Using dynamic domains for flexible rule authoring in WebSphere Operational Decision Management

Ensuring rule accuracy is an important part of rule development in WebSphere Operational Decision Management 7.5. This tutorial demonstrates how a dynamic domain populated from an Excel file can be used to reduce inaccuracies when authoring rules. Using domains also eliminates dependencies on IT support for code changes because business users can extend the set of values in the domain by simple edits to the spreadsheet in the Decision Center. Further, because adding a domain value does not result in changes to service contracts, redeployment of rule services is not necessary.

INTRODUCTION	4
WHAT ARE DYNAMIC DOMAINS?	
DADT 1 CDEATING A DIVIANIC DOMAIN	
PART I. CREATING A DYNAMIC DOMAIN	0
STEP 1: IMPORT PACKAGE AND REVIEW RULES	6
The HelloRequest and HelloResponse classes	
STEP 2: CREATE A DOMAINS.XLSX FILE DEFINING THE DOMAIN	
STEP 5: CREATE NEW VIRTUAL BOINTENTRY	11
STEP 5: CREATE THE DOM CLASS CALLED LANGUAGET THE CLASS	
STEP 6: CREATE A NEW MEMBER OF HELLOREQUEST CLASS	17
STEP 7: MODIFY THE CORRESPONDING RULES	22
STEP 8: RERUN THE TEST	23
STEP 9: ADD NEW VALUES TO THE SPREADSHEET	24
STEP 10: PUBLISH TO DECISION CENTER	
PART 2. MODIFYING DYNAMIC DOMAINS IN THE DECISION CENTER	28
STEP 1: LOG INTO THE DECISION CENTER	28
STEP 2: CREATE RESOURCES SMART FOLDER	30
STEP 3: MODIFY DOMAINS.XLSX IN THE RESOURCES FOLDER	
STEP 4: RELOAD DYNAMIC DOMAINS	
STEP 5: ADD THE NEW VALUE TO THE DECISION TABLE	
CONCLUSION	
Figure 1. Specifying a string when defining a rule	5
Figure 2. Selecting a language from a predefined list	5
Figure 3. Selecting an import source	6
Figure 4. Selecting the archive file to import	7
Figure 5. The HelloRequest and Hello Response classes	
Figure 6. The Business Object Model (BOM)	9
Figure 7. The Greetings decision table	9
Figure 8. Decision Validation Services test scenarios	
Figure 9. Results of the DVS test execution	10
Figure 10. The domains.xlsx file	
Figure 11. Creating a new BOM entry	
Figure 12. Creating the LanguageType class	
Figure 13. The LanguageType class	
Figure 14. Creating a domain for the class	14
Figure 15. Selecting the domain type (Excel)	14
Figure 16. Mapping the columns of the Excel file	
Figure 17. The domain values	

riguit 17.		. 10
Figure 18.	Mapping the BOM class to the XOM	. 17
Figure 19.	Removing the original verbalization for the HelloRequest class	. 17
Figure 20.	Creating the languageVirtual member	. 18
Figure 21.	Selecting the languageVirtual member	19
Figure 22.	Creating a default verbalization for languageVirtual	19
Figure 23.	Editing the verbalization	20
U	e	

Figure 24.	Editing the verbalization term	20
Figure 25.	The final verbalization	21
Figure 26.	Defining the getter and setter for the languageVirtual member	21
Figure 27.	The existing rule contains an invalid type (string)	22
Figure 28.	Selecting 'Edit Condition Column'	22
Figure 29.	Replacing string with LanguageType	22
Figure 30.	The modified decision table	23
Figure 31.	Setting up and running the DVS test	24
Figure 32.	Adding new values in the Excel file	24
Figure 33.	Synchronizing new domain values with the class	25
Figure 34.	Updating the decision table	25
Figure 35.	Starting the sample server	26
Figure 36.	Configuring the connection to the Decision Center	26
Figure 37.	Connecting to the Decision Center	27
Figure 38.	A successful synchronization with Decision Center	28
Figure 39.	Logging into the Decision Center	28
Figure 40.	Selecting a project in Decision Center	28
Figure 41.	Accessing the Greetings decision table in Rule Designer	29
Figure 42.	The Greetings decision table	29
Figure 43.	The Create Smart Folder icon	30
Figure 44.	Creating a Resources folder	30
Figure 45.	Specifying a Query for the Resources folder	31
Figure 46.	The domains.xlsx file in the Resources folder	31
Figure 47.	Downloading the domains.xlsx file	32
Figure 48.	Adding a new rows to the Excel file	32
Figure 49.	Editing the link to the domains.xlsx file	32
Figure 50.	Uploading the new domains.xlsx file	33
Figure 51.	The uploaded domains.xlsx file	33
Figure 52.	Reloading the dynamic domain	34
Figure 53.	Selecting the Greetings decision table	34
Figure 54.	Editing the decision table	34
Figure 55.	Selecting the new values in the Languages column	35
Figure 56.	Entering the translated values in the Greeting column	35
Figure 57.	Using the new values in rule definitions	36

Introduction

This article demonstrates how to create and modify a dynamic domain when working with business rules in WebSphere Operational Decision Management V7.5. An example rule is defined to expect a value of type string. The rule is then converted so that the expected values are constrained by an enumerated domain. The domain is dynamically populated by an Excel file. This article provides steps for converting and testing the rule in Rule Designer, and shows how the domain can be modified or extended by a business user in the Decision Center.

This article assumes the following prerequisites:

- Familiarity with rule design and development in WebSphere Operational Decision Management V7.5
- Practical experience with Rule Designer
- A successful installation of IBM WebSphere Operational Decision management V7.5
- An installed copy of Microsoft Excel 2003 or 2007

What are dynamic domains?

WebSphere Operational Decision Management V7.5 supports the use of a variety of domains for rule development. A domain places a restriction on type elements in the Business Object Model (BOM). WebSphere Operational Decision Management supports static, dynamic, enumerated, and complex domains. For example, enumerated domains can be used in rule editing to present valid choices to the rule author. A dynamic domain is an enumerated domain that makes use of values in an Excel file or other data source. The domain is dynamically populated from the data source. This provides the flexibility to make on-the-fly changes and then synchronize the domain and the data source.

Why use dynamic domains?

Dynamic domains offer a number of advantages to both rule developers and business users:

- By constraining a business user's selection to a predefined set of values, rule developers can help prevent errors.
- Dynamic domains can be easily extended or modified by business users, whereas static enumerations require IT support for changes.
- Adding a value to a dynamic domain does not require a change to a rule service contract.

Preventing errors. An important best practice in rule design is creating rules that ensure accuracy and consistency when they are used. Enforcing consistency upfront can prevent downstream validation errors during rule execution. For example, suppose that one

clause of a rule expects the name of a language, specified as a string. The rule might state that if the language of a request is English, the response is to be transmitted in that language. Other language strings expected might be "German" or "Italian", but unless the string is entered accurately the rule is invalid.

Figure 1. Specifying a string when defining a rule

```
Content

[definitions]

if

the language of the request [±] is ▼ <a string> [±] ×

then

<select an action> ×

[else]
```

During rule authoring, business users can accidentally misspell the string or use a mix of upper and lower cases. An exact match on the string is necessary for successful rule execution; user errors will result in the rule misfiring or not firing at all.

To tighten the constraints on a rule so that errors are less likely, a string can be replaced with an enumerated list. This improves rule accuracy but extensions to the list require additional coding. This can cause a delay while business users wait for the change to be implemented, tested, and released for use.

The recommended best practice is to represent the expected values in a dynamic domain. A dynamic domain is a predefined set of values that can be easily extended by business users without the need for an additional cycle of development and test time. In this example, a domain is used to constrain the choices to a predefined list of languages for selection.

Figure 2. Selecting a language from a predefined list

Content	
[definitions] if the language of i then <select action="" an=""> × [else]</select>	s <u></u> Engush French German Hindi Italian
Cancel Previous	Portuguese Spanish the language of

Freeing business users to extend valid parameters for rules. If an Excel file is the underlying data source of a domain, business users authoring rules in the Decision Center can easily add or remove values. A change can be introduced by making a simple edit to the spreadsheet and then updating the corresponding decision table. This flexibility removes dependencies on developers to make required code changes and can result in reduced time for changes to be deployed.

Eliminating the need for changes to service contracts. An important benefit of using domains instead of static enumerations, apart from reduced dependency on developers, is that adding a domain value does not result in changes to service contracts. Therefore developers do not need to redeploy the rule service or change the service client(s). Conversely, when using dynamic domains the service contract is less self-descriptive as it does not describe the possible domain values. The values would need to be communicated to the service client outside of the contract.

Part 1. Creating a dynamic domain

This section introduces the example rule project and describes the steps needed to convert a rule so that the expected parameter uses the enumerated type LanguageType instead of the type string.

Step 1: Import package and review rules

- 1. Open the Rule Designer. From the File menu, select Import...
- 2. Select Existing Projects into Workspace and Click Next.

Figure 3. Selecting an import source

E Import
Select Create new projects from an archive file or directory.
Select an import source: type filter text General Archive File Stisting Projects into Workspace File System Preferences File System Preferences File System Preferences Operational Decision Management Operational Decisio
Cancel

3. On the Import Projects dialog, select "Select archive file". Browse to the location of the downloaded zip package. Click **Finish**.

Figure 4. Selecting the archive file to import

🚝 Import			
Import Projects Select a directory to searc	ch for existing Eclipse projects.		
 Select root directory: Select archive file: Projects: 	C:\Download C:\Download\hello-rules.zip		Browse Browse
hello-rules (hello-r	ules)		Select All
hello-xom (hello-x	om)		Deselect All
			Refresh
	rkspace		
Working sets	napaco.		
Add project to work	ing sets		
Working sets:		*	Select
?	< Back Next >	Finish	Cancel

The HelloRequest and HelloResponse classes

The Execution Object Model (XOM) in our example consists of 2 Java classes, a request and a response. The HelloRequest class takes a language (a string); the HelloResponse class returns a message in the language specified by the request.

Figure 5. The HelloRequest and Hello Response classes



A Business Object Model (BOM) is generated from the XOM with these 2 classes.

Figure 6. The Business Object Model (BOM)

🟗 Rule Explorer 🛛 🗖 🗖	Greetings	00 virtual 000 000 000 000 000 000 000 000 000 0	hello-xom 🛛	
← → ি ↓ 2 ⊂ 4 ∨ ← → hellocequest	Class Hell General Inform Name: Namespace: Superclasses: Interfaces: Deprecated	oRequest (pa mation HelloRequest com.ibm.wodm.tutorik java.lang.Object	a Change Change	bm.wodm.tutorial) Class Verbalization Remove the verbaliz Generate automatic Term: hello request i the hello request, a f
	Members Specify the mer Ianguage name C HelloReque	nbers of this class. est()	New Delete	✓ Domain Create and edit a domain f

The Messages rule package contains a Greetings decision table as shown in Figure 7.

Figure 7. The Greetings decision table

🔃 Gr	🚺 Greetings 🕱				
<i>]</i>					
×	×				
	Language Greeting				
1	English	Hello			
2	2 Spanish Hola				
3	3 German Hallo				
4	Hindi Namaste				
5	5 French Bonjour				
6	Italian	Ciao			

There are 6 rules in the rule set (each row corresponding to a rule). If the language specified in the request is English, the greeting returned is "Hello". If the language requested is German, the greeting is "Hallo". In this initial decision table, the language is of type string. Later in the tutorial, we will constrain the language to be a predefined set of values of type LanguageType.

The project includes a Decision Validation Services (DVS) test configuration with two scenarios and expected results (in hello-rules/testsuite.xlsx) as shown in Figure 8.

Figure 8. Decision Validation Services test scenarios

Α	В	С	D	E
	Create your so	enarios		
	Click here to a	iccess the he	elp sheet	
			the requ	Jest
	Scenario ID	description	language	name
	Scenario 1		English	John Doe
	Scenario 2		French	John Doe

	Α	В	С		
1					
2		Fill only the cells for the results you want to test			
3		Click here to a	access the help sheet		
4					
5		Scenario ID	the hello message of the response equals		
9		Scenario 1	Hello John Doe		
10		Scenario 2	Bonjour John Doe		

To execute the DVS test suite, right-click on DVS test.launch in hello-rules and run as 'DVS test'. Figure 9 shows the result of the test suite execution:

Figure 9. Results of the DVS test execution

WebSphere. Exe	cution Report				October 17, 2012 2:55:11 PM E
🚖 Summary					
Execution		Local			
Decimal Precis	sion	2 digits are used			
Scenarios		2			
Tests		2			
Success Rate		100%			
Failures		0			
Errors		0			
🚖 Details for	all Scenarios				
Name	Success Rate	Tests	Failures	Errors	Message
Scenario 1	100%	1	0	0	
Scenario 2	100%	1	0	0	

Step 2: Create a domains.xlsx file defining the domain

Next, create an Excel spreadsheet for the dynamic domain. For this article, the file domains.xlsx has already been created under the resources folder with the following content:

A	В	С
Values	BOM to XOM	Label
ENGLISH	return "English";	English
FRENCH	return "French";	French
SPANISH	return "Spanish";	Spanish
GERMAN	return "German";	German
HINDI	return "Hindi";	Hindi
ITALIAN	return "Italian";	Italian

Figure 10. The domains.xlsx file

The Excel domain provider handles the link between the values in the Excel spreadsheet and the BOM. The Excel file must have one row for each value of the domain provider. The file must conform to this structure:

- The Values column contains the values of the domain provider.
- The BOM to XOM mapping column indicates the BOM to XOM mapping for each domain value.
- The Label column lists each language as it will be displayed when authoring a rule (the verbalization of the domain value).

Step 3: Create new virtual BOM entry

1. Right-click the bom directory and select **New -> BOM entry**.

2. In the BOM entry dialog, enter virtual for the Name field, and select the radio button "Create an empty BOM entry". Click **Finish**.

New BOM Entry		
BOM Entry Create a new BOM entry.		28
BOM folder:	/hello-rules/bom	Browse
Folder:		Browse
Name:	virtual	
 Create a BOM entry from a XOM ● Create an empty BOM entry 	R	
?	< Back Next > Finish	Cancel

Figure 11. Creating a new BOM entry

Step 4: Create new BOM class called LanguageType class

1. From the Rule Explorer, double-click the virtual BOM entry just created. Click New Class.

Figure 12. Creating the LanguageType class

🕼 virtual 🛛		F
BOM Entry: virtual		
▼ Tasks		
i <u>Verbalize</u> the elements of this BOM entry.	i <u>Create add and remove methods</u> for all attrib	utes of type Collection.
${f i}$ - Update the dynamic domains of this BOM entry.		
Business Object Model Entry		E
የ፲፮ BOM Entry: virtual		New Class
		New Package
		Delete
	N	Edit
	Re and a second	

2. Enter LanguageType as the name of the class, and click Finish. The result is shown in Figure 13.

Figure 13. The LanguageType class

NT virtual 🛛		
BOM Entry: virtual		
▼ Tasks		
 <u>Verbalize</u> the elements of this BOM entry. <u>Update</u> the dynamic domains of this BOM entry. 	i <u>Create add and remove methods</u> for all attributes of	type Collection.
Business Object Model Entry		1= ti
Emerger Som Entry: virtual		New Class New Package Delete
		Edit

Step 5: Create the domain for the class

1. Double click Language Type. Select Create a domain.

Figure 14. Creating a domain for the class

n virtual 🔀			
Class Langu	адеТуре (рас	kage: defa	ult)
General Informa	tion		▼ Class Verbalization
			i This class is not verbalized. <u>Create</u> a default verbalization.
Name: La	nguageType		
Namespace:		Change	Generate automatic variable
Superclasses: ja	va.lang.Object	Change	_
Interfaces:		Change	
Deprecated			
▼ Members			▼ Domain
Specify the membe	rs of this class.		Create and edit a domain for this class.
		New	🥙 <u>Create</u> a domain.
		Delete	

2. Under Dynamic Domains, choose Excel and click Next.

Figure 15. Selecting the domain type (Excel)

🖶 Domains	_ 🗆 🗙
Domain Type	
Create a dynamic domain from an Excel file.	
	R
<back next=""> Finish</back>	Cancel

- 3. Enter the name of the Excel file that defines the domain. Indicate Languages as the name of the sheet. *Be sure to select the "Table with header" checkbox*. Enter Values for the Value column. Enter BOM to XOM for the BOM to XOM column. Select Label as the Label column.
- 4. Click **Finish**.

Figure 16. Mapping the columns of the Excel file

Excel	Excel		
Configure the domain pro	vider to map the columns	in your Excel file.	
Excel file:	domains.xlsx	_	
Sheet:	Languages	▼ Table with header	
Value column:	Values	T	
BOM to XOM column:	BOM to XOM	•	
Locale	Label column	Documentation column	
English (default)	Label		
Arabic			
Chinese			
Chinese (Taiwan)			
Dutch		•	
?	Back Next >	Finish Cancel	

The domain values now appear as members of the class.

Figure 17. The domain values

🕅 virtual 🖾			
Class LanguageType (package: default)			
General Information	 Class Verbalization 		
Name: LanguageType Namespace: Change Superclasses: java.lang.Object Interfaces: Change Deprecated	 This class is not verbalized. <u>Create</u> a default verbalization. Generate automatic variable 		
▼ Members	▼ Domain		
Specify the members of this class.	Create and edit a domain for this class.		
SF ENGLISH SF FRENCH SF GERMAN Delete	 <u>Edit</u> the domain. <u>Remove</u> the domain. 		
♥ HINDI SF ITALIAN	Domain type: Excel		
SF SPANISH	Synchronize with dynamic values.		

- 5. Expand the BOM to XOM Mapping section. You may need to maximize the tab and scroll down to see this section.
- 6. Enter java.lang.String as the Execution name for the BOM to XOM Mapping.

Figure 18. Mapping the BOM class to the XOM

 BOM to XOM Ma Edit the mapping be 	apping tween this BOM class and the XOM.
Execution name:	java.lang.String
Extender name:	

7. Save the project.

Step 6: Create a new member of HelloRequest class

1. From the bom directory, navigate to the HelloRequest class, double click language, and remove the verbalization. Select the checkbox "Ignore for DVS" so that language is not used during testing.

Figure 19. Removing the original verbalization for the HelloRequest class



2. Save your work.

TIP: You will see errors in the Rule Explorer view. However, these errors will be resolved when the remaining steps are completed.

- 3. Create a new member of the HelloRequest class called languageVirtual. Doubleclick the HelloRequest class. Under the Members section, click New. It is of type LanguageType (instead of string). Click **Finish**.
- 4. Save your work.

Figure 20. Creating the languageVirtual member

🚝 New Member	٢			
Member Create a membe	er.			M
Type • Attribute	C Constructor	C Method		
Name: language	Virtual			
Type: Language	еТуре			Browse
Name	Туре	Domain		Add
				Remove
				Up
				Down
				Edit,
?			inish	Cancel

5. Double-click on the member languageVirtual.

Figure 21. Selecting the languageVirtual member

	197 Hello-xolli Ka	
←⇒ & ₽₂ = ♦ ▼	Class HelloRequest (package: com.i	bm.wodm.tutorial)
	General Information Name: HelloRequest Namespace: com.ibm.wodm.tutorial Change Superclasses: java.lang.Object Change Interfaces: Change Deprecated	Class Verbalization Remove the verbalization. Generate automatic variable Term: hello request i the hello request, a hello request, the hello requests
G HelloResponse G HelloResponse G HelloResponse G HelloResponse G HelloResponse G domains.xlsx G domains.xlsx G domains.xlsx	Members Specify the members of this class. Inquage I	Domain Create and edit a domain for this class. Create a domain. Categories

6. Under the Member Verbalization section, click Create to create a default verbalization.

Figure 22. Creating a default verbalization for languageVirtual

🕅 *hello-xom 🛛			
Member lang	guageVirtual (class: com	.ibm.wodm	.tutorial.HelloRequest)
General Informa	tion		▼ Member Verbalization
Name: language Type: Language Class: com.ibm. I Read/Write Static Deprecated I Ignore for DVS	Virtual Type wodm.tutorial.HelloRequest O Read Only O Write Only Final Update object state	Browse	A This member is not verbalized. Create a default verbalization.

7. Modify the verbalization by clicking "Edit the subject used in phrases." **Figure 23. Editing the verbalization**

ĥΩ *hello-xom ⊠	-
Member languageVirtual (class: com.ibm.	wodm.tutorial.HelloRequest)
Name: languageVirtual Type: LanguageType Class: com.ibm.wodm.tutorial.HelloRequest Ø Read/Write Ø Read Only Ø Static Final Deprecated Update object state Ignore for DV5	 Remove the verbalization. <u>Create</u> a navigation phrase. <u>Create</u> an action phrase. <u>Edit</u> the subject used in phrases. <u>Action : "set the language virtual of a hello request to a LanguageTy</u> Template: set the language virtual of {this} to {language virtual} Navigation : "the language virtual of a hello request" Template: {language virtual of a hello request

8. In the dialog box, change languageVirtual to language. Click OK. Figure 24. Editing the verbalization term

🚝 Edit Term	×
Term: language virtual (i) the language, a language, the languages	
Singular: anguage	Plural languages
Definite article	Indefinite article
Plural: the	Plural:
?	OK Cancel

9. Remove the word virtual from the Action clause so that the final verbalization matches Figure 25.

Figure 25. The final verbalization

 Member Verbalization 	
× <u>Remove</u> the verbalization.	
🐈 <u>Create</u> a navigation phrase.	
💠 <u>Create</u> an action phrase.	
🖉 Edilis the subject used in phrases.	
 Action : "set the language of a hello request to a LanguageType 	" ×
 Action : "set the language of a hello request to a LanguageType Template: set the language of {this} to {language} 	" × 40
 Action : "set the language of a hello request to a LanguageType Template: set the language of {this} to {language} Navigation : "the language of a hello request" 	• × @ ×

10. Under the BOM to XOM Mapping section, define the getter and setter for the new member as shown in Figure 26. You may need to maximize the tab and scroll down to see this section.

T! 0(T 0 1 /1		0 (1 1	T 7 0 / 1 1
Figure 26.	Defining the g	effer and setter	for the langu	age Virtual member
			TOT THE MANAGE	

BOM to XOM N Edit the mapping b	Mapping between this BOM memb	per and the XOM.	
	rts.		
_return t	this.language;	I	
z			
▼ Setter			

11. Save your work.

_this.language=value;

Thus far, you created a new attribute that represents the underlying "language", defined getters and setters for it, verbalized it, and made it of type LanguageType. You also removed the verbalization from the original "language".

Step 7: Modify the corresponding rules

The original rules are expecting a string. Therefore the rules must be adjusted to now expect input of type languageType.

1. Under the rules folder, double click the Greetings rule package. String is now an invalid type.

0			
×	✓ the language of 'the requ	lest' is <u><a string<="" u=""></u>	
	Language	O Greeting	
1	"English"	Hello	
2	"Spanish	s or the request is 	
3	"German ● Inv	valid type 'string', it is not assignable from type 'LanguageType'	
4	"Hindi"	Namaste	
5	"French"	Bonjour	
6	"Italian"	Ciao	

Figure 27. The existing rule contains an invalid type (string)

2. To correct this, right-click on the column header Language and select Edit Condition Column...

Figure 28. Selecting 'Edit Condition Column'

the language of 'the request' is <u></u>				
Langu	Edit Condition Column	Greeting		
Engl	Tormat	Hello		
Spar	a Sort Ascending	Hola		
Gern	1ª, Sort Descending	Hallo		
Hin -		Namaste		
Frer	C Insert Condition Column Before	Bonjour		
Itali	C> Insert Condition Column After	Ciao		
	X Remove Condition Column			
	Ctrl+Shift+O			
-				

3. Edit the Condition Column and indicate LanguageType instead of string. Figure 29. Replacing string with LanguageType

🕌 Condition Column			×	
Condition Definition				
S Expression has errors. Edit the expression to correct errors before applying this text.				
_ Test				
the language of 'the	request' is			
i Validate the expression before editi	ing sub-colump prope	Ca LanguageType>		
1 validate the expression before edit	ing sab-colamn prope	An object>		
Properties		English		
Title: Language		French		
		🕘 German		
Check interval	Ascending	Hindi		
		Italian		
Expression Placeholders		 Spanish 		
	4	° (a) the language of 		
a string [a string]	Sub-column Title:			

- 4. Click Apply and then click OK.
- 5. Double click on each string in the Language column and choose the appropriate value from the dropdown (English for "English", and so on).
- 6. Save your work.

The resulting Decision table should look like Figure 30.

Figure 30. The modified decision table

×	the language of 'the request' is I	italian
	Language 📀	Greeting
1	English	Hello
2	Spanish	Hola
3	German	Hallo
4	Hindi	Namaste
5	French	Bonjour
6	Italian	Ciao
. 7		

Step 8: Rerun the test

In this step, you rerun the DVS test as a regression test to ensure that the change to the dynamic domain does not impact the expected results.

- 1. From the main menu, select **Run -> Run Configurations...**
- 2. Select DVS_test under **DVS Excel file**.
- 3. Verify the path of the Excel file containing the test suite and indicate the desired location of the HTML output report.

🚝 Run Configurations		×
Create, manage, and run	configurations	
Yeye filter text Apache Tomcat DVS Archive DVS Excel File DVS test - with Dy DVS_test Eclipse Annication	Name: DVS_test Excel File A JRE Classpath L Extraction B DVS Configuration Com Source: Excel File: \hello-rules\testsuite.xlsx Rule Project: hello-rules Code: the co	mon Browse Browse
Clipse Data Tools Clipse Data Tools Clipse Data Tools Generic Server HTTP Preview J2EE Preview J2EE Preview J32ava Applet Java Applet Java Application Ju Junit Ju Junit Ju Junit V JUnit Filter matched 22 of 22 items	Output: HTML Report: \hello-rules\report1023.html Working directory: • • Default: \${workspace_loc:hello-rules} • Other: • Workspace File System. Apply. •	Browse
?	Ru	n Close

Figure 31. Setting up and running the DVS test

4. Verify the test results by viewing the HTML report with a browser.

Step 9: Add new values to the spreadsheet

1. First open the spreadsheet domains.xlsx and create a new row for each new value. In this example, a row has been added for the Alsatian and Portuguese languages.

Figure 32. Adding new values in the Excel file

А	В	С
Values	BOM to XOM	Label
ENGLISH	return "English"	English
FRENCH	return "French";	French
SPANISH	return "Spanish";	Spanish
GERMAN	return "German";	German
HINDI	return "Hindi";	Hindi
ITALIAN	return "Italian";	Italian
PORTUGUESE	return "Portuguese";	Portuguese
ALSATIAN	return "Alsatian";	Alsatian

- 2. Save the file and refresh the resources folder.
- 3. Under the folder bom -> virtual, double-click the LanguageType class. Under Domain type: Excel, click Synchronize. The new values appear in the list of class members.

Figure 33. Synchronizing new domain values with the class



TIP: After any updates to a domain, refresh the resources folder containing the Excel sheet, and use the Synchronize link to synchronize the updates with the class.

4. Return to the Greetings decision table in the rule package. Use the dropdown menu to select the new languages and then type the expected response in the Greeting column.

Figure 34. Updating the decision table

🔢 *Greetings 🕱				
×	x v the language of 'the request' is 			
	Language	Greeting		
1	English	Hello		
2	Spanish	Hola		
3	German Hallo			
4	Hindi	Namaste		
5	French Bonjour			
6	Italian Ciao			
7	Alsatian	Salu		
8	Portuguese	Ola		
_				

You can then extend your test suite by adding scenarios to cover the newly added languages.

Step 10: Publish to Decision Center

1. From the Start menu, start the Sample server if it is not already started.





2. Right-click the project and select **Decision Center -> Connect**. Enter the URL used to access the Decision Center and supply your authentication information. Click the Connect button.

Figure 36. Configuring the connection to the Decision Center

ŧ			_ 🗆 🗵
Decision Ce	enter configuration		
(i) Connection	with Decision Center successfully establishe	ed.	
Connection -			
URL:	http://localhost:9080/teamserver		•
User name:	rivi		•
Password:	****		
Data source:			•
			Connect
Synchronizatio	on Filter		
Filter rule	elements to be synchronized using the spec	ified query	
			~
Project config	uration		
Create a r	new project on Decision Center		
🔘 Synchronia	ze with the existing Decision Center project		
			7
			1
(?)		Einish	Cancel

3. When the Remote Operation is complete, click Finish.

Figure 37. Connecting to the Decision Center

틎 Configure project			_ 🗆 🗵
	joing		
🔲 Always run in background			
ß	Run in Background	Cancel	Details >>

4. When synchronization is complete click OK.



Part 2. Modifying dynamic domains in the Decision Center

Using dynamic domains allows business users to implement and deploy a change to a business rule quickly without the aid of IT support. Rules developed with enumerations would require code modifications to accommodate new values; with dynamic domains a business user can make simple edits to the underlying Excel spreadsheet that defines the domain. The new values are immediately available for testing and deployment, and eventual synchronization with the Rule Designer.

The following steps demonstrate how a business user can add a new value to a predefined list of languages. The steps assume that the project "practice-hello-rules" has been synchronized with the Decision Center and is ready to use.

Step 1: Log into the Decision Center

1. Enter your username and password on the Decision Center sign-in page.

inguice of hogging into the	Decision Center
I	
	Sign in to Decision Center Username: Password: Remember my information Sign In

Figure 39. Logging into the Decision Center

2. On the Home tab, select your project from the Project in use menu.

Figure 40. Selecting a project in Decision Center

🕙 Decision Center - Mozilla Firefox		'×
Eile Edit View History Bookmarks Tools Help		
🜐 Decision Center +		
Contraction (Contraction Contraction Contractic Con		⋒
🕗 Most Visited 📋 Getting Started 🔊 Latest Headlines		
IBM, Decision Center	Options About Print View Help	•
Home Explore Compose	Query Analyze Project Configure	
	Project: hello-rules Get Link	
Welcome to the Decision Cent	er Home Page	
Project in use: 🐵 hello-rules	×	
Branch in use: 🖷 main 🔻		
Current action: Work upon branch 💌		
Help Help is available by clicking Help in the top online help	o banner. Also, question marks are available throughout Decision Center to access specific to	pi
<u> 1</u>		

3. On the Explore tab, under Business Rules, review the values of the Greetings decision table imported from Rule Designer. Click the Greetings link.

Figure 41. Accessing the Greetings decision table in Rule Designer

Home Explore Compose Query Analyze Project			
Smart Folders 🛛 💀 🛛 🖗	Business Rules > messages		
 Business Rules messages 1 Ruleflows 1 Templates 	🔊 New 🔍 Details 🔨 Edit 💥 Delete 🗈 Copy Display by 10 💌		
Simulations	Actions Name		
Test Suites	🗖 🔍 📉 📴 <u>Greetings</u>		
Resources 1	\sim		

Figure 42. The Greetings decision table

EGreetings (Decision Table)

🎝 New 🔨 Edit 💢	Delete 🛅 Copy 🔒 Lock 👩 Unlock	🔗 Release lock 🤅) History ⑦ Help	
🔲 Properties		🖉 Table		
Name	Greetinas		Language	
Status	Now	1	English	
Status	14644	2	Spanish	
Priority		3	German	
Expiration Date	None	4	Hindi	
Effective Date	None	5	French	
	None	6	Italian	
Locale	English (United States)	7	Alsatian	
Categories	Any	8	Portuguese	
Template			14	
Active	Тпію	Attached	items	

Step 2: Create Resources Smart folder

1. On the Explore tab, click the Create Smart Folder icon. Create a new smart folder called Resources.

Figure 43. The Create Smart Folder icon



2. For Properties, enter Resources as the folder name. Click Next.

Figure 44. Creating a Resources folder

Step 1: Properties	Properties	
Step 2: Query		
Step 3: Displayed Properties	Name*	Resources
Step 4: Documentation	Include Dependencies	
Step 5: Version aformation	Group	<none></none>
	Cancel	Previous Next Finish

3. For Step 2. Query, indicate that all resources are to be searched and displayed. Click **Finish**.

Home Explore	Compose* Query Analyze
	Project: practice-he
Step 1: Properties	Querv
Step 2: Query	y
Step 3: Displayed Properties	Find all business rules
Step 4: Documentation	all business rules
Step 5: Version Information	all decision tables us all decision trees all functions all resources
	all rule 🖑 tifacts all rule packages

Figure 45. Specifying a Query for the Resources folder

The domains.xlsx file should be visible as the element in this new folder.

Figure 46. The domains.xlsx file in the Resources folder

IBM。Decision C	enter	
Home Expl	ore Cor	mpose Query Analyze Project
Smart Folders	💀 l 🔁	Resources
 Business Rules Ruleflows 1 Templates Simulations Test Suites Resources 1 	Q, (New Q Details Edit Edit Copy Display by 10 Actions Name Actions Name Actions Name

Step 3: Modify domains.xlsx in the Resources folder

In this step we download the Excel file to our local disk and add a row for a new language, Swedish. We then upload this file to the Decision Center and refresh the project.

- 1. Click the download icon next to the file name. Save the file to your local disk and then open it for editing.
- 2. Add a new row for the Swedish language.

Figure 47. Downloading the domains.xlsx file

Resources		
🕹 New 🔍 Det	ails 📏 Edit 💢 Delete 🛅 (
Display by 10 💌		
Actions	Name	
	🗋 domains.xlsx (8.61 KB)	

3. Edit the file, adding a new row for the new language. Save the file.

	А	В	С
1	Values	BOM to XOM	Label
2	ENGLISH	return "English"	English
3	FRENCH	return "French";	French
4	SPANISH	return "Spanish";	Spanish
5	GERMAN	return "German";	German
6	HINDI	return "Hindi";	Hindi
7	ITALIAN	return "Italian";	Italian
8	PORTUGUESE	return "Portuguese";	Portuguese
9	ALSATIAN	return "Alsatian";	Alsatian
10	SWEDISH	return "Swedish";	Swedish
11			

Figure 48. Adding a new rows to the Excel file

4. Back in the Decision Center, click the domain.xlsx link. Click Edit.

Figure 49. Editing the link to the domains.xlsx file

Odomains (Resource)

🔄 New 🔨 Edit 💥 Dele	ete 🖹 Copy 👌 Lock 👩 Unlock 🔗 Release loc
Properties	
File	domains.xlsx (8.61 KB)
Group	
Created By	rivi
Last Changed By	rivi
Last Changed On	Oct 22, 2012 5:17:36 PM EDT
Created On	Oct 22, 2012 5:17:36 PM EDT
Туре	Resource
Folder	/

5. Browse and select the updated domains.xlsx file to upload it back into the Decision Center.

Figure 50. Uploading the new domains.xlsx file

Properties

File	domains.xlsx (8.34 KB)	
		Browse
Folder	/	
Group	<none></none>	

Figure 51. The uploaded domains.xlsx file

IBM。Decision Center		
Home Explore Co	mpose Query Analyze Project	
Smart Folders 🛛 🖓	Resources	
 Business Rules Ruleflows 1 Templates Simulations Test Suites Resources 1 	 New Q Details Edit Delete Copy Display by 10 Actions Name Actions Q Mains.xlsx (8.61 KB) 	

Step 4: Reload dynamic domains

From the Project tab select Reload Dynamic Domains to refresh the project. Figure 52. Reloading the dynamic domain

Home Explore Compose Query Analyze Project
Project
Manage Project
Manage Subbranches and Baselines Create, delete, and rename subbranches and baselines for the current project
Merge Branches Merge the working branch with another branch
Edit Project Dependencies View and edit the projects/baselines this project depends on
Edit Project Options View and edit the options of this project
Business Object Model
View BOM Path View the BOM path entries of this project
Reload Dynamic Domains All the dynamic domains of the BOM will be reloaded

Step 5: Add the new value to the decision table

The new value Swedish can now be used when authoring a rule. To make it available in the Decision Center, update the decision table.

1. Under Business Rules, click the Greetings link and then click Edit.

Figure 53. Selecting the Greetings decision table



Figure 54. Editing the decision table

Home Explo	re Compose Query	Analyze Pr	roject Configure	
Explore > Details			Project: Copy of hello-rules G	et l
Greetings (De	ecision Table)			
	*			
💩 New 📉 🚫 Edit 💥 🛛	Delete 🖹 Copy 🔂 Lock 👩 Unloc	🛙 🖗 Release lock 🔇) History 🕐 Help	
Properties		🖉 Table		
Name	Greetings	🔥 col: 1, lin	e: 1: Rows have gap(s).	
Status	New			_
Priority			Language 💧	
Expiration Date	None	1	English	_
Effective Date	None	3	German	_
Locale	English (United States)	4	Hindi	
	English (Onliced Scates)	5	French	
Categories	Any	6	Italian	
Template		7	Alsatian	
Active	True	8	Portuguese	
Folder	/messages	Attached	Items	
Group				

2. Select Step 2: Table and use the dropdown menu to reveal the refreshed list of predefined values for languages. In the Language column, select Swedish.

Figure 55.	Selecting the	e new val	ues in th	he Languages	column
Ober D. Tabl	-				

Step 2, Table					
Step 3: Tags	Prg Prg Prg 22 24 22 All				
Step 4: Override Rules	\sim the language of the request is Swedish [and/or]				
Step 5: Documentation					
Step 6: Version Information	i Use this editor to edit the selected cell.				
	Language				
	1 English				
	2 Spanish				
	3 German				
	4 Hindi				
	5 French				
	6 Italian				
	7 Alsatian				
	8 Portuguese				
	9 Swedish				
	10 Alsatian				
	11 English				
	12 French				
	13 German				
	14 Hinai Italian				
	15 Dortuguese				
	16 Snanish				
	Swedish				
	Cancel Previous Next Finish				

- 3. In the Greeting column, indicate the translated value to be displayed when Swedish is requested.
- 4. Click Finish and save your work.

Figure 56. Entering the translated values in the Greeting column

Table

	1 - 16 16 - 20	All 🔚 💪 🔯	
Action editor			
$\times \cdot$	$\propto \sqrt{1+1}$ set the helio message of the response to Hei + " " + the name of the request [±]		
i Use this editor to edit the selected cell.			
	Lanquage	Greeting	
	English	Lielle	
1	English		
Z	Spanish	Hola	
3	German	Hallo	
4	Hindi	Namaste	
5	French	Bonjour	
6	Italian	Ciao	
7	Alsatian	Salu	
8	Portuguese	Ola	
9	Swedish	Hej	
10			

The new value is immediately available for use in rule authoring.

Home Explore	Compose* Query Analyze Project Configure
	Project: Copy of hello-rules Editing: New Action Rule (Actio
Step 1: Properties	Content
Step 2: Content	
Step 3: Tags	[definitions]
Step 4: Override Rules	the language of the request is 🗙
Step 5: Documentation	Alsatian
Step 6: Version Information	then English < <select action="" an=""> × French [else] German Hindi</select>
	Cancel Previou Portuguese Spanish Swedish the language of

Figure 57. Using the new values in rule definitions

Conclusion

This article demonstrates the advantages of using dynamic domains in WebSphere Operational Decision Management V7.5. The use of a dynamic domain ensures a measure of error prevention during rule authoring and offers benefits over use of static enumerations. By using a simple Excel file as the underlying data source for a domain, rule developers can provide business users with the flexibility to modify the domain from within the Decision Center without the need for IT support. And because adding a new domain value does not result in changes to service contracts, developers do not need to redeploy the rule service or change the service clients.