.compare($COLUMN_TITLE, op, value);		decision.compare($COLUMN_TITLE, expression);	if (decision.isTraceOn()) decision.log("Conclusion: " + $COLUMN_TITLE + " " + op.toString() + " " + value); decision.assign($COLUMN_TITLE,op,value); return null;		decision.assign($COLUMN_TITLE,value); if (decision.isTraceOn()) decision.log($COLUMN_TITLE + ": " + value); return null;	if (decision.isTraceOn()) decision.log($COLUMN_TITLE + " := " + value); decision.assign($COLUMN_TITLE,value); return null;	if (method != n //decision.log method); decision.exe } return null;
Oper op	String value	String expression	Oper op	String value	String value	String value	St
Decision Variable		Variable	Decision Variable		Decision Variable	Decision Variable	Execute

From DSL to Decision Models

DSL
(API)

1. Domain-Specific Classes
   - Classes/Sub-classes
   - Attributes
2. Domain-Specific Classes
   - Methods
   - If/Then/Else
   - Loops
   - …
3. Navigating through graphs and collections

Decision Modeling Constructs
(Diagrams & Decision Table)

1. Domain-Specific Glossary
   - Business Concepts
   - Decision Variables
2. Domain-Specific Decision Tables
   - Conditions
   - Actions
   - Hit Policies
   - Aggregation
   - …
3. Process Diagrams

DSL Examples

- Decision Modeling in Constraint Programming
- Decision Modeling in Geospatial Domain
- Decision Modeling for Dynamic Web Interaction

DSL Example
"Constraint Programming API"

- JSR-331 "Java Constraint Programming API" – an official Java Community Process (JCP) standard www.jcp.org
- JSR-331 covers key concepts for representation and resolution of constraint satisfaction and optimization problems:
  - Class **Var** for constraint variables and all operators on them
  - Class Constraint for different constraints such as "var1 I greater than var2" or "all variables in the array are different"

Use Case "Staff Rostering"

- As the manager, you are required to hire and set the weekly schedule for your employees as follows:
  - Total employees required

Mon	Tue	Wed	Thu	Fri	Sat	Sun
5	8	9	10	16	18	12

  - Available employees:

Employee Type	Total	Cost per Day
F/T	14	$100
P/T	4	$150

- What is the minimal staffing cost?

	M	T	W	T	F	S	S
FT	5	8	9	10	14	14	12
PT	0	0	0	0	2	4	0

Solution using Java API (1)

```
Problem p = ProblemFactory.newProblem("EmployeeRostering1");
// Define FT and PT variables
int maxFT = 14;
int maxPT = 4;
Var monFT = p.variable("MonFT", 0, maxFT);
Var monPT = p.variable("MonPT", 0, maxPT);
Var tueFT = p.variable("TueFT", 0, maxFT);
Var tuePT = p.variable("TuePT", 0,maxPT);
Var wedFT = p.variable("WedFT", 0, maxFT);
Var wedPT = p.variable("WedPT", 0, maxPT);
Var thuFT = p.variable("ThuFT", 0, maxFT);
Var thuPT = p.variable("ThuPT", 0, maxPT);
Var friFT = p.variable("FriFT", 0, maxFT);
Var friPT = p.variable("FriPT", 0, maxPT);
Var satFT = p.variable("SatFT", 0, maxFT);
Var satPT = p.variable("SatPT", 0, maxPT);
Var sunFT = p.variable("SunFT", 0, maxFT);
Var sunPT = p.variable("SunPT", 0, maxPT);
```

Solution using Java API (2)

```
// Post daily constraints
p.post(monFT.plus(monPT),"=",5);    // monFT + monPT = 5
p.post(tueFT.plus(tuePT),"=",8);
p.post(wedFT.plus(wedPT),"=",9);
p.post(thuFT.plus(thuPT),"=",10);
p.post(friFT.plus(friPT),"=",16);
p.post(satFT.plus(satPT),"=",18);
p.post(sunFT.plus(sunPT),"=",12);
```

```
// Define costs
int[] costs = {100,150,100,150,100,150,100,150,100,150,100,150,100,150};
Var[] vars =
{monFT,monPT,tueFT,tuePT,wedFT,wedPT,thuFT,thuPT,friFT,friPT,satFT,satPT,sunFT,sunPT};
Var totalCost = p.scalProd(costs, vars);
p.add("TotalCost",totalCost);
```

Solution using Decision Model (1)

```
Problem p = ProblemFactory.newProblem("EmployeeRostering1");
// Define FT and PT variables
int maxFT = 14;
int maxPT = 4;
Var monFT = p.variable(
Var monPT = p.variable(
Var tueFT = p.variable("
Var tuePT = p.variable("
Var wedFT = p.variable(
Var wedPT = p.variable(
Var thuFT = p.variable("
Var thuPT = p.variable("
Var friFT = p.variable("Fr
Var friPT = p.variable("Fr
Var satFT = p.variable("S
Var satPT = p.variable("S
Var sunFT = p.variable("
Var sunPT = p.variable("
```

Glossary glossary

Decision Variable	Business Concept	Attribute	Domain
Mon FT	Roster	monFT	0-14
Mon PT		monPT	0-4
Tue FT		tueFT	0-14
Tue PT		tuePT	0-4
Wed FT		wedFT	0-14
Wed PT		wedPT	0-4
Thu FT		thuFT	0-14
Thu PT		thuPT	0-4
Fri FT		friFT	0-14
Fri PT		friPT	0-4
Sat FT		satFT	0-14
Sat PT		satPT	0-4
Sun FT		sunFT	0-14
Sun PT		sunPT	0-4
Total Cost		totalCost	0-20000

Solution using Decision Model (2)

```
// Post daily constraints
p.post(monFT.plus(monPT),"=",5);    // monFT + monPT = 5
p.post(tueFT.plus(tuePT),"=",8);
p.post(wedFT.plus(wedPT),"=",9);
p.post(thuFT.plus(thuPT),"=",10);
p.post(friFT.plus(friPT),"=",16);
p.post(satFT.plus(satPT),"=",18);
p.post(sunFT.plus(sunPT),"=",12);
```

Mon	Tue	Wed	Thu	Fri	Sat	Sun
5	8	9	10	16	18	12

DecisionTable EmployeeDailyDemand

ActionXoperYcompareZ

Variable	Arith Oper	Variable	Compare Oper	Value
Mon FT	+	Mon PT	=	5
Tue FT	+	Tue PT	=	8
Wed FT	+	Wed PT	=	9
Thu FT	+	Thu PT	=	10
Fri FT	+	Fri PT	=	16
Sat FT	+	Sat PT	=	18
Sun FT	+	Sun PT	=	12

Solution using Decision Model (3)

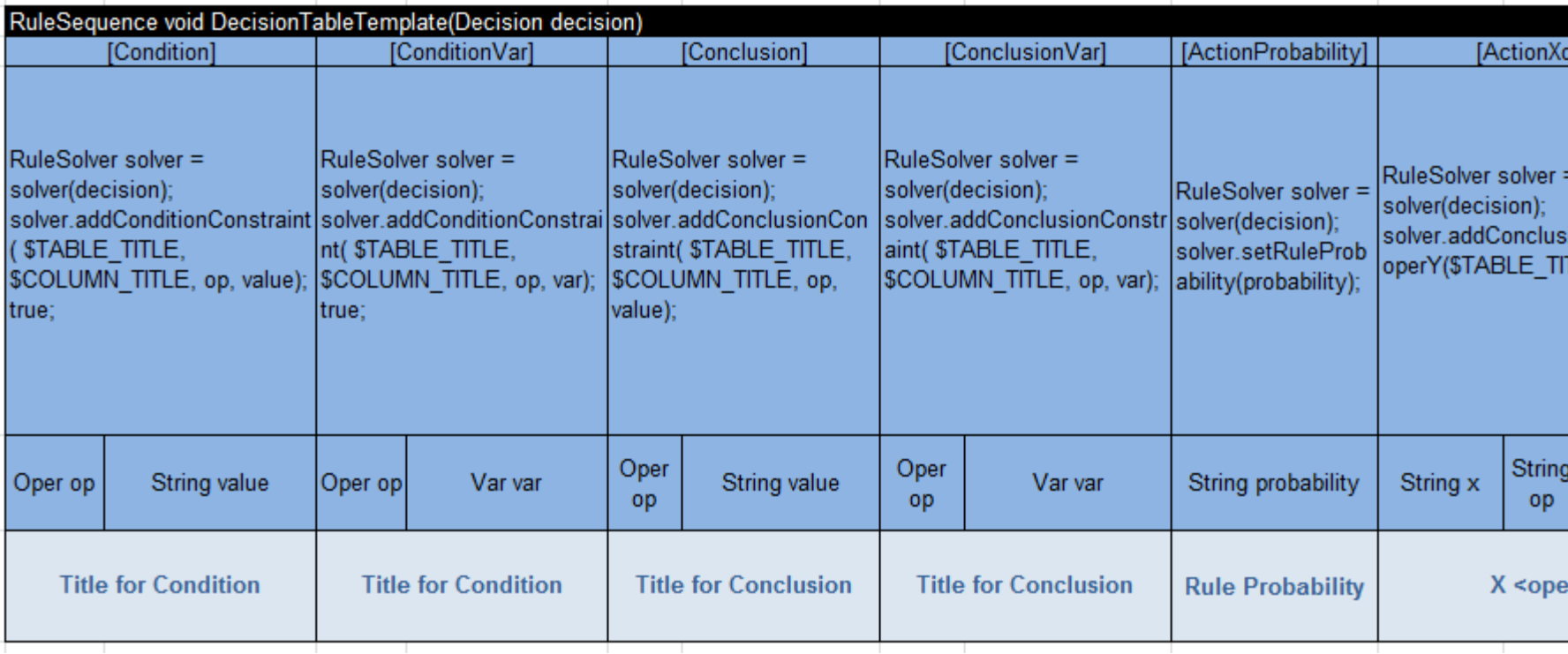
```
// Define costs
int[] costs = {100,150,100,150,100,150,100,150,100,150,100,150,100,150};
Var[] vars =
{monFT,monPT,tueFT,tuePT,wedFT,wedPT,thuFT,thuPT,friFT,friPT,satFT,satPT,sunFT,sunPT};
Var totalCost = p.scalProd(costs, vars);
p.add("TotalCost",totalCost);
```

Employee Type	Total	Cost per Day
F/T	14	$100
P/T	4	$150

DecisionTable DefineTotalCost		
ActionScalProd		
Name of the Scalar Product	**Numbers**	**Variables**
Total Cost	100,150,100,150,100,150, 100,150,100,150,100,150, 100,150	Mon FT, Mon PT, Tue FT, Tue PT, Wed FT, Wed PT, Thu FT, Thu PT, Fri FT, Fri PT, Sat FT, Sat PT, Sun FT, Sun PT

How did we moved from Java to Decision Tables?

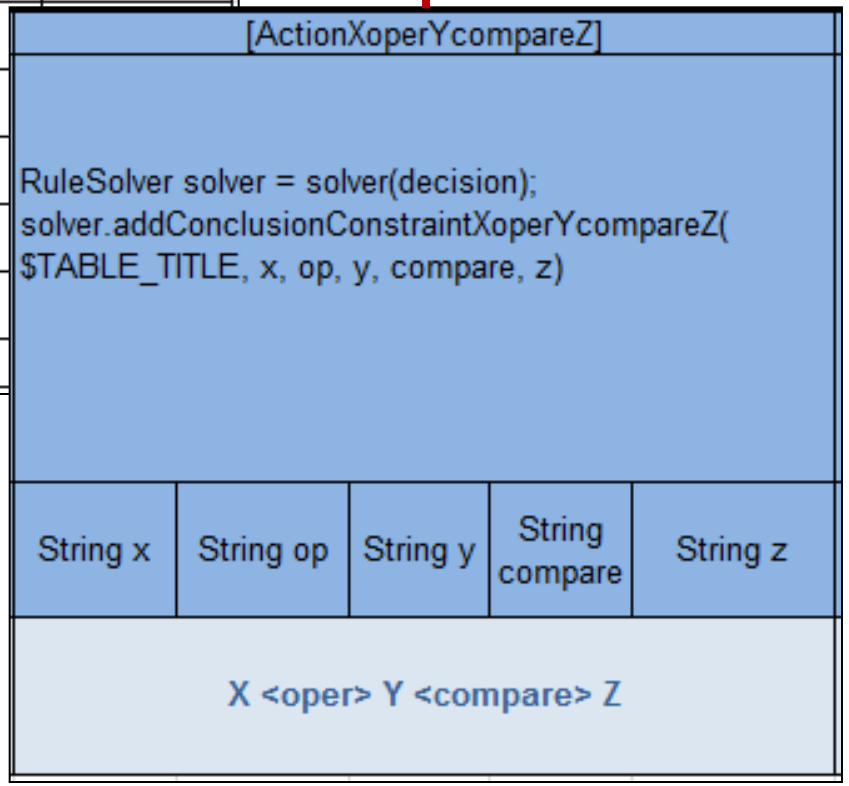
- Providing business-oriented decision modeling facilities on top of the JSR-331
- We utilized the OpenRules Decision Table Templatization Mechanism



RuleSequence void DecisionTableTemplate(Decision decision)											
[Condition]		[ConditionVar]		[Conclusion]		[ConclusionVar]		[ActionProbability]	[ActionXo		
RuleSolver solver = solver(decision); solver.addConditionConstraint($TABLE_TITLE, $COLUMN_TITLE, op, value); true;		RuleSolver solver = solver(decision); solver.addConditionConstraint($TABLE_TITLE, $COLUMN_TITLE, op, var); true;		RuleSolver solver = solver(decision); solver.addConclusionConstraint($TABLE_TITLE, $COLUMN_TITLE, op, value);		RuleSolver solver = solver(decision); solver.addConclusionConstraint($TABLE_TITLE, $COLUMN_TITLE, op, var);		RuleSolver solver = solver(decision); solver.setRuleProbability(probability);	RuleSolver solver = solver(decision); solver.addConclusoperY($TABLE_TI		
Oper op	String value	Oper op	Var var	Oper op	String value	Oper op	Var var	String probability	String x	String op	
Title for Condition		Title for Condition		Title for Conclusion		Title for Conclusion		Rule Probability	X <ope		

Above Example

Decision Table EmployeeDailyDemand				
ActionXoperYcompareZ				
Variable	Arith Oper	Variable	Compare Oper	Value
Mon FT	+	Mon PT	=	5
Tue FT	+	Tue PT	=	
Wed FT	+	Wed PT	=	
Thu FT	+	Thu PT	=	
Fri FT	+	Fri PT	=	
Sat FT	+	Sat PT	=	
Sun FT	+	Sun PT	=	



- Templates are created by programmers
- End users (decision table designers) never look at the code

How to create a Domain-Specific Decision Modeling Framework?

1. Start with a domain-specific language (Java API)
2. Design Decision Tables oriented to a subject matter expert in this particular domain
3. Implement new decision templates with conditions and actions that support such domain-specific decision tables
    - Usually it requires a creation of another layer of Java that hides the complexity of the original API
    - For example, OpenRules developed a special component (Rule Solver) built on the top of the JSR-331 standard

DSL Examples

- Decision Modeling in Constraint Programming
- **Decision Modeling in Geospatial Domain**
- Decision Modeling for Dynamic Web Interaction

Decision Modeling Framework for Geospatial Applications

- Domain: **Geospatial Applications**
  - Deals with objects and algorithms for processing linear geometries (points, lines, areas, etc.) on the 2-dimensional Cartesian plane
  - A real-world application will be described by Alex Karman (Revolutionary Machines)

JTS (Java Topology Suite)

- JTS is a de-facto standard open source Java library with a complete, consistent, and robust Java API
- It covers various spatial predicates and operators (see on the right):
- But JTS is too complex for business users

Equals
Disjoint
Touches
Contains
Covers
Intersects
Within
Covered By
Crosses
Overlaps

Domain-Specific Business Expressions

- Typical business conditions expressed by end users as:

HRR has at least 5 HSAs in it
HRR has fewer than 5 HSAs in it
HRR overlaps at least two Counties
At least one hospital is within 5 km from the Airport
Between 5 and 15 Hospitals are > 25 km from the Airport
No Hospital is within 10 km from the airport
More than 2 hospitals within 20 km from the Residence
More than 1200 residences within 20 km from the Hospital

HRR – Hospital Referral Region

HSA – Hospital Service Area

- How to present such rules in a “human” way that still can be executed by a computer?

Decision Modeling Exa

New Custom Type of Decision Tables

New Custom Condition

JTS Operators

New Custom Expression (mini DSL)

DecisionTableSpatialEntityToEntityRules							
C#	ConditionEntityToEntity					Conclusion	
#	Relationship Between Two Entities					Score	
	Main Entity Type	Relationship	Related Entity	Oper	Value		
0						=	0
1	HRR	Contains		Is	TRUE	+=	2
2	HRR	Touches	HSA	Is	TRUE	+=	1
3	HRR	Is Disjoint From	HSA	Is	FALSE	+=	3
4	HRR+5	Contains	HSA	Is	TRUE	+=	1
5	Route	Crosses	HSA	Is	TRUE	+=	2
6	HRR	Overlaps	County	Is	TRUE	-=	2
7	Airport	Distance	Hospital	<	250	+=	1
8	HRR	Area		<	25	+=	1
9	County	Area		<	25	+=	1
10	HRR	Is Among 25 Closest To	Hospital	Is	TRUE	+=	1
11	Facility	Is Part Of	University	Is	TRUE	+=	1
12	University	Comprises	ility	Is	TRUE	+=	4

Implementation Template

DecisionTableSpatialEntityToEntityRules

C#	ConditionEntityToEntity					Conclusion
	Relationship Between Two Entities					
#	Main Entity Type	Relationship	Related Entity Type	Oper	Value	**Spatial Significance Score**
0						
1	HRR	Contains				
2	HRR	Touches				
3	HRR	Is Disjoint From				
4	HRR+5	Contains				
5	Route	Crosses				
6	HRR	Overlaps				
7	Airport	Distance				
8	HRR	Area				
9	County	Area				
10	HRR	Is Among 25 Closest To				
11	Facility	Is Part Of				
12	University	Comprises				

Rules void DecisionTableSpatialTemplate(Decision decision)

[ConditionEntityToEntity]

GeoDatabase.conditionEntityToEntity (decision,mainEntityType, relationship, relatedEntityType, oper, value);

String mainEntityType	String relationship	String relatedEntityType	String oper	String value

Relationship Between Two Entities

Main Entity Type	Relationship	Related Entity Type	Oper	Value

DSL Examples

- Decision Modeling in Constraint Programming
- Decision Modeling in Geospatial Domain
- **Decision Modeling for Dynamic Web Interaction**

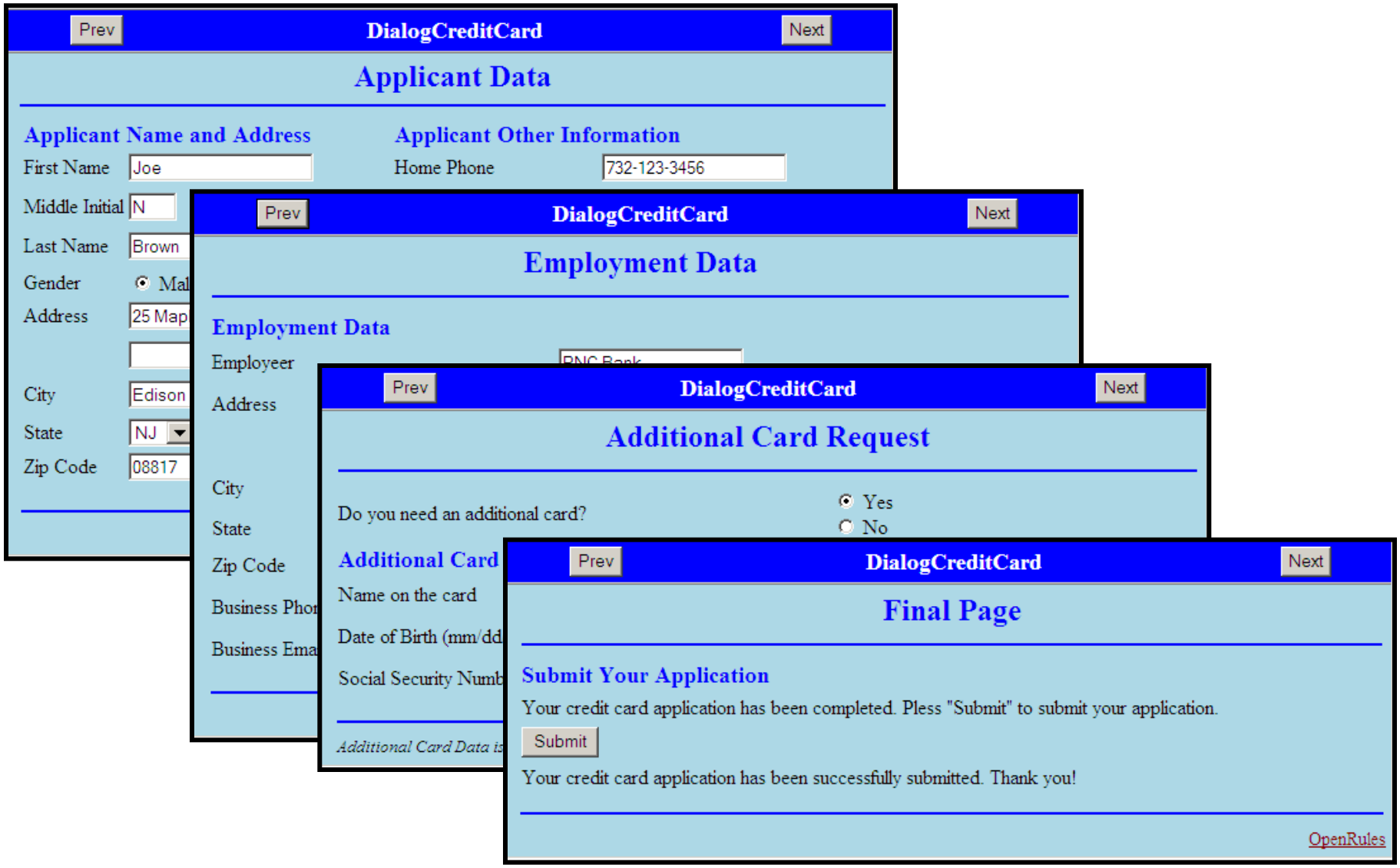
Decision Modeling Framework for Web Questionnaires

- Domain: **Dynamic Web Applications** with complex interaction logic
  - Allows non-technical people to develop and maintain web-based **questionnaires** (dialogs) using only Excel and without a need to learn complex web programming techniques
  - A real-world web application for a large US bank will be described tomorrow by Erik Marutian

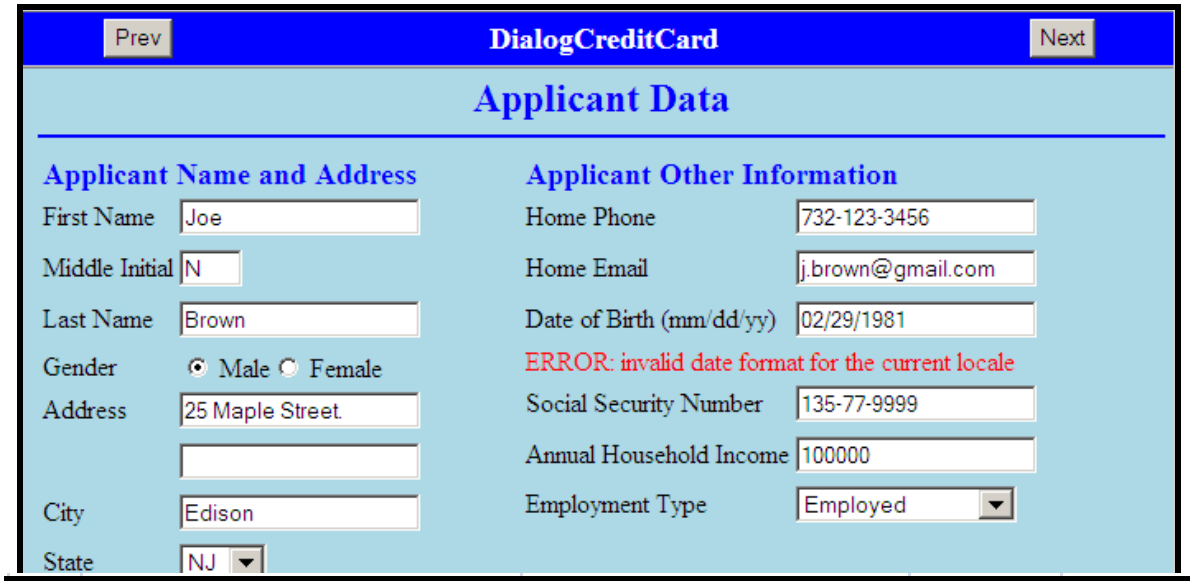
OpenRules Dialog™

- A questionnaire is described by a subject matter expert in terms of **Pages**, **Sections**, and different types of **Questions** and **Answers**
- **Layouts** of pages, sections, questions, and complex **relationships** between them can be expressed in a very natural way using simple and intuitive Excel tables.

Example: Credit Card Application



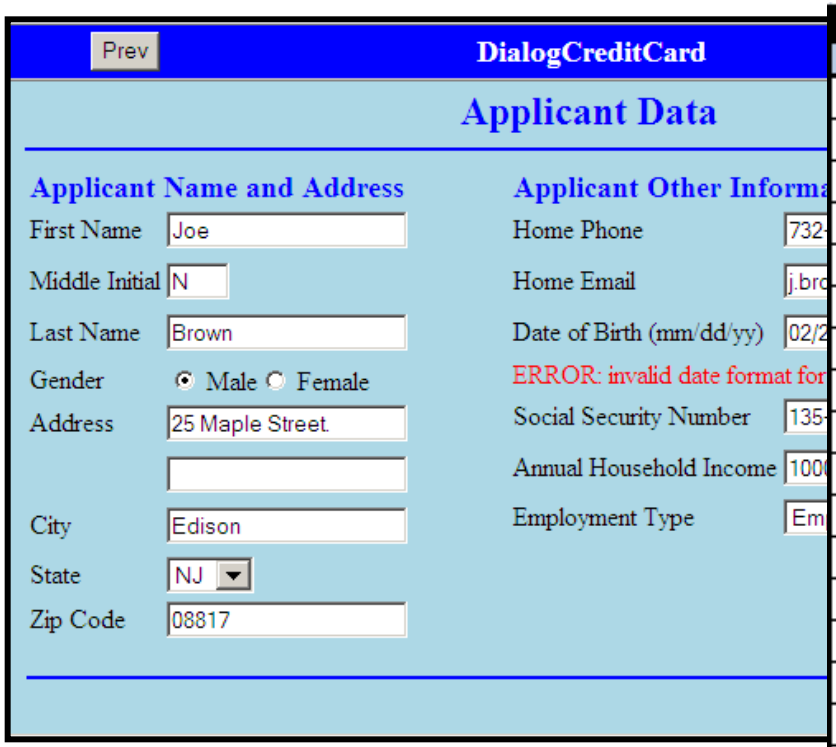
Pages



Rules pages extends pagesTemplate

#	Page ID	Page Name	Hidden	Section Column 1	Section Column 2
1	ApplicantData	Applicant Data		ApplicantNameAddress	ApplicantOtherInfo
2	ApplicantEmployment	Employment Data		EmploymentData	
3	AdditionalCardRequest	Additional Card Request		RequestAdditionalCard	
4				Additional Card Data	
5	FinalPage	Final Page		FinalSection	

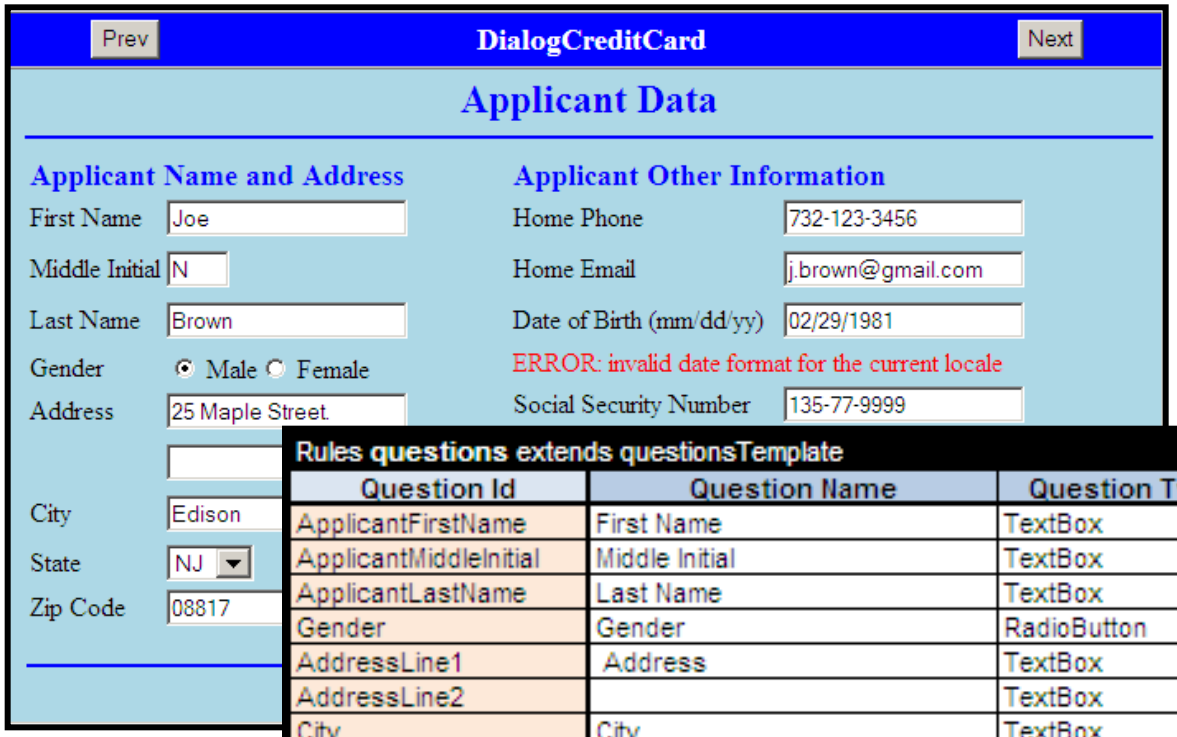
Sections



Rules sections extends sectionsTemplate

#	Section ID	Section Name	Hidden	Question Column 1
1	ApplicantNameAddress	Applicant Name and Address		ApplicantFirstName
1				ApplicantMiddleInitial
1				ApplicantLastName
1				Gender
1				AddressLine1
1				AddressLine2
1				City
1				State
1				ZipCode
2	ApplicantOtherInfo	Applicant Other Information		HomePhone
2				HomeEmail
2				DOB
2				SSN
2				Income
2				EmploymentType
3	EmploymentData	Employment Data		Employeer
3				BusinessAddressLine1
3				BusinessAddressLine2
3				BusinessCity
3				BusinessState
3				BusinessZipCode
3				BusinessPhone
3				BusinessEmail
4	RequestAdditionalCard			NeedAdditionalCard
4	Additional Card Data	Additional Card Data	Yes	NameOnTheCard
4				AdditionalDOB
4				AdditionalSSN
5	FinalSection	Submit Your Application		Completed
5				SubmitApplication
5				Submitted

Questions



Rules questions extends questionsTemplate

Question Id	Question Name	Question Type	Size	Hidden	Validation
ApplicantFirstName	First Name	TextBox			
ApplicantMiddleInitial	Middle Initial	TextBox	2		
ApplicantLastName	Last Name	TextBox			
Gender	Gender	RadioButton			
AddressLine1	Address	TextBox			
AddressLine2		TextBox			
City	City	TextBox			
State	State	ComboBox			
ZipCode	Zip Code	TextBox			Range 1 99999
HomePhone	Home Phone	TextBox			REGEX [0-9]{3}-[0-9]{3}-[0-9]{4}
HomeEmail	Home Email	TextBox			EMAIL
DOB	Date of Birth (mm/dd/yy)	TextBox			DATE
SSN	Social Security Number	TextBox			SSN
Income	Annual Household Income	TextBox			Range 10000 10000000
EmploymentType	Employment Type	ComboBox			
NeedAdditionalCard	Do you need an additional card?	RadioButtonVertical Submit			
NameOnTheCard	Name on the card	TextBox			
AdditionalDOB	Date of Birth (mm/dd/yy)	TextBox			DATE
AdditionalSSN	Social Security Number	TextBox			SSN
Employeer	Employeer	TextBox			
BusinessAddressLine1	Address	TextBox			

Navigation Rules

Rules **navigateRules** extends navigateDialogTemplate					
C1	C2	C3			A1
IF Current Page is	**AND Action is**	**AND**			**THEN Go to Page**
		Answer to Question	**Is or IsNot**	**Value**	
ApplicantData	Next	EmploymentType	IsNot	Employed	AdditionalCardRequest

Rules navigateRules extends navigateDialogTemplate							
C1	C2	C3			C4	A1	A4
IF Current Page is	**AND Action is**	**AND**			**AND Condition is true**	**THEN Go to Page**	**AND Set Dialog Status**
		Answer to Question	**Is or IsNot**	**Value**			
TaxpayerGeneralInformation	Next	MarriedFillingJointly	IsNot	Yes		IncomeData	
IncomeData	Prev	MarriedFillingJointly	IsNot	Yes		TaxpayerGeneralInformation	

IncomeData	Next				{
					double income = d.getDoubleAnswer("TaxableIncome");
					income >=50000;
					}
						IncomeData	SORRY, YOU CAN NOT USE 1040EZ FORM (your taxable income should be less than $50,000)

IncomeData	Next				{
					double interest = d.getDoubleAnswer("TaxableInterest");
					interest >= 1500;
					}
						IncomeData	SORRY, YOU CAN NOT USE 1040EZ FORM (your taxable interest > $1,500)

Page Update Rules

Rules updateRules extends updateDialogTemplate									
C1	C2	C3			A1		A2		A4
IF Current Page is	AND Action is	AND			THEN Hide/Show Question		AND Hide/Show Section		AND Set Dialog Status
		Answer to Question	Is or IsNot	Value	Hide	Question	Hide	Section	
AdditionalCardRequest		NeedAdditionalCard	Is	No			Hide	Additional Card Data	Additional Card Data is hidden
AdditionalCardRequest		NeedAdditionalCard	Is	Yes			Show	Additional Card Data	Additional Card Data is shown
FinalPage	Submit				Show	Submitted			

Answers

Data Answers answers		
Question Id	**Default Answer**	**Possible Answers**
EmploymentType	Employed	employmentTypes
NeedAdditionalCard	No	yesno
State	MA	USstates
BusinessState	MA	USstates
ZipCode	11371	
Income	100000	
BusinessZipCode	11371	
Gender	Female	

Data PossibleAnswers possibleAnswers	
id	choices
ID	**Answers**
employmentTypes	Employed
	Unemployed
	Full Time Student
	Retired
gender	Male
	Female
yesno	Yes
	No
USstates	AL
	AK
	AR
	AZ
	CA

Auto-Responses

Rules **autoResponses** extends autoResponsesTemplate	
Question Id	**Auto Response**

AdjustedGrossIncome	{
	double wages = d.getDoubleAnswer("Wages");
	double taxableInterest = d.getDoubleAnswer("TaxableInterest");
	double unemploymentCompensation = d.getDoubleAnswer("UnemploymentCompensation");
	double answer = wages + taxableInterest + unemploymentCompensation;
	format(answer);
	}

A	{
	double wages = d.getDoubleAnswer("Wages");
	format(wages + 250);
	}

Many More Domain-Specific Features

- Custom Layouts for Pages, Sections, Questions
- Problem-specific question properties
- Support for child-parent relationships
  - E.g. Hide all children of the question
- Saving/Loading dialog to/from external sources
- Embedded Support for HTML, Style Sheets, and Java Scripts

OPEN RULES

Conclusion

Programmers	Business Analysts
GPL (such as Java)	BRDMS (DMN)
JSR-331	Rule Solver
JTS (Java Topology Suite)	Spatial Decision Templates
Web App Development Techniques	Rule Dialog
DSL (Domain-specific APIs)	Domain-specific Decision Models

Q&A

Web: www.OpenRules.com

Email:

support@openrules.com

jacobfeldman@openrules.com