

Course Guide

Managing Decisions in IBM Operational Decision Manager V8.9

Course code WB401 / ZB401 ERC 1.1



October 2017 edition

Notices

This information was developed for products and services offered in the US.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

*IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
United States of America*

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some jurisdictions do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you provide in any way it believes appropriate without incurring any obligation to you.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to actual people or business enterprises is entirely coincidental.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the web at "Copyright and trademark information" at www.ibm.com/legal/copytrade.shtml.

© Copyright International Business Machines Corporation 2017.

This document may not be reproduced in whole or in part without the prior written permission of IBM.

US Government Users Restricted Rights - Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

Trademarks	xvi
Course description	xvii
Agenda	xix
Unit 1. Introducing IBM Operational Decision Manager V8.9.....	1-1
How to check online for course material updates	1-2
Unit objectives	1-3
Topics	1-4
1.1. What is a business rule?.....	1-5
What is a business rule?	1-6
What is a business rule?	1-7
From business policy to business rules	1-8
Discussion	1-9
Discussion: What is a rule?	1-10
1.2. Why the need for decision management?	1-11
Why the need for decision management?	1-12
Why is there a need for decision management?	1-13
What is decision management?	1-15
1.3. What is operational decision management?.....	1-17
What is operational decision management?	1-18
Challenges to decision automation challenges	1-19
Using operational decision management	1-20
What is operational decision management?	1-22
Synchronized business and IT cycles	1-23
1.4. Introducing IBM Operational Decision Manager.....	1-24
Introducing IBM Operational Decision Manager	1-25
Operational Decision Manager (1 of 2)	1-26
Operational Decision Manager (2 of 2)	1-27
Operational Decision Manager roles and activities	1-29
Decision Center: Authoring and maintaining business rules	1-30
Decision Server: Development and execution	1-31
Decision Server Insights	1-32
1.5. ODM packaging	1-34
ODM packaging	1-35
Options for using ODM	1-36
Operational Decision Manager on-premises editions	1-37
IBM ODM on Cloud: Three runtime environments	1-39
Business Rules Service on IBM Bluemix	1-40
1.6. Integration with IBM product family.....	1-41
Integration with IBM product family	1-42
Synergies with other IBM products	1-43
1.7. Operational Decision Manager roles	1-45
Operational Decision Manager roles	1-46
Operational Decision Manager user roles	1-47
Interaction between roles	1-48
Business analyst	1-49
Policy manager	1-50
Rule author	1-51

Architect	1-52
Developer	1-53
Administrator	1-54
Discussion	1-55
Discussion: Who is in charge of what?	1-56
1.8. Governance and decision management	1-57
Governance and decision management	1-58
Applying governance to decision management	1-59
Governance in Operational Decision Manager	1-61
Unit summary	1-62
Review questions (1 of 2)	1-63
Review questions (2 of 2)	1-64
Review answers (1 of 2)	1-65
Review answers (2 of 2)	1-66
Exercise: Operational Decision Manager in action	1-67
Exercise objectives	1-68
Exercise workflow	1-69
Unit 2. Modeling for business rules	2-1
Unit objectives	2-2
Topics	2-3
2.1. Approaching rules as a business analyst	2-4
Approaching rules as a business analyst	2-5
Approaching rules as a business analyst	2-6
Overview of business analyst tasks	2-7
Who is involved and when?	2-8
2.2. Starting with models	2-9
Starting with models	2-10
What is a model?	2-11
Types of models	2-13
2.3. Modeling the business process	2-14
Modeling the business process	2-15
Modeling the process	2-16
Eliciting the process	2-17
Discussion	2-18
Discussion: Draw a process flow	2-19
Discussion review: A possible process flow	2-20
2.4. Defining use cases	2-21
Defining use cases	2-22
Defining use cases	2-23
What is a use case?	2-24
Alternative scenarios	2-25
Example: Use case for renting a car	2-26
Discussion	2-27
Discussion: Create a use case	2-28
Discussion review: A possible use case description	2-29
Discussion review: A possible use case template	2-30
2.5. Modeling the rule vocabulary	2-31
Modeling the rule vocabulary	2-32
From decisions to discovery	2-33
Intertwined discovery of vocabulary and rules	2-34
Discover the vocabulary and rules	2-35
Identifying objects in use cases	2-37
Classes	2-38
Class diagrams	2-39
Attributes	2-40

Class methods	2-41
Class relationships	2-42
Relationships	2-43
Cardinality in associations	2-44
Specifying association	2-45
Aggregation	2-46
Composition	2-47
Inheritance	2-48
Generalization	2-49
Inheritance and association	2-50
Inheritance in rules	2-51
What is a “good” business object model?	2-52
Ensuring completeness in the model	2-53
Exercise: Building the model on paper	2-54
Exercise objectives	2-55
Exercise review: A possible solution	2-56
2.6. Implementing the model	2-58
Implementing the model	2-59
Implementing the model	2-60
From discovery to implementation	2-61
What happens next?	2-62
Using an agile development approach	2-63
Unit summary	2-65
Review questions	2-66
Review answers (1 of 2)	2-67
Review answers (2 of 2)	2-68
Exercise: Implementing the model	2-69
Exercise objectives	2-70
Exercise review (1 of 2): A possible solution	2-71
Exercise review (2 of 2)	2-72

Unit 3. Understanding decision services.....	3-1
Unit objectives	3-2
Topics	3-3
Recall: Agile business rule development	3-4
3.1. Designing the business rule application	3-5
Designing the business rule application	3-6
Designing the business rule application	3-7
Rule project structure (1 of 2)	3-8
Rule project structure (2 of 2)	3-9
Decision services and classic rule projects (1 of 2)	3-10
Decision services and classic rule projects (2 of 2)	3-11
Working with decision services in Decision Center	3-12
Modular structure	3-13
Decision services: Modular structure (1 of 2)	3-14
Decision services: Modular structure (2 of 2)	3-15
3.2. Setting up a decision service	3-16
Setting up a decision service	3-17
Decision service map	3-18
Create a decision service rule project in Rule Designer	3-19
Decision service rule project templates	3-20
Decision service projects	3-21
Decision service setup: Import XOM	3-22
Decision service setup: Create a BOM	3-23
Decision operations	3-25
Operation Map	3-26

3.3.	Design signature	3-27
	Design signature	3-28
	Decision operations and signatures	3-29
	Design signature	3-30
3.4.	Orchestrating	3-31
	Orchestrating	3-32
	Orchestrate	3-33
	Rule packages (1 of 2)	3-34
	Rule packages (2 of 2)	3-35
	Orchestrating ruleset execution	3-36
	Ruleflow tasks	3-38
	Example: Car insurance	3-39
3.5.	Sharing and synchronizing rule projects	3-40
	Sharing and synchronizing rule projects	3-41
	Synchronization	3-42
	Synchronizing between business and IT users	3-44
	Implementation approaches for business rule management	3-45
	Unit summary	3-46
	Review questions	3-47
	Review answers	3-48
	Exercise: Setting up a decision service	3-49
	Exercise objectives	3-50
Unit 4.	Working with the BOM	4-1
	Unit objectives	4-2
	Topics	4-3
4.1.	Creating rule vocabulary	4-4
	Creating rule vocabulary	4-5
	Working collaboratively on the BOM	4-6
	Rule vocabulary	4-7
	Creating vocabulary	4-8
	Verbalization	4-9
	Class	4-10
	Members	4-11
	Verbalizing BOM members	4-12
4.2.	Categories	4-13
	Categories	4-14
	Categories	4-15
	Category semantics	4-16
4.3.	Refactoring	4-17
	Refactoring	4-18
	Refactoring	4-19
4.4.	Using domains	4-20
	Using domains	4-21
	Domains	4-22
	Main types of domains	4-23
	Dynamic domains	4-24
	Using Excel for dynamic domains	4-25
	Unit summary	4-26
	Review questions	4-27
	Review answers	4-28
	Exercise: Working with the BOM	4-29
	Exercise objectives	4-30
Unit 5.	Introducing Decision Center	5-1
	Unit objectives	5-2

Topics	5-3
5.1. Decision Center overview.....	5-4
Decision Center overview	5-5
Decision Center overview	5-6
Decision Center Repository	5-7
Accessing the Decision Center consoles	5-8
Decision Center relationship to rule-based applications	5-9
5.2. Tour of the Decision Center Enterprise console.....	5-10
Tour of the Decision Center Enterprise console	5-11
Enterprise console: Rule authoring and administration	5-12
Enterprise console environment and Home tab	5-13
Exploring rules	5-14
Creating a rule: Using the Compose tab wizard	5-15
Editing a rule: Quick edit and the Edit menu	5-17
Working in the rule editor	5-18
5.3. Tour of the Decision Center Business console.....	5-19
Tour of the Decision Center Business console	5-20
Decision Center Business console	5-21
Business console Home tab	5-22
Library and Work	5-24
Working with decision services	5-25
Business console Decision Artifacts tab	5-26
Viewing a rule	5-27
Creating rules and decision tables	5-28
Editing an existing rule or decision table	5-29
Using the rule editor	5-30
Working with snapshots	5-31
Social features: Following	5-33
Social features: Streams and comments	5-35
Unit summary	5-37
Review questions	5-38
Review answers	5-39
Exercise: Exploring the Decision Center Business console	5-40
Exercise objectives	5-41
Unit 6. Introducing rule authoring.....	6-1
Unit objectives	6-2
Topics	6-3
6.1. From business policy to business rule	6-4
From business policy to business rule	6-5
Rule language evolves during a project (1 of 2)	6-6
Rule language evolves during a project (2 of 2)	6-8
When rule authoring occurs	6-9
Who writes rules?	6-10
Where are rules written?	6-11
6.2. Rule structure	6-12
Rule structure	6-13
Parts of a rule (1 of 3)	6-14
Parts of a rule (2 of 3)	6-16
Parts of a rule (3 of 3)	6-17
Rules must include an action	6-18
Differences between policies and rules	6-19
Detailed rule structure	6-20
Discussion	6-21
Discussion question: What is the underlying policy for this rule?	6-22
Discussion: Structuring your rules	6-23

6.3. Rule artifacts	6-24
Rule artifacts	6-25
Types of rule artifacts	6-26
Unit review (1 of 2)	6-27
Unit review (2 of 2)	6-28
Unit summary	6-29
Review questions (1 of 2)	6-30
Review questions (2 of 2)	6-31
Review answers (1 of 2)	6-32
Review answers (2 of 2)	6-33
Exercise: Understanding the case study	6-34
Exercise objectives	6-35
Unit 7. Discovering and analyzing rules.....	7-1
Unit objectives	7-2
Topics	7-3
7.1. Overview of discovery and analysis	7-4
Overview of discovery and analysis	7-5
Goals of rule discovery and analysis	7-6
7.2. Agile Business Rule Development (ABRD)	7-7
Agile Business Rule Development (ABRD)	7-8
Challenges to implementing business rule applications	7-9
Agile Business Rule Development (ABRD)	7-10
ABRD phases of development	7-11
ABRD tracks	7-12
ABRD methodology	7-13
Agile ruleset development	7-14
7.3. Rule discovery	7-15
Rule discovery	7-16
Rule discovery	7-17
Step 1: Select discovery roadmap	7-18
Roadmaps	7-19
Planning rule discovery time	7-20
Step 2: Select rule standards	7-21
Business rule classification (1 of 4)	7-22
Business rule classification (2 of 4)	7-23
Business rule classification (3 of 4)	7-24
Business rule classification (4 of 4)	7-25
Discussion	7-26
Discussion: Classify your rules	7-27
Step 3: Discover the rules through the use case roadmap	7-28
Decision points table example	7-29
Decision points table: Classification complexity	7-30
Formalizing decisions as rules with rule templates	7-31
Rule documents	7-32
Step 4: Authenticate the rules	7-33
Step 5: Discover vocabulary	7-34
Step 6: Create test scenarios	7-35
Exercise: Discovering rules	7-36
Exercise objectives	7-37
7.4. Rule analysis	7-38
Rule analysis	7-39
Rule analysis	7-40
Step 1: Make rules atomic	7-41
Step 1: Make rules atomic: Action enablers and inference rules	7-42
Step 1: Make rules atomic: Constraints and guidelines	7-43

Step 2: Identify rule patterns	7-44
Step 3: Remove redundancy and overlaps	7-45
Step 3a: Remove redundancy	7-46
Step 3b: Remove overlaps	7-47
Step 4: Resolve inconsistency	7-49
Step 5: Ensure completeness among rules	7-50
Step 6: Identify dependencies	7-51
Example: Rule dependency table	7-52
Step 7: Refine process	7-53
Step 8: Optimize the rules	7-54
Unit summary	7-55
Review questions	7-56
Review answers	7-57
Exercise: Analyzing rules	7-58
Exercise objectives	7-59
Unit 8. Working with conditions in rules	8-1
Unit objectives	8-2
Topics	8-3
8.1. Rule structure review	8-4
Rule structure review	8-5
Parts of a rule (rule structure)	8-6
8.2. Rule condition examples.....	8-7
Rule condition examples	8-8
Rule condition examples (1 of 3).....	8-9
Rule condition examples (2 of 3).....	8-10
Rule condition examples (3 of 3).....	8-11
8.3. Working with BAL constructs to build condition statements	8-12
Working with BAL constructs to build condition statements	8-13
Business Action Language (BAL) constructs and operators	8-14
Types of conditions	8-15
Comparison conditions	8-16
Count conditions	8-17
Existence conditions	8-18
Evaluate set membership conditions	8-20
Negative conditions	8-21
BAL number operators	8-22
BAL arithmetic operators	8-23
Discussion	8-24
Discussion: Working with conditions in rules	8-25
8.4. Using Boolean logic in rules	8-26
Using Boolean logic in rules	8-27
Boolean logic in rules: ANDs, ORs, and NOTs	8-28
The meaning of OR	8-29
The meaning of AND	8-30
Mixing ANDs and ORs	8-31
Negating conditions	8-32
Negating AND (1 of 2)	8-33
Negating AND (2 of 2)	8-34
Negating OR	8-35
Negating "there is at least one"	8-36
Stating that something is true for all Xs	8-37
8.5. Writing condition statements in the Business console	8-38
Writing condition statements in the Business console	8-39
Review: BOM relationships in vocabulary lists (1 of 2)	8-40
Review: BOM relationships in vocabulary lists (2 of 2)	8-41

Choosing the correct vocabulary	8-42
Where to find terms and phrases in the vocabulary menu	8-43
Unit summary	8-44
Review questions (1 of 2)	8-45
Review questions (2 of 2)	8-46
Review answers (1 of 3)	8-47
Review answers (2 of 3)	8-48
Review answers (3 of 3)	8-49
Exercise: Working with conditions in rules	8-50
Exercise objectives	8-51
Exercise review	8-52
Exercise review: Using a constant versus using a variable	8-53
Exercise review: Advanced practice	8-54
Unit 9. Working with definitions in rules	9-1
Unit objectives	9-2
Topics	9-3
9.1. Understanding definitions and variables.....	9-4
Understanding definitions and variables	9-5
What is a definition?	9-6
What is a variable? (1 of 2)	9-7
What is a variable? (2 of 2)	9-8
Defining variables in a rule (1 of 3)	9-9
Defining variables in a rule (2 of 3)	9-10
Defining variables in a rule (3 of 3)	9-11
9.2. Types of variables.....	9-12
Types of variables	9-13
Types of variables	9-14
Ruleset variables	9-15
Using ruleset variables with ruleset parameters	9-16
Ruleset variables with ruleset parameters: Example	9-17
Automatic variables	9-19
Rule variables	9-21
Where you use variables	9-22
Initializing variables	9-23
9.3. Working with definitions	9-24
Working with definitions	9-25
Definitions and rule variables overview	9-26
Defining a rule variable	9-27
Defining preconditions	9-28
Comparing rules with and without preconditions	9-29
When to define variables	9-30
9.4. Types of variable definitions.....	9-31
Types of variable definitions	9-32
Defining variables: Constants	9-33
Defining variables: Expressions	9-34
Defining variables: Objects (1 of 2)	9-35
Defining variables: Objects (2 of 2)	9-36
Defining variables: Collections of objects	9-37
9.5. Using BAL constructs in variables	9-38
Using BAL constructs in variables	9-39
Using BAL constructs in variables	9-40
Using the in <collection> construct	9-41
Using the where <test> construct	9-42
Using the from <relation> construct	9-43
9.6. Working with collections of objects	9-44

Working with collections of objects	9-45
BAL constructs to use with collections	9-46
Defining a collection of objects	9-47
Using a defined collection in conditions	9-48
Using a collection in action statements	9-49
Discussion	9-50
Discussion: Working with definitions in rules	9-51
9.7. Mechanics of defining variables	9-52
Mechanics of defining variables	9-53
Defining variables in the Business console (1 of 3)	9-54
Defining variables in the Business console (2 of 3)	9-55
Defining variables in the Business console (3 of 3)	9-56
Using a variable	9-57
Unit summary	9-58
Review questions (1 of 2)	9-59
Review questions (2 of 2)	9-60
Review answers (1 of 2)	9-61
Review answers (2 of 2)	9-62
Exercise: Working with definitions in rules	9-63
Exercise objectives	9-64
Exercise overview: Reviewing the BOM before you start	9-65
Exercise review: Solution 1	9-66
Exercise review: Solution 2	9-67
Exercise review: Solution 3	9-68
Exercise review: Comparison of solutions 1 and 2	9-69
Exercise review: Comparison of solutions 2 and 3	9-70
Unit 10. Writing complete rules.....	10-1
Unit objectives	10-2
Topics	10-3
10.1. Understanding rule actions.....	10-4
Understanding rule actions	10-5
What is an action?	10-6
Rule actions	10-7
Examples of rule actions	10-8
Rules that include only actions	10-9
Multiple actions	10-10
Else statements	10-11
When to use else statements	10-12
10.2. Rules that perform computations.....	10-14
Rules that perform computations	10-15
Rules that perform computations	10-16
Computation examples (1 of 2)	10-17
Computation examples (2 of 2)	10-18
10.3. Handling errors.....	10-19
Handling errors	10-20
Identifying errors	10-21
Effect of errors	10-22
Avoiding ambiguity with parentheses (1 of 2)	10-23
Avoiding ambiguity with parentheses (2 of 2)	10-24
Use of parenthesis () in rule action	10-25
Reviewing written rules: Verifying accuracy	10-26
Unit summary	10-27
Review questions (1 of 2)	10-28
Review questions (2 of 2)	10-29
Review answers (1 of 2)	10-30

Review answers (2 of 2)	10-31
Exercise: Writing complete rules	10-32
Exercise objectives	10-33
Unit 11. Authoring decision tables	11-1
Unit objectives	11-2
Topics	11-3
11.1. Introducing decision tables	11-4
Introducing decision tables	11-5
Introducing decision tables	11-6
11.2. Decision tables overview	11-7
Decision tables overview	11-8
About decision tables	11-9
Example: Symmetrical rules	11-10
Example: Symmetrical rules in a decision table	11-11
Structure of a decision table	11-12
Columns for conditions and actions	11-13
Discussion	11-14
Discussion: Decision tables in your work	11-15
11.3. Working with decision tables	11-16
Working with decision tables	11-17
Working with decision tables: Overview	11-18
Defining conditions (1 of 2)	11-19
Defining conditions (2 of 2)	11-20
Splitting conditions	11-21
Defining actions (1 of 2)	11-22
Defining actions (2 of 2)	11-23
Building decision tables by adding columns (1 of 2)	11-24
Building decision tables by adding columns (2 of 2)	11-25
Viewing the meaning of individual rules in decision tables	11-26
Editing tools and options	11-27
Editing cell values	11-28
Editing in place	11-29
Changing the operator	11-30
Custom values for cells	11-31
Entering domain values	11-32
Empty cells	11-33
Disabling action cells	11-34
Preconditions for a decision table (1 of 2)	11-35
Preconditions for a decision table (2 of 2)	11-36
Decision table properties	11-37
Overlap and gap checks	11-38
Decision tables and Excel	11-39
Unit summary	11-40
Review questions	11-41
Review answers	11-42
Exercise: Authoring decision tables	11-43
Exercise objectives	11-44
Exercise: Authoring rules: Putting it all together	11-45
Exercise introduction	11-46
Unit 12. Running tests and simulations	12-1
Unit objectives	12-2
Topics	12-3
12.1. Overview of testing and simulation	12-4
Overview of testing and simulation	12-5

What is testing and simulation?	12-6
Reports	12-7
KPI results	12-8
12.2. Working with scenarios	12-9
Working with scenarios	12-10
What are scenarios?	12-11
Scenario file data	12-12
Excel scenario files and the BOM	12-13
Relationships with the BOM	12-14
Scenarios	12-15
Expected results (1 of 2)	12-16
Expected results (2 of 2)	12-17
Understanding results	12-18
12.3. Defining tests	12-19
Defining tests	12-20
Defining tests in the Business console	12-21
Generating a scenario file for testing (1 of 3)	12-22
Generating a scenario file for testing (2 of 3)	12-23
Generating a scenario file for testing (3 of 3)	12-24
Reports for test results	12-25
12.4. Defining simulations	12-26
Defining simulations	12-27
Defining simulations	12-28
Metrics	12-29
KPIs	12-31
Data	12-32
Report formats	12-33
12.5. Collaborating with developers	12-34
Collaborating with developers	12-35
Collaborating with developers	12-36
Unit summary	12-37
Review questions	12-38
Review answers	12-39
Exercise: Running tests and simulations in the Business console	12-40
Exercise objectives	12-41
Unit 13. Working with Decision Center administrative tools	13-1
Unit objectives	13-2
Topics	13-3
13.1. Managing decision services with the Enterprise console	13-4
Managing decision services with the Enterprise console	13-5
Administration tools in the Enterprise console	13-6
Opening projects in Enterprise console	13-7
Importing and exporting projects in Decision Center (1 of 2)	13-8
Importing and exporting projects in Decision Center (2 of 2)	13-9
Erasing projects in Decision Center	13-10
13.2. Branch management	13-11
Branch management	13-12
Branches	13-13
Snapshots	13-14
Managing branches in the Business console	13-15
13.3. Managing users and permissions	13-16
Managing users and permissions	13-17
Mandatory role definitions to access Decision Center	13-18
Custom users and groups	13-19
User management in Business console	13-21

Working with LDAP: Overview	13-22
Working with LDAP: Process	13-23
Creating an LDAP connection in Business console	13-24
Importing groups and users from LDAP	13-25
Assigning permissions to imported groups	13-26
Assigning users to groups	13-27
Unit summary	13-28
Review questions	13-29
Review answers	13-30
Exercise: Working with management features in Decision Center	13-31
Exercise objectives	13-32
Exercise: Managing user access in Decision Center	13-33
Exercise introduction	13-34

Unit 14. Introducing decision governance.....	14-1
Unit objectives	14-2
Topics	14-3
14.1. What is decision governance?	14-4
What is decision governance?	14-5
What is decision governance?	14-6
Why decision governance is required	14-7
Traditional approach to maintenance	14-8
Decision management increases communication	14-9
Decision governance goal	14-11
Definition of project governance	14-12
Business Decision Management group	14-13
Governance and agile development	14-15
Implementing governance	14-16
14.2. Operational Decision Manager support for governance	14-18
Operational Decision Manager support for governance	14-19
Decision lifecycle in Operational Decision Manager (1 of 2)	14-20
Decision lifecycle in Operational Decision Manager (2 of 2)	14-21
Discussion	14-22
How can you apply governance?	14-23
14.3. Introducing the decision governance framework	14-24
Introducing the decision governance framework	14-25
Decision governance framework overview	14-26
Recall: Operational Decision Manager tools	14-27
Using the decision governance framework	14-29
Governance in the Business console	14-30
States and user roles	14-31
Releases	14-32
Release governance	14-33
Release sequence	14-34
Change activity governance	14-35
Validation activity governance	14-36
Deploying releases	14-37
Unit summary	14-38
Review questions (1 of 2)	14-39
Review questions (2 of 2)	14-40
Review answers (1 of 2)	14-41
Review answers (2 of 2)	14-42
Exercise: Working with the decision governance framework	14-43
Exercise objectives	14-44

Unit 15. Course summary	15-1
Unit objectives	15-2
Course objectives	15-3
Course objectives	15-4
Earn an IBM Badge	15-5
To learn more on the subject	15-6
Enhance your learning with IBM resources	15-7
Unit summary	15-8
Course completion	15-9
Appendix A. List of abbreviations	A-1
Appendix B. IBM ODM on Cloud.....	B-1
Appendix C. Business Rules Service on IBM Bluemix.....	C-1

Trademarks

The reader should recognize that the following terms, which appear in the content of this training document, are official trademarks of IBM or other companies:

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide.

The following are trademarks of International Business Machines Corporation, registered in many jurisdictions worldwide:

Bluemix®

ILOG®

Orchestrate®

SPSS®

Express®

Insight™

Rational®

WebSphere®

IBM Bluemix™

Notes®

Redbooks®

z/OS®

Intel, Intel Xeon and Xeon are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Microsoft and Windows are trademarks of Microsoft Corporation in the United States, other countries, or both.

Java™ and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

VMware is a registered trademark or trademark of VMware, Inc. or its subsidiaries in the United States and/or other jurisdictions.

Social® is a trademark or registered trademark of TWC Product and Technology, LLC, an IBM Company.

Other product and service names might be trademarks of IBM or other companies.

Course description

Managing Decisions in IBM Operational Decision Manager V8.9

Duration: 5 days

Purpose

This course introduces business analysts to IBM Operational Decision Manager V8.9. You learn the concepts and skills that are necessary to capture, author, validate, and manage business rules with Operational Decision Manager.

IBM Operational Decision Manager provides a complete platform for managing business rules by helping to automate and govern decisions across processes and applications. This course focuses on the iterative nature of working collaboratively with business policy experts and development teams on rule projects. Through instructor-led presentations and hands-on lab exercises, you learn about the core features of Operational Decision Manager. You also receive intensive training in modeling business rule vocabulary, rule discovery, rule authoring, and rule governance and management. The course uses realistic scenarios and a case study to illustrate the principles and good practices for discovering, analyzing, and authoring business rules.

The lab environment for this course uses Windows 2012 Server R2 Standard.

Audience

This course is designed for business analysts.

Prerequisites

- Experience with identifying and defining business policies and rules
- A basic understanding of business models

Objectives

- Describe the benefits of implementing an Operational Decision Manager solution
- Identify the main user roles and tasks that are involved in designing and developing an Operational Decision Manager solution
- Explain modeling concepts and the UML notation that is relevant to modeling for business rules
- Define and implement object models for business rules
- Set up the rule authoring environment in Designer by working with decision services and synchronizing across development and business environments
- Transform business policy into rule statements and make sure that they form a complete and coherent set of rules

- Use the Operational Decision Manager rule editors to author business rules and decision tables
- Run tests and simulations in the Decision Center Business console to validate decision logic and rule changes
- Work with Decision Center decision service administration tools
- Manage user access and permissions in the Business console
- Use Operational Decision Manager tools to support decision governance

Agenda

**Note**

The following unit and exercise durations are estimates, and might not reflect every class experience.

Day 1

- (00:30) Course introduction
- (02:00) Unit 1. Introducing IBM Operational Decision Manager V8.9
- (01:30) Exercise 1. Operational Decision Manager in action
- (01:30) Unit 2. Modeling for business rules
- (01:00) Exercise 2. Building the model on paper
- (01:30) Exercise 3. Implementing the model

Day 2

- (01:00) Unit 3. Understanding decision services
- (01:30) Exercise 4. Setting up a decision service
- (01:00) Unit 4. Working with the BOM
- (01:00) Exercise 5. Working with the BOM
- (01:00) Unit 5. Introducing Decision Center
- (01:30) Exercise 6. Exploring the Decision Center Business console

Day 3

- (01:00) Unit 6. Introducing rule authoring
- (00:30) Exercise 7. Understanding the case study
- (01:00) Unit 7. Discovering and analyzing rules
- (01:30) Exercise 8. Discovering rules
- (01:30) Exercise 9. Analyzing rules
- (01:00) Unit 8. Working with conditions in rules
- (01:30) Exercise 10. Working with conditions in rules

Day 4

- (01:00) Unit 9. Working with definitions in rules
- (01:00) Exercise 11. Working with definitions in rules
- (01:00) Unit 10. Writing complete rules
- (01:45) Exercise 12. Writing complete rules
- (00:30) Unit 11. Authoring decision tables
- (01:00) Exercise 13. Authoring decision tables
- (01:45) Exercise 14. Authoring rules: Putting it all together

Day 5

- (01:00) Unit 12. Running tests and simulations
- (00:45) Exercise 15. Running tests and simulations in the Business console
- (01:00) Unit 13. Working with Decision Center administrative tools
- (00:30) Exercise 16. Working with management features in Decision Center
- (01:30) Exercise 17. Managing user access in Decision Center
- (01:45) Unit 14. Introducing decision governance
- (01:30) Exercise 18. Working with the decision governance framework
- (00:30) Unit 15. Course summary

Unit 1. Introducing IBM Operational Decision Manager V8.9

Estimated time

02:00

Overview

This unit introduces you to Operational Decision Manager and describes the advantages of implementing a decision management solution in your organization.

How you will check your progress

- Checkpoint
- Exercise



How to check online for course material updates



Note: If your classroom does not have internet access, ask your instructor for more information.

Instructions

1. Enter this URL in your browser:
ibm.biz/CloudEduCourses
2. Find the product category for your course, and click the link to view all products and courses.
3. Find your course in the course list and then click the link.
4. The wiki page displays information for the course. If the course has a course corrections document, this page is where it is found.
5. If you want to download an attachment, such as a course corrections document, click the **Attachments** tab on the page.

Comments (0) Versions (1) **Attachments (1)** About
6. To save the file to your computer, click the document link and follow the prompts.

Unit title

© Copyright IBM Corporation 2017

Figure 1-1. How to check online for course material updates

Unit objectives

- Explain the benefits of using Operational Decision Manager
- Identify the need for governance
- Map the various roles that are involved in a decision management solution to roles in your organization
- Identify the tasks that are performed on various Operational Decision Manager modules, and which user roles perform them

Unit title

© Copyright IBM Corporation 2017

Figure 1-2. Unit objectives

Topics

- What is a business rule?
- Why the need for decision management?
- What is operational decision management?
- Introducing IBM Operational Decision Manager
- ODM packaging
- Integration with IBM product family
- Operational Decision Manager roles
- Governance and decision management

Unit title

© Copyright IBM Corporation 2017

Figure 1-3. Topics

1.1. What is a business rule?

What is a business rule?

Unit title

© Copyright IBM Corporation 2017

Figure 1-4. What is a business rule?

What is a business rule?

- A business rule is a statement of business logic that:
 - Business users can author and understand
 - Applications can invoke for execution
- From the business perspective:
 - A business rule is a precise statement that describes, constrains, or controls some aspect of your business
- From the IT perspective:
 - Business rules are a package of executable business policy statements that can be invoked from an application

Unit title

© Copyright IBM Corporation 2017

Figure 1-5. What is a business rule?

How you define a business rule can depend on your perspective. Generally, a business rule is a statement of business logic that can be understood both by the business users who author the rules, and by the application that invokes the rules for execution.

From the business perspective, rules relate to the expression of decisions that are needed to respond to a business request. If you are a business user, you might define a business rule as a precise statement that describes, constrains, or controls some aspect of your business.

From the development or IT perspective, rules relate to the implementation of a decision system, and integration into an existing infrastructure, such as application-based or service-oriented architecture. A developer sees business rules as a package of executable business policy statements that can be invoked from an application.

From business policy to business rules

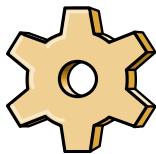
- Rules formalize business policy as “if-then” statements
- Example:

Business policy



When customers spend more than \$1500 in a single transaction, they should be upgraded

Formal rule



If the customer's category is Gold and the value of the customer's shopping cart is more than \$1500

Then change the customer's category to Platinum

Unit title

© Copyright IBM Corporation 2017

Figure 1-6. From business policy to business rules

Business policies are statements that are used for decisions, such as pricing for insurance or loan underwriting, eligibility approvals for social or health services, or product recommendations for online purchases. A single policy can require hundreds of rules to implement it.

Business rules formalize a business policy into a series of “if-then” statements.

Discussion

What is a rule?

Unit title

© Copyright IBM Corporation 2017

Figure 1-7. Discussion

This discussion is an opportunity to discover more about the work experience and expertise of your fellow students.

If you are a classroom or online student, be prepared to share your ideas with the class.

If you are a self-paced student, you can complete this activity as an independent exercise.

Discussion: What is a rule?

1. Write your definition of a business rule.

2. Provide examples of business rules, from either your domain or from the following application types:
 - Insurance: Online quotation
 - Financial services: Loan application
 - Telecommunications: Choosing a rate plan

Unit title

© Copyright IBM Corporation 2017

Figure 1-8. Discussion: What is a rule?

Now that you have the IBM definition of a rule, take a few minutes to write down your own definition, along with some examples of rules that are used in your organization.

For classroom and online students: When you are done with your notes, discuss your ideas with the class.

For self-paced students: Take some time to consider these questions and write your notes here.

1.2. Why the need for decision management?

Why the need for decision management?

Unit title

© Copyright IBM Corporation 2017

Figure 1-9. Why the need for decision management?



Why is there a need for decision management?

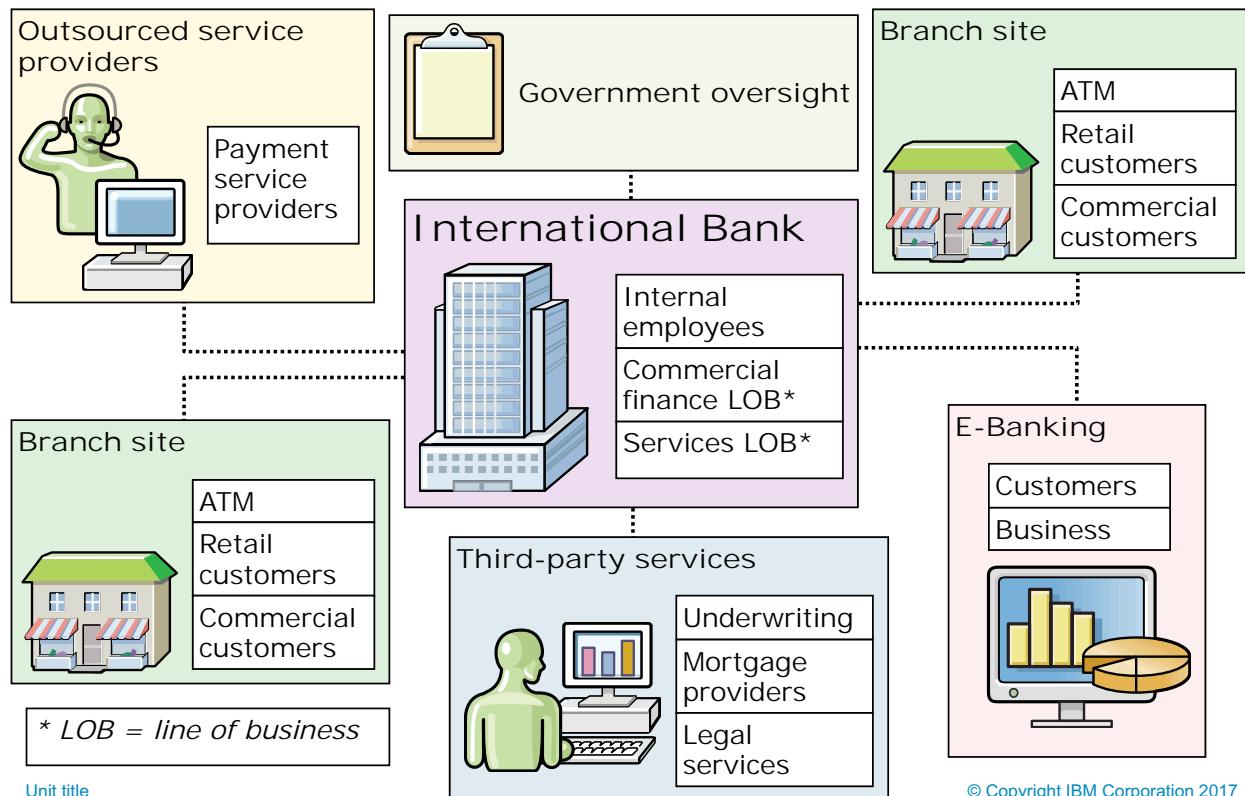


Figure 1-10. Why is there a need for decision management?

Every business, government, and industry is challenged with thousands of daily operational decisions.

Business rules have been talked about for years, but decision management is more than just rules.

Decision management combines business expertise with technology, and thus becomes an extension of the decisions business people would make if they had unlimited time to consider those decisions.

As illustrated on this slide, the financial industry is a typical example of the current business environment for many domains, which includes:

- Multiple channels and platforms
- Government regulations that vary according to geography
- A broad network of relationships and interactions between employees, customers, suppliers, and partners, along with internal and external processes and systems

Within this network, businesses make thousands, even millions, of *repeated* and *repeatable* decisions. Those decisions need to be consistent across the organization, but might vary according to the context. The dynamic nature of this environment requires business agility to respond to change by making better decisions in real time.

For example, as illustrated on this slide, consider the financial industry. Whether the decision is to underwrite a mortgage, extend credit, or provide a loan, every day this organization makes thousands, even millions, of decisions that are repeated and repeatable.

Regardless of the business domain, the right decision must be based on the *complete* business context. The decision might differ from one situation to the next. For each business situation, you want a decision that leads to the next best course of action within the business process.

Decision management focuses on how to *improve repeatable* decisions through automation. Obviously, *not every business decision can be automated*. But, for *repeatable* decisions, the technology makes it possible to record the rules you use in your business every day, codify those rules in a user interface, and make them easy to change and manage.



Questions

How does this picture relate to your industry and organizational structure?

What is decision management?

- A business discipline that combines business expertise with software
 - Automate, optimize, and govern *repeatable* business decisions
 - Capture, change, and govern decision logic in a controlled and scalable way
 - Automate decisions so that they can be called in real time by processes, applications, and other business solutions

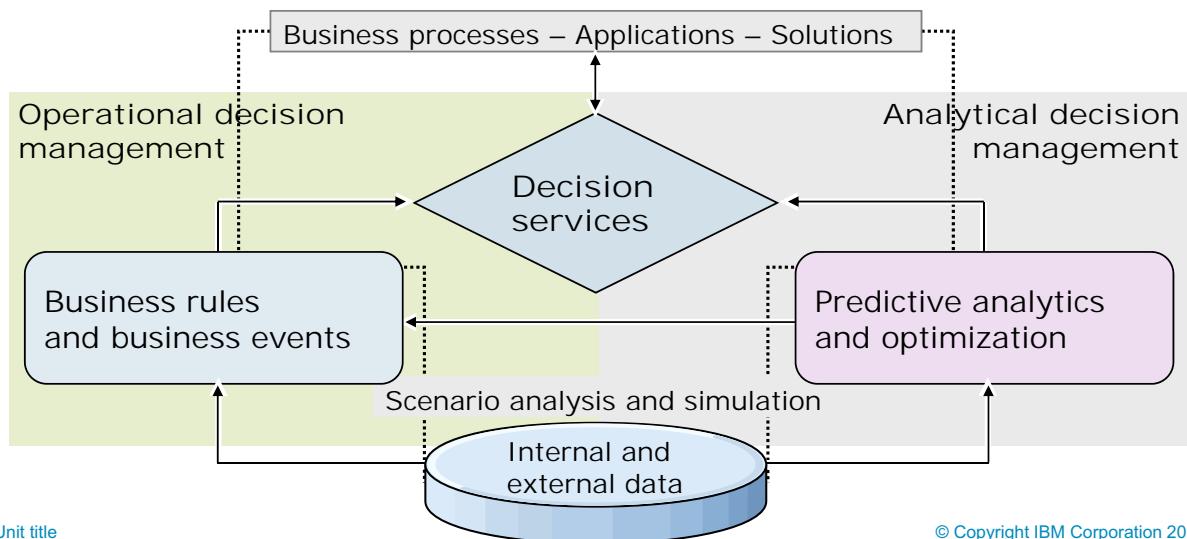


Figure 1-11. What is decision management?

Decision management is an extension of the decisions business people would make if they had unlimited time to consider those decisions. Not every business decision can be automated. However, for repeatable decisions, the technology makes it possible to record the rules you use in your business every day. It also makes it possible for you to codify those rules in a user interface, and easily change and manage those rules so that the system can decide for you.

As shown in this diagram, decision services encapsulate the behavior of the automated decision. Automation of the decision logic means that those decisions can be called in real time by processes, applications, and other business solutions.

The key enablers of decision management are generally viewed to be business rules and predictive analytics. Integration between these two entry points provides a complete approach to decision management.

- (*Right side of screen*) In **analytical decision management**, decision history is analyzed to build a model that can be used to predict the best decision response for the future. Decisions can be optimized for the specific contexts in which they are being made to ensure the best possible decision at the current moment and situational context. You can also use data to discover insights and continuously improve decisions over time.
- (*Left side of screen*) In **operational decision management**, policy, suggested practices, and business experience are used to write rules that describe decision making and identify

situations that require a response. In many cases, the results from analytical decision management can enhance the operational decisions.

To measure the effectiveness of decision management, you define key performance indicators (KPIs) that relate to the overall business goals. By using KPIs with scenario analysis and simulation, you can assess how decision changes might affect the behavior of business systems.



Note

Analytical decision management is not the focus of this course. But take a minute just to look at what it covers.

1.3. What is operational decision management?

What is operational decision management?

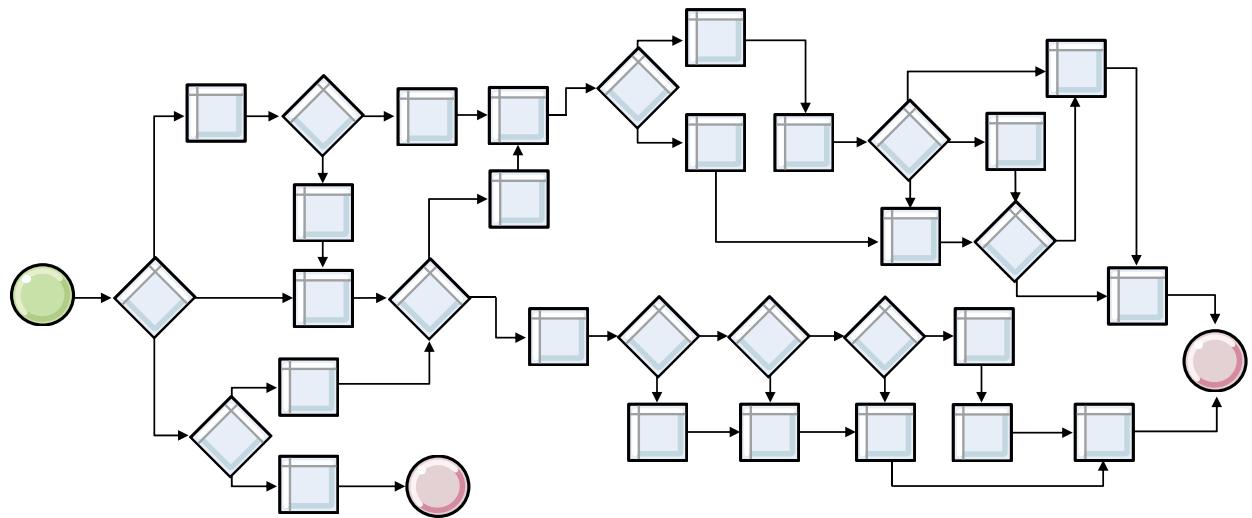
Unit title

© Copyright IBM Corporation 2017

Figure 1-12. What is operational decision management?

Challenges to decision automation challenges

- Decisions are locked in processes and applications
- Programming skills are required to create and modify decision logic
- IT bandwidth limits the speed of business change
- Manual intervention increases costs and reduces customer satisfaction



Unit title

© Copyright IBM Corporation 2017

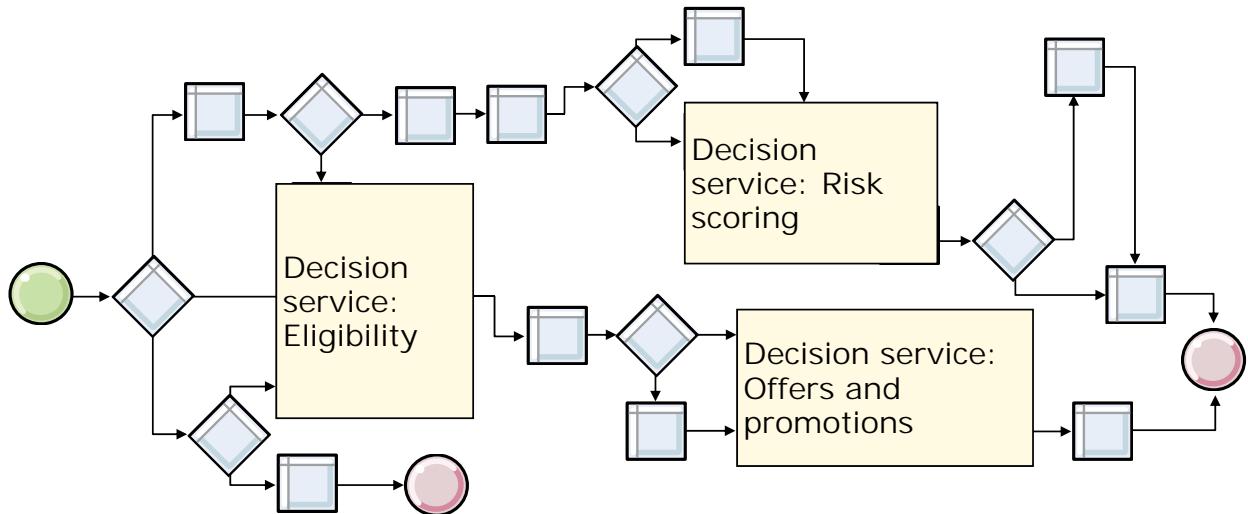
Figure 1-13. Challenges to decision automation challenges

Business agility depends on responsive, intelligent decision automation. However, organizations might have millions of rules in spreadsheets. Or, the automation of business policies might be hardcoded in the application. If so, rules might be invisible, incorrect, and unmanageable without help from the development team. Changes to policy might take months to implement, which can severely limit an organization's ability to be flexible.

Also, millions of events that potentially require a business response might be flowing freely through the IT infrastructure. Without decision management tools, business users cannot identify and respond to the volume and complexity of these situations by themselves.

Using operational decision management

- Facilitates reuse of decision assets across processes
- Empowers business users to own, author, and update decision logic
- Increases response time to changing market conditions
- Maximizes automation and straight-through processing



Unit title

© Copyright IBM Corporation 2017

Figure 1-14. Using operational decision management

With operational decision management, the decision logic is externalized from processes and applications. This externalization of decision logic from processes and applications provides visibility for business users and enables reuse across business systems.

As depicted on this slide, many of the tasks in a process can be automated and replaced with decision services.

The separation of decision logic from application code puts business experts in control of the business logic instead of the IT team. Business experts can define and manage the business logic themselves. This separation reduces the amount of time and effort that are required to update that business logic in production systems. It also increases the ability of an organization to respond to changes in the business environment.

Decision automation also reduces the load on people, which frees them to use their skills more effectively and maximizes straight-through processing. Automation provides faster, consistent, predictable, and traceable decision making.

To manage decisions, you must first recognize the context in which decisions are made. Modeling is really the key. Decisions are always made within the context of a business process, so you start with process modeling and management. Within the process model, you can then identify the decision points and capture the individual rules that support that decision.

You might wonder why it is necessary to manage rules separately from processes. Strong interplay exists between business process management and decision management. However, even if a process is not fully automated, you can still identify tasks within the process that would be better implemented as a decision point. Therefore, you would improve business performance.

What is operational decision management?

- Systematic use of technology to manage the process of making operational decisions across critical business systems
- Externalizes decision logic from application code
 - Decision logic is managed independently from the application
 - Changes to business policy do not affect application code
- Empowers business users to maintain business rules directly with limited dependence on the IT department

Unit title

© Copyright IBM Corporation 2017

Figure 1-15. What is operational decision management?

Operational decision management typically involves high-volume, repeatable decisions that are based on business rules. It also incorporates business events. This style of decision management is generally, but not always, used for high-volume, repeatable decisions, which are often based on hundreds or thousands of business rules.

Operational decisions rely on known business expertise, such as corporate policies and external regulations. Operational decisions also rely on enterprise knowledge:

- Corporate suggested practices (explicit expertise that defines how the business is run)
- “Know-how” (implicit knowledge that is used in running the business that is not codified)

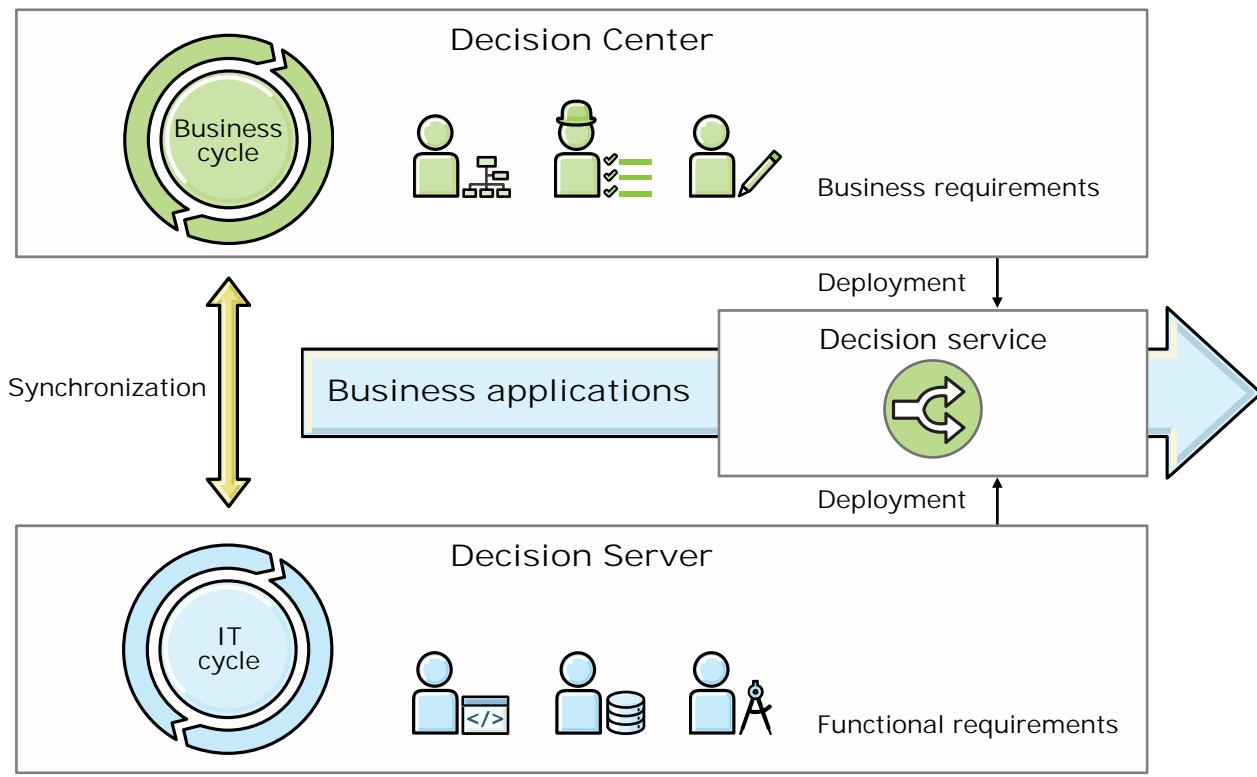
The ability to deal with change in operational systems is directly related to the decisions that those systems can make. Every transaction, order, customer interaction, or process depends on decisions, which depend, in turn, on particular internal or external requirements and situational contexts. Any change to those requirements or situations can affect decisions, including those decisions that are handled automatically within business systems.

With Operational Decision Manager, you manage decisions separately from business applications.

IBM Training



Synchronized business and IT cycles



Unit title

© Copyright IBM Corporation 2017

Figure 1-16. Synchronized business and IT cycles

Decision management and application development lifecycles can evolve in parallel.

The lifecycles for decision management are much shorter than application development lifecycles. The separation of decisions from processes and applications allows these lifecycles to be managed asynchronously. Decisions can evolve as the business context requires, without putting an extra load on the application development team.

For example, application developers might work in a semiannual cycle: a new application version is developed every six months in response to changing application infrastructure and other core business requirements. At the same time, policy managers might work on a weekly cycle to deliver decision updates in response to an evolving market, changing regulatory environment, or new patterns of events. Changes to rules do not affect code, which means that policy managers can review and even modify policies without first needing to contact the development team. Each time the application evolves, the decision management environment is synchronized with the application.

Within your organization, you can negotiate the degree of dependence between business and development teams. Dependency can range from limited review by business users of the decisions that developers implement, to giving business users complete control over the specification, creation, testing, and deployment of the rules.

1.4. Introducing IBM Operational Decision Manager



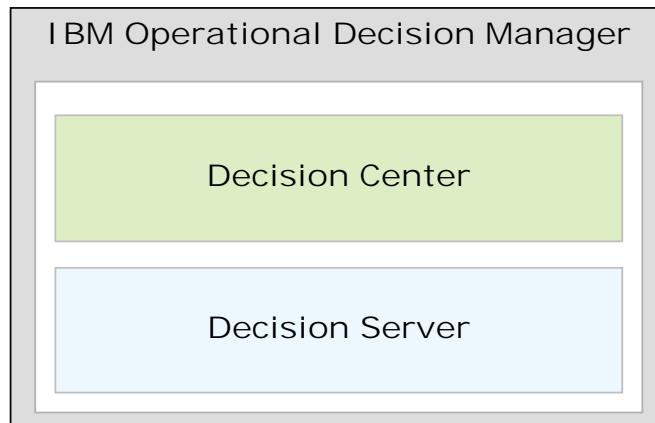
Unit title

© Copyright IBM Corporation 2017

Figure 1-17. Introducing IBM Operational Decision Manager

Operational Decision Manager (1 of 2)

- Provides two main environments:
 - Decision Center for business users
 - Decision Server for technical users



Unit title

© Copyright IBM Corporation 2017

Figure 1-18. Operational Decision Manager (1 of 2)

Operational Decision Manager provides comprehensive capabilities to intelligently automate a wide range of decisions across business processes and applications.

- Provides organizations the ability to build highly flexible, adaptable solutions that can detect and react to data patterns as they occur within a specified time period
- Provides the appropriate decision response to transactional and process-oriented business systems
- Improves the quality of transaction and process-related decisions that are made repeatedly
- Determines the appropriate course of action, according to the specific context of each situation
- Provides an environment for business experts to author and maintain decision logic in partnership with IT
- Provides an integrated management repository for rule-based and event-based decisions
 - This repository supports governance and change management
- Executes real-time decisions precisely and reliably based on the context of specific interactions

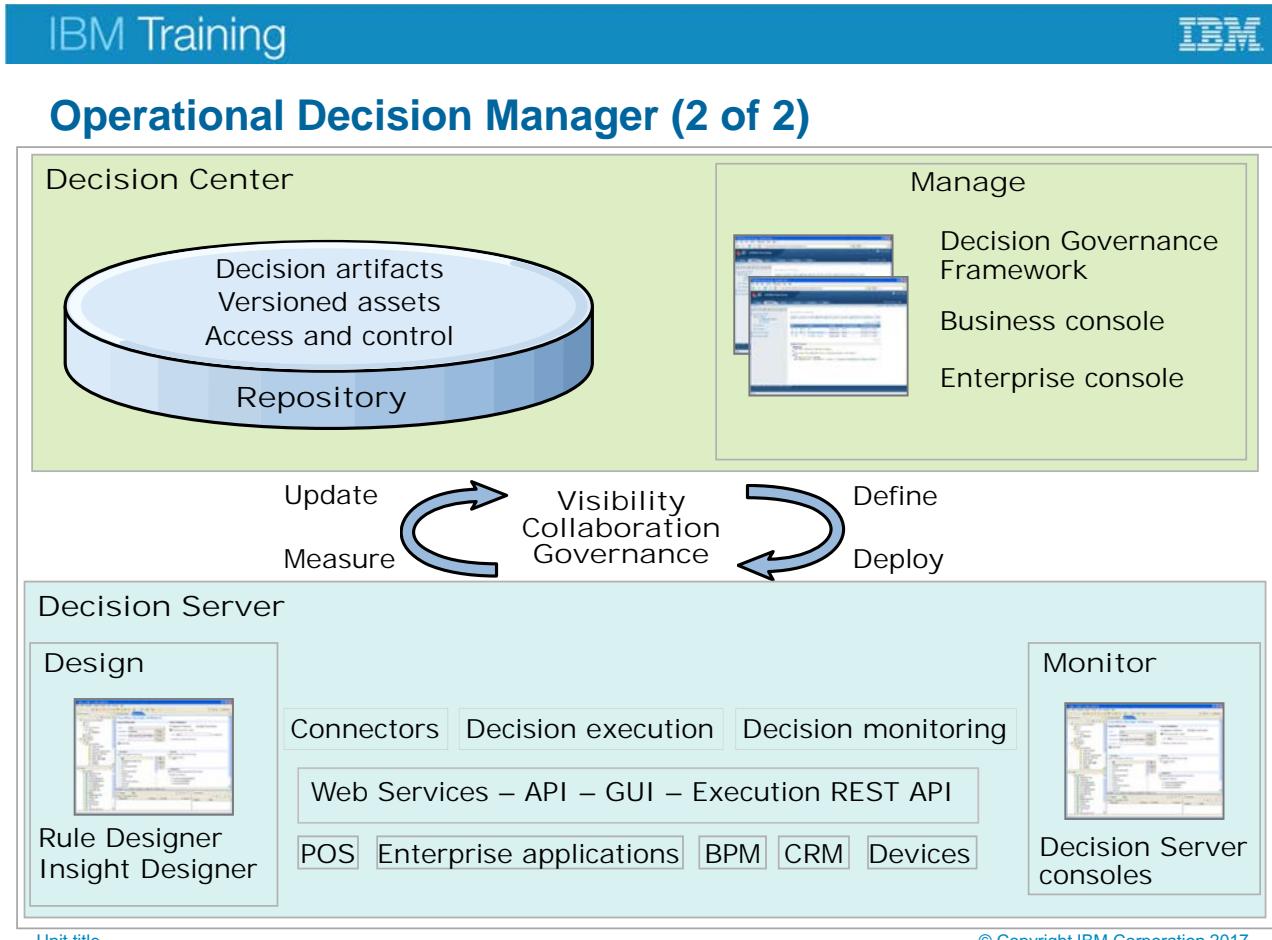


Figure 1-19. Operational Decision Manager (2 of 2)

Business rules and the associated business object model are initially created and edited in the Eclipse-based Rule Designer by developers, and later created and edited in the Decision Center web console by business users. The rules are stored in a repository and are deployed to the Decision Server as RuleApps for testing and execution.

As you can see in this diagram, Operational Decision Manager consists of a set of modules that operate in different environments. However, these modules also work together to provide a comprehensive platform for managing and executing business rules and events.

Decision Center provides all the tools for business users to define and govern business rule and business event-based decision logic.

Through the capabilities of Decision Center, the entire organization is aligned in the implementation of automated decision services.

Decision artifacts for (both rules and event logic) are stored in a centralized *repository* with version control, release management, and secure access.

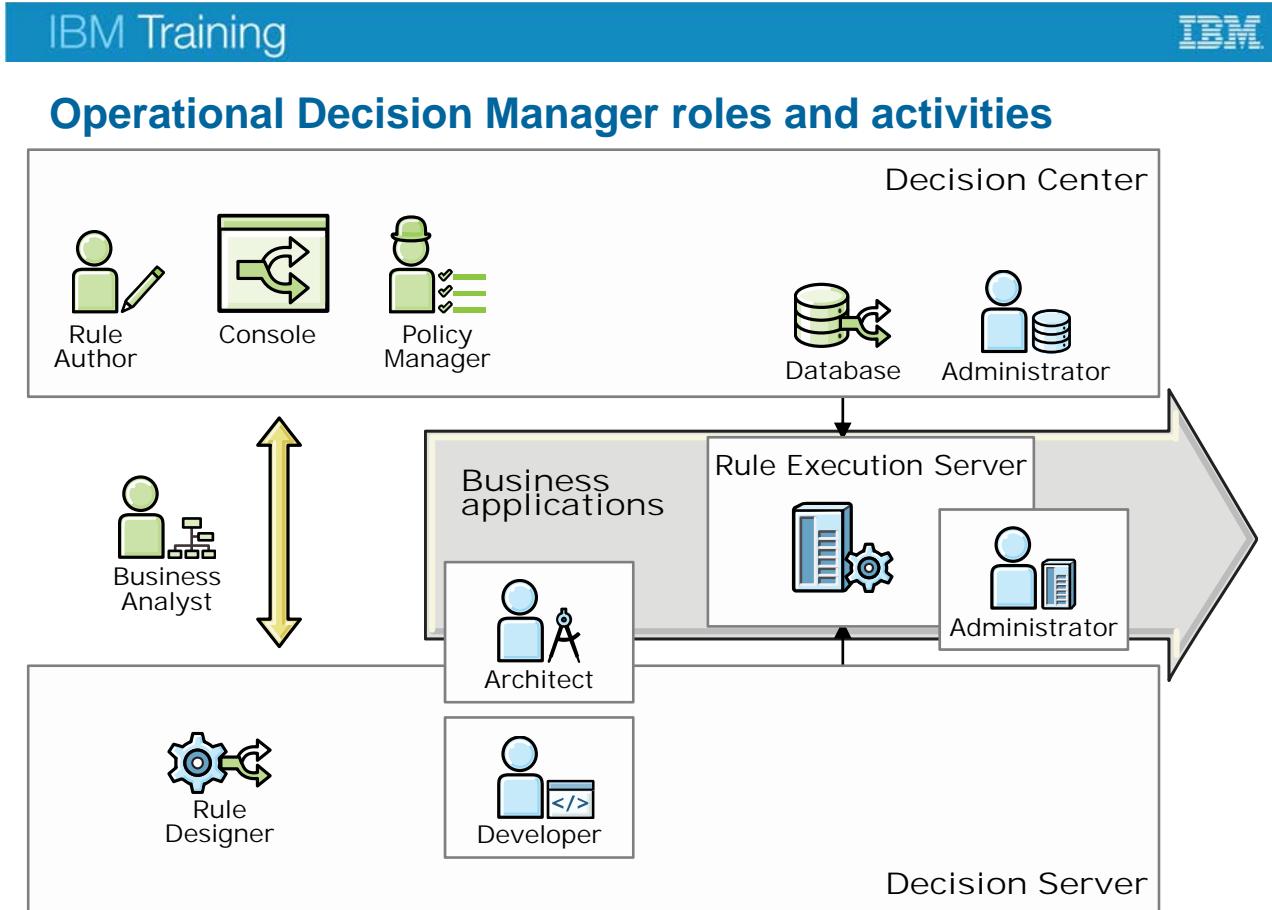
Decision Center also comes with various user interfaces:

- Business console, which provides a social collaboration environment for rule authoring and validation, deployment, and release management.

- Enterprise console, which is a web console that provides a full set of authoring and management capabilities.

Decision Server is for technical users and it contains all runtime components and Eclipse-based development tools.

- Rule Designer and Insight Designer are integrated as Eclipse plug-ins so that you can design, develop, and synchronize with the business environment.
- Decision Server is also the execution environment for business rules and events, and it provides execution management and monitoring to detect event-based patterns, process hundreds, or even thousands of business rules, and determine how to respond.



Unit title

© Copyright IBM Corporation 2017

Figure 1-20. Operational Decision Manager roles and activities

The Operational Decision Manager modules are aimed at specific user roles, which are based on their varied skill sets. Synchronization mechanisms allow developers and business users to collaborate on the same project, while working in their own environments and at their own pace.

This graphic shows an overview of the different modules that are used for business rules. You see the environment in which they are used, and how they work together through synchronization and deployment.

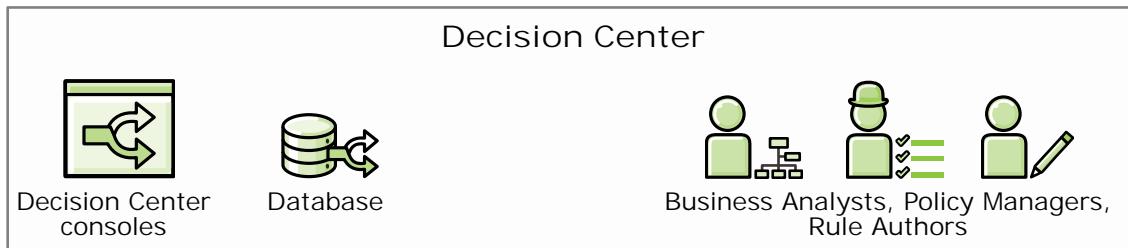
Operational Decision Manager has three main environments:

- The development environment in Designer
- The business rule management and authoring environment, in Decision Center
- The rule execution environment, which can be managed through Rule Execution Server

Next, you learn more about the modules.

Decision Center: Authoring and maintaining business rules

- Decision Center is the main authoring and management environment for business users
 - Decision Center Business console
 - Decision Center Enterprise console
 - Decision Center database



Unit title

© Copyright IBM Corporation 2017

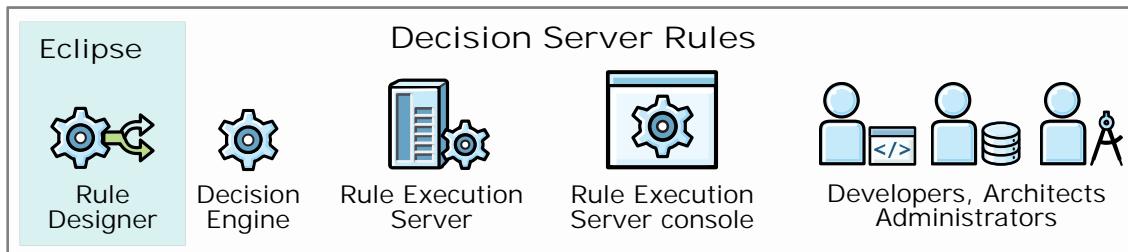
Figure 1-21. Decision Center: Authoring and maintaining business rules

Decision Center encompasses the modules that are shown here:

- Decision Center consoles
 - Business console
 - Enterprise console
- Decision Center database

Decision Server: Development and execution

- Decision Server is the technical environment for development, deployment, and execution of decision services
 - Rule Designer
 - Decision Engine
 - Rule Execution Server
 - Rule Execution Server console



Unit title

© Copyright IBM Corporation 2017

Figure 1-22. Decision Server: Development and execution

Decision Server encompasses the tools for development, deployment, integration, and managed execution of decision services.

Decision Server Insights

- Decision Server Insights provides context awareness and situation detection through complex event processing
 - Insights Designer
 - Insight Server
 - Insight Inspector



Unit title

© Copyright IBM Corporation 2017

Figure 1-23. Decision Server Insights

Decision Server Insights has a structure similar to ODM Decision Server Rules. It includes:

- Insights Designer: A development environment in Eclipse
- Insight Server: A runtime environment that handles complex event processing and agent execution

Monitoring tools are also available, including Insight Inspector, which is a browser-based tool for visualizing event activity.

Insight Designer offers the following features:

- Eclipse interface to develop rule-based, event-driven solutions
- Develop solutions that capture business models and logic through natural-language editors
- Solutions route events to entities through agents or services and use business rules to process responses
- Solutions include model definition, business rules, and analytics
- Connectivity definitions determine inbound and outbound endpoints to receive and send events between solution and external systems

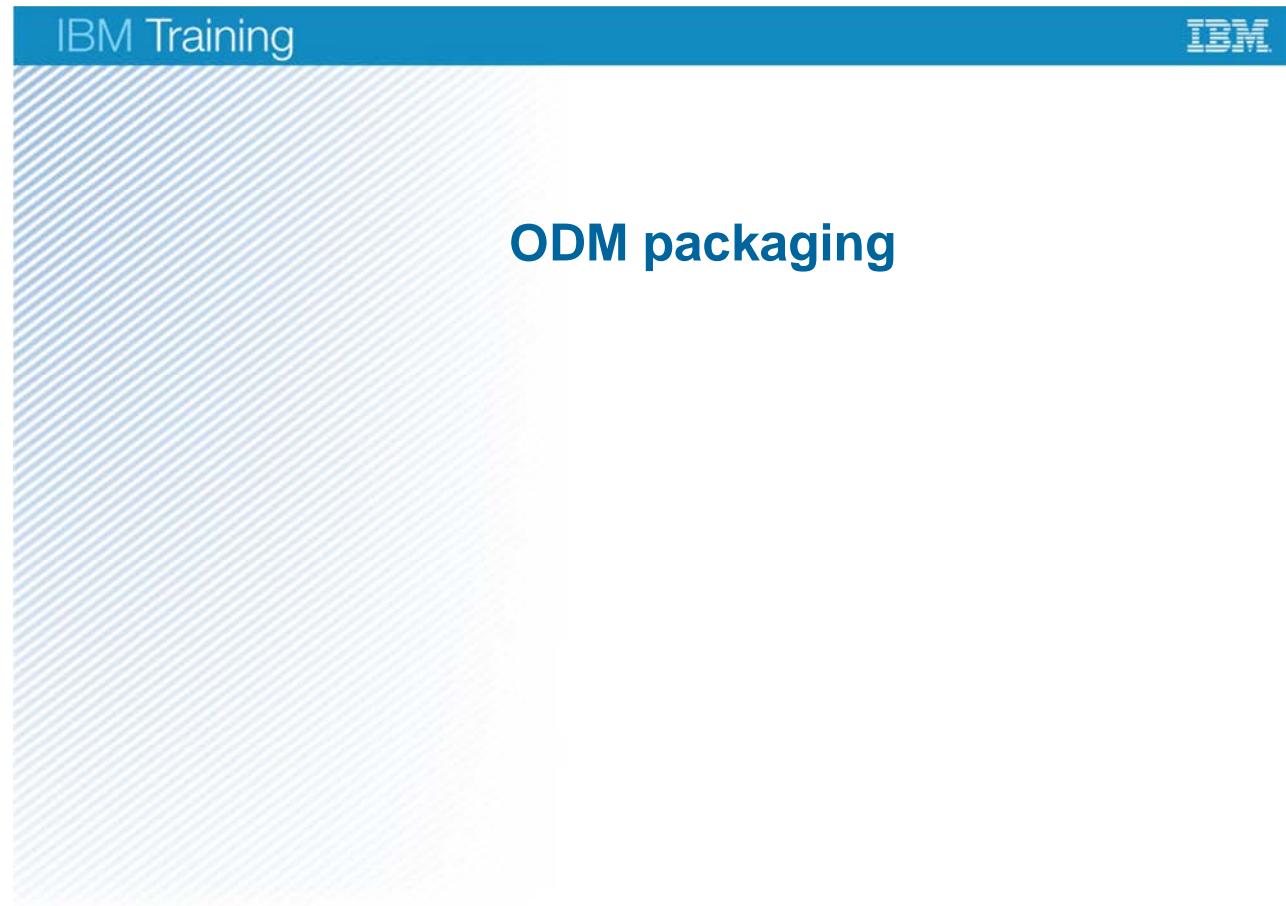
Insight Server Runtime is based on WebSphere Liberty and Extreme Scale:

- Elastic and scalable in-memory compute and data grid
- Maintains a stateful context of business entities
- Applies event-processing logic at the time of interaction

Insights Inspector provides the capability to:

- Visualize timeline of event activity for entities

1.5. ODM packaging



Unit title

© Copyright IBM Corporation 2017

Figure 1-24. ODM packaging

Options for using ODM

ODM is available in these environments:

- On premises
 - Complete installation in your environment
- ODM on Cloud
 - Hosted environment for Decision Server and Decision Center
- Business Rules Service on IBM Bluemix
 - Hosted environment for Decision Server only

Unit title

© Copyright IBM Corporation 2017

Figure 1-25. Options for using ODM

You can work with ODM by installing it in your on environment (on premises), or by using ODM on Cloud or the Business Rules Service on IBM Bluemix.

This course is based on the on-premises version of ODM only. However, the information that you learn here can be applied to all environments.

Operational Decision Manager on-premises editions

Advanced

- Decision Server Rules
- Decision Server Insights
- Decision Center

Standard

- Decision Server Rules
- Decision Center

Express

- Decision Server Rules
- Decision Center
- *Same functionality as Standard, but with license restrictions*

Unit title

© Copyright IBM Corporation 2017

Figure 1-26. Operational Decision Manager on-premises editions

IBM Operational Decision Manager Express offers the same functionality as Standard, but with licensing restrictions that make it an affordable entry point into rules-based decision management capabilities.

IBM Operational Decision Manager Advanced allows organizations to take advantage of the data available to them to enrich and improve their business decisions. With real-time, actionable insight capabilities, companies can now bring together data from multiple sources to identify meaningful patterns and trends that can be applied to operational decisions. As a result, an organization can create and shape business moments by automating decisions in context.

For more information about the IBM Operational Decision Manager packaging, see the product documentation.

Operational Decision Manager (ODM) has three editions to meet client needs:

- **Operational Decision Manager Advanced** provides a full-featured decision management platform that includes Decision Server Insights. You can address the entire decision automation scope, which includes business rules and decision insights capabilities.
- **Operational Decision Manager Standard** includes business rules only. You can use the complete business rules management system (BRMS) from IBM to develop business rule applications.

- **Operational Decision Manager Express** offers the same functionality as Standard, but with licensing restrictions that make it an affordable entry point into rules-based decision management capabilities.

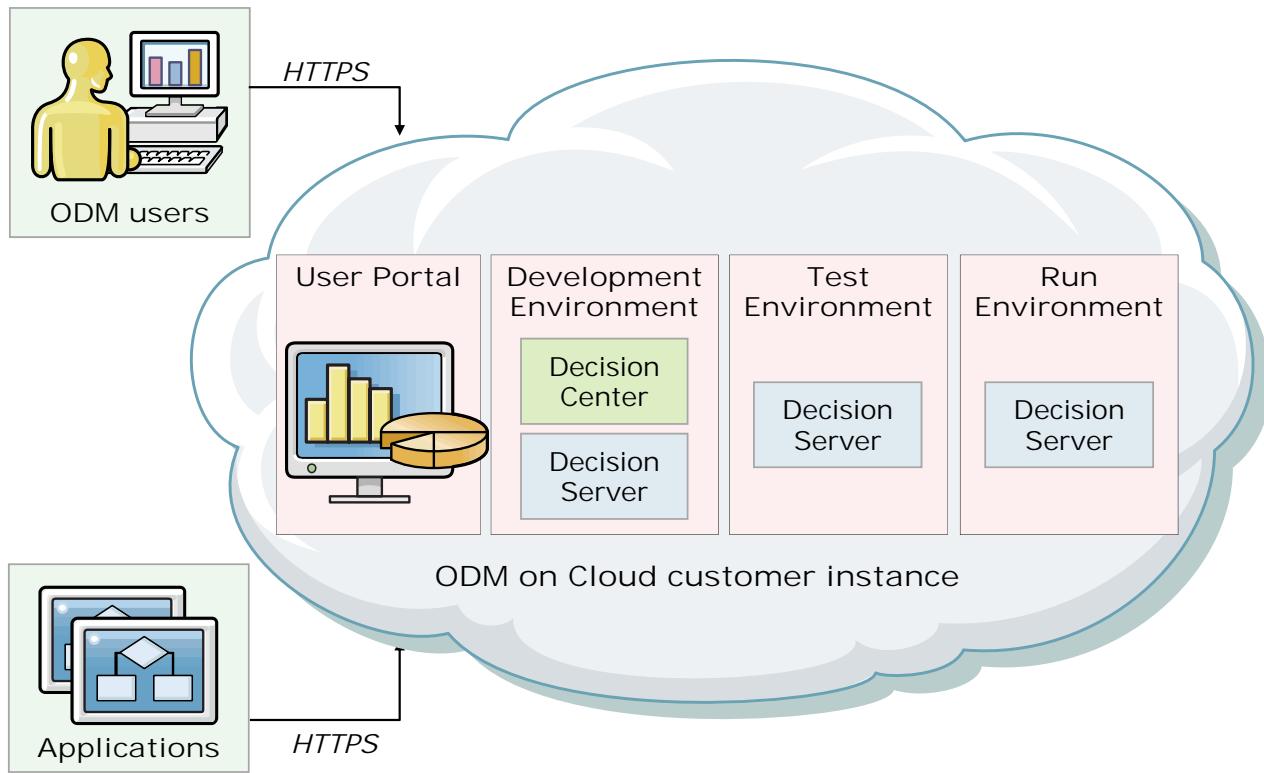
This edition remains the low-cost-of-ownership BRMS solution of IBM. You can get started with small-sized to medium-sized applications. **Operational Decision Manager Express** entitles you to use only the business rules capabilities, but not the events capabilities, within a limited configuration. In particular, you cannot use **Operational Decision Manager Express** in a cluster environment.

This course uses Operational Decision Manager Standard.

For more information about the IBM Operational Decision Manager packaging, see the product documentation.



IBM ODM on Cloud: Three runtime environments



© Copyright IBM Corporation 2017

Figure 1-27. IBM ODM on Cloud: Three runtime environments

In addition to the on-premises set of Operational Decision Manager product offerings, IBM offers IBM ODM on Cloud.

IBM ODM on Cloud is a subscription ODM service that offers a cloud-based, collaborative, and role-based environment.

With cloud computing, users can access applications or computing resources as services from anywhere through their connected devices by using a simplified user interface. Data and services can then be accessed from the cloud through connected devices over the internet.

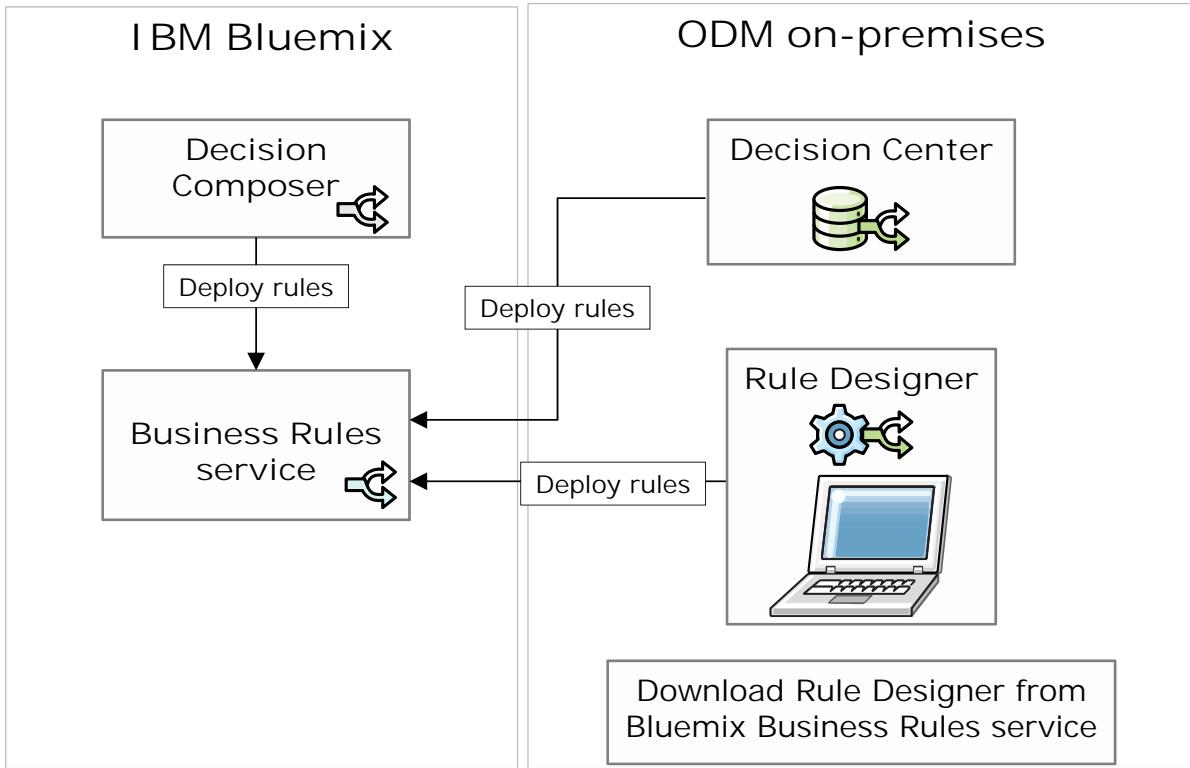
IBM ODM on Cloud offers three runtime environments for decision management: development, testing, and running.

Business users work mainly in the Decision Center consoles that are available in the cloud.

For more information about IBM ODM on Cloud, see Appendix B: IBM ODM on Cloud.



Business Rules Service on IBM Bluemix



Unit title

© Copyright IBM Corporation 2017

Figure 1-28. Business Rules Service on IBM Bluemix

Bluemix is a pay-as-a-service solution, intended to remove steps for managing and maintaining service installation and infrastructure.

The Bluemix Business Rules service provides a development environment to manage decision lifecycle from modeling to deployment, and the execution environment to orchestrate and automate decision logic.

It can be valuable hosting decisions here with cloud-based applications.

The Business Rules service eases the deployment and execution of rules by removing the need to install and maintain the IBM ODM components on an on-premises application server instance. In addition, the Business Rules service is also compatible with the on-premises version of ODM. Also, with the Business Rules service, you don't need to install and maintain an on-premises database to persist rule data. The infrastructure for storage and ruleset execution is provided.

The Business Rules service also provides Decision Composer, which is an experimental tool for decision modeling.

For more information about the Business Rules Service, see Appendix C: Business Rules Service on IBM Bluemix.

1.6. Integration with IBM product family

Integration with IBM product family

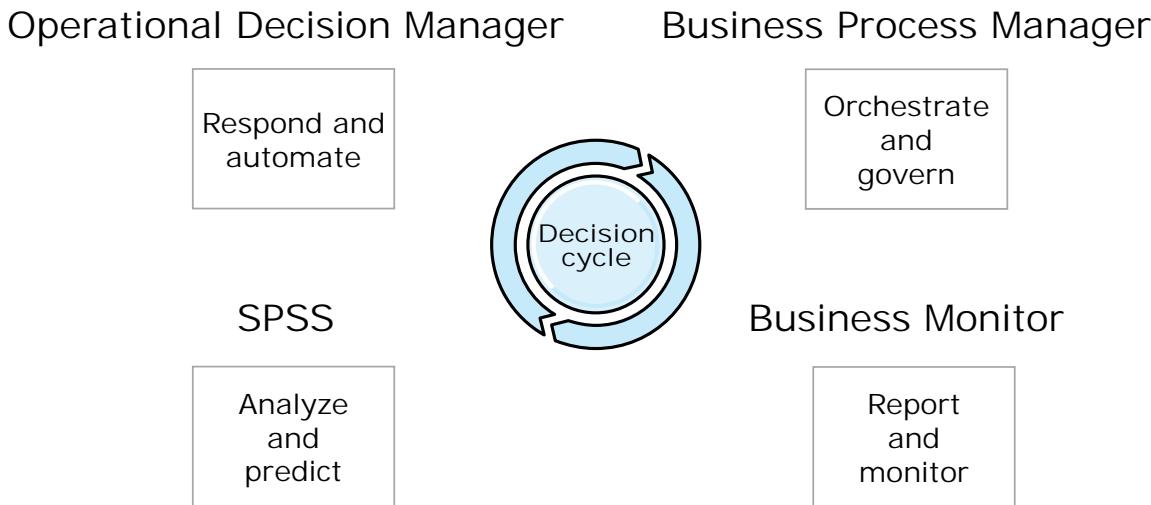
Unit title

© Copyright IBM Corporation 2017

Figure 1-29. Integration with IBM product family

Synergies with other IBM products

- Operational Decision Manager complements other IBM products to empower businesses to automate, manage, and improve the decision cycle



Unit title

© Copyright IBM Corporation 2017

Figure 1-30. Synergies with other IBM products

If you are already familiar with IBM offerings, you probably recognize these products that can be integrated with Operational Decision Management. For more information about how these products complement each other, see the IBM Knowledge Centers for these products.

IBM offerings include the following products.

- For operational decision management:
 - Operational Decision Manager
 - Operational Decision Manager for z/OS
- For analytical decision management:
 - SPSS Decision Management
 - Integration with IBM SPSS provides business predictive analytics for continuous decision improvement.
 - Allows rule-based decision services to use analytic models, for example, in determining fraud or risk, or best offers.
- Business Process Manager orchestrates decision services within processes.

- IBM Process Designer and Integration Designer include wizards to easily integrate existing business rule applications into business processes.
- Integration with IBM Business Monitor provides real-time monitoring of business decisions for decision improvement.

1.7. Operational Decision Manager roles

Operational Decision Manager roles

Unit title

© Copyright IBM Corporation 2017

Figure 1-31. Operational Decision Manager roles

Operational Decision Manager user roles

Business users	
 Business analyst	Bridges between the business side and technical side of a business rule application
 Policy manager	Business expert and owner of business policy
 Rule author	Business domain expert who updates and reviews rules
Technical users	
 Architect	Manages overall deployment, organization of rules, and optimization of rule execution
 Developer	Develops, tests, and deploys business rule applications and event applications
 Administrator	Installs and configures rule management and execution environments

Unit title

© Copyright IBM Corporation 2017

Figure 1-32. Operational Decision Manager user roles

During the development of a decision management solution, various skills are required at different stages of the lifecycle. In ODM, these skills are grouped into a set of business and technical roles.

Developing and maintaining a decision management solution involves various skills that are grouped into two categories:

- Business users: Business analysts, policy managers, and rule authors develop and maintain the decision logic.
- Technical users: Architects, developers, and administrators develop and maintain the rule application.

These roles are explained in more detail in the upcoming slides.

Interaction between roles

- Roles do not correspond to individuals, but to activities and responsibilities
- Tasks might not correspond to a single position in your organization
 - A business expert might be involved in the technical side of things
 - A developer might also be the person who authors and manages the rules
- Communication between the business and technical roles is vital

Unit title

© Copyright IBM Corporation 2017

Figure 1-33. Interaction between roles

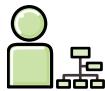
Roles refer to tasks and responsibilities rather than individuals.

Roles might not correspond to a single position in your organization, so crossover between departments might make it difficult to discern who fits into a particular role. For example, a business policy expert might be involved in the technical side of things, and a developer might also be the person who writes and manages the rules.

Expect extensive interaction between business roles and technical roles, particularly during the early stages of developing an application. Having this expectation can ensure that the implementation meets the business view.

Business analyst

- Responsibilities:
 - Designing a formal specification for the rules, with validation from both developers and policy managers
 - Defining the vocabulary that is used in rules
 - Writing and organizing business rules so that rule authors can maintain them
 - Validating that rule execution yields the expected results
- Tools: Rule Designer and Decision Center



Unit title

© Copyright IBM Corporation 2017

Figure 1-34. Business analyst

Business analysts act as a bridge between the business and technical sides of a business rule application.

Business analysts are involved in rule discovery tasks, including process modeling, writing use cases, and defining the vocabulary that is used in rules. They also work with developers to ensure that the implementation matches the business requirements.

Depending on their level of technical knowledge, business analysts can do tasks that are currently described as developer tasks. However, business analysts generally do not write code.

Business analysts can use Rule Designer or Decision Center.

Policy manager

- Responsibilities:
 - Participating in the design of a formal specification for the rules
 - Defining vocabulary elements with the help of business analysts
 - Creating and updating rules
 - Reviewing how the execution of rules is orchestrated
 - Reporting on the status of the business policy
 - Testing rules to ensure that they are written correctly
 - Running simulations to ensure that the rules give the intended business outcome
 - Managing multiple releases
- Tools: Decision Center Business and Enterprise Consoles



Unit title

© Copyright IBM Corporation 2017

Figure 1-35. Policy manager

Policy managers are owners of the decisions within an organization, and are involved in the review, validation, and authoring activities of the decision lifecycle.

A policy manager is a business expert who is responsible for defining business policy definitions, participating in rule discovery and validation of results, and reviewing how the execution of rules is organized. In some organizations, policy managers are also known as the subject matter expert. Examples of policy managers include actuaries, underwriters, and compliance officers for insurance companies or personnel who are in charge of underwriting or pricing in mortgage providers.

Policy managers work with business analysts during the initial discovery phase to validate that business requirements are captured accurately.

Policy managers work in Decision Center.

Rule author

- Responsibilities:
 - Updating and sometimes creating rules
 - Reviewing the business rules
- Tools: Decision Center Business and Enterprise Consoles



Unit title

© Copyright IBM Corporation 2017

Figure 1-36. Rule author

A rule author is a business domain expert who formulates policies into business rules. Rule authors work in Decision Center to update and create rules. They also work with queries and reports to review business rules and event rules.

Architect

- Responsibilities:
 - Managing the overall deployment organization of the rules and making sure that their execution is optimized
 - Defining the project organization so that it is convenient for developers and business users alike
 - Defining the granularity of the rule applications and how they fit into the wider business process
- Tools: Rule Designer



Unit title

© Copyright IBM Corporation 2017

Figure 1-37. Architect

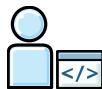
Architects work in Rule Designer. Their responsibilities are listed on the slide.

An architect ensures overall deployment, organization of rules, and optimization of rule execution.

The architect defines the types of rules that are used, and orchestrates their execution in a business rule application. The architect also ensures coherent rule deployment across several business rule applications.

Developer

- Responsibilities:
 - Developing, testing, debugging, and deploying business rule applications
 - Writing the code for rule execution
- Tools: Rule Designer



Unit title

© Copyright IBM Corporation 2017

Figure 1-38. Developer

Developers are familiar with object models, APIs, and the development environment (Java EE application servers, Java SE, and z/OS platforms). Developers are involved in the design, author, test, integration, and deployment activities of the decision lifecycle.

Developers work in Rule Designer to do these tasks:

- Work with business analysts to implement business rule vocabulary
- Set up the authoring environment for rule authors
- Write the invocation code for rule execution
- Write complex rules that business users cannot write
- Implement customizations to meet specific needs

Administrator

- Responsibilities:
 - Deploying and configuring the server and database for Decision Center and Rule Execution Server
 - Managing user access to Decision Center and Rule Execution Server
 - Configuring trace data sources for testing purposes
 - Deploying applications
 - Redeploying rulesets and event assets as changes are made
 - Generating detailed execution reports
 - Tracking and monitoring rule execution
 - Restoring a particular application state
- Tools: Servers for Decision Center or runtime environments



Unit title

© Copyright IBM Corporation 2017

Figure 1-39. Administrator

Administrators install and configure rule management and execution environments. Their responsibilities are listed on the slide.

System administrators work on the servers to ensure that they run smoothly. These servers can be for Decision Center or runtime environments.

Discussion

Who is in charge of what?

Unit title

© Copyright IBM Corporation 2017

Figure 1-40. Discussion

In this activity, consider the questions on the following slide as they relate to your own organization.

If you are a classroom and online student, be prepared to share your ideas with the class.

If you are a self-paced student, you can complete this activity as an independent exercise.

Discussion: Who is in charge of what?

1. Who is in charge of capturing rules from the business policies?
2. Who is in charge of authoring rules?
3. Who is in charge of deploying rules?
4. How does your current role relate to the BRMS roles?

Unit title

© Copyright IBM Corporation 2017

Figure 1-41. Discussion: Who is in charge of what?

Take a moment to think about these questions and map the people from your organization to the role descriptions.

If your organization is involved in an active decision management project, the results from this exercise can be useful to the project manager. It is important to identify, early in a project, whom to include on the rule management team after the solution goes live.

For classroom and online students, when you are done writing your notes, discuss your ideas with the class.

For self-paced students, take time to consider these questions and make some notes.

1.8. Governance and decision management

Governance and decision management

Unit title

© Copyright IBM Corporation 2017

Figure 1-42. Governance and decision management

Applying governance to decision management

- Governance is management of the decision logic lifecycle
 - Govern lifecycle from initial development to deployment and maintenance
- Provides an organizational framework to prevent problems:
 - Defines expectations
 - Assigns roles and responsibilities
 - Verifies performance
- Goals:
 - Efficient collaboration between business and IT teams
 - Ability to demonstrate that an organization accomplished what it said it would accomplish

Unit title

© Copyright IBM Corporation 2017

Figure 1-43. Applying governance to decision management

The advantages of implementing a decision management solution can be lost if governance is not included. Governance is the management of the decision logic lifecycle, from initial development to deployment and maintenance.

Implementing decision management requires that development teams collaborate regularly with business users, starting with the initial design phases of a project. The business team knows which actions must be taken in response to the various business events, and which requirements form the basis of decisions. Modeling makes it easier for the business team to understand the business logic, and how it can be implemented, which makes the system more maintainable.

Governance processes, change management, and testing features also alleviate fear of the potential side effects of rule changes. Agile development involves daily interaction between these teams. Regardless of the development methodology that your organization uses (such as waterfall), both the development and business teams must interact regularly. This regular interaction helps to ensure that developers understand “what was meant” by business users, not just “what was said.”

Applying governance to decision management encompasses people, processes, and goals across your organization. By defining expectations, assigning responsibilities, and verifying performance, governance provides an organizational framework that prevents potential problems. It facilitates

efficient collaboration between business and IT teams, and makes it possible for an organization to demonstrate that it accomplished what it said it would accomplish.

Governance in Operational Decision Manager

- Operational Decision Manager supports governance and change management through the decision governance framework
 - The decision governance framework is a ready-to-use method for applying governance principles to a BRMS
- Decision governance is based on:
 - The states of releases and activities within a decision service lifecycle
 - The roles of users who work on these releases and activities

Unit title

© Copyright IBM Corporation 2017

Figure 1-44. Governance in Operational Decision Manager

In Operational Decision Manager, the decision governance framework is a built-in framework to help your organization apply governance and change management principles.

This framework is defined around the change management activities and releases of a decision service lifecycle. The decision service includes all the rules and projects that are required to produce a decision.

The decision service and the decision governance framework are specially designed to support rule authors who work in the Decision Center Business console.

You learn more about decision services and the decision governance framework later in this course.

Unit summary

- Explain the benefits of using Operational Decision Manager
- Identify the need for governance
- Map the various roles that are involved in a decision management solution to roles in your organization
- Identify the tasks that are performed on various Operational Decision Manager modules, and which user roles perform them

Unit title

© Copyright IBM Corporation 2017

Figure 1-45. Unit summary

Review questions (1 of 2)

1. True or False: Operational decision management combines business expertise with technology to automate repeated and repeatable decisions.
2. True or False: Because the lengths of application lifecycles and decision lifecycles usually take about the same amount of time, developers can use Operational Decision Manager to update rules every six months, when the application infrastructure is updated.



Unit title

© Copyright IBM Corporation 2017

Figure 1-46. Review questions (1 of 2)

Write your answers here:

1.

2.

Review questions (2 of 2)

3. Operational Decision Manager Standard includes which components?
 - A. Decision Server Rules
 - B. Decision Server Insights
 - C. Decision Center
 - D. All of the above

4. Which of the following roles are involved during the early stages of a business rule application development project?
Select all that apply.
 - A. Policy manager
 - B. Business analyst
 - C. Architect
 - D. Developer



Unit title

© Copyright IBM Corporation 2017

Figure 1-47. Review questions (2 of 2)

Write your answers here:

3.

4.

Review answers (1 of 2)

1. True or False: Operational decision management combines business expertise with technology to automate repeated and repeatable decisions.

The answer is True.



2. True or False: Because the lengths of application lifecycles and decision lifecycles usually take about the same amount of time, developers can use Operational Decision Manager to update rules every six months, when the application infrastructure is updated.

The answer is False. Business policies evolve more rapidly than the application infrastructure. By using Operational Decision Manager, you can manage the decision lifecycle and the application infrastructure lifecycle asynchronously.

Unit title

© Copyright IBM Corporation 2017

Figure 1-48. Review answers (1 of 2)

Review answers (2 of 2)



3. Operational Decision Manager Standard includes which components?

- A. [Decision Server Rules](#)
- B. Decision Server Insights
- C. [Decision Center](#)
- D. All of the above

The answer is A and C.

4. Which of the following roles are involved during the early stages of a business rule application development project?
Select all that apply.

- A. [Policy manager](#)
- B. [Business analyst](#)
- C. [Architect](#)
- D. [Developer](#)

The answer is A, B, C, and D.

Unit title

© Copyright IBM Corporation 2017

Figure 1-49. Review answers (2 of 2)

Exercise: Operational Decision Manager in action

Unit title

© Copyright IBM Corporation 2017

Figure 1-50. Exercise: Operational Decision Manager in action

Exercise objectives

- Explain the general workflow in Operational Decision Manager for working with business rule projects
- Identify the Operational Decision Manager tools that apply to your role in your organization



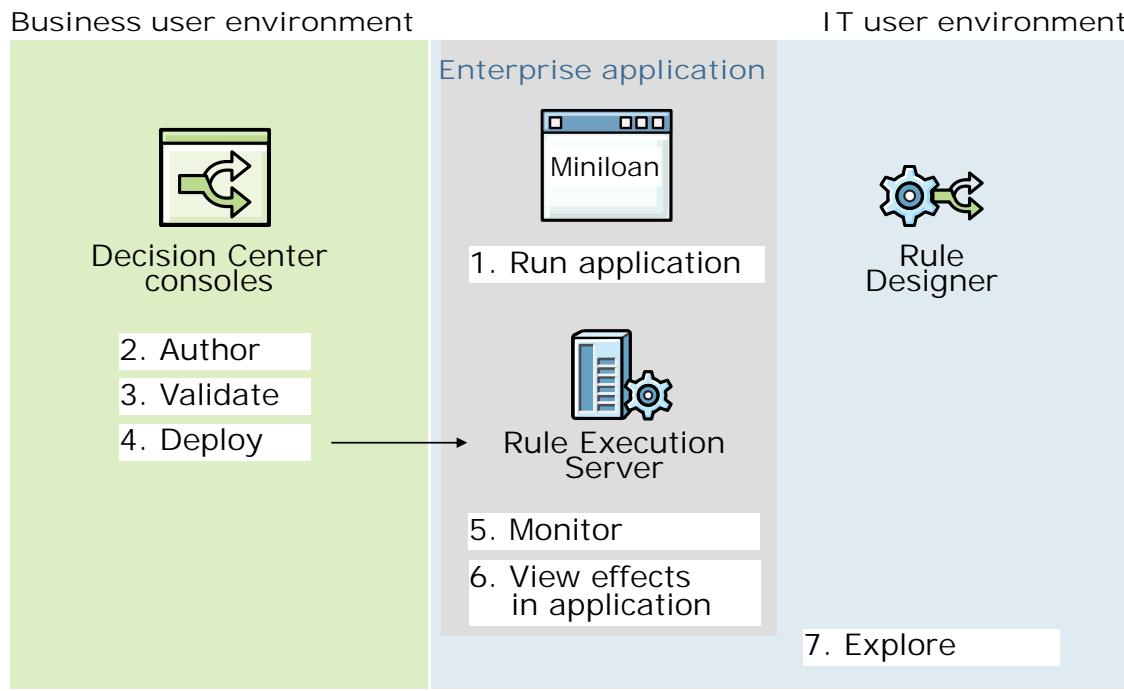
Unit title

© Copyright IBM Corporation 2017

Figure 1-51. Exercise objectives



Exercise workflow



Unit title

© Copyright IBM Corporation 2017

Figure 1-52. Exercise workflow

This diagram depicts the workflow that you follow during the exercise. After you run the application that calls business rules, you modify the rules and redeploy them to see how easily change can be applied.

Take a moment to read through the steps that are outlined here, which describe the workflow that you see in the exercise.

Unit 2. Modeling for business rules

Estimated time

01:30

Overview

This unit and the next several units introduce you to business rule and decision service concepts and tasks, such as modeling, rule projects, and BOMs. In this unit, you learn how to get started with a business rule project. The unit walks you through the tasks that you participate in during the development of the business rule application, and shows you how to apply modeling concepts and UML when specifying rule vocabulary requirements.

How you will check your progress

- Checkpoint
- Exercises

Unit objectives

- Model a business process
- Define use cases
- Explain UML notation and the modeling concepts that are relevant to modeling business rule vocabulary
- Describe the interaction between the business and development teams during the initial phases of business rule application development
- Use business policy documents and use cases to create a class diagram that models the business domain
- Identify the connection between the model and the vocabulary that is used to author rules
- Implement the business model in Rule Designer
- Write rules against the model to test for completeness

Unit title

© Copyright IBM Corporation 2017

Figure 2-1. Unit objectives

Topics

- Approaching rules as a business analyst
- Starting with models
- Modeling the business process
- Defining use cases
- Modeling the rule vocabulary
- Implementing the model

Unit title

© Copyright IBM Corporation 2017

Figure 2-2. Topics

2.1. Approaching rules as a business analyst

Approaching rules as a business analyst

Unit title

© Copyright IBM Corporation 2017

Figure 2-3. Approaching rules as a business analyst

Approaching rules as a business analyst

- To implement rules in your organization, where do you start?
- Business analysts are concerned with these questions:
 - Where can they find the rules?
 - How do they know when rules change?
 - What is the business vocabulary?
 - Where do their business objects come from?
 - How do they write new rules?



Unit title

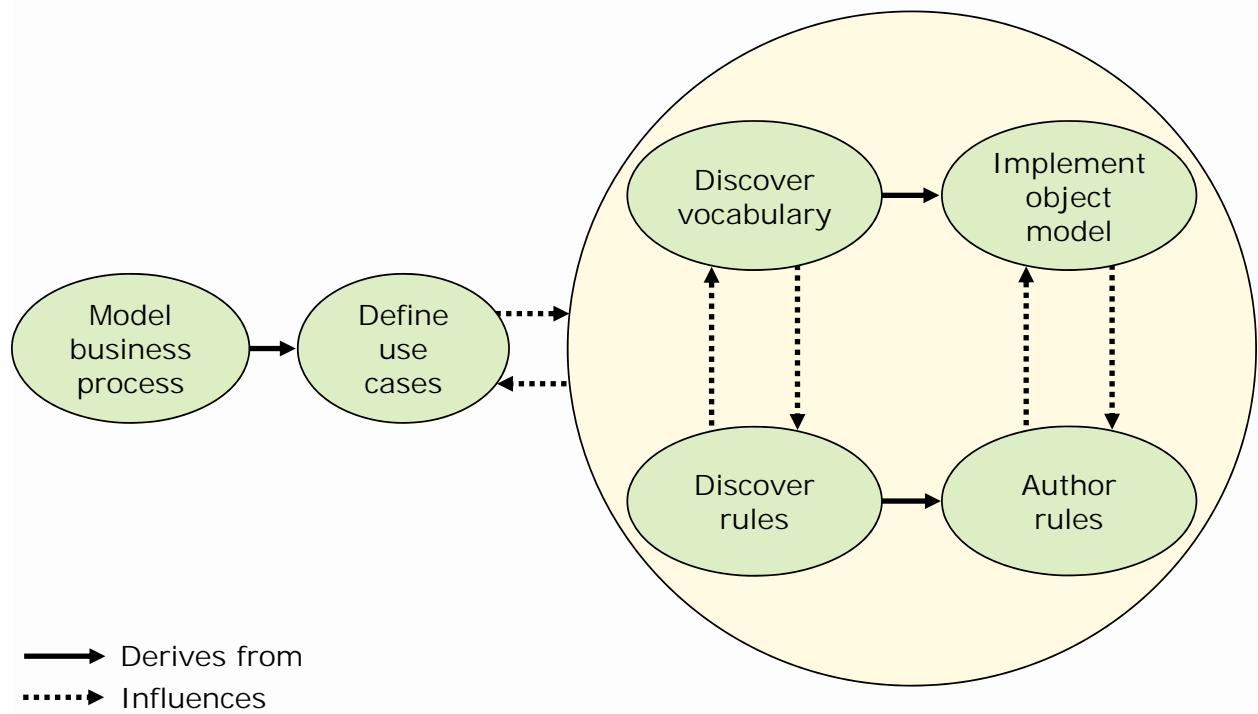
© Copyright IBM Corporation 2017

Figure 2-4. Approaching rules as a business analyst

As a business analyst just getting started with rules, you might wonder where to begin. To answer initial questions, you work closely with various members of both the business and development teams.



Overview of business analyst tasks



[Unit title](#)

© Copyright IBM Corporation 2017

Figure 2-5. Overview of business analyst tasks

This diagram provides an overall look at the breakdown of tasks in a business rule project.

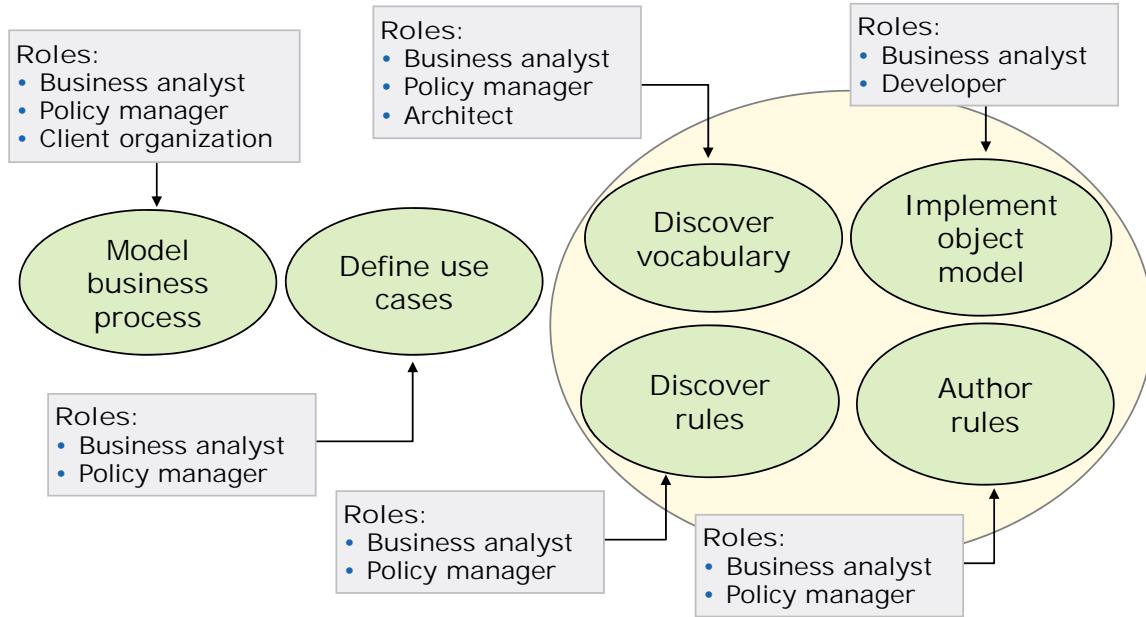
A business rule application project starts “on paper” with the business analysts, policy managers, and architect all working together to understand the business, and to identify and define business policies. Business analysts then work with developers to move the policies into the Operational Decision Manager environment as business rules.

Notice in this diagram that the starting point is the business process model. As the solid lines indicate, you derive use cases from the process model. From the use cases, you can discover both the vocabulary and rules. If process modeling is outside the scope of your role in your organization, discovery tasks might be the point at which you get involved in the project.

The dotted arrows indicate that vocabulary and rules are intertwined so you discover them together. Likewise, implementation of the vocabulary is intertwined with implementation of the rules. The vocabulary determines what rules you can write, and as you start writing, you discover what vocabulary is missing. The iterative nature of discovery and implementation is demonstrated during the exercises.

Who is involved and when?

- Business analysts must interact with other roles in the organization throughout the process



Unit title

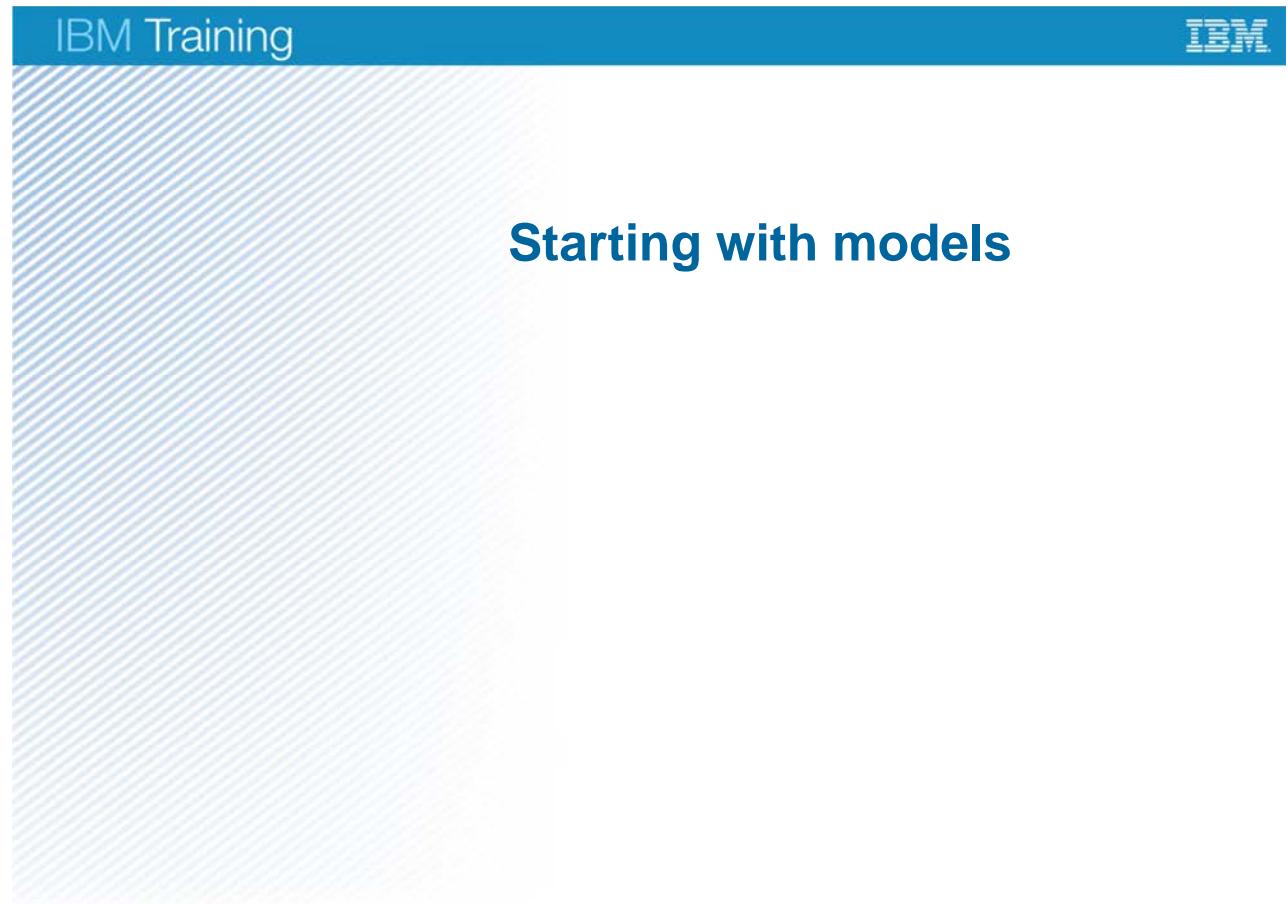
© Copyright IBM Corporation 2017

Figure 2-6. Who is involved and when?

This graphic looks at the task overview again. This slide provides an overview of the interaction between roles at various phases of the project, and shows which roles are involved and when. Notice that the business analyst is involved at every stage in the process.

The iterative nature of developing a business rule application means that business analysts must be prepared to meet several times with various stakeholders, particularly in the early stages of the project.

2.2. Starting with models



Unit title

© Copyright IBM Corporation 2017

Figure 2-7. Starting with models

What is a model?

- Models are simplified descriptions of complex entities or processes
- Models provide the overall view:
 - Define the scope of what the system should do
 - Identify what your business does and how it does it
 - Ensure that you establish the correct context
- Goals:
 - Formalize the business process flow
 - Determine the decision points
 - Define the underlying object model
 - Detail the rule execution logic

Unit title

© Copyright IBM Corporation 2017

Figure 2-8. What is a model?

Models are a good starting point to help you define the scope of what you want the business rule application to do. Modeling a business process provides an overview of the whole process to help you determine which parts of the process include decisions that can be implemented with business rules. By defining what your business does and how it does it, you establish the context for your rules. Without this context, you might end up with the wrong rules.

The main purpose of modeling is communication. Models help ensure that business context and rules are captured in a clear and unambiguous way that business policy managers can validate and developers can implement.

When meeting with various stakeholders, keep in mind these goals:

- Formalize the business process flow
 - Identifying the business objectives, the activities of the process, and the actors or roles that do those activities provides the business context for the rules. The rules are aimed towards achieving the objectives.
- Identify the decision points
 - Decisions that are made within the process can usually be broken down into smaller decisions, which are often the underlying rules.

- Define the underlying object model
 - Since business rules are written about “things” or data, modeling the business helps identify the data that makes up the underlying object model, which is the basis for the rule vocabulary.
- Detail the rule execution logic
 - Rules influence the business process flow. Wherever decisions show up in the process, the results can provide the basis for taking alternative paths through the process flow.

Types of models

- Different types of models capture different aspects of your business

Types of models	
Process models	<ul style="list-style-type: none"> Describe how things are done and the sequence of activities Process maps, activity diagrams
Functional models	<ul style="list-style-type: none"> Describe how actors interact with the system UML use case diagrams
Structural models	<ul style="list-style-type: none"> Identify all of the business entities or objects that are used and how they are related to each other UML class diagrams

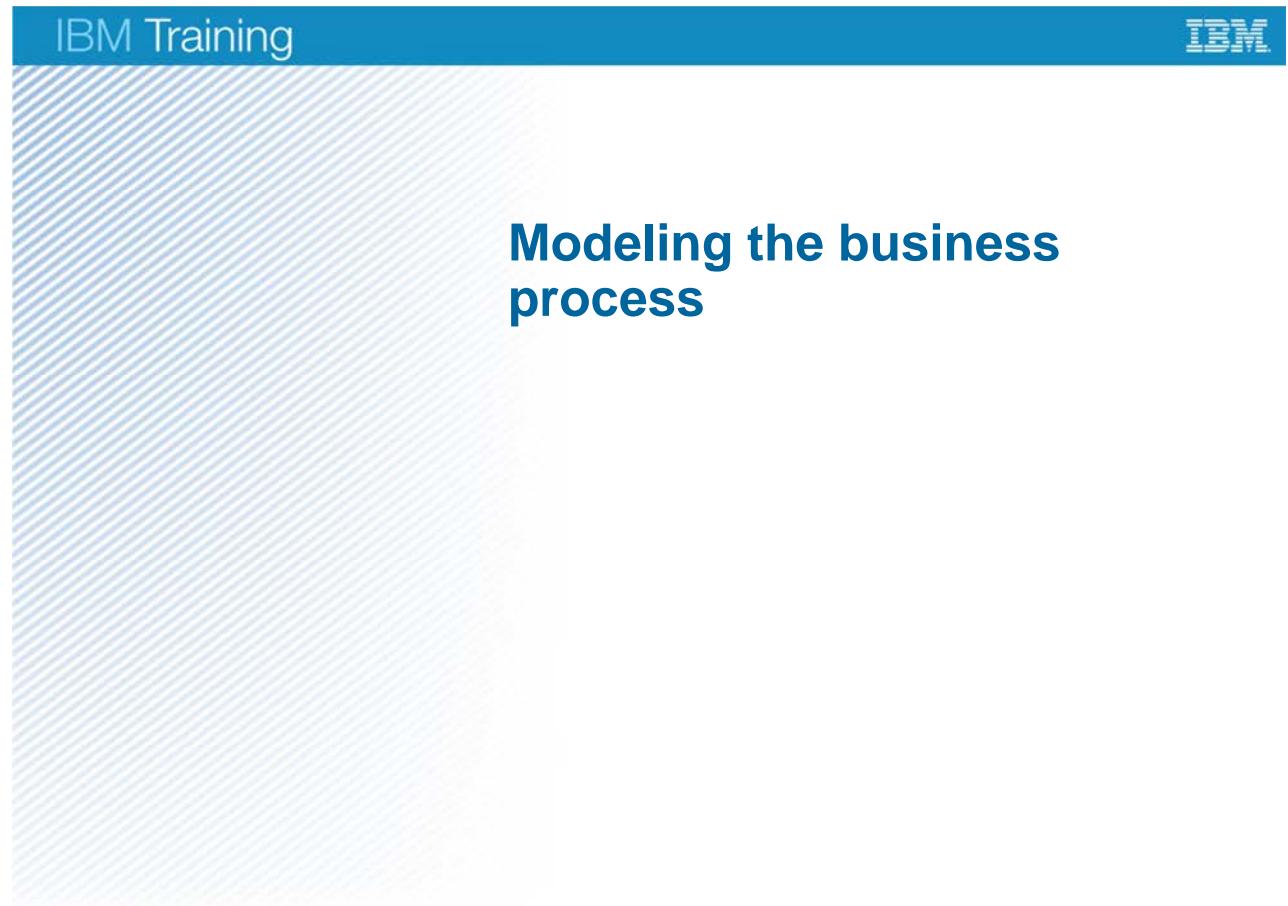
Unit title

© Copyright IBM Corporation 2017

Figure 2-9. Types of models

You can use several types of models to represent different aspects of your business. Using various types helps ensure that you capture a complete view of the business.

2.3. Modeling the business process

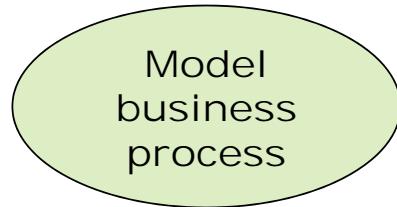


Unit title

© Copyright IBM Corporation 2017

Figure 2-10. Modeling the business process

Modeling the process



- **Roles:**

- **Policy manager:** Outlines what gets done at each step of the process, and how it gets done (business policies and rules)
- **Business analyst:** Elicits and models the business process



Unit title

© Copyright IBM Corporation 2017

Figure 2-11. Modeling the process

Business analysts work with policy managers to elicit the process model, and ensure that the activities at each step in the process are clearly defined.

Eliciting the process

- Business analysts work with policy managers to:
 - Identify the business objective
 - Identify activities of the process
 - Identify the actors and roles that do the activities
- Business process model is key for completing remaining tasks
 - Use the model to identify which steps involve decisions that can be automated

Unit title

© Copyright IBM Corporation 2017

Figure 2-12. Eliciting the process

The business process is a set of procedures or activities that collectively accomplish some particular business objective. Your goal when working with the policy managers is to identify that objective, identify the activities of the process, and identify the actors or roles that complete those activities.

After the process model is complete, you can use it to identify decision points.

Discussion

Draw a process flow

Unit title

© Copyright IBM Corporation 2017

Figure 2-13. Discussion

If you are a classroom or online student, discuss with your instructor how to draw a business process diagram, following the instructions on the next slide.

If you are a self-paced student, you can follow the instructions on the next slide as an independent exercise.

Discussion: Draw a process flow

- As a class, work out with your instructor a process model for the following business process:

Industry	An insurance company that offers car insurance coverage
Business objective	<ul style="list-style-type: none"> Create a repeatable, efficient process for processing car insurance coverage requests, determine acceptable risk levels, and issue policies: <ul style="list-style-type: none"> Low-risk applications are approved Medium-risk applications are sent to an underwriter for manual approval High-risk applications are rejected Provide a quick response to applicants, and offer competitive terms for approved car insurance policies

Unit title

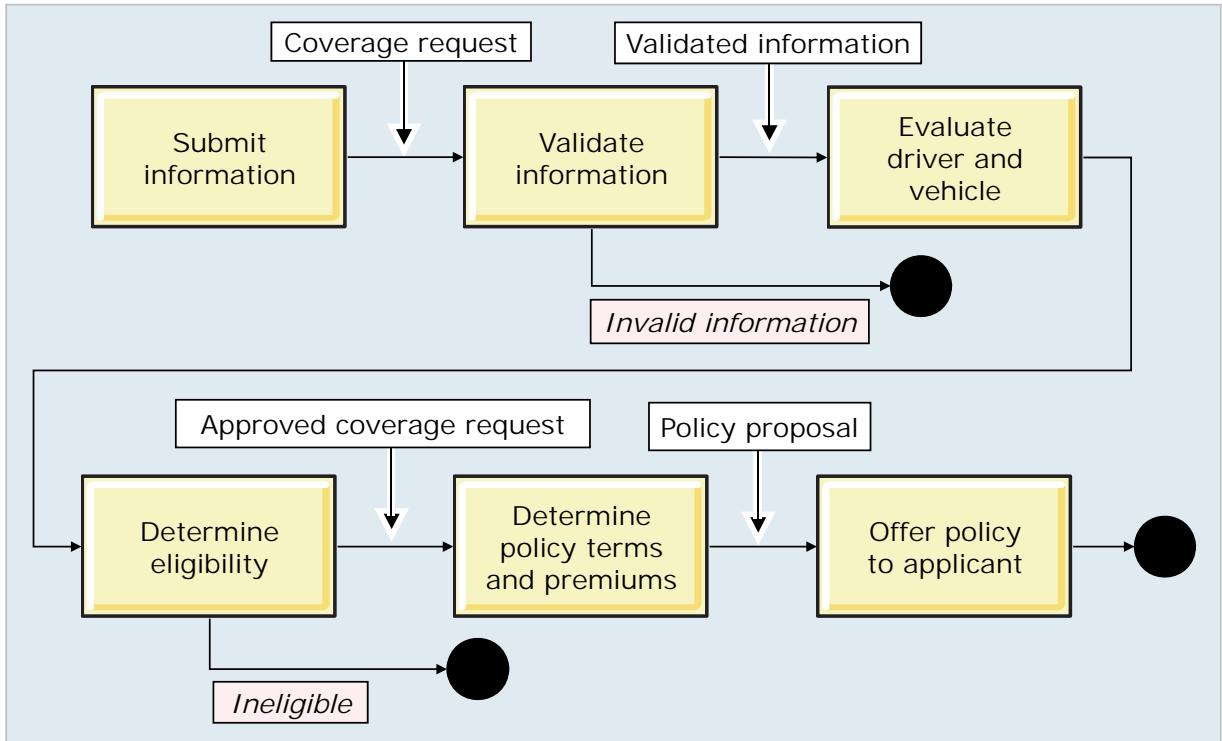
© Copyright IBM Corporation 2017

Figure 2-14. Discussion: Draw a process flow

Draw a business process diagram that:

- Defines the steps of getting a loan
- Determines the order in which the tasks are done
- Identifies who does each task

Discussion review: A possible process flow



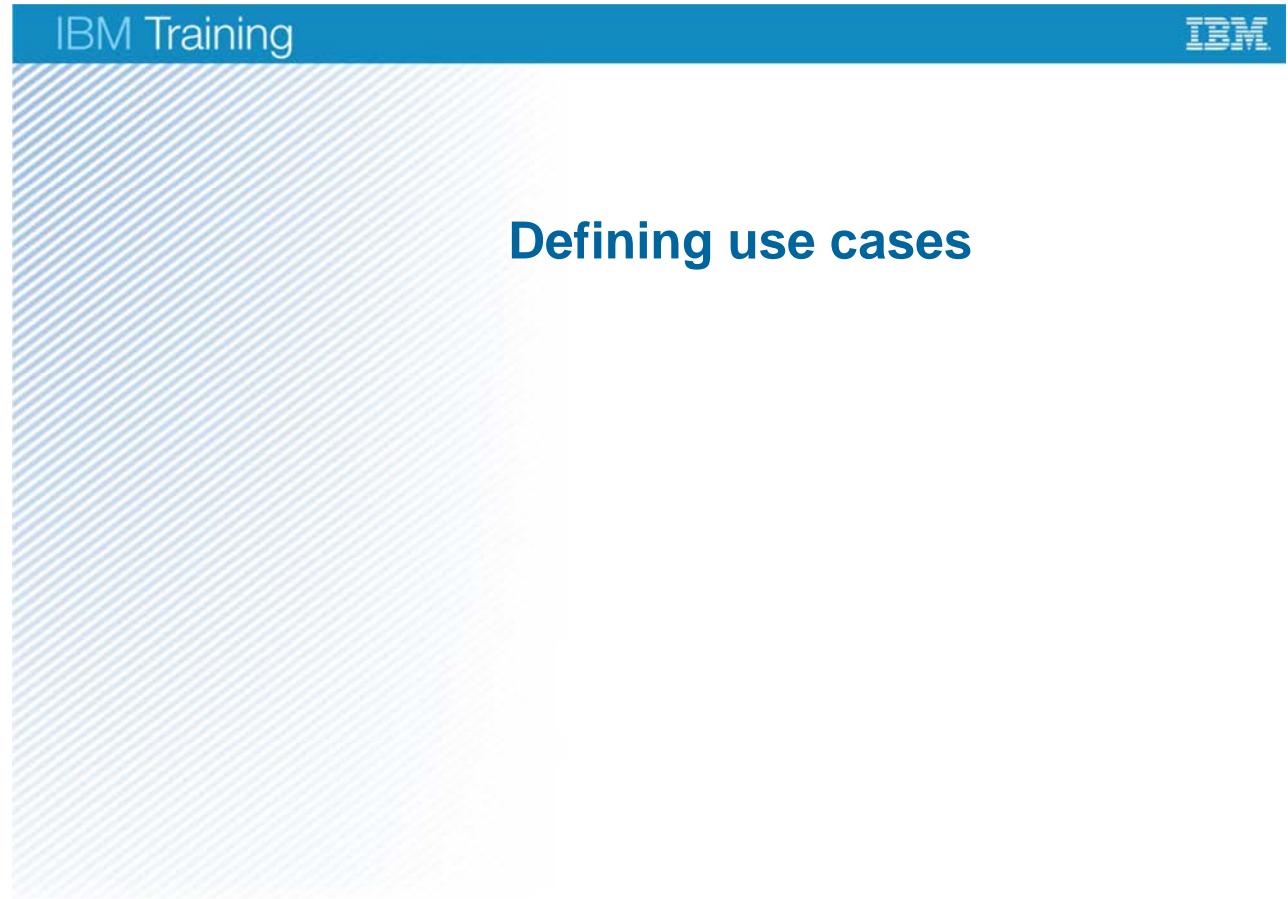
Unit title

© Copyright IBM Corporation 2017

Figure 2-15. Discussion review: A possible process flow

This slide shows a possible process flow for the car insurance application business process.

2.4. Defining use cases

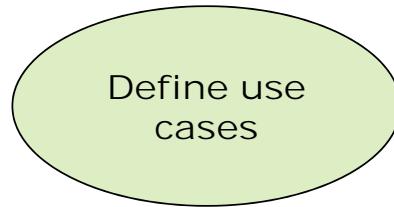


Unit title

© Copyright IBM Corporation 2017

Figure 2-16. Defining use cases

Defining use cases



- **Roles:**
 - **Business analyst:** Defines the use cases
 - **Policy manager:** Validates the use cases



Unit title

© Copyright IBM Corporation 2017

Figure 2-17. Defining use cases

The next step is to take the business process model from the previous step, and derive the use cases to specify the expected behavior (or behavioral requirements) of the application. If process modeling is not a task that you are involved in, you might start from use cases instead.

Again, business analysts and policy managers work together on this step.

What is a use case?

- Step-by-step description of how users (actors) interact with the system to do a particular task
 - Each use case describes a single interaction
- Each use case can involve several actors
 - An actor can be a person or another system
- Main scenario is the “happy path” or description of a successful interaction
 - Each use case can have many scenarios

Unit title

© Copyright IBM Corporation 2017

Figure 2-18. What is a use case?

A use case is a step-by-step text description of how a user interacts with the system to complete a particular task. To identify use cases clearly, you can use long names to describe them.

Each use case can involve several actors. An actor can be a person or a thing (like an external application or system). For example, a use case that describes an online shopping transaction would show the customer as an actor, and might also include an external credit card verification system as an actor.

Write the main scenario to describe the “happy path” through the process, or the steps that lead to a successful interaction. The scenario represents a concrete example with the related data.

Scenarios can also be used later for testing and validating the rules.

Alternative scenarios

- What if ...
 - Personal information is not complete
 - Borrower has poor credit rating
 - Borrower is already a customer in good standing
- Alternative scenarios include:
 - Failure scenarios, where the process terminates without achieving the business objectives of the actors
 - Special treatment scenarios
 - Exceptional cases
- Scenarios can be used later for testing and validating rules

Unit title

© Copyright IBM Corporation 2017

Figure 2-19. Alternative scenarios

Along with the main scenario, the use case can include alternative scenarios, which account for problems or exceptions. Alternative scenarios can include the following types of situations:

- Failure
 - The process terminates without achieving the business objectives of the actors.
 - Examples: Personal information is invalid or incomplete, credit rating is unacceptable.
- Special treatment
 - Example: VIP customers or loyal customers.
- Exceptional cases
 - Example: If non-permanent residents have collateral, set aside the requirement for permanent residency status.

Scenarios can be used later for testing and validating the rules.

Example: Use case for renting a car

Use case name	Rent a car in a branch office
Description	A customer enters a branch office and asks for a car
Actors	Booking clerk, customer, customer database, car group database
Preconditions	The booking clerk is logged on the system
Success end conditions	<ul style="list-style-type: none"> • The customer has the key of the car with a signed contract • The customer has no contract because of high risk
Failed end conditions	The customer's reservation could not be completed
Main success scenario	<ol style="list-style-type: none"> 1. The booking clerk enters data that is related to the reservation: car group, start and end dates, pickup and return branches 2. The booking clerk enters the data for the customer and the billing information 3. The system verifies the quality of the data 4. The system verifies the credit limit of the customer 5. The system computes the car availability and the price
Alternative	
Input	Customer information and reservation information
Output	A contract or a refusal

Unit title

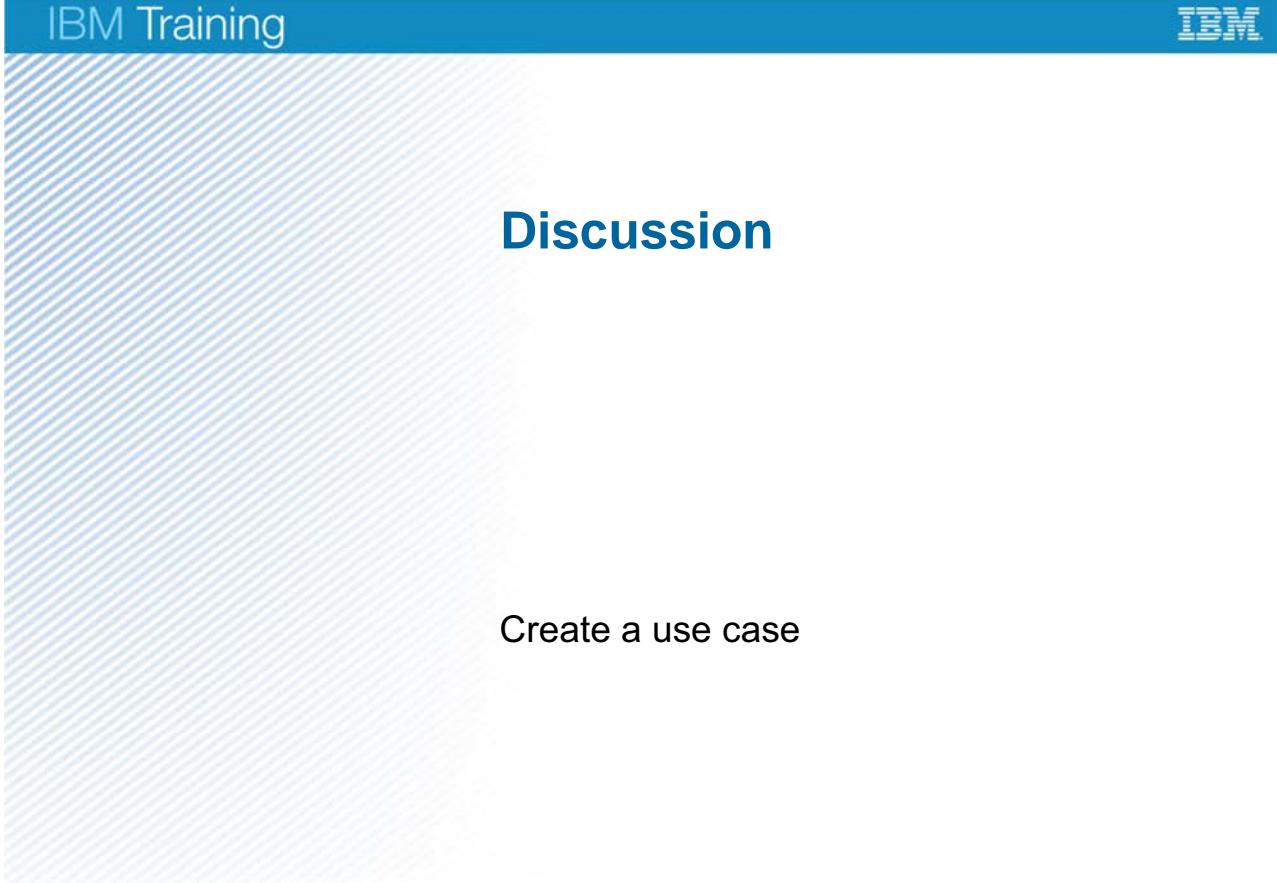
© Copyright IBM Corporation 2017

Figure 2-20. Example: Use case for renting a car

The example here outlines the “happy path,” or main scenario, for renting a car.

Note the use case name, description, actors, and main success scenario:

- The use case name, “Rent a car in a branch office,” is descriptive and specific.
- The human actors include the rental car booking agent and the customer who wants to rent a car.
- System actors include the customer database and the car group database.
- The main success scenario outlines the main steps that are involved when a customer goes to a car rental company branch office and requests a car rental.



The slide features a blue header bar with the text "IBM Training" on the left and the IBM logo on the right. The main content area has a light gray background with a subtle diagonal striped pattern. A large, bold, dark blue heading "Discussion" is centered at the top. Below it, a dark blue sub-heading "Create a use case" is also centered. The overall design is clean and professional, typical of an IBM training presentation.

Unit title

© Copyright IBM Corporation 2017

Figure 2-21. Discussion

For classroom and online versions of the course, discuss with your instructor how to create the use case by following the instructions on the next slide.

For self-paced versions of this class, you can follow the instructions on the next slide as an independent exercise.

Discussion: Create a use case

- As a class, take a few minutes to discuss the steps that are involved in handling a car insurance coverage request
- Based on those steps, complete this use case outline

Use case name	
Description	
Actors	
Main success scenario	
Alternative scenarios	

Unit title

© Copyright IBM Corporation 2017

Figure 2-22. Discussion: Create a use case

In this activity, take a few minutes to write down your notes on the steps that are involved in a car insurance coverage request.

When you are finished with your notes, use your ideas to complete the use case outline on the slide.

First, complete the use case name and description. Then, write down the actors in the scenario, whether they are human or a system. Next, write down the steps in the main success scenario. Finally, write down steps in any alternative scenarios that you defined.

For classroom and online students: Be prepared to share your work and ideas with the instructor and other students.

For self-paced students: You can work on this task as an independent activity.

Discussion review: A possible use case description

1. User accesses application.
2. System prompts user for personal information and vehicle details.
3. User enters name, address, amount of car insurance coverage, and other information.
4. System validates information.
5. Information is OK; system analyzes driver and vehicle information, and assesses eligibility for insurance coverage.
6. Coverage is approved; system determines policy terms and premiums.
7. System offers policy to applicant.

Unit title

© Copyright IBM Corporation 2017

Figure 2-23. Discussion review: A possible use case description

The use case description that is in this slide is the narrative that you enter in the main scenario section of the use case template, as shown next.

System actions are shown in a blue font.

Discussion review: A possible use case template

Name	Apply for a loan	Version	1.4
Description	An insurance agent enters a coverage request from a driver into system. System checks the eligibility of the driver and vehicle for insurance coverage, determines policy terms and premiums, and offers the policy to applicant.		
Actors	Primary: Insurance agent; Secondary: Department of Motor Vehicles records.		
Pre-conditions	Department of Motor Vehicles system is available.		
Post-conditions	A Policy is created and saved with status approved.		
Main scenario			
1.	User accesses application.		
2.	System prompts user for personal information and coverage details.		
3.	User enters borrower name, address, amount of coverage wanted, vehicle type, and other information.		
4.	...		
Alternative scenarios			
4.a	Driver license number field is invalid.		
4.a.1	System asks user to reenter driver license number.		
...	...		
7.a	Vehicle is more than 40 years old.		
7.a.1	System rejects application and informs user.		

Unit title

© Copyright IBM Corporation 2017

Figure 2-24. Discussion review: A possible use case template

This example shows a typical use case format. The use case template includes fields for the use case name, version, description, actors, and any preconditions and postconditions. It also includes a section for the main scenario, and a section for alternative scenarios.

Use case *diagrams* are not explained in this unit. While use case diagrams are useful for providing a high-level view of the interactions between users and the system, the details pertinent to identifying automated decision points are contained in the use case narrative.

2.5. Modeling the rule vocabulary

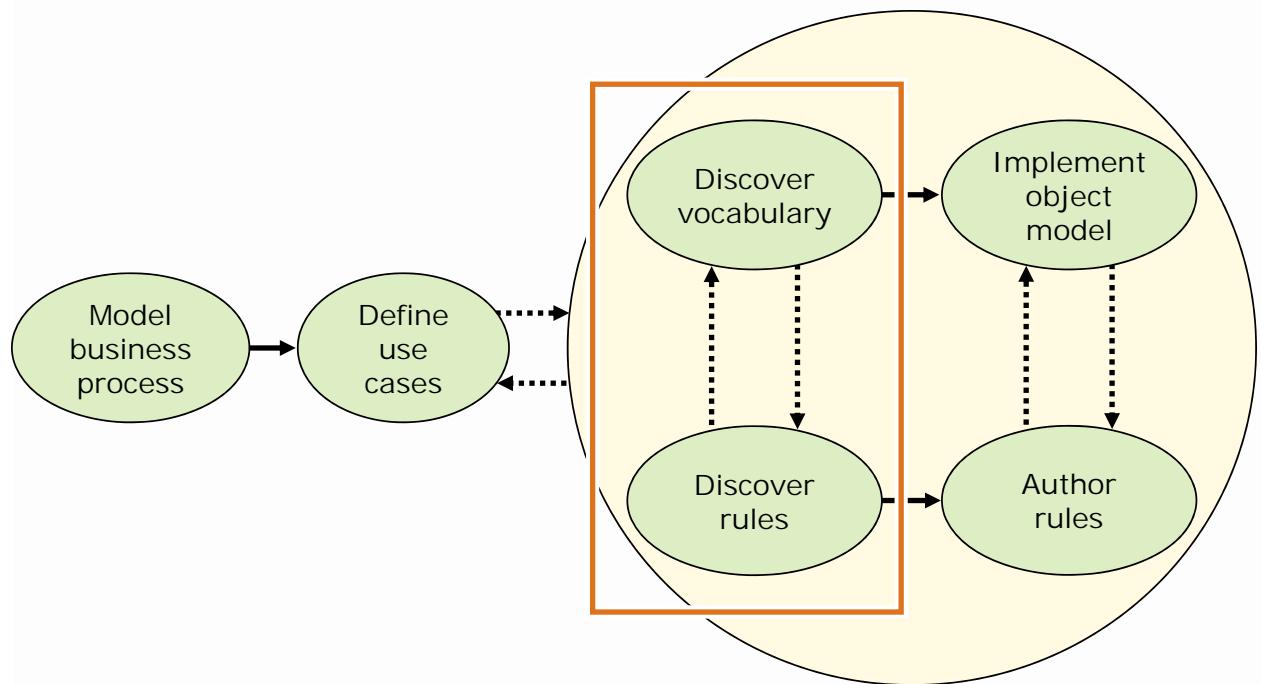
Modeling the rule vocabulary

Unit title

© Copyright IBM Corporation 2017

Figure 2-25. Modeling the rule vocabulary

From decisions to discovery



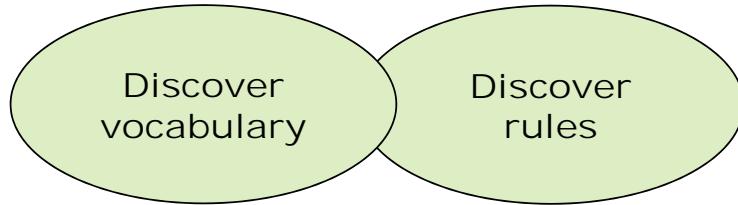
Unit title

© Copyright IBM Corporation 2017

Figure 2-26. *From decisions to discovery*

After you establish the business context with the process model and use cases, the discovery phase can begin. The discovery phase includes both discovering vocabulary and discovering rules.

Intertwined discovery of vocabulary and rules



- **Roles:**

- **Business analyst:** Models the business view through the object model
- **Architect:** Defines the project organization, the types of rules that are used



Unit title

© Copyright IBM Corporation 2017

Figure 2-27. Intertwined discovery of vocabulary and rules

The discovery phase involves the business team and the architect from the development team, so for these tasks, business analysts work with both policy managers and the development team.

Discovering vocabulary is intertwined with rule discovery, so the discovery phase involves both activities together.

Discover the vocabulary and rules

- Discovering vocabulary:
 - Identify the “things” (objects) about which decisions are made
 - Create structural model or class diagram
- Discovering rules:
 - Identify decisions and underlying rules
 - Formalize and document the rules
- Rules and objects intertwined:
 - Objects make up the rule vocabulary
 - Writing the rules helps complete the vocabulary
- Initial discovery of rules and objects is done on paper

Unit title

© Copyright IBM Corporation 2017

Figure 2-28. Discover the vocabulary and rules

Discovering vocabulary involves identifying the “things” or objects about which decisions are made, and creating a structural or object model to describe their relationships.

Discovering rules involves formalizing and documenting the rules that are identified from business policies by using the vocabulary that is provided in the object model. For example, to write a rule about customers, orders, or accounts, your vocabulary or object model must include customer, order, and account objects.

The object model that you build can be handed over to developers, who can get started with implementing it. After the model is implemented in Rule Designer, you can start writing rules in the rule editors.

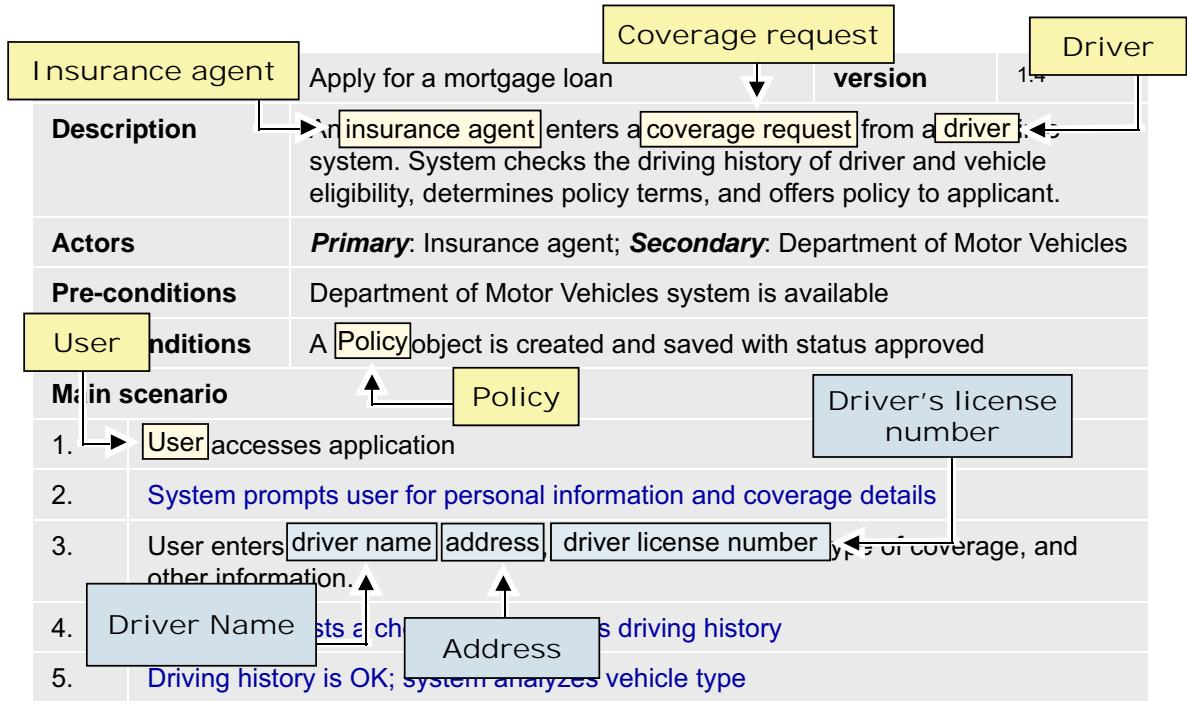
Your object model becomes the source of vocabulary in the rule editors. Rules and objects are intertwined, and rules are written about objects. Rules cannot be expressed precisely without an object model.

As you and your rule authors start writing rules, it becomes apparent whether your original model was missing vocabulary. When you try to express rules, you might find that some rules reference data that is not captured in the model, or you might uncover inconsistencies. In that case, you must go back to the developers to have them fix the model.

This section concentrates on how you can use UML class diagrams to provide a complete and detailed model of your vocabulary. If you provide a comprehensive model to developers, it can minimize the number of times you must go back to the developers for fixes.

Identifying objects in use cases

- Pick out nouns and noun phrases as candidate classes from the use cases



Unit title

© Copyright IBM Corporation 2017

Figure 2-29. Identifying objects in use cases

To discover your vocabulary, go through the use cases and look for nouns and adjectives, which are often cues as to what to include in the vocabulary model.

In the example here, yellow boxes highlight nouns that are potential vocabulary, such as *Driver*, *User*, and *Policy*. The blue boxes also identify nouns and adjectives, but these nouns and adjectives are attributes of other nouns. For example, *Driver Name* is a noun that helps to describe a driver.

Before you decide which vocabulary to include from this use case, review some object modeling terminology, in particular, *classes* and *objects*.

Classes

- Classes represent a set of objects that share a common structure and behavior
 - Examples: Bank Customer, Account
- Classes have *attributes* and *operations*
- **Attributes:** The descriptive characteristics of the objects that are relevant to the business domain
 - Bank Customer attributes: name, address, age
 - Account attributes: account type, amount
- **Operations:** Actions that are linked to the class
 - Bank Customer operation: deposit, withdraw

Unit title

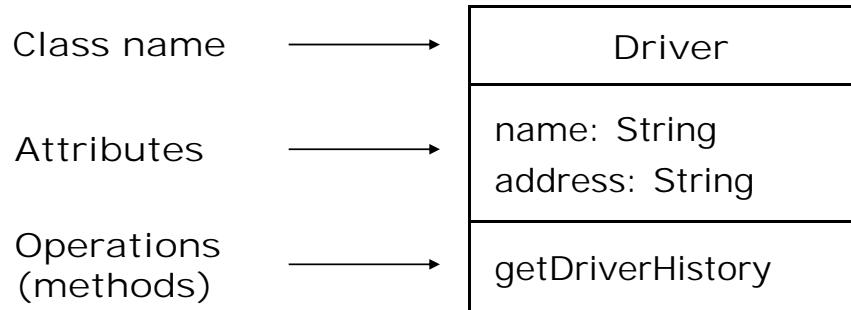
© Copyright IBM Corporation 2017

Figure 2-30. Classes

A class represents a set of objects that share structure and behavior. An object is a specific instance of a class. The class is like a template, whereas the object is the actual thing. For example, if *Person* is a class, you are an object of type *Person*.

Classes have attributes and operations or methods. Attributes are the characteristics of the object relative to the business domain. Operations are actions that the object can do. For example, if you have a Bank Customer class, what attributes would the objects in that class have? What about a Bank Account? What operations can that class of objects do? The answers that are relevant to the business are the ones that identify the attributes and operations to include in your model so that you can write rules about them.

Class diagrams



Unit title

© Copyright IBM Corporation 2017

Figure 2-31. Class diagrams

The class diagram on this slide is a simple example. If you are not familiar with this notation, notice the format, which is a box with the class name capitalized, and a separate section for the attributes and methods.

An object model is made up of class diagrams just like the one shown here.

Attributes

- Attributes describe and distinguish instances of classes
 - Each attribute has a name and a type (range of values)
 - Correspond to database table columns
- Retain only attributes that are relevant to the application
 - Name
 - Address
 - Social Security number
 - Annual income
 - Loan amount
 - Status

Property
-type: String -location: String -assessmentValue: float -assessmentDate: Date

Borrower
-name: String -address: String -ssn: String -annualIncome: float

MortgageRequest
-loanAmount: float -loanPurpose: String -status: short -term: short

Unit title

© Copyright IBM Corporation 2017

Figure 2-32. Attributes

The attributes in a class diagram have both a name and a *type*. If you are unsure about what “type” to use when preparing the class diagram, developers can help you.

Returning to the earlier question about which adjectives from the use case to include in the model, notice here the exaggerated list of adjectives that might apply to a borrower. For example, you know that your rules about borrowers consider their annual income, but never consider eye color or shoe size. Therefore, in the class diagram, only the relevant attributes are included.

Class methods

- Some attributes are *computed*
 - Borrower: credit percentage, projected debt load
 - Loan: monthly payment
- Computation logic for attributes is encoded as *operations* or *methods* that are attached to classes
- Some methods use data from several classes:
 - To determine which class “owns” such methods, first determine what data the method uses or produces

Unit title

© Copyright IBM Corporation 2017

Figure 2-33. Class methods

When you start creating class diagrams, you might find that some of the attributes come from computed values. For example, if a borrower has an *age* attribute, that value would be calculated based on the date of birth and the current year. Another example might be monthly payments, which are calculated based on a percentage of a loan.

That type of computation logic can be implemented by using rules, which allows more agility. It is good practice to use rules to implement computational logic that changes more than one time a year.

Class relationships

- Class diagrams describe the types of objects in the business domain and their relationships
- Example: Relationship between Customer classes and Order classes
 - A customer places an order



Unit title

© Copyright IBM Corporation 2017

Figure 2-34. Class relationships

Class diagrams depict not only the classes in your business domain, but also the relationships between them.

The line from one class to another indicates that one object is aware of another object when doing a specific task. For example, a class diagram for a customer who places an order shows an association between the `Customer` class and the `Order` class, as seen here.

Relationships

Relationships between classes include:

- Association
- Part-whole (“has-a”) relationships
 - Aggregation
 - Composition
- Inheritance or parent-child (“is-a”) relationships:
 - Specialization (subclasses)
 - Generalization (superclasses)

Relationships can affect how rules are written

- When a rule applies to a parent class, it automatically applies to the child classes, but some rules might apply only to specific child classes
- Rules about objects that are involved in aggregation or composition relationships might need to account for those relationships

Unit title

© Copyright IBM Corporation 2017

Figure 2-35. Relationships

One of the main purposes of using UML notation is to depict the different types of relationships between classes.

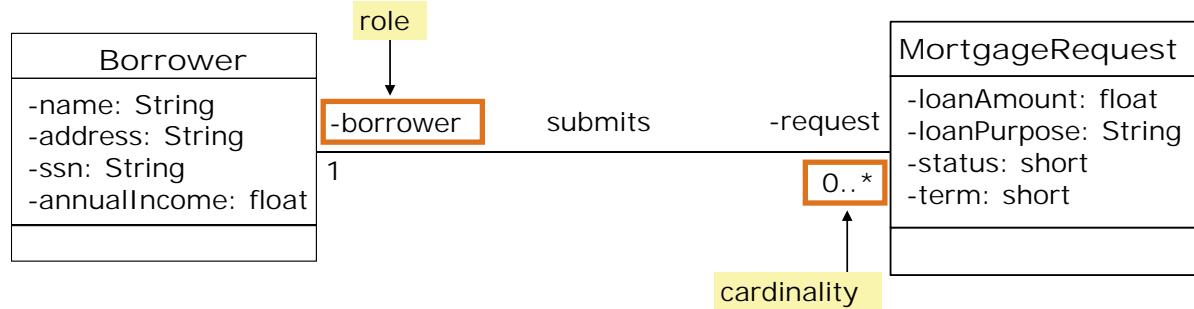
There are several types of relationships, including:

- Association, which indicates that classes are aware of each other during a task
- Aggregation and composition, which indicate a grouping of classes to create another class
- Inheritance, which references a parent-child relationship

Understanding these relationships can affect the way that you write your rules. Rules that apply to one class might affect or extend to other classes, depending on the relationship.

Cardinality in associations

- Indicates the number of relationships that can exist between classes
 - 0..1:** Zero or one (optional, single-valued)
 - 1:** Exactly one (mandatory, single-value)
 - 0..*:** Zero or more (optional, multi-value)
 - 1..*:** One or more (mandatory, multi-value)
 - n..m:** Between n and m
- Example of one-to-many relationship between the borrower and the MortgageRequest



Unit title

© Copyright IBM Corporation 2017

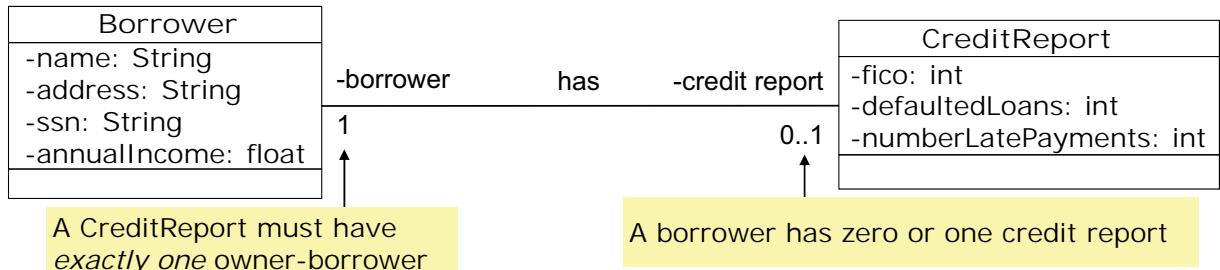
Figure 2-36. Cardinality in associations

When you depict association in a class diagram, you use numbers or *cardinality* to indicate how many relationships can exist between these classes.

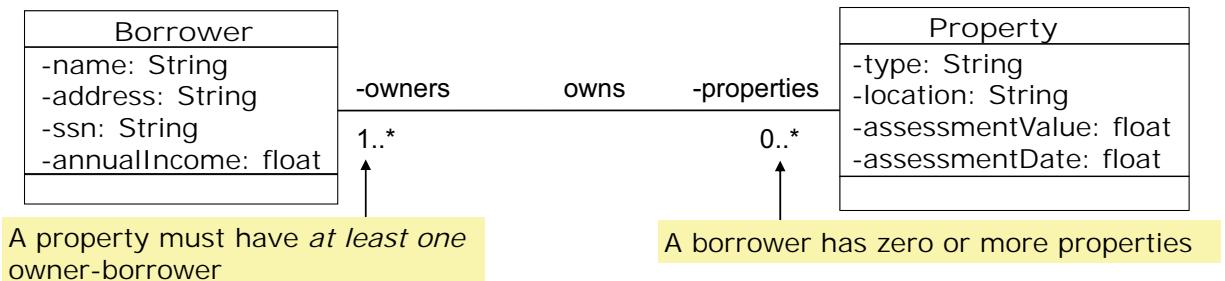
You can assign names to relationships. For example, here you see that the name of the association is *submits*. Also, you can indicate what *role* a class plays in the relationship. In this example, a **Borrower** class can have zero or more **MortgageRequests**, while the **MortgageRequest** class must have exactly one **Borrower**, which indicates that you cannot have a mortgage without a **Borrower**. The direction is from **Borrower** to **MortgageRequest**.

Specifying association

- A Borrower has a CreditReport
 - Each CreditReport is associated with exactly one Borrower



- A Borrower owns zero or more Properties
 - Each **Property** is owned by one or more **Borrowers**



Unit title

© Copyright IBM Corporation 2017

Figure 2-37. Specifying association

These examples further illustrate relationships with roles and cardinality.

A Borrower can have zero or more CreditReports, but each CreditReport is associated with **exactly one** Borrower.

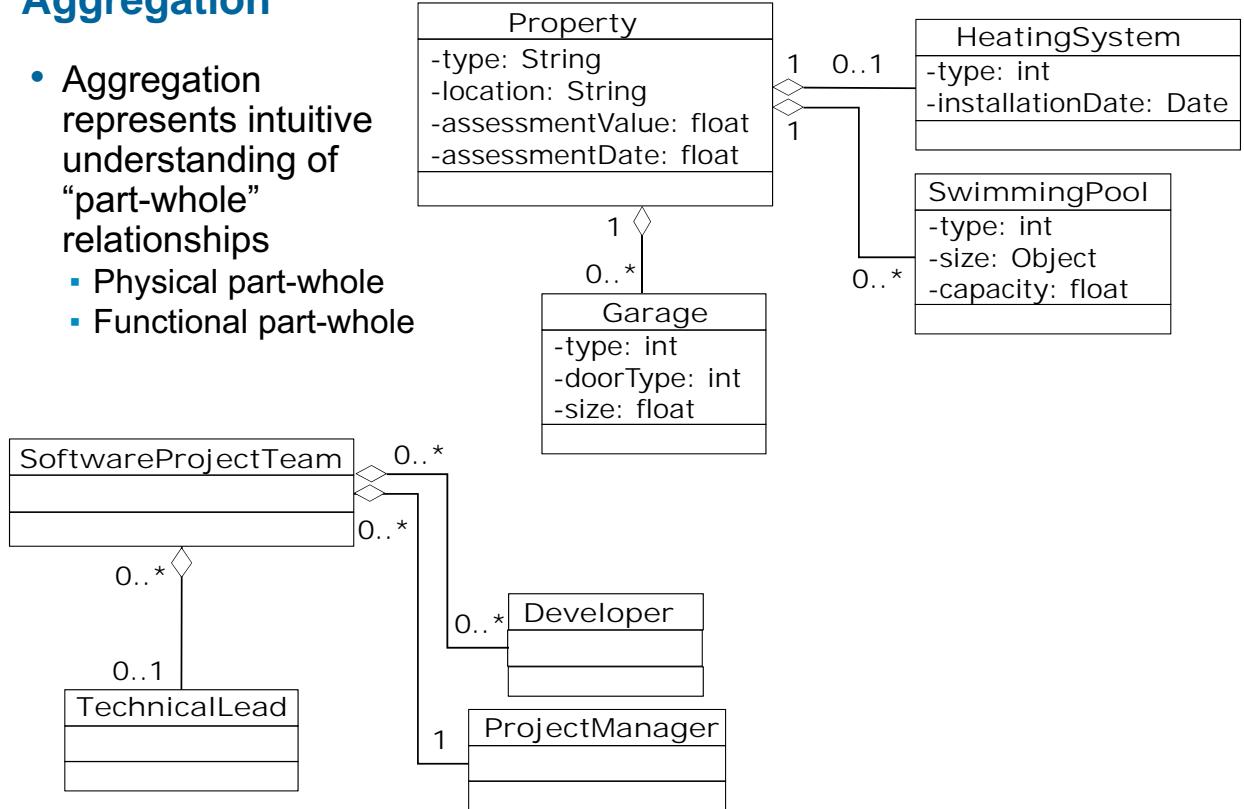
A Borrower owns zero or more Properties, and each Property is owned by one or more Borrowers.

In the first example, the class diagram specifies that a Borrower can have zero or more CreditReports, but each CreditReport is associated with **exactly one** Borrower.

The next example shows that a Borrower can own zero or more Properties, while Properties can have one or more owners. Notice that the role of the Borrower in this relationship is identified as *owner*.

Aggregation

- Aggregation represents intuitive understanding of “part-whole” relationships
 - Physical part-whole
 - Functional part-whole



Unit title

© Copyright IBM Corporation 2017

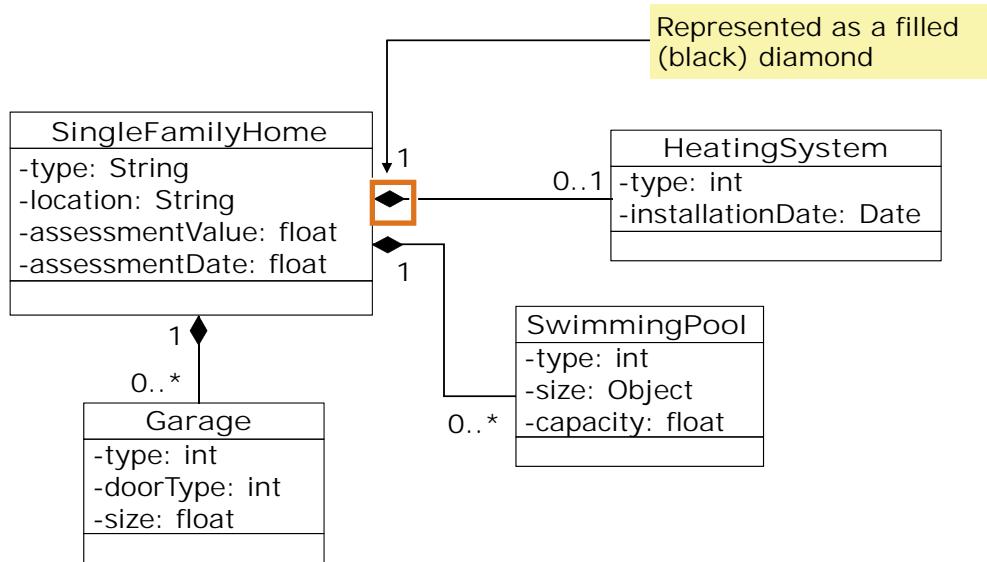
Figure 2-38. Aggregation

Aggregation and composition are considered part-whole (or “has-a”) relationships, in which several objects are grouped to form another object.

With aggregation, the relationship can be physical or functional. For example, a university campus can be composed of several buildings. If you close the university, the buildings can continue to exist. The “part” can continue to exist even if the “whole” dies.

Composition

- Composition is an aggregation with a twist:
 - The “whole” owns the part (exclusivity)
 - The part dies with the “whole” (nested lifecycles)



Unit title

© Copyright IBM Corporation 2017

Figure 2-39. Composition

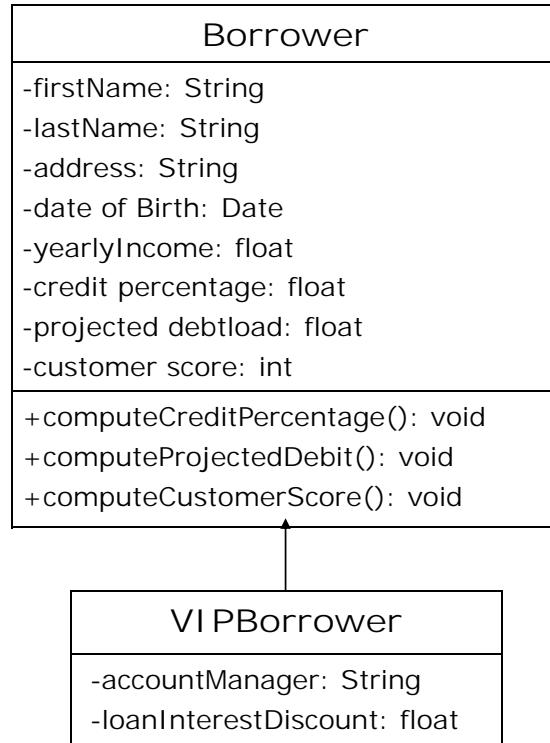
Composition is when an object is made up of other objects, but the parts cannot exist separately from the “whole.”

For example, a house is composed of a roof, walls, doors, windows, and rooms. By themselves, these parts have no function. You cannot have a room without the walls, nor can you have free-standing windows. However, when grouped, they can serve a purpose. The “whole” owns the parts, and the parts do not function separately from the “whole.”

Inheritance

- Specialization
 - Classes can be specialized into hierarchies of subclasses
 - Subclasses inherit all attributes and associations of a parent class, plus other characteristics
 - Example: VIPBorrower inherits everything in Borrower plus some attributes specific to VIP

- Generalization
 - The root of a class hierarchy is called the superclass
 - Example: Borrower is the parent class



Unit title

© Copyright IBM Corporation 2017

Figure 2-40. Inheritance

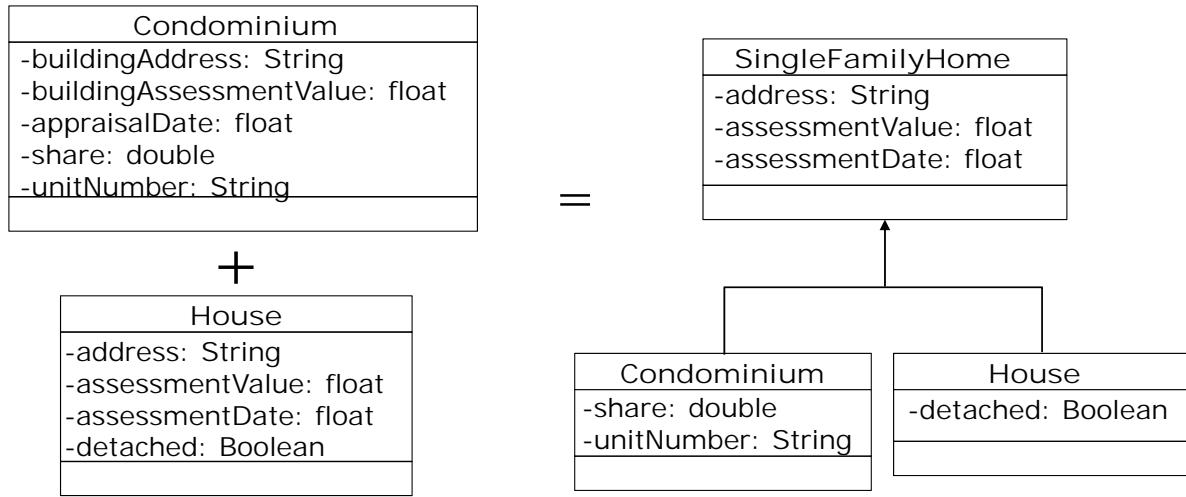
Inheritance involves the concept of parent-child (or “is-a”) relationships, and supports the concept of reuse. A child class, or subclass, inherits the attributes and operations from its parent class, the superclass.

A child class is a specialized version of the parent class. The child class includes all the attributes and associations of the parent class, but can also include more specialized characteristics, behavior, or associations.

For example, if `Borrower` is the parent class, a specialized or child class might be `VIP Borrower`, which inherits all the attributes and methods of `Borrower`, plus some attributes specific to `VIP Borrowers`.

Generalization

- When you have two or more classes with common properties and behavior, you can group them under a superclass
 - Simplified modeling
 - Reuse of attributes, operations, and relationships from a parent class in child classes



Unit title

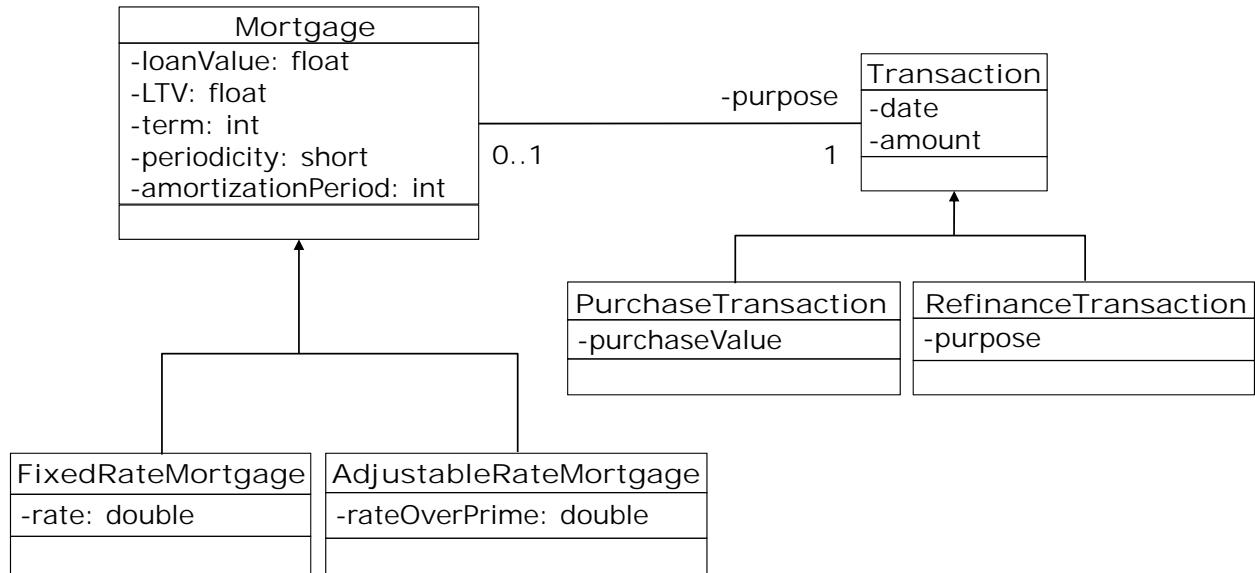
© Copyright IBM Corporation 2017

Figure 2-41. Generalization

When you have two or more classes with common properties and behavior, you can group them under a superclass. Generalization simplifies modeling by capturing attributes, operations, and relationships in a parent class that can be reused in child classes. In this example, condominium and house both share common properties and behavior and can be extracted and generalized under a superclass called SingleFamilyHome.

Inheritance and association

- Inherited classes also inherit associations with other classes and their child classes



Unit title

© Copyright IBM Corporation 2017

Figure 2-42. Inheritance and association

This example shows that child classes inherit the relationships of the parent class. In this example, both the `FirstRateMortgage` and `AdjustableRateMortgage` have a relationship with `Transactions`.

Inheritance in rules

- Object model must include all classes that are used in vocabulary
- Example: Eligibility
 - Classes can be specialized into hierarchies of subclasses
 - Eligibility rules apply to all objects of type Borrower
 - Some eligibility rules are specific to CoBorrowers
 - Some eligibility rules apply only to objects of type TemporaryResidentBorrower
 - To write rules that apply to distinct types of borrowers, your object model must include Borrower, CoBorrower, and TemporaryResidentBorrower

Unit title

© Copyright IBM Corporation 2017

Figure 2-43. Inheritance in rules

To ensure that you have the required vocabulary to write rules that are about specialized (or child) classes, your object model must include not only the parent classes, but also the child classes. For example, you might have a set of eligibility rules that include rules that apply to all objects of type Borrower. However, some of the rules might be specific to CoBorrowers. Some might apply only to objects of type TemporaryResidentBorrower. To write rules that apply to distinct types of borrowers, your object model must include Borrower, CoBorrower, and TemporaryResidentBorrower.

What is a “good” business object model?

- Supports the business rules
 - All rules can be written against it intuitively

- Can be implemented easily and efficiently
 - All of the underlying business data is available, either in raw form or through some computation
 - All of the data is efficiently accessible



Negotiated between business analysts and developers

Unit title

© Copyright IBM Corporation 2017

Figure 2-44. What is a “good” business object model?

The best way to know whether your data model is complete is to implement it and start writing rules against it. As you review your model, ask yourself if it includes all the vocabulary sufficient to formulate your rules.

The goal of using a class diagram to model your vocabulary is to be able to express your requirements to the development team in a clear and precise way. Does the class diagram clearly show developers what to implement? If developers understand your requirements correctly, they can implement your model accurately so that rule authors can start writing rules.

Ensuring completeness in the model

- To know whether your model is complete, you must start writing rules
- Example: Consider this policy that restricts the value of an automobile that the insurance company is willing to cover:
Insurance requests are rejected if the vehicle value is more than \$120000
 - To write this rule in the rule editors, the term “value” must be part of the vocabulary
 - Is “value” in your model?**
- If your rules include vocabulary that is not found in the model, update the model
 - Many rule-model iterations can be done before implementation
 - After the model is implemented in the tool, the vocabulary is available for rule authoring in the Operational Decision Manager rule editors
 - By working with the rules, rule authors discover whether vocabulary is missing
 - Developers implement missing vocabulary in Rule Designer

Unit title

© Copyright IBM Corporation 2017

Figure 2-45. Ensuring completeness in the model

Completing your model might require several iterations of trying to write rules in the rule editors, discovering missing vocabulary, asking developers to implement missing vocabulary in Rule Designer, and then working with updated implementations.

The best way to ensure that your model supports your rules is by trying to write your rules against the implemented model. However, you can try to minimize the number of iterations by trying to validate your model even before developers implement it.

Exercise: Building the model on paper

Unit title

© Copyright IBM Corporation 2017

Figure 2-46. Exercise: Building the model on paper

Exercise objectives

- Use business policy documents and use cases to create a class diagram that models the business domain
- Identify the connection between the model and the vocabulary that is used to author rules

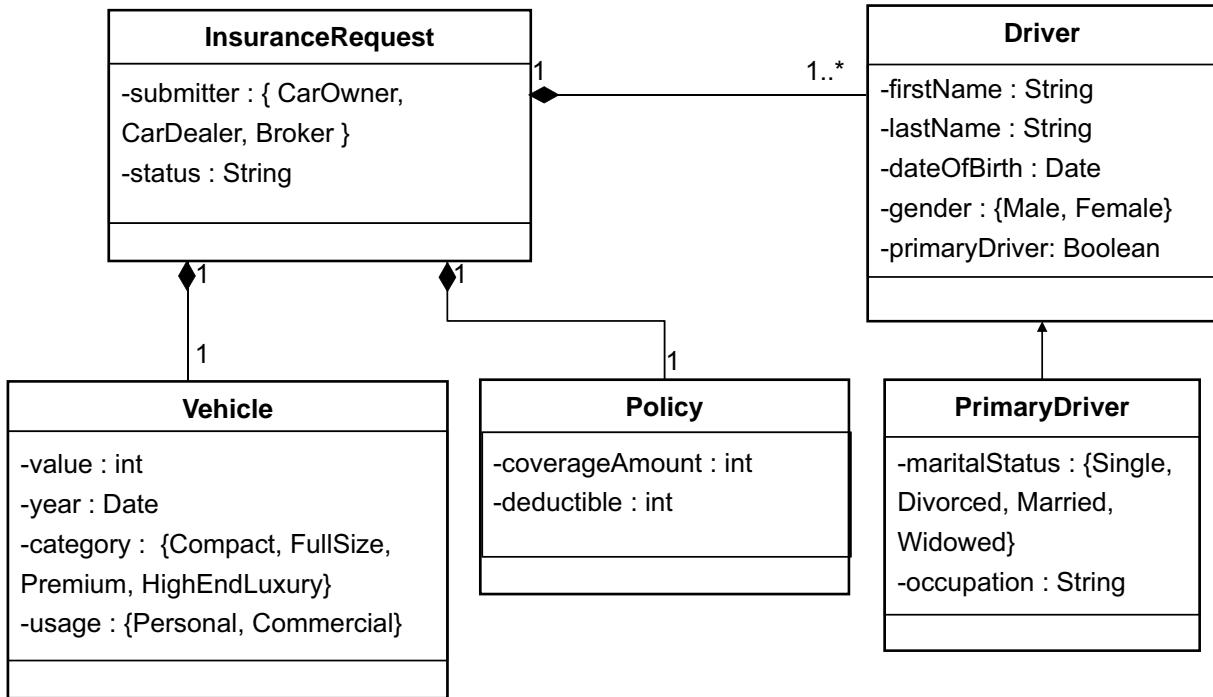


Unit title

© Copyright IBM Corporation 2017

Figure 2-47. Exercise objectives

Exercise review: A possible solution



Unit title

© Copyright IBM Corporation 2017

Figure 2-48. Exercise review: A possible solution

This diagram is a possible solution for the in-class discussion section of [Exercise 2, "Building the model on paper"](#).

In this example solution:

- **InsuranceRequest** is a composition
 - **Policy**, **Vehicle**, and **Driver** exist only in relation to **InsuranceRequest**
- **PrimaryDriver** is a subclass of **Driver**
 - **PrimaryDriver** is a *specialization* of **Driver** (inheritance)
 - As a specialization, **PrimaryDriver** inherits the attributes of **Driver** and allows for more specialized attributes, such as occupation
- Cardinality:
 - There must be one **InsuranceRequest** (cardinality: 1)
 - There must be one **Vehicle** (cardinality: 1) and one **Policy** (cardinality: 1) per **InsuranceRequest**
 - There can be multiple instances of **Driver** (cardinality: 1..*)
- Class and attribute names follow general Java naming conventions

- Attributes are listed with their types
 - Types in braces ({}), indicate **domains**, which are covered in a later unit. A domain represents a set of values.
 - Domains are represented in the rule editor as a list of values that you can choose from when authoring a rule, such as **{Personal, Commercial}** for **usage**.

This diagram is used in the next exercise, which is [Exercise 3, "Implementing the model"](#).

2.6. Implementing the model

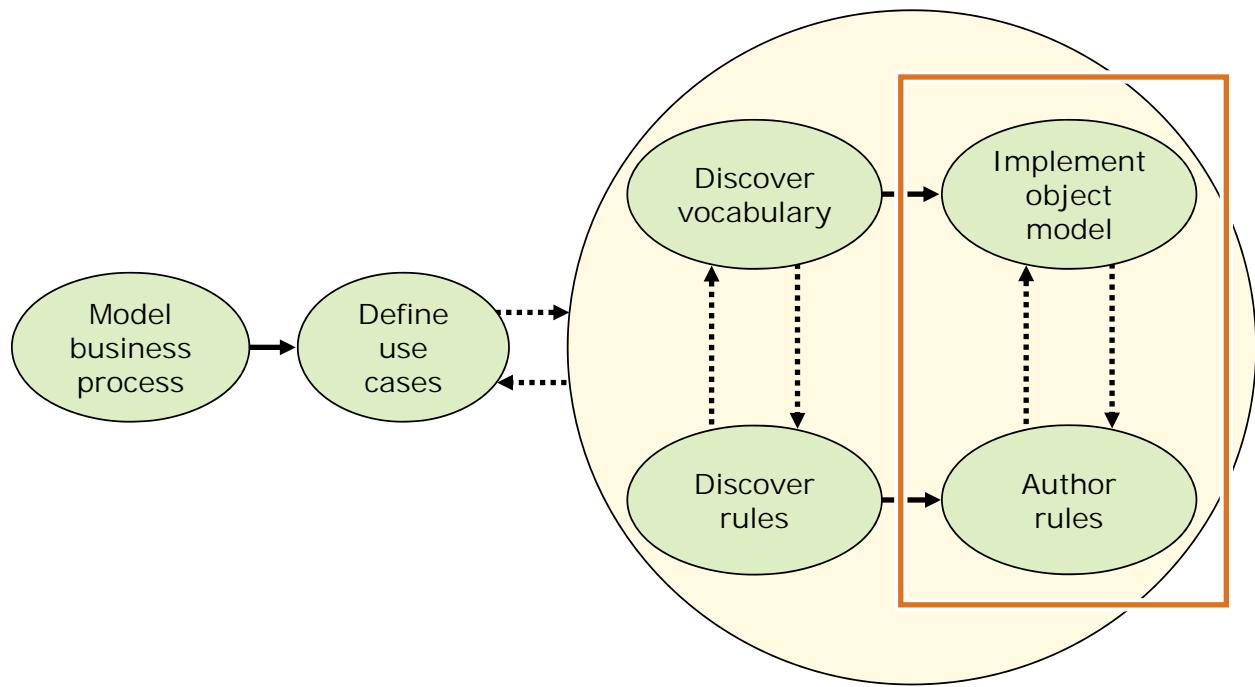
Implementing the model

Unit title

© Copyright IBM Corporation 2017

Figure 2-49. Implementing the model

Implementing the model



Unit title

© Copyright IBM Corporation 2017

Figure 2-50. Implementing the model

Just as vocabulary and rule discovery are intertwined, vocabulary and rule implementation are also intertwined.

Agile business rule application development involves short cycles of discovery and implementation. As soon as some vocabulary and rules are discovered, they can be implemented in the tools.

From discovery to implementation

- As soon as initial versions of rules and vocabulary are validated with policy managers, developers can start implementation
- Development activities to set up the rule authoring environment include:
 - Implementing the object model (class diagram) in either Java code or XML
 - Creating a *ruleflow* that is based on decision execution logic
 - Defining *ruleset variables* that are based on input and output objects so the application can communicate with the rule engine

Unit title

© Copyright IBM Corporation 2017

Figure 2-51. From discovery to implementation

Developers can start preparing the rule authoring environment while you are validating initial versions of the vocabulary and rules with policy managers.

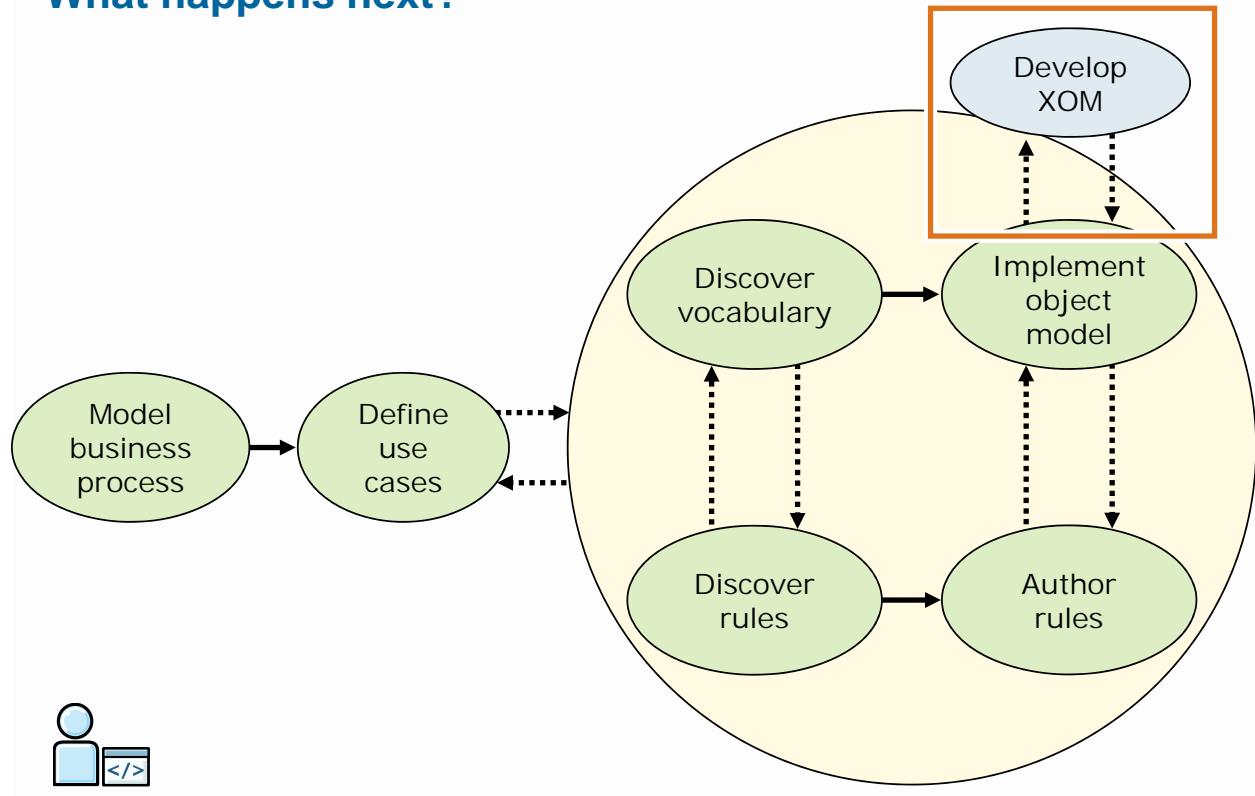
Based on discussions with you, the developers implement an executable version of your object model in Java code or XML, which is what enables rules to be executed. They also reproduce your object model as a business object model (BOM). Operational Decision Manager provides default vocabulary when the BOM is created, but you must review it. After the model is implemented, you can review the vocabulary in the Rule Designer Vocabulary view.

The decision logic is implemented as a *ruleflow*, which graphically outlines the sequence in which rules are evaluated to produce the decision.

You also collaborate with developers on how to implement the communication between the application and the rules. You use *ruleset variables* to pass objects to the rule engine, and return the decision to the application. Based on the type of input, developers can determine the best way to write the rules and optimize rule execution. For example, if rules are evaluated against only one customer at a time, you might write them differently than if groups of customers are provided as input.



What happens next?



Unit title

© Copyright IBM Corporation 2017

Figure 2-52. What happens next?

This graphic shows the next step, “Develop XOM,” in which developers implement the execution object model (XOM). The XOM must be developed before the rules can run.

IBM Training



Using an agile development approach

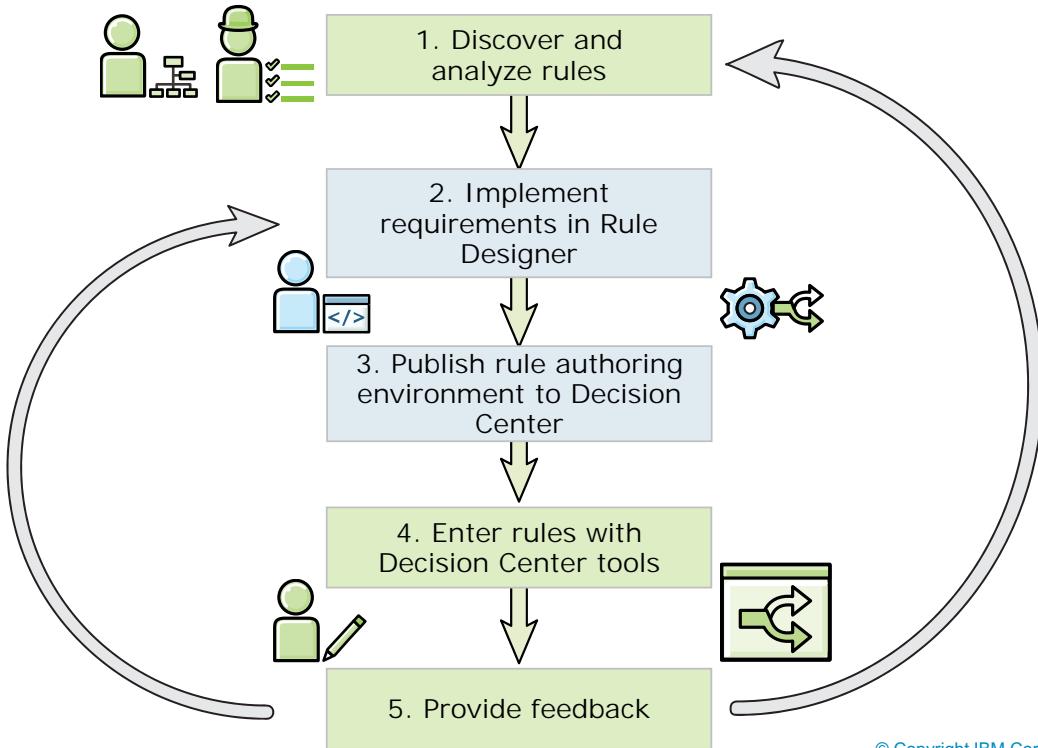


Figure 2-53. Using an agile development approach

Decision management projects require extensive collaboration between business and technical stakeholders in the project. To ensure that the rules were discovered and analyzed correctly, the business users must start entering rules into the tool as soon as possible. This need requires that developers prepare the rule authoring environment early in the development cycle, even before all the rules are discovered.

The following development tasks are illustrated on this slide:

1. Early results from rule discovery, including vocabulary models and initial rules, are passed to the developers as requirements.
2. Based on these requirements and discussion with the business analyst, developers work in Rule Designer to design and set up the rule authoring environment.
3. When the rule authoring environment is ready, it is published to Decision Center so that business users can access it through the Decision Center consoles. Both developers and business users can continue to work simultaneously on the same project but in their separate environments on separate tasks.
4. As rule authors begin entering rules in Decision Center, they might find problems or discover new requirements.

5. The rules might look good on paper, but trying to implement them in the tool can uncover possible design or analysis issues.

By using an agile or iterative approach, the business users can provide early feedback on the project implementation, which allows the developers to respond more easily to requirement or model changes. The project might go through several cycles of implementation and feedback loops, but these iterations become less frequent as the project stabilizes.

Unit summary

- Model a business process
- Define use cases
- Explain UML notation and the modeling concepts that are relevant to modeling business rule vocabulary
- Describe the interaction between the business and development teams during the initial phases of business rule application development
- Use business policy documents and use cases to create a class diagram that models the business domain
- Identify the connection between the model and the vocabulary that is used to author rules
- Implement the business model in Rule Designer
- Write rules against the model to test for completeness

Unit title

© Copyright IBM Corporation 2017

Figure 2-54. Unit summary

Review questions

1. True or False: Business users are not involved in initial phases of developing a business rule application because their input is required only when a project is near completion.
2. True or False: Discovering rule vocabulary is a developer task that is done entirely in Rule Designer.
3. True or False: Business users can use UML class diagrams to communicate rule vocabulary requirements to the development team.
4. True or False: Because defining the BOM and vocabulary is intertwined with rule discovery, you cannot ensure that your BOM is complete until you start writing rules.



Unit title

© Copyright IBM Corporation 2017

Figure 2-55. Review questions

Write your answers here:

- 1.
- 2.
- 3.
- 4.

Review answers (1 of 2)

1. True or False: Business users are not involved in initial phases of developing a business rule application because their input is required only when a project is near completion.
The answer is False. Business users are experts in business policy and are the main sources for discovering rules and rule vocabulary.

2. True or False: Discovering rule vocabulary is a developer task that is done entirely in Rule Designer.
The answer is False. Discovering rule vocabulary starts “on paper” with business models and extensive interviews between business analysts and policy managers.



Unit title

© Copyright IBM Corporation 2017

Figure 2-56. Review answers (1 of 2)

Review answers (2 of 2)

3. True or False: Business users can use UML class diagrams to communicate rule vocabulary requirements to the development team.
The answer is True.

4. True or False: Because defining the BOM and vocabulary is intertwined with rule discovery, you cannot ensure that your BOM is complete until you start writing rules.
The answer is True.



Unit title

© Copyright IBM Corporation 2017

Figure 2-57. Review answers (2 of 2)

Exercise: Implementing the model

Unit title

© Copyright IBM Corporation 2017

Figure 2-58. Exercise: Implementing the model

Exercise objectives

- Implement the business model in Rule Designer
- Write rules against the model to test for completeness

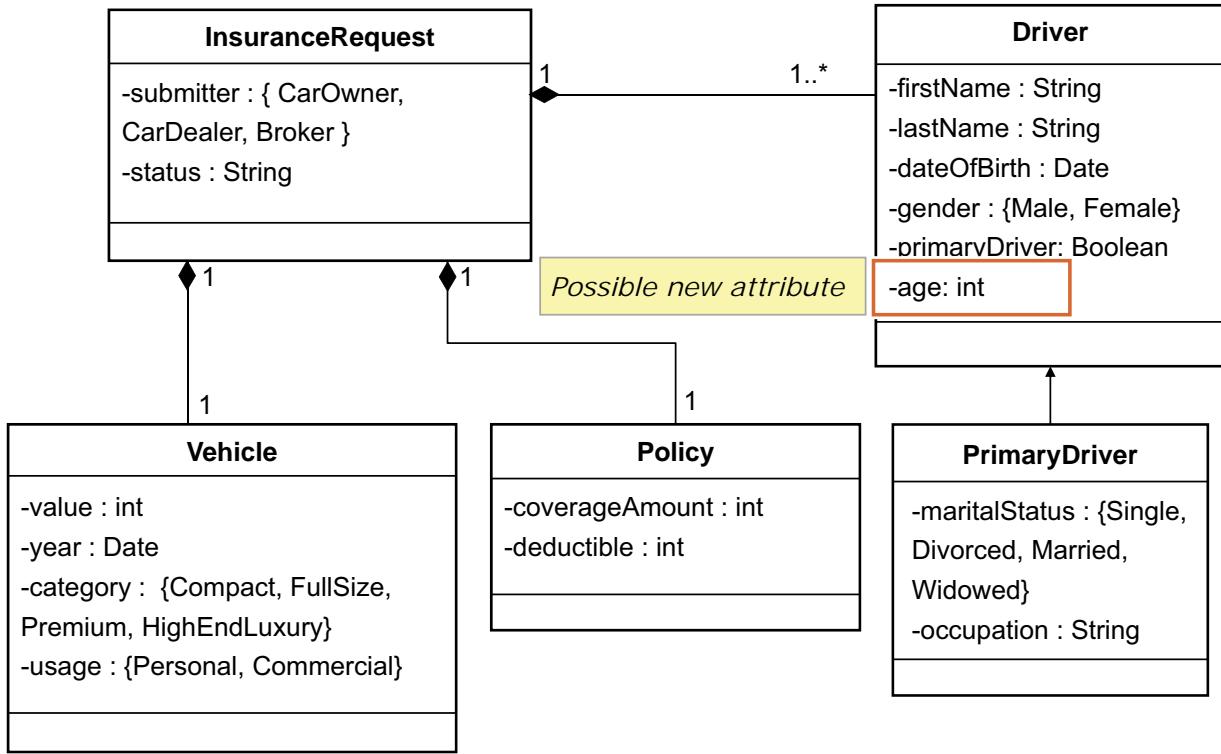


Unit title

© Copyright IBM Corporation 2017

Figure 2-59. Exercise objectives

Exercise review (1 of 2): A possible solution



Unit title

© Copyright IBM Corporation 2017

Figure 2-60. Exercise review (1 of 2): A possible solution

In [Exercise 3, "Implementing the model"](#), you created a BOM based on the class diagram from [Exercise 2, "Building the model on paper"](#). You then started authoring rules that are based on the Exercise 2 use case. However, “age” did not appear in the vocabulary list, so you were not able to create rules that included the age of the driver.

This class diagram shows an update to the **PrimaryDriver** class: a possible new attribute called “age.” As noted in Exercise 3, the business user can ask the development team for help with adding “age” to the vocabulary list:

- The business user realizes that “age” is missing from the vocabulary.
- The business user asks the development team for help in creating vocabulary for “age.”
- Developers update the code so that “age” is included in the vocabulary. In this case, the developer:
 - Writes a function that calculates the age of the driver, as based on the `dateOfBirth` attribute
 - Updates the vocabulary in the BOM to include the age of the driver
- The business user can then author rules that include the age of the driver.

Exercise review (2 of 2)

In this exercise, you did the following tasks:

- Modeled the rule vocabulary by working with use cases and business policies
- Implemented your class diagram as a Business Object Model (BOM) in Rule Designer
- Authored some rules to test the BOM completeness
- Discovered missing vocabulary

Unit title

© Copyright IBM Corporation 2017

Figure 2-61. Exercise review (2 of 2)

This slide summarizes what you did in [Exercise 3, "Implementing the model"](#).

Unit 3. Understanding decision services

Estimated time

01:00

Overview

This unit continues the focus on introducing you to business rule concepts and practices. You learn how to create a decision service.

How you will check your progress

- Checkpoint
- Exercise

Unit objectives

- Describe the decision service rule project structure
- Explain how ruleflows orchestrate rule execution
- Describe synchronization mechanisms between the business and development environments

Unit title

© Copyright IBM Corporation 2017

Figure 3-1. Unit objectives

In this unit, you learn about how the rule authoring environment is set up in Rule Designer. Seeing this process can help you understand when you must collaborate with developers during the design and development of the business rule application.

Unit 2, "Modeling for business rules" introduced the top-down approach of business rule application development. In that unit, you started on paper with a class diagram, and then you implemented your object model in Rule Designer without worrying about code.

This unit shows you how developers approach a project, by starting in Rule Designer with the code implementation, which is through either Java objects or XML data. Developers generate a BOM from that code, and create the other artifacts that are required to make the data model work.

Topics

- Designing the business rule application
- Setting up a decision service
- Design signature
- Orchestrating
- Sharing and synchronizing rule projects

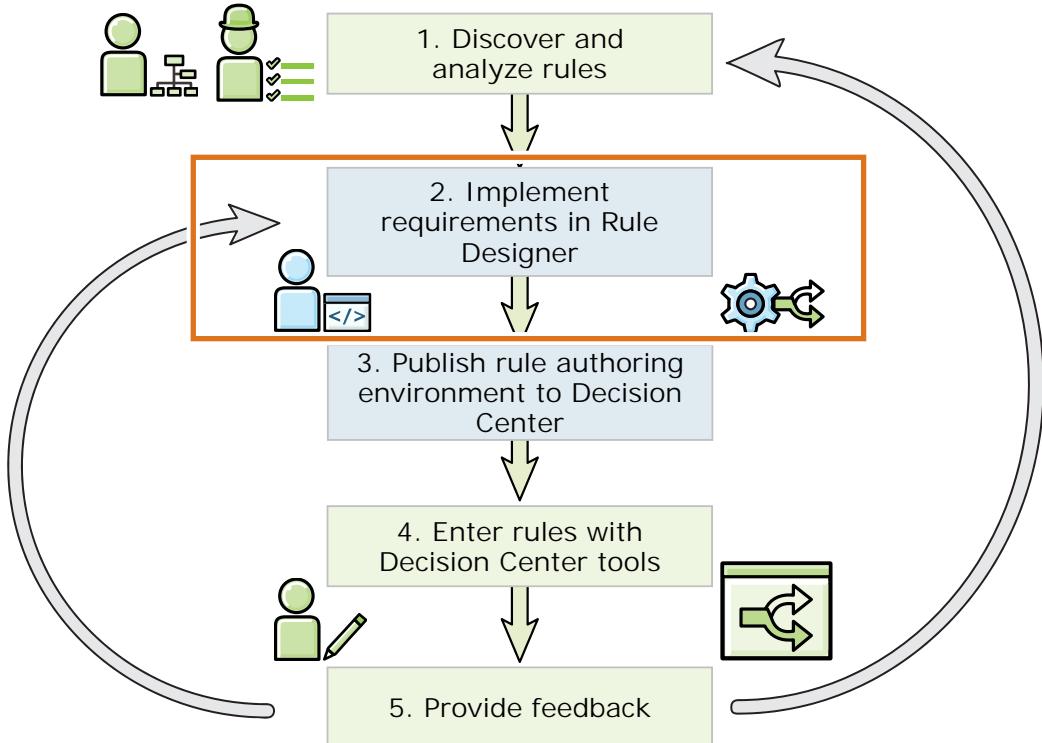
Unit title

© Copyright IBM Corporation 2017

Figure 3-2. Topics



Recall: Agile business rule development



Unit title

© Copyright IBM Corporation 2017

Figure 3-3. Recall: Agile business rule development

As explained in the previous unit, architects and business users work together to discover and analyze rules (Step 1).

Based on these requirements and discussion with the business analyst, developers work in Rule Designer to design and set up the rule authoring environment. As a first step, developers prepare a *decision service*. Decision services organize *rule projects*, which are used as containers for the newly discovered rules and vocabulary (Step 2). A decision service has a *main rule project* in its project hierarchy, which serves as an entry point into the decision service. The main rule project references all of the other rule projects in the decision service.

This unit focuses on how rules are structured and managed through decision services.

3.1. Designing the business rule application

Designing the business rule application

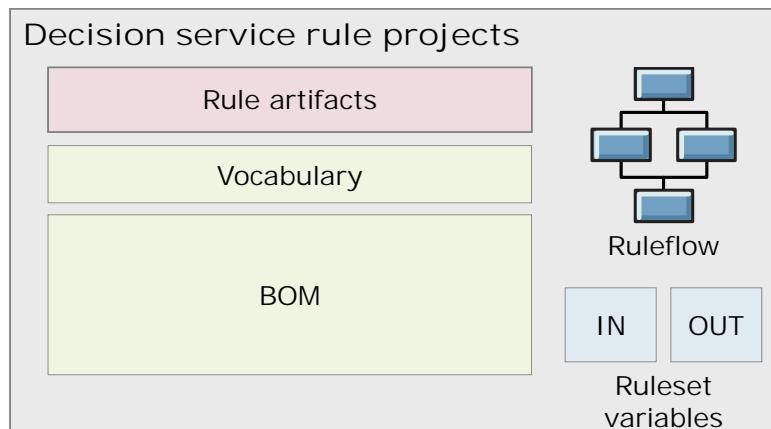
Unit title

© Copyright IBM Corporation 2017

Figure 3-4. Designing the business rule application

Designing the business rule application

- Developers and architects design the business rule application according to the models and requirements that you provide
- The first development task is to set up rule authoring environment for business users by creating ***decision services***, which contain ***rule projects***



Unit title

© Copyright IBM Corporation 2017

Figure 3-5. Designing the business rule application

From the development perspective, the business rule application starts in Rule Designer. Developers and architects design the business rule application that is based on the models and requirements that you provide.

To set up the rule authoring environment infrastructure for business users, developers create ***decision services***, which contain ***rule projects***. Rule projects are the basic containers for rule authoring artifacts.

Decision service rule projects contain everything that is required for business users to start authoring rules in the rule editors. Understanding how decision services are structured can help you better understand rule authoring.

Decision service rule projects include:

- Business object model (BOM) and vocabulary
- Ruleflow, which outlines the general sequence in which business logic must be evaluated within the ruleset to produce the correct decision (The ruleflow outline can be defined even before the rules are written.)
- Ruleset variables, which describe the format of input that is required to make a decision and how the decision results should be returned. (These variables are called the *ruleset signature*.)

Rule project structure (1 of 2)

- A standard rule project template contains these folders:

Folder	Contains
rules	Stores rule artifacts, including action rules, decision tables, decision trees, and ruleflows
bom	Stores the business object model and the vocabulary that is used in the rules
queries	Stores queries that can be run to find certain rules and apply actions on those rules
resources	Stores any type of file that is not part of the rule model
templates	Stores templates, or partially filled rules, that can serve as a starting point when creating rules

Unit title

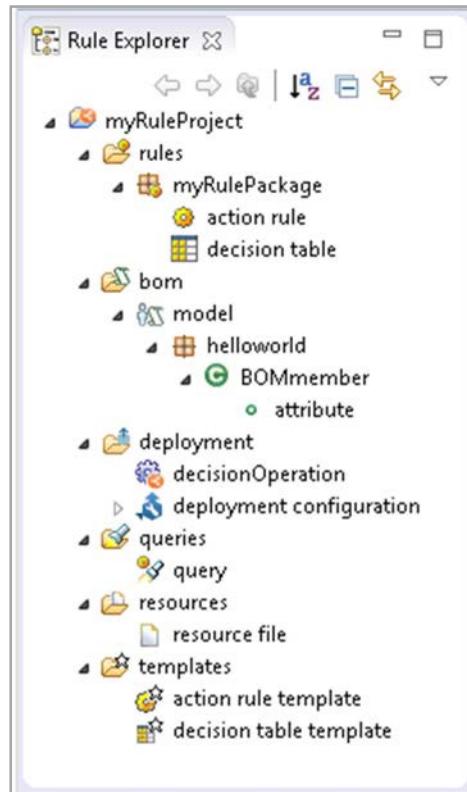
© Copyright IBM Corporation 2017

Figure 3-6. Rule project structure (1 of 2)

By default, a standard rule project includes these folders.

Rule project structure (2 of 2)

- Created in Rule Designer
- Set of folders to contain all rule authoring artifacts



Unit title

© Copyright IBM Corporation 2017

Figure 3-7. Rule project structure (2 of 2)

As shown here, the rule project structure is a set of folders that contain all the artifacts that you need to start rule authoring. This slide has an example of a decision service rule project.

Normally, developers prepare rule projects, complete with the business object model (BOM), the supporting code, and some templates. Rule projects can be managed through decision services or by using classic rule projects.

After the rule projects are set up, business users can then start authoring rules.

Decision services and classic rule projects (1 of 2)

- Two different ways to manage rule projects: classic rule projects and decision services
 - These two approaches share features, but manage them in different ways
- Classic rule projects
 - Stores the business logic of rule application
 - Contains rule artifacts, business object model (BOM), vocabulary, and reference to execution object model (XOM)
 - Not shown by default in the Business console
- Decision services
 - Allows several rule projects to be grouped, deployed, and managed as one entity
 - Has one main rule project that serves as entry point into decision service
 - Decision service can contain several projects that have rules and other elements
 - Classic rule projects can be migrated into decision services

Unit title

© Copyright IBM Corporation 2017

Figure 3-8. Decision services and classic rule projects (1 of 2)

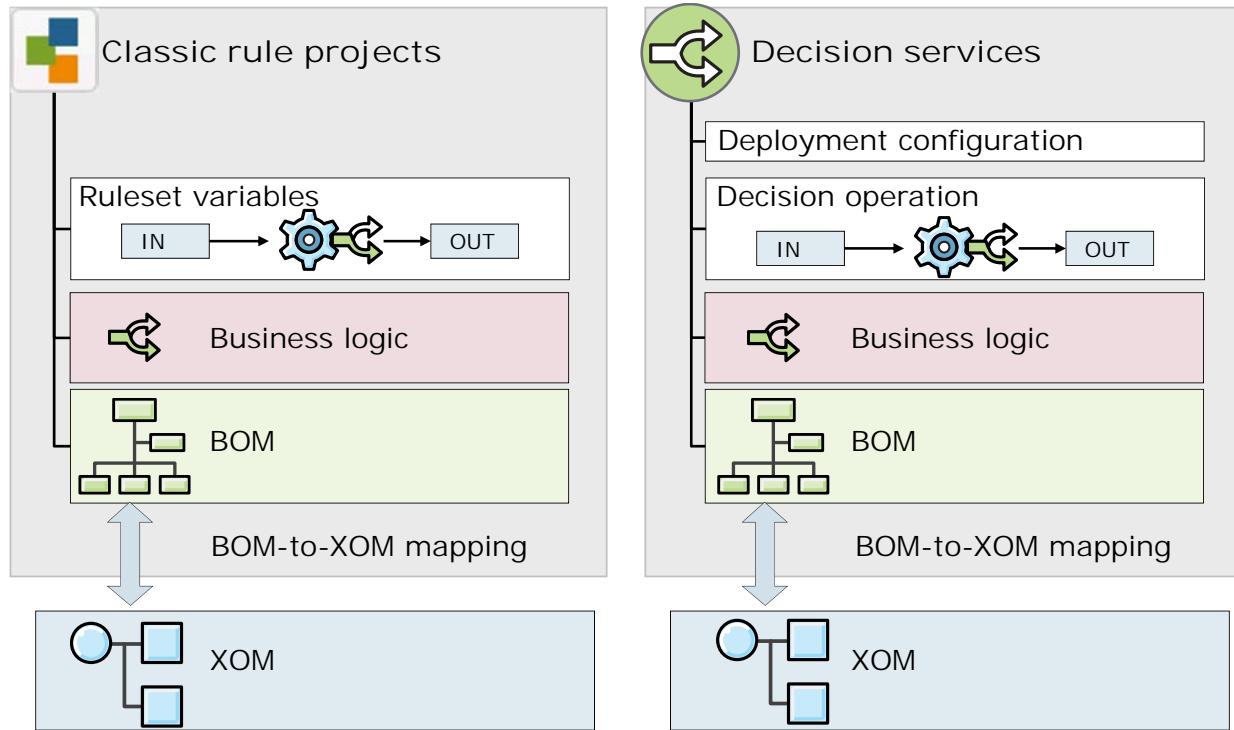
Operational Decision Manager offers two approaches to maintaining business logic: classic rule projects and decision services. Both classic rule projects and decision services start with the rule project as the basic container. While they share certain features, these features are managed in different ways.

A classic rule project stores the business logic of the rule application. It contains all of the artifacts that are needed to author rules: the business object model (BOM), vocabulary, and a reference to the execution object model (XOM). Rules are stored in folders that are called rule packages. By default, classic rule projects are not accessible in the Business console. However, the Business console can be configured to display classic rule projects.

A decision service is an approach to rule project management that contains one or more rule projects. You can use decision services to group several rule projects so that they can be managed and deployed as one entity. A decision service has one main rule project, which is used as the main reference for managing the decision service, and it can contain other standard rule projects. You can easily move among the projects to make changes, and deploy the service to a test or production environment.

If classic rule projects meet certain criteria, they can be migrated into decision services. Decision services cannot be migrated to standard rule projects.

Decision services and classic rule projects (2 of 2)



Unit title

© Copyright IBM Corporation 2017

Figure 3-9. Decision services and classic rule projects (2 of 2)

This diagram highlights some of the similarities and differences between classic rule projects and decision services.

- Classic rule projects
 - Design a hierarchy of rule projects to organize artifacts
 - Rulesets are managed within rule projects
 - Groups of rulesets are deployed together within RuleApp
 - No governance over modifications

A decision service contains a set of related rule projects. Decision services incorporate governance features so that all related rule projects are managed and deployed as a single unit.

- Each ruleset within the decision service is defined as a *decision operation* with a unique signature of input and output parameters
 - The decision operation also includes ruleset parameters to pass data from the calling application to the ruleset, and to retrieve data from the ruleset.
 - Multiple decision operations can share a BOM and rule project hierarchy, and be managed and deployed as a single unit.

Working with decision services in Decision Center

- Decision services can be governed or ungoverned
 - Work on governed decision services through the decision governance framework
 - Working on ungoverned decision services is similar to working on classic rule projects
 - Both the Business console and the Enterprise console support work on governed and ungoverned decision services
- Business console
 - Primary interface for working with decision services
 - Use decision governance framework to manage and perform changes activities, validation activities, and deployment for governed decision services
- Enterprise console
 - Cannot edit or create rule artifacts in governed decision services, but some configuration tasks can be done (managing branches and user permissions)

Unit title

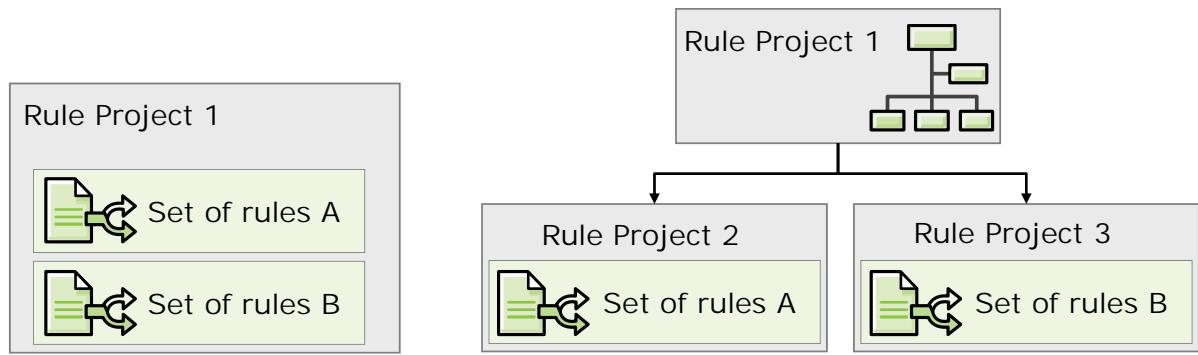
© Copyright IBM Corporation 2017

Figure 3-10. Working with decision services in Decision Center

Decision service governance and the decision governance framework are covered in greater detail in Unit 14, "Introducing decision governance" and in [Exercise 18, "Working with the decision governance framework"](#).

Modular structure

- Projects can reference other projects
 - Facilitates dependencies between your rules and data
 - Facilitate the assignment of permissions in Decision Center



Unit title

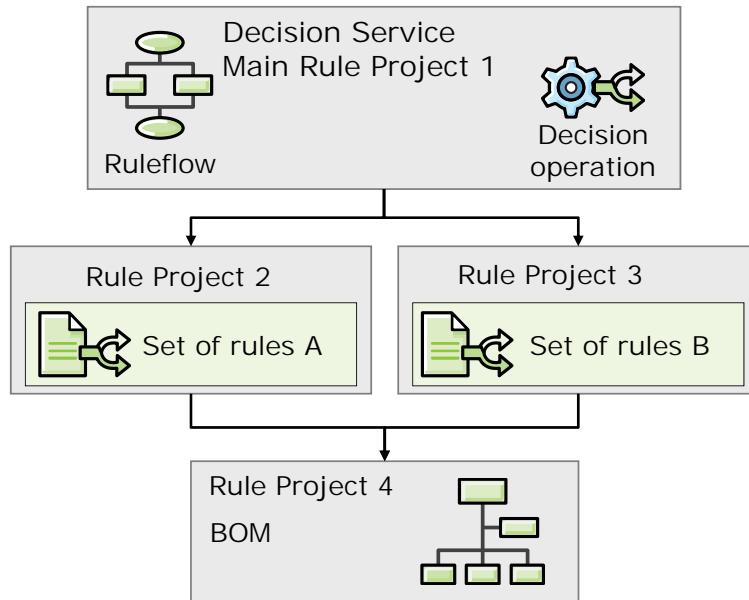
© Copyright IBM Corporation 2017

Figure 3-11. Modular structure

Rule projects can reference each other. Rules in one project can be dependent upon rules or other artifacts in a separate project.

Decision services: Modular structure (1 of 2)

- Principles of modular structure apply to decision services
 - Use **Main Rule Project** as top-level project in hierarchy
 - Use **Standard Rule Project** for child projects



Unit title

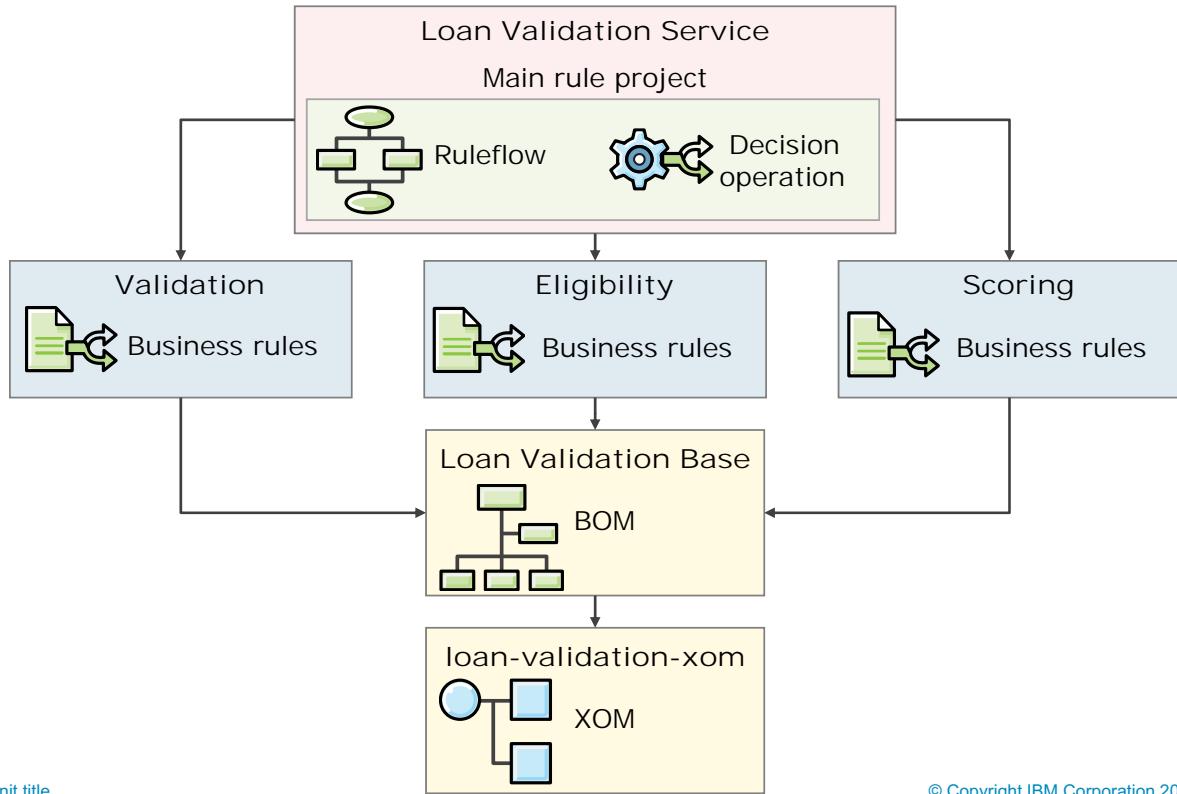
© Copyright IBM Corporation 2017

Figure 3-12. Decision services: Modular structure (1 of 2)

To support the decision governance framework, you can define a set of rule projects as a decision service. By defining project dependencies between the decision service projects, the business users can manage the lifecycle of all projects within the decision service.



Decision services: Modular structure (2 of 2)



Unit title

© Copyright IBM Corporation 2017

Figure 3-13. Decision services: Modular structure (2 of 2)

This diagram shows another example of a decision service structure.

A decision service must have a main rule project, which is the top-level project in the rule project hierarchy. The main rule project serves as the entry point to the decision service. Main rule projects also reference the other (child) rule projects in the decision service. Child projects in a decision service are called standard rule projects.

In this diagram, **Loan Validation Service** is the main rule project. It references the child standard rule projects, which are called **Validation**, **Eligibility**, and **Scoring**. These rule projects help organize the business rules for the decision service, and they reference the business object model (BOM). The BOM for **Loan Validation Service** is managed in its own rule project, called **Loan Validation Base**, which is dedicated to organizing and managing the BOM. Finally, the BOM in **Loan Validation Base** references the decision service XOM, which is managed in a separate project that is called **loan-validation-xom**.

3.2. Setting up a decision service

Setting up a decision service

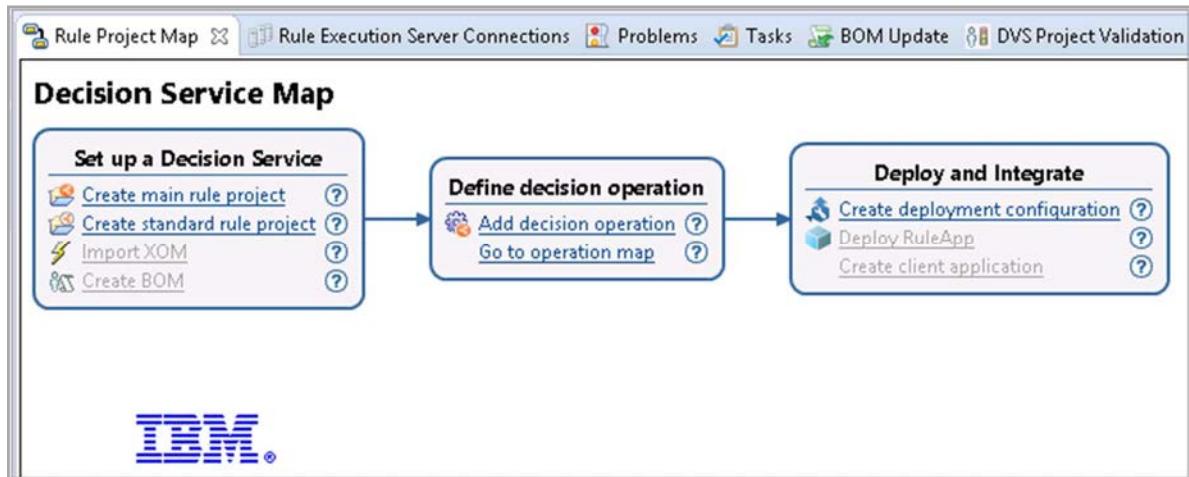
Unit title

© Copyright IBM Corporation 2017

Figure 3-14. Setting up a decision service

Decision service map

- In Rule Designer, the Decision Service Map outlines the main phases and tasks for setting up a decision service
- Decision Service Map example:



Unit title

© Copyright IBM Corporation 2017

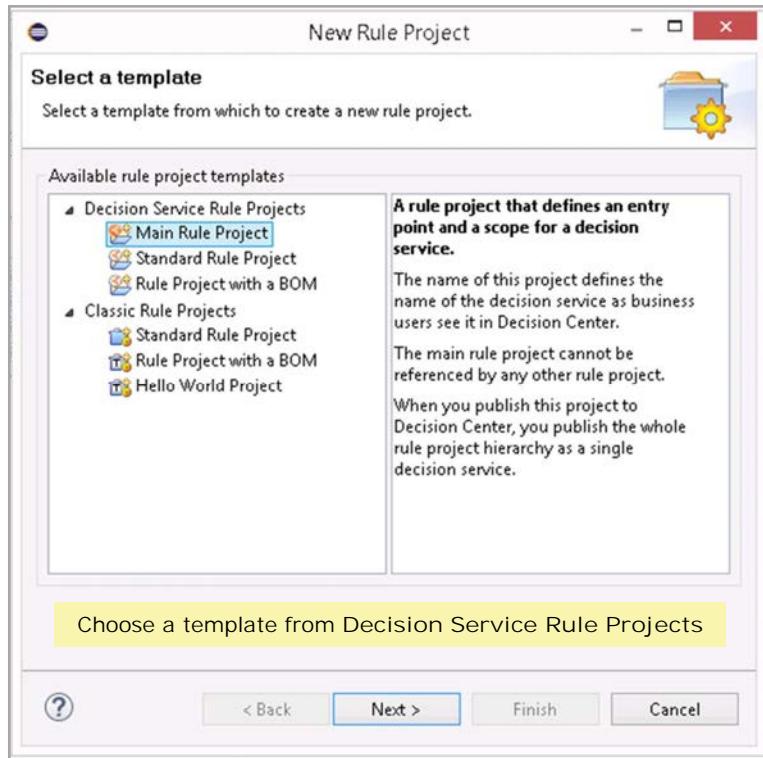
Figure 3-15. Decision service map

Rule Designer includes a Decision Service Map that helps set up a decision service. You work with this map during the exercises.

Rule Designer also provides the Rule Project Map for classic rule projects. However, the focus in this course is on decision services.



Create a decision service rule project in Rule Designer



Unit title

© Copyright IBM Corporation 2017

Figure 3-16. Create a decision service rule project in Rule Designer

To create a decision service rule project, you use the New Rule Project wizard in Rule Designer to choose from the **Decision Service Rule Projects** templates that are provided.

If the decision service has only one rule project, you must define it as the main rule project.

Decision service rule project templates

- Decision service rule project templates contain these folders:

Folder	Contains
rules	Stores rule artifacts, including action rules, decision tables, decision trees, and ruleflows
bom	Stores the business object model and the vocabulary that is used in the rules
deployment	Stores the decision operations and deployment configurations
queries	Stores queries that can be run to find certain rules and apply actions on those rules
resources	Stores any type of file that is not part of the rule model
templates	Stores templates (partially filled rules) that can serve as a starting point when creating rules

Unit title

© Copyright IBM Corporation 2017

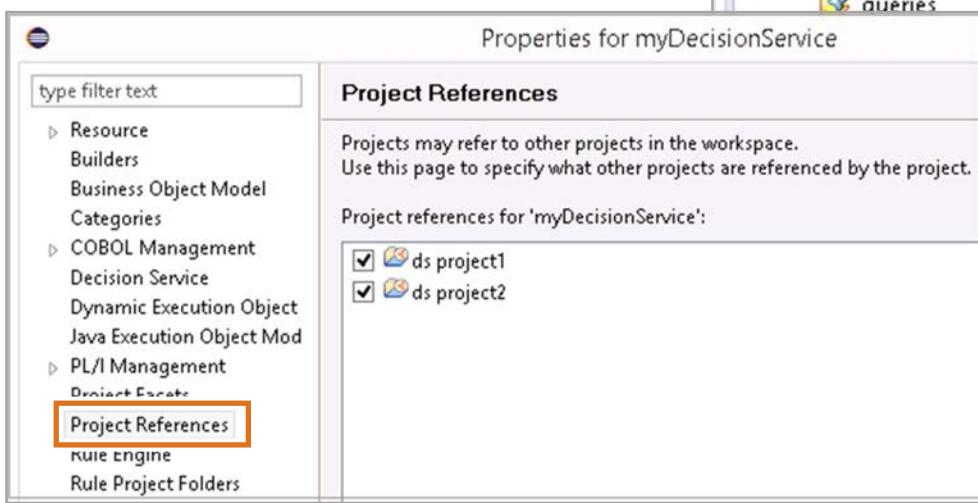
Figure 3-17. Decision service rule project templates

By default, a standard decision service rule project includes the basic rule project folders, plus the **deployment** folder.



Decision service projects

- A main decision service rule project is distinguished from other projects with an icon
- Define project references to other projects in the Properties window



© Copyright IBM Corporation 2017

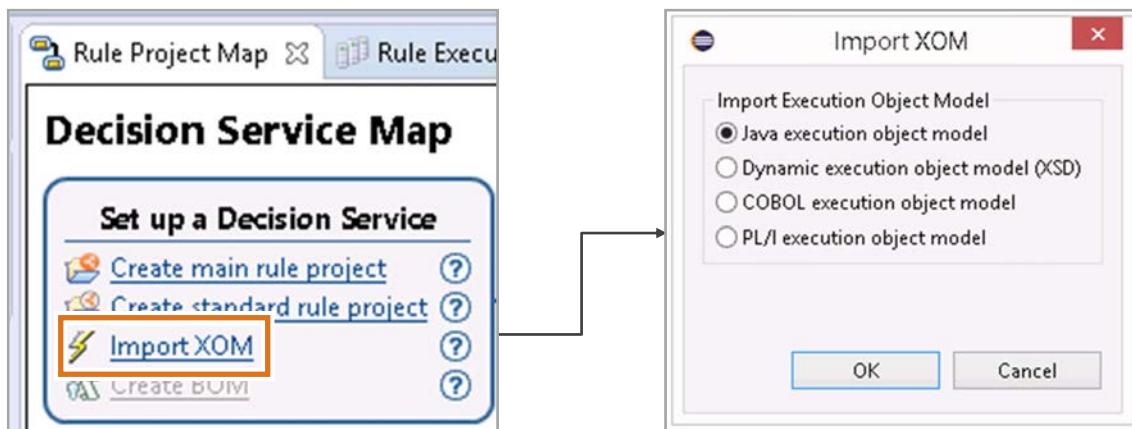
Figure 3-18. Decision service projects

The project hierarchy is defined by the main decision service project. If other rule projects are included in the decision service, they are referenced by the main project. This hierarchy simplifies management of the decision service lifecycle. For example, when you synchronize a decision service with Decision Center, all dependent projects are automatically included. When business users open a decision service in the Business console, depending on permissions, they can see any of the projects included in the decision service.



Decision service setup: Import XOM

- Developers write the execution object model (XOM), which makes rule execution possible
- The rule authoring environment must reference the project that contains the XOM
- XOM can be written in various formats, such as Java or XML



Unit title

© Copyright IBM Corporation 2017

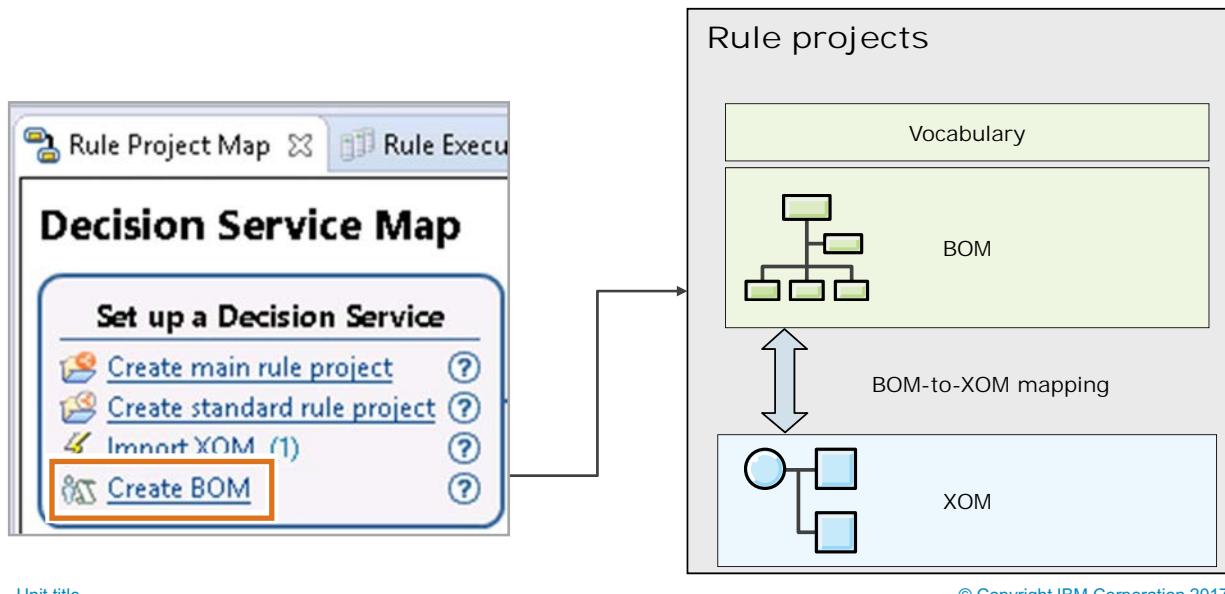
Figure 3-19. Decision service setup: Import XOM

The business logic can be implemented by using either Java objects or an XML schema (XSD). As mentioned earlier, the implementation code is called the execution object model (XOM). The XOM makes rule execution possible. While business rules use the vocabulary from the BOM, each business element in a rule must have a corresponding XOM implementation.

Regardless of how developers implement the XOM, whether in Java or XML, they can build the BOM to match your original models and requirements. If your data model is as complete as possible when you pass it to developers, the implementation of that model becomes more usable for rule authoring. Clear and complete requirements mean fewer iterations between you and developers to stabilize the BOM.

Decision service setup: Create a BOM

- BOM is “business view” of code
 - Automatically generate BOM from XOM in Rule Designer
- Vocabulary that is used in the rules depends on BOM members



Unit title

© Copyright IBM Corporation 2017

Figure 3-20. Decision service setup: Create a BOM

After developers write the code for the XOM, the BOM can be automatically generated in Rule Designer. The BOM is a business view of the code. The link between the BOM and XOM is called BOM-to-XOM mapping. Each business element that is used in a rule must have a corresponding XOM implementation.

Vocabulary that is used in the rules comes directly from the BOM. As soon as the BOM is defined, rule authoring can begin. Designer can automatically “translate” the names of BOM elements from programming language into natural language terms.

After you and the other business users start working with the BOM, you will likely uncover missing vocabulary or other fixes that require changes to the code in the XOM. You might go back and forth quite a few times between business and development teams to get the implementation right.



Note

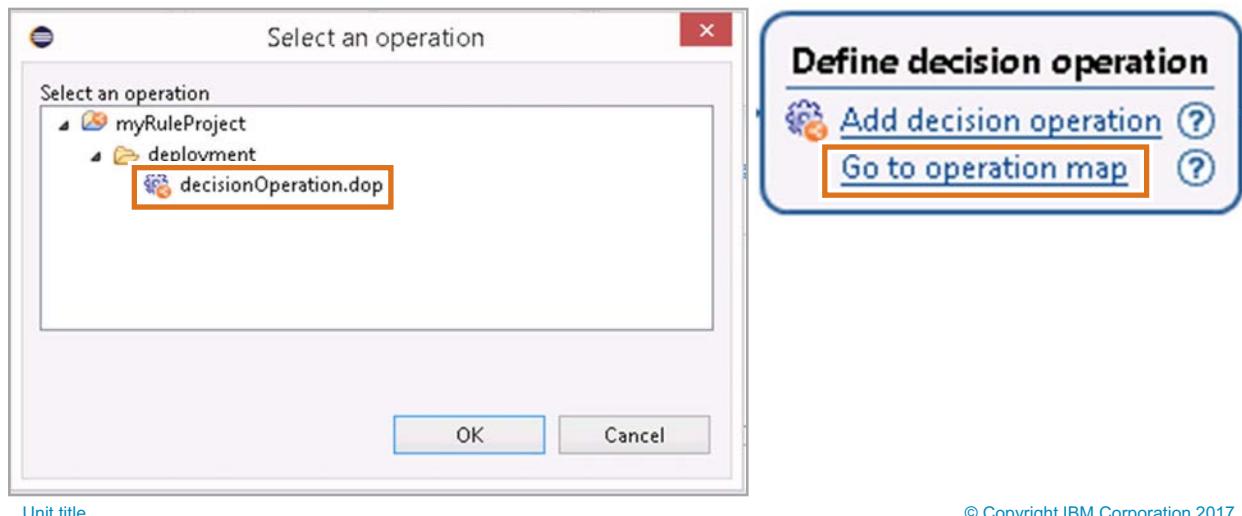
Tip: Changes to the BOM, such as new vocabulary, also incur changes to the XOM. Making sure that your requirements are clear, complete, and can facilitate implementation so that the BOM stabilizes quickly and rule authoring can proceed smoothly.

You learn more about the BOM and vocabulary later.



Decision operations

- After defining the decision service projects, you add a decision operation
 - Stored in the **deployment** folder
 - File extension: .dop
- After the decision operation is defined, you can go to the Operation Map for your decision operation



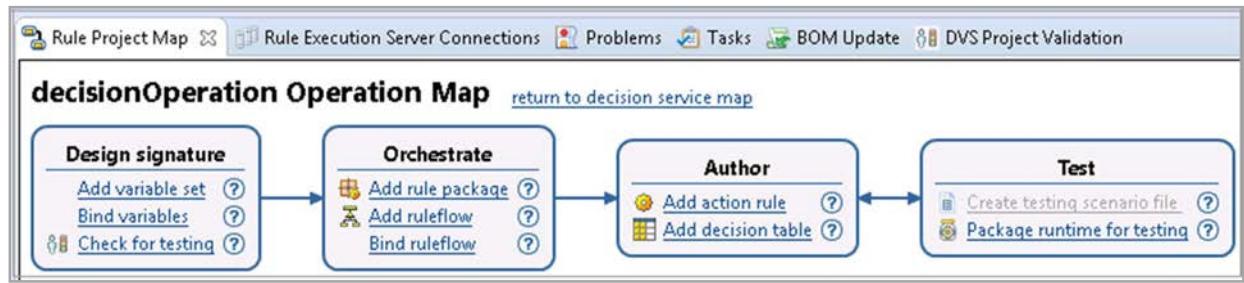
© Copyright IBM Corporation 2017

Figure 3-21. Decision operations

The Decision Service Map can be opened by clicking the **Go to operation map** link the **Define decision operation** task.

Operation Map

- The Operation Map guides you through these tasks:
 - Design signature, including definition of ruleset variables
 - Orchestrate, including rule packages and ruleflow
 - Author (rules and decision tables)
 - Test



Unit title

© Copyright IBM Corporation 2017

Figure 3-22. Operation Map

3.3. Design signature



Unit title

© Copyright IBM Corporation 2017

Figure 3-23. Design signature

Decision operations and signatures

- Each decision operation groups your rules into a ruleset
 - Specifies the rules that you want to deploy from the containing rule project, or from any of its dependent projects
 - Stand-alone, executable container that corresponds to a decision

- Each decision operation has a unique set of variables
 - Variables are bound to ruleset parameters
 - Used to exchange data between the application and the rules

- Developers and business analysts decide together on ruleset parameters
 - Which objects are expected as input from the application and are required to make the business decision?
 - Which objects are returned to the application as output that contains the results of rule processing?



Unit title

© Copyright IBM Corporation 2017

Figure 3-24. Decision operations and signatures

You use the content of the rule project to extract the set of rules, or *ruleset*, that the decision engine uses to produce a decision.

A *ruleset* is a stand-alone, executable container that corresponds to a decision.

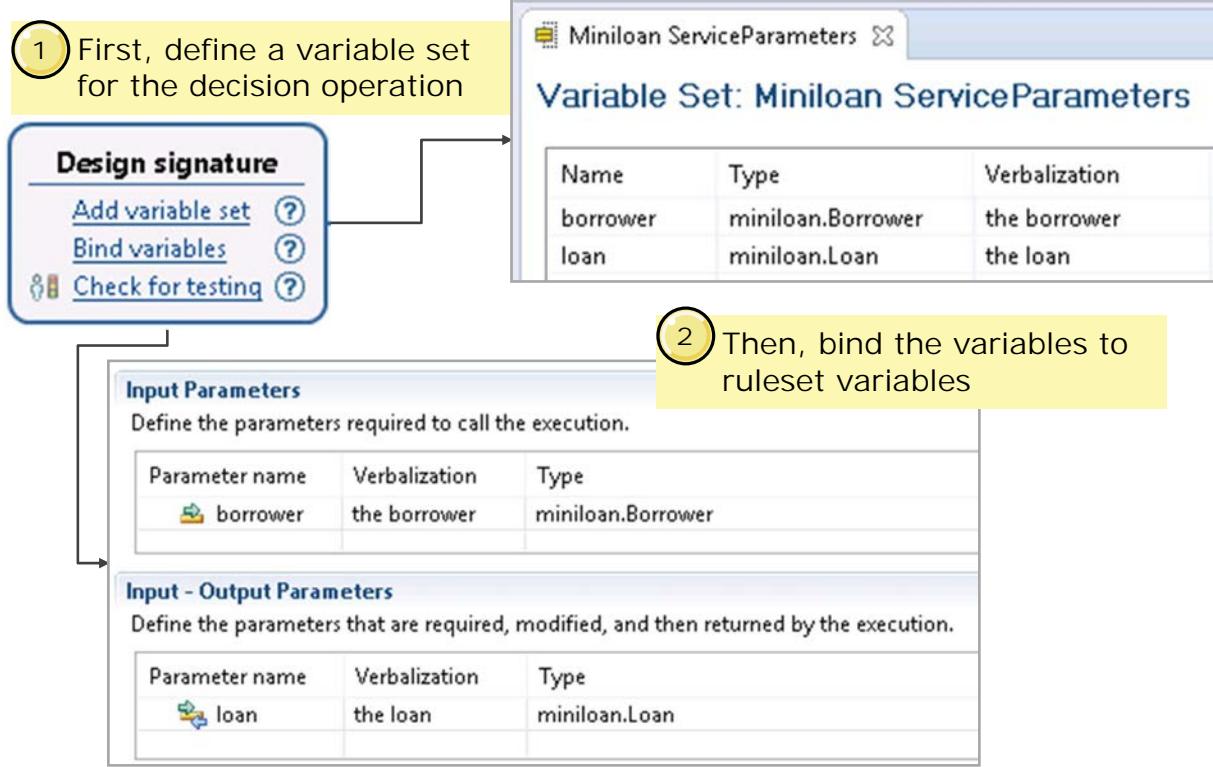
The decision engine evaluates the rules in the ruleset against a set of objects that are passed from the calling application through ruleset parameters.

You use ruleset variables to exchange objects between the application and the decision engine. After the BOM is created, you can choose which objects from the BOM are passed between the calling application and the decision service.

You work with the developers to define what information you require as a decision result and what information must be provided from the application in order for the decision engine to produce that result. This communication is implemented by defining variables for the decision operation.



Design signature



Unit title

© Copyright IBM Corporation 2017

Figure 3-25. Design signature

You bind the variables to input and output parameters to create a *signature* for your decision operation.

The signature indicates the type of data, which is either XML or Java objects, and which direction the data is sent.

- **IN:** The object is passed to the rule engine.
- **OUT:** The object is passed from the rule engine to the application.
- **IN_OUT:** The object is passed to the rule engine, and then returned to the application with some decision results.

3.4. Orchestrating

Orchestrating

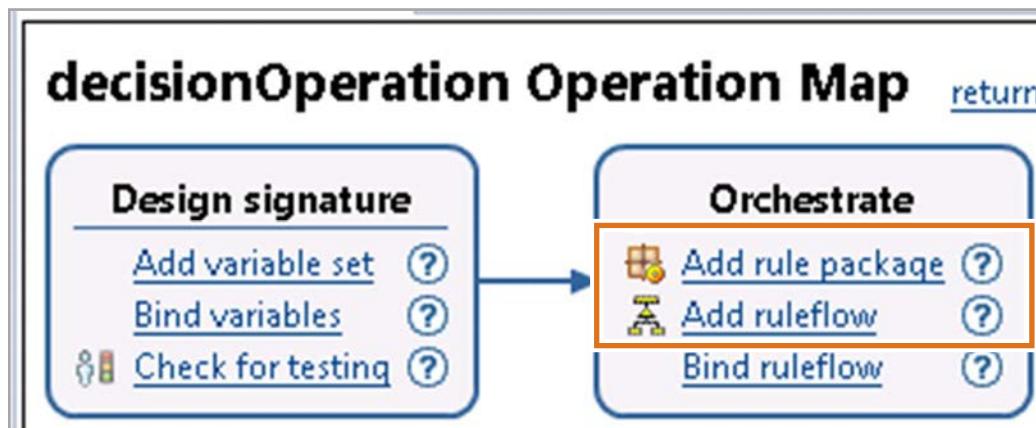
Unit title

© Copyright IBM Corporation 2017

Figure 3-26. Orchestrating

Orchestrate

- Orchestration involves:
 - Logically organizing rules into *rule packages* or folders
 - Graphically outlining the flow of rule execution with a *ruleflow*



Unit title

© Copyright IBM Corporation 2017

Figure 3-27. Orchestrate

In the **Orchestrate** part of the Rule Project Map, you organize your rules into rule packages (folders) and create a ruleflow.

Rules are designed to produce decisions, but they have no notion of sequence. The ruleflow is used to enforce the sequence in which rules are selected, and the method that the rule engine uses to evaluate them. Rule packages organize the rules logically.

Rule packages (1 of 2)

- Logical partitions
 - Multiple users can work in the same repository without hindering each other
 - Rules are stored together in a single repository
- Package structure is entirely user-defined
 - Developers and business users discuss naming and organization to ensure that the structure makes sense to all stakeholders
- Packages provide organization and structure for your rules
 - Good rule organization is one of the most important outcomes of a BRMS project

Unit title

© Copyright IBM Corporation 2017

Figure 3-28. Rule packages (1 of 2)

Packages provide a logical means of partitioning the rules so that multiple users can work in the same repository without hindering each other. Make the package structure sufficiently fine-grained to allow rule authors to work exclusively in a few packages for relatively long periods.

The package structure is entirely user-defined, so you can work with developers to ensure that the naming and organization make sense to everyone in the rule management team.



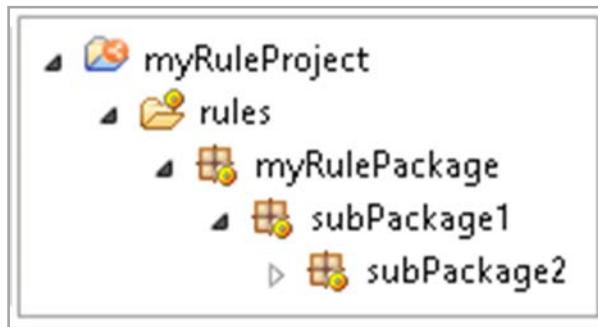
Important

When naming rules and packages, keep in mind that Windows has a 256-character file name length limitation, which can be an issue for deeply nested packages and folders.

Good rule organization is one of the most important outcomes of a BRMS project. When you define the organization in naming of rule packages, make sure that what you use makes sense to everyone in the rule management team.

Rule packages (2 of 2)

- Rule packages are folders that organize rules into logical groups
- A single package is equivalent to a stand-alone segment of business logic
- Rule packages can contain other packages nested within them
 - Nested packages can contain all types of rule artifacts
 - You can create package hierarchy by separating the names with a dot (period): `myRulePackage.subPackage1.subPackage2`



Unit title

© Copyright IBM Corporation 2017

Figure 3-29. Rule packages (2 of 2)

As you saw earlier, the rule project contains a folder that is called `rules`. Within that folder, you can organize your rule artifacts into a hierarchy of rule packages. Physically, all rules are stored together in a single repository, so the rule packages separate the rules into *logical* groups.

Packages can contain rules, decision tables, decision trees, ruleflows, and other packages. You can make each package equivalent to a stand-alone segment of business logic.

For example, if you have a set of business rules that are related to pricing, you can group them in a rule package named `pricing`.

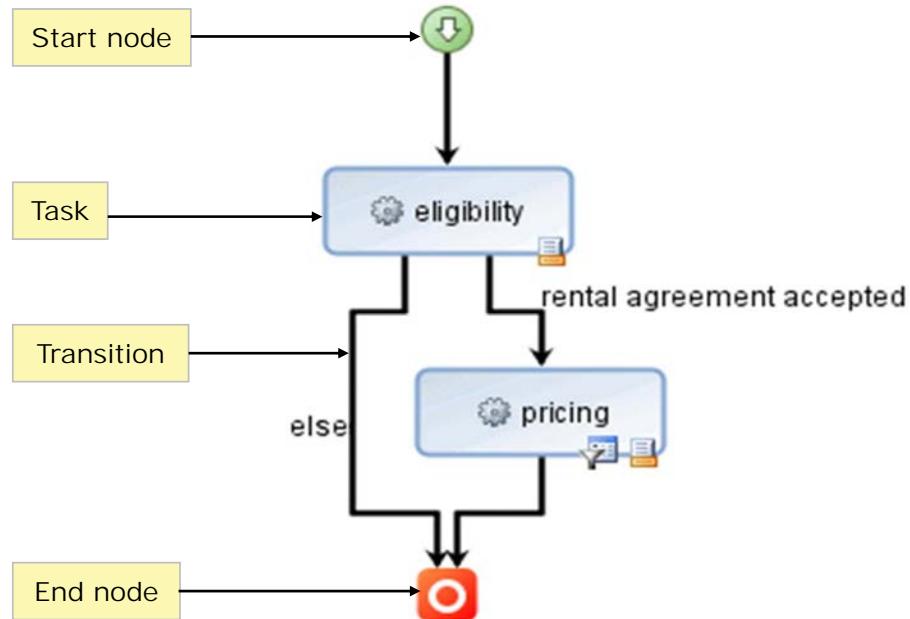
If the pricing rules can be further separated according to location or product, you can create nested packages, such as `pricing.Europe` or `pricing.myproduct`.

However, rule content should be contained within the lowest subpackage level. A package typically does not contain a mix of individual rules and subpackages.

As a suggested practice, map one rule package to one task in a ruleflow.

Orchestrating ruleset execution

- Rule execution is controlled through a ruleflow



Unit title

© Copyright IBM Corporation 2017

Figure 3-30. Orchestrating ruleset execution

Ruleflows provide a graphical outline of the business logic that is contained in a ruleset.



Reminder

A ruleset is a stand-alone, executable container that corresponds to a decision. It contains the set of rules and rule artifacts that the rule engine executes. Rules that pertain to a specific business decision are organized in a ruleflow and then extracted into the ruleset.

The application sends input to the rule engine and expects a single business decision in return. However, that business decision might break down into a series of smaller decisions. The ruleflow helps visualize the smaller underlying decisions by organizing the smaller decisions as *tasks*.

Tasks are linked together with unidirectional *transitions*. You set conditions on transitions to control the routing logic and define a specific path to follow through the ruleflow from one task to another. You define these conditions in the same way as conditions in a business rule.

Creating ruleflows is an iterative process. You work with developers in Rule Designer to outline the general flow of tasks as the starting point. However, as business users create and edit rules, the

organization of tasks might change. Developers might also need to include some additional processing to produce the expected result.

Ruleflow tasks

- Groups of rules that produce a decision
- You can set task properties to determine which rules to use and how the rules execute
- Three types of ruleflow tasks: rule task, action task, subflow task

Rule task	<ul style="list-style-type: none"> Contains an ordered list of rules or rule packages to execute When a rule task is done, the referenced rules or packages are executed in the specified order
Action task	<ul style="list-style-type: none"> Contains rule action statements to execute
Subflow task	<ul style="list-style-type: none"> References another ruleflow to execute The referenced ruleflow can be any other ruleflow within the rule project

Unit title

© Copyright IBM Corporation 2017

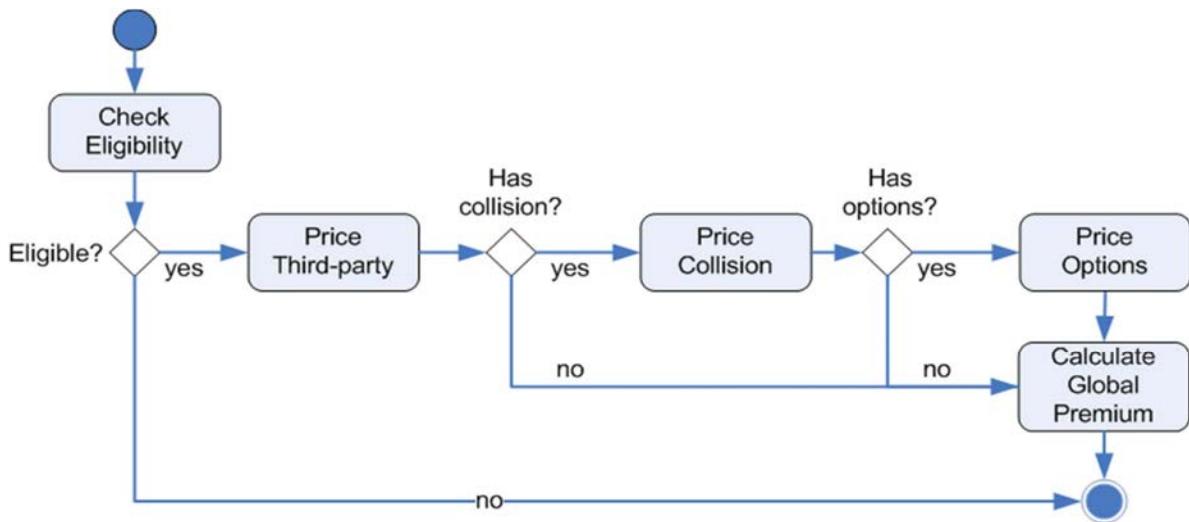
Figure 3-31. Ruleflow tasks

By grouping the rules into tasks, you can define the order in which each individual rule is evaluated. Each task produces a decision. The way that the tasks are ordered creates the series of decisions that determine the outcome of the ruleflow.

There are three types of ruleflow tasks: rule tasks, action tasks, and subflow tasks.

Example: Car insurance

- Underwriting car insurance is based on a series of decisions to determine eligibility and pricing
- In the case study, the rules are grouped into packages that correspond to these ruleflow tasks:



Unit title

© Copyright IBM Corporation 2017

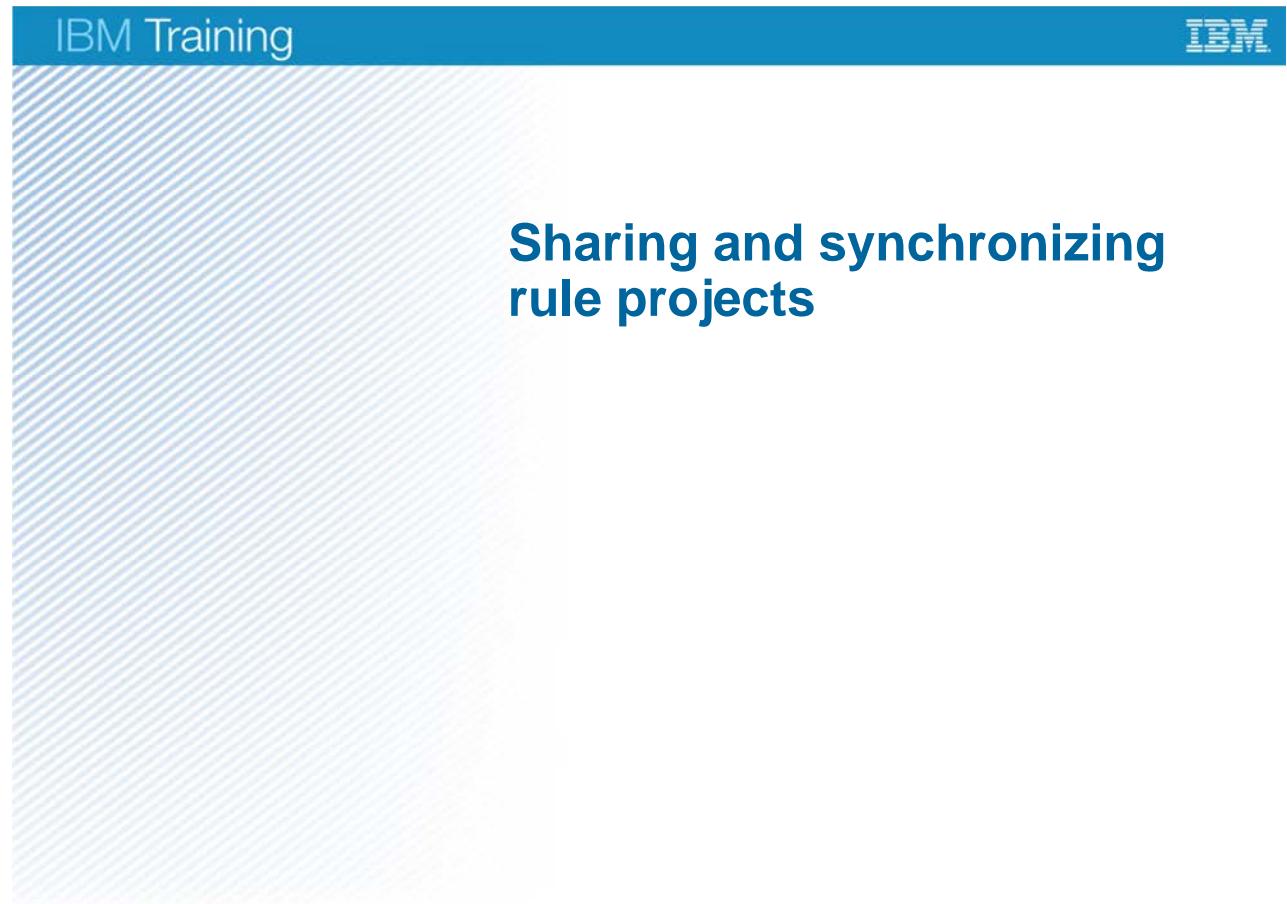
Figure 3-32. Example: Car insurance

In the car insurance case study, a series of smaller decisions must first be made regarding eligibility and pricing before an overall decision is returned about whether to insure the vehicle and at what price.

Each set of smaller decisions is grouped as a task in the ruleflow.

According to the routing logic depicted with transitions, when one of the tasks fails, ruleset execution ends. The overall decision is based on the success or failure of the intermediate decisions.

3.5. Sharing and synchronizing rule projects

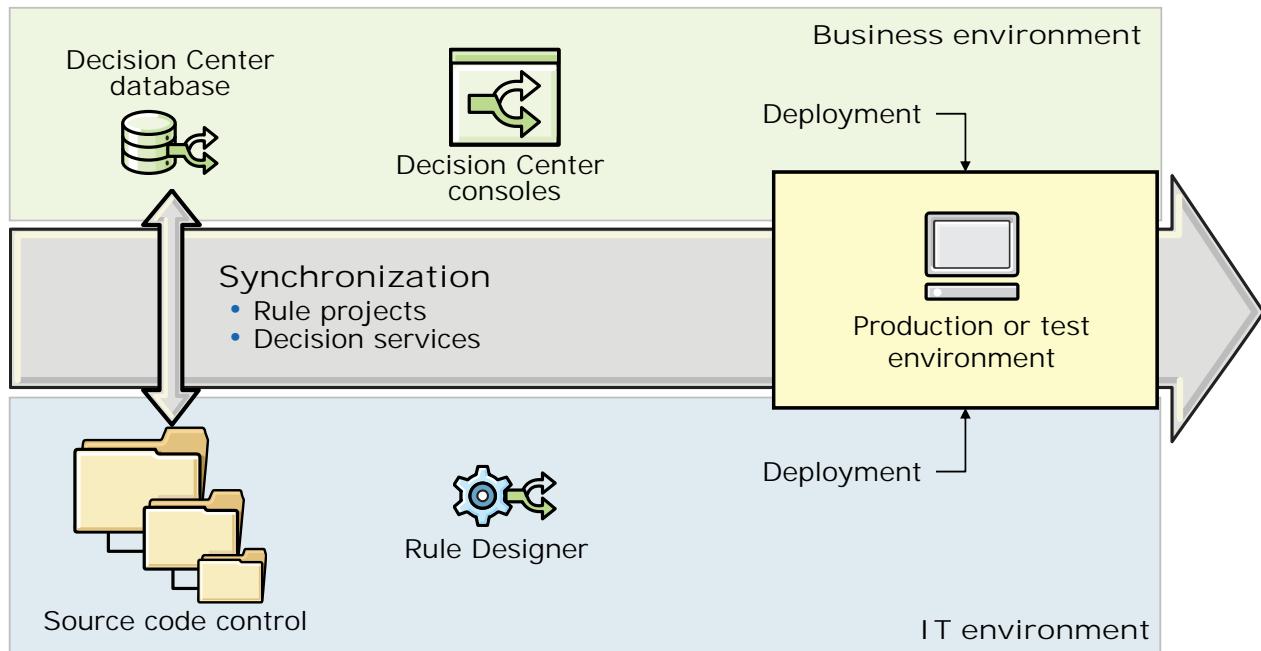


Unit title

© Copyright IBM Corporation 2017

Figure 3-33. Sharing and synchronizing rule projects

Synchronization



Unit title

© Copyright IBM Corporation 2017

Figure 3-34. Synchronization

Collaborative work between developers and business users is managed across the business and development environments through synchronization mechanisms.

Changes that are made in Decision Center are synchronized through Decision Center and stored in the Decision Center Repository.

Rule Designer uses a source code control system for storing rule artifacts. Collaborative work between developers and business users is accomplished through synchronization in Rule Designer.

After a rule project is set up, it can be published from Rule Designer to Decision Center so that business users can also start working on the rules in Decision Center.

As soon as the project is published to Decision Center, control of the rule project passes to business users and Decision Center administrators. Developers can continue to work on other aspects of the business rule application while business users work on the rules.

Rule Designer and Decision Center store rule artifacts separately. Development and business teams can continue to work on the same projects, but in separate environments.

Synchronization mechanisms allow multiple copies of a project, which can be reconciled into a single “master” version that can be stored in the Decision Center Repository.

The suggested approach to sharing rule projects between business users and developers is to consider the version in the Decision Center Repository as the master source. Any Rule Designer copy would be considered a temporary copy of the project.

Synchronizing between business and IT users

- Synchronization between Decision Center and Rule Designer is controlled from Rule Designer
 - Publish an existing rule project to Decision Center
 - Create a rule project from an existing Decision Center project
 - Synchronize the Rule Designer and Decision Center copies of the project to account for changes on either side

Unit title

© Copyright IBM Corporation 2017

Figure 3-35. *Synchronizing between business and IT users*

Synchronization is required whenever one group, whether business or IT, makes an update that the other group must incorporate into their work.

For example, when business users who are working in Decision Center identify changes or fixes that must be made to the vocabulary, their copy of the project must be synchronized with the copy in Rule Designer. Developers can then modify the BOM and publish the project back to Decision Center. Synchronization can also be a business user task.

Implementation approaches for business rule management

- Developer-centric approach:
 - Master copies managed and stored through Designer by using source code control
 - Developers implement changes that business users identify
 - Developers publish the updated implementation back to Decision Center
- Business-user-centric approach:
 - Business users make rule changes, and deploy rule changes to production
 - Alternative: Business users deploy to a staging area where rules are tested and then deployed to production by developers

Unit title

© Copyright IBM Corporation 2017

Figure 3-36. Implementation approaches for business rule management

The way that you implement business rule management with Rule Designer and Decision Center depends on who the owners of the business policy are, and whether you use the Decision Center Repository as production storage.

Unit summary

- Describe the decision service rule project structure
- Explain how ruleflows orchestrate rule execution
- Describe synchronization mechanisms between the business and development environments

Unit title

© Copyright IBM Corporation 2017

Figure 3-37. Unit summary

Review questions

1. True or False: From the development perspective, the business rule application starts in Rule Designer and is designed according to the models and requirements that you provide.
2. True or False: To prepare the rule authoring environment, developers create rule projects and decision services with all the business artifacts that are needed to start writing rules, including a BOM and perhaps some rule templates.
3. True or False: After developers start working on the decision service, all collaboration with business users can stop.



Unit title

© Copyright IBM Corporation 2017

Figure 3-38. Review questions

Write your answers here:

- 1.
- 2.
- 3.

Review answers

1. True or False: From the development perspective, the business rule application starts in Rule Designer and is designed according to the models and requirements that you provide.
The answer is True.

2. True or False: To prepare the rule authoring environment, developers create a rule project with all the business artifacts that are needed to start writing rules, including a BOM and perhaps some rule templates.
The answer is True.

3. True or False: After developers start working on the decision service, all collaboration with business users can stop.
The answer is False. Collaboration between development and business teams on a decision service is ongoing, and is facilitated through the Rule Designer synchronization mechanism.



Unit title

© Copyright IBM Corporation 2017

Figure 3-39. Review answers

Exercise: Setting up a decision service

Unit title

© Copyright IBM Corporation 2017

Figure 3-40. Exercise: Setting up a decision service

Exercise objectives

- Set up a decision service for rule authoring
- Create a ruleflow
- Share and synchronize rule projects between the business and development environments



Unit title

© Copyright IBM Corporation 2017

Figure 3-41. Exercise objectives

Unit 4. Working with the BOM

Estimated time

01:00

Overview

This unit continues the focus on introducing you to business rule and decision service concepts and tasks. You learn how to work with the BOM and the vocabulary to ensure that the BOM implementation matches requirements and supports your rules.

How you will check your progress

- Checkpoint
- Exercise

Unit objectives

- Explain the relationship between the BOM and the rule vocabulary
- Verbalize the BOM
- Create categories to simplify rule authoring
- Define domains

Unit title

© Copyright IBM Corporation 2017

Figure 4-1. Unit objectives

Topics

- Creating rule vocabulary
- Categories
- Refactoring
- Using domains

Unit title

© Copyright IBM Corporation 2017

Figure 4-2. Topics

4.1. Creating rule vocabulary

Creating rule vocabulary

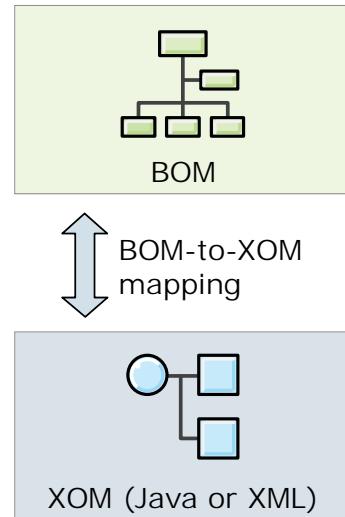
Unit title

© Copyright IBM Corporation 2017

Figure 4-3. Creating rule vocabulary

Working collaboratively on the BOM

- Top-down approach
 - Reflects business perspective for defining the BOM
 - Create the BOM without worrying about how the business elements are implemented
- Bottom-up approach
 - Reflects development perspective for defining the BOM
 - Write the code first to create the XOM in Java or XML
 - Then, generate the BOM from the XOM
- Both options require extensive interaction between business analysts and developers
 - Expect several iterations to stabilize the BOM
- Rule Designer supports the bottom-up approach



Unit title

© Copyright IBM Corporation 2017

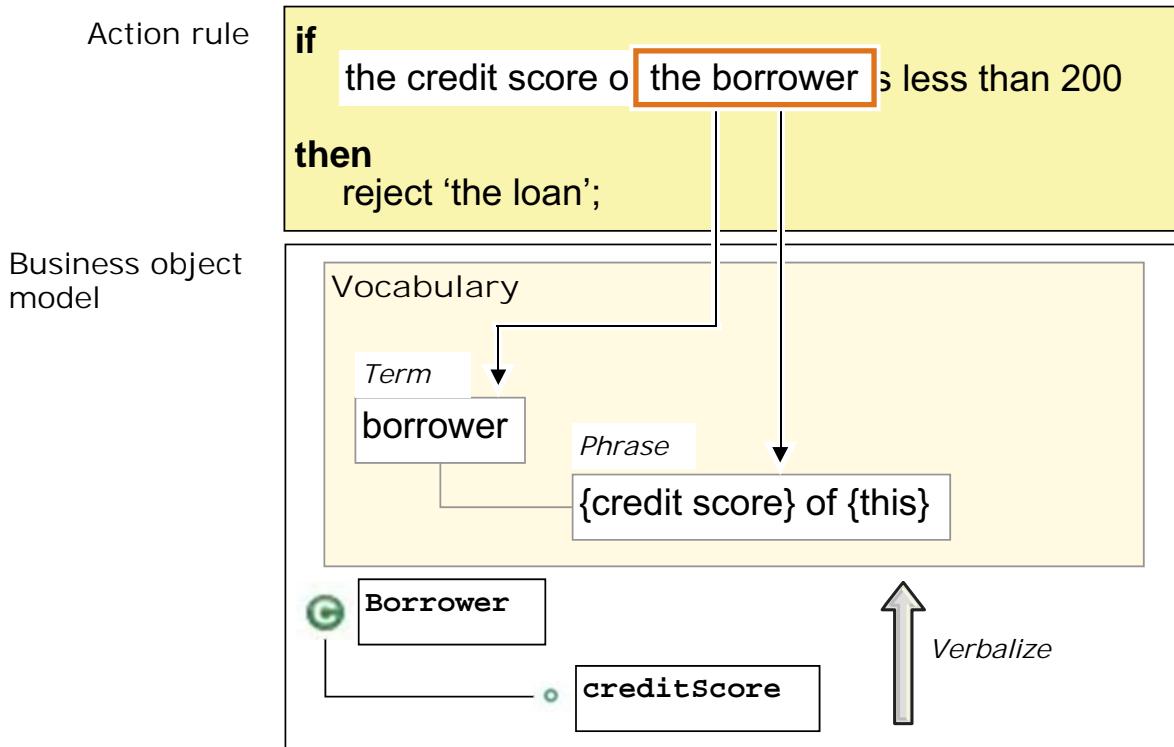
Figure 4-4. Working collaboratively on the BOM

While it is possible to create a BOM directly in Rule Designer (top-down), this approach is not encouraged because the BOM must still be implemented in a programming language.

Rule Designer is designed to support development of the XOM first (bottom-up). Developers can implement the XOM according to requirements such as a class diagram from the business analysts, and then generate the BOM from the XOM.



Rule vocabulary



Unit title

© Copyright IBM Corporation 2017

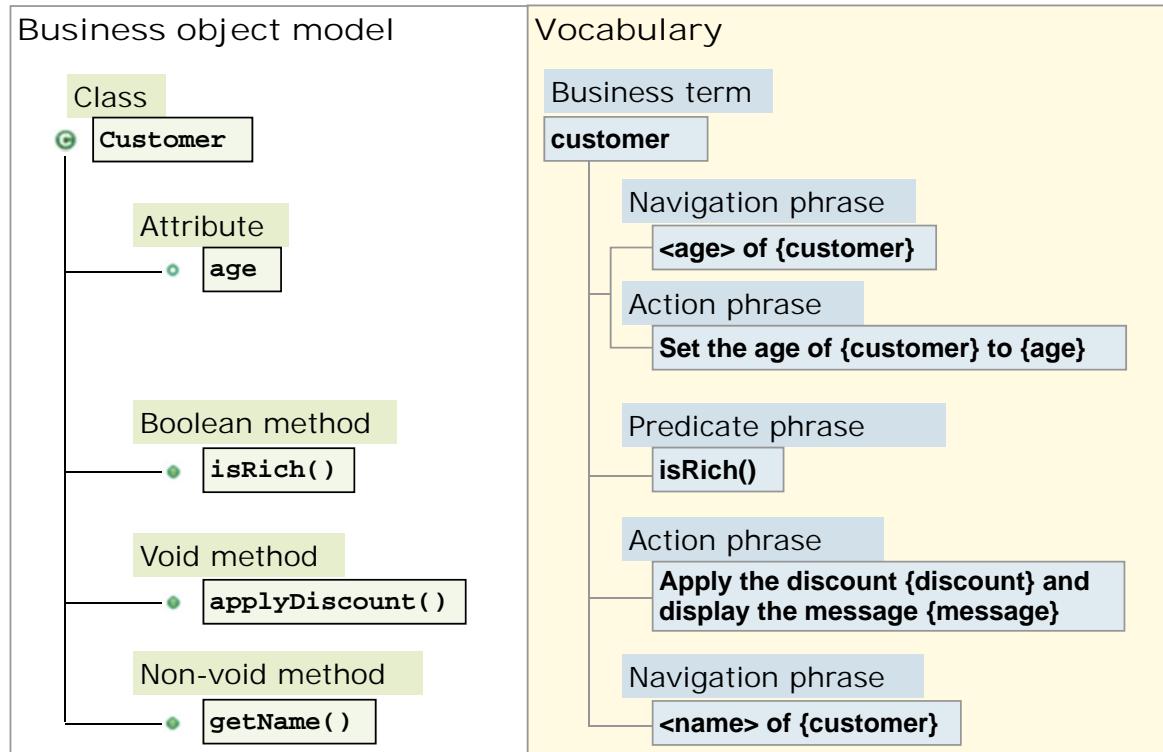
Figure 4-5. Rule vocabulary

The vocabulary in the rules comes directly from the BOM.

The BOM defines the business objects that you can write about in rules.



Creating vocabulary



Unit title

© Copyright IBM Corporation 2017

Figure 4-6. Creating vocabulary

To create the vocabulary, you *verbalize* the BOM.

To verbalize the BOM, you attach natural language terms and phrases to the elements of the BOM. The set of terms and phrases defines the rule vocabulary.

Verbalization

- Verbalizations are included in the vocabulary lists of rule editors
 - BOM must be defined and verbalized before you can start writing rules in the rule editors
 - Only verbalized business elements are accessible in the rule editors, and can be used in your business rules
- Rule Designer can provide a default verbalization of BOM elements
 - Review default vocabulary manually for clarity or customization

Unit title

© Copyright IBM Corporation 2017

Figure 4-7. Verbalization

Before you write rules in the rule editors, the BOM must be defined and verbalized.

Rule Designer can provide a default verbalization for BOM elements, but you should review the default vocabulary for clarity.

Class

- Class verbalizations are called **terms**
- Terms can be edited to correct plural form and articles
- Defaults typically require attention for irregular English-language nouns and non-English-language articles
- Examples of default verbalization:
 - LoanReport => loan report, loan reports
 - branch => branch, branchs

Unit title

© Copyright IBM Corporation 2017

Figure 4-8. Class

Class verbalizations are called *terms*, and terms can be edited as needed so that they include the correct plural form and articles.

The default verbalization of irregular English-language nouns and articles in languages other than English often must be edited to reflect the correct forms.

Members

- Member verbalizations consist of subjects and phrases
- Defaults typically require attention on methods
- Examples of default verbalization:
 - `getIncome()`, `setIncome()` \Leftrightarrow income \Rightarrow the income of ...
 - `getYearlyIncome()`, `setYearlyIncome()` \Leftrightarrow yearlyIncome \Rightarrow the yearly income of ...
 - `{this}.applyDiscount({0})` \Rightarrow apply a discount of {0} to {this}

Unit title

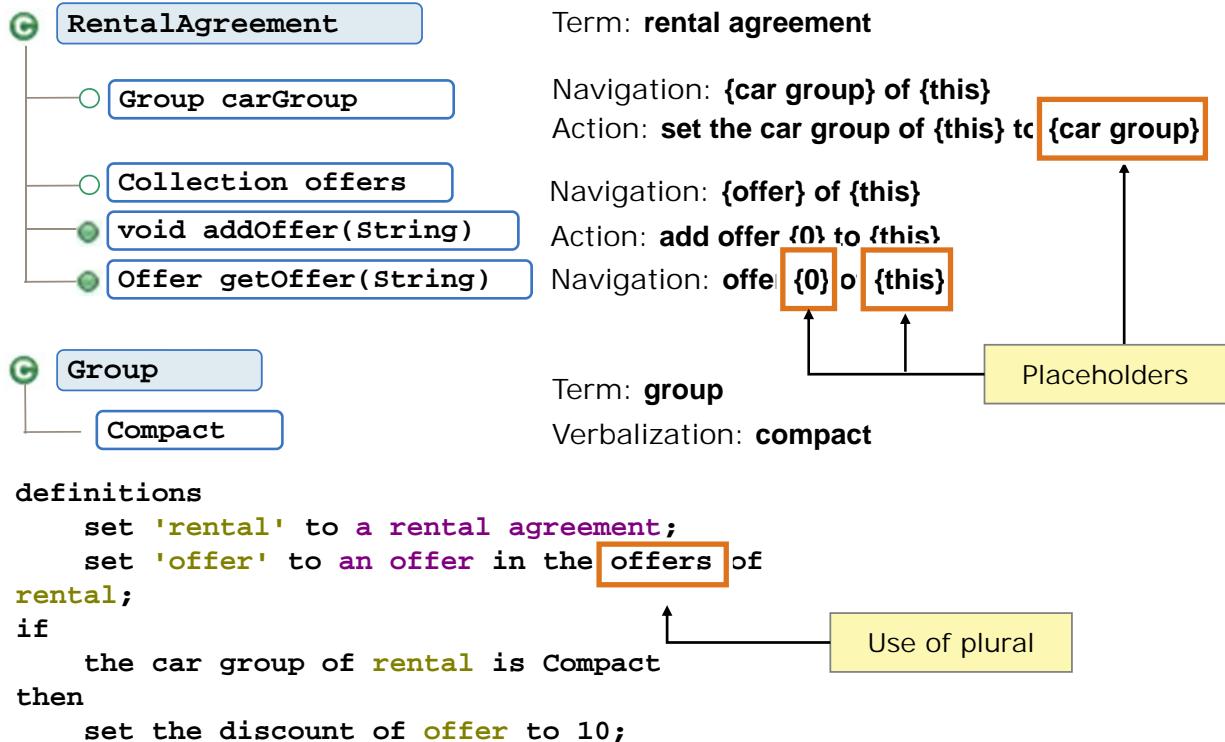
© Copyright IBM Corporation 2017

Figure 4-9. Members

Member verbalizations consist of subjects and phrases. The default member verbalizations often require a review so that the methods express the correct behavior.



Verbalizing BOM members



Unit title

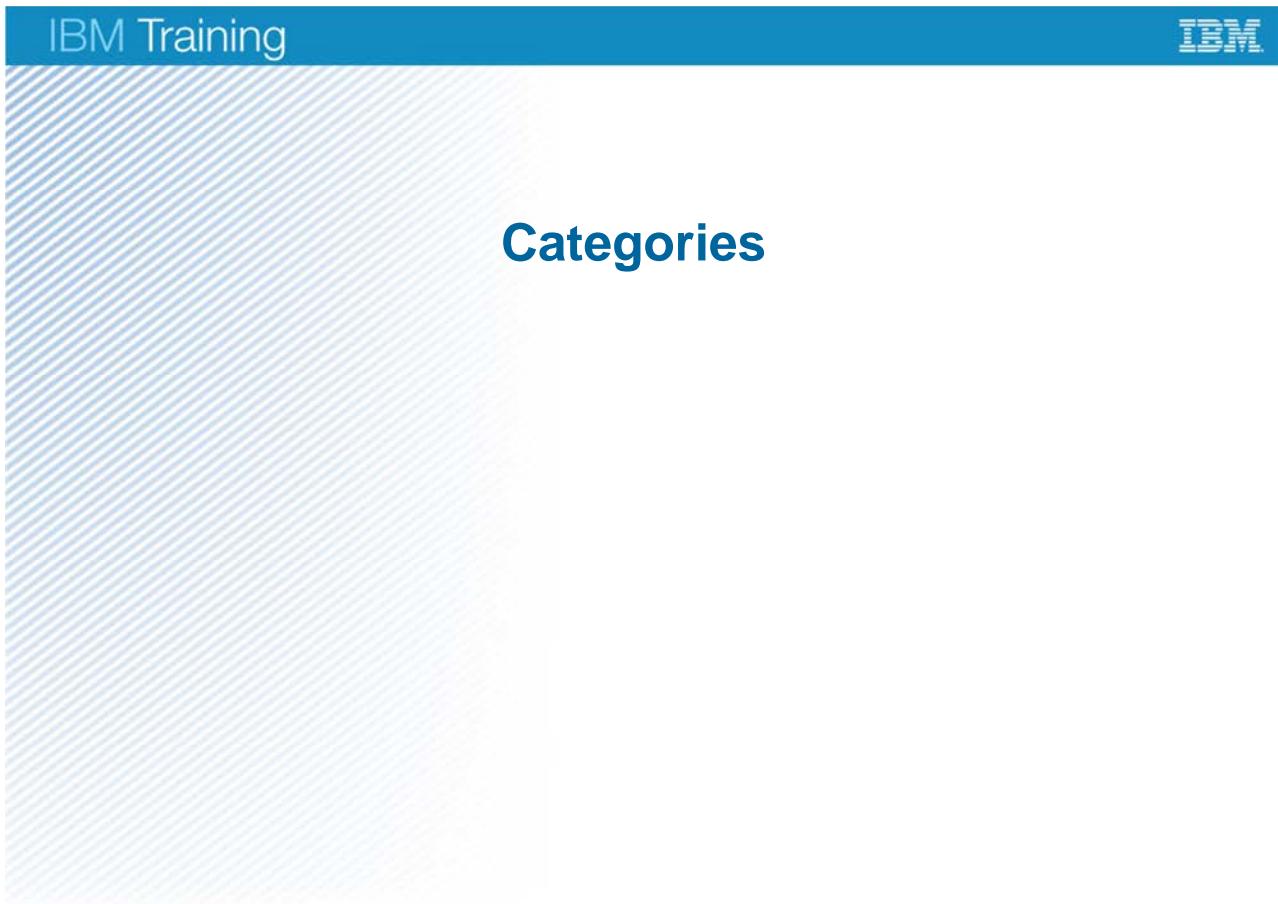
© Copyright IBM Corporation 2017

Figure 4-10. Verbalizing BOM members

The terms and phrases that you assign to the BOM elements show up as vocabulary in the rule editor vocabulary lists.

You validate vocabulary to ensure that rules are expressed in grammatically correct sentences.

4.2. Categories



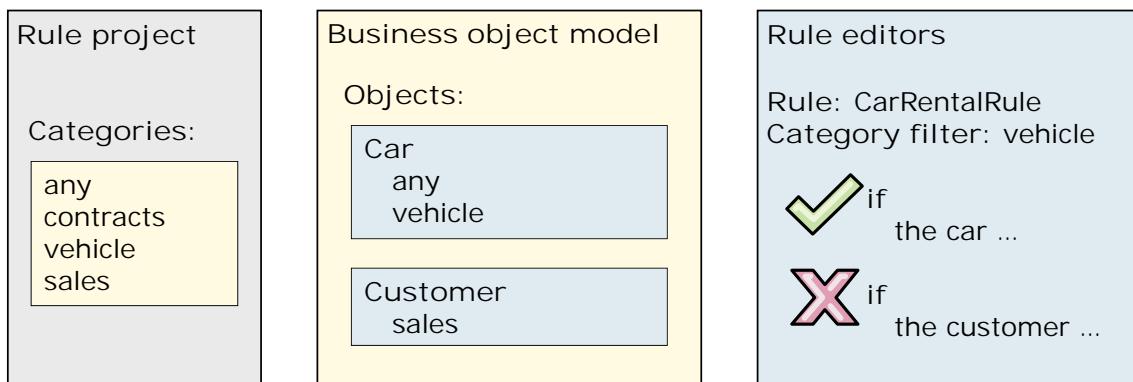
Unit title

© Copyright IBM Corporation 2017

Figure 4-11. Categories

Categories

- Tags on the BOM elements can be used as a filter in the rules
 - If a category is “hidden” for a specific rule, business elements that are tagged with that category are invisible in the rule editor
 - If the category filter is in use, only BOM elements with categories that match the rule category show up in the vocabulary list when that rule is edited
 - Simplifies rule authoring by removing vocabulary that is relevant to the rule from the list of choices in the rule editors
- Example:



Unit title

© Copyright IBM Corporation 2017

Figure 4-12. Categories

Categories can be helpful when you have a large vocabulary. Categories simplify the authoring task by acting as filters. They reduce the number of elements available as vocabulary in the rule editors.

A category is a tag or identifier that can be applied to business classes and members and filtered in business rules. You set categories to specify whether a business element can be used in a specific rule. By default, the predefined category “any” is set on all business elements, meaning that all business elements can be used in all business rules that are linked to that BOM.

For example, as shown here, the `Car` class is assigned the category “vehicle” in the BOM. The rule project also has a business rule, `CarRentalRule`. `CarRentalRule` uses the “vehicle” category filter. From that business rule, you can see all the classes that are assigned the “vehicle” category, which in this case is the `Car` class.

If you then define a category that is named “sales” and assign it to a `Customer` class, the `Customer` class is not visible in the rule editor vocabulary list for the `CarRentalRule` action rule. If you try to use the `Customer` class in the business rule anyway, a warning is raised.

Category semantics

- Predefined category “any” is set on all business elements
- When no category is defined (including “any”), the element cannot be used in the rules

If the category of BOM element is:	And the rule category filter is:	Then, in rule editor, the element is:
any	any (or another category)	visible
any (or another category)	any	visible
category1 category2	category1	visible
category1 category2	category3 category4	hidden
Empty (no category)	any (or another category)	hidden
any (or another category)	Empty (no category)	hidden

Unit title

© Copyright IBM Corporation 2017

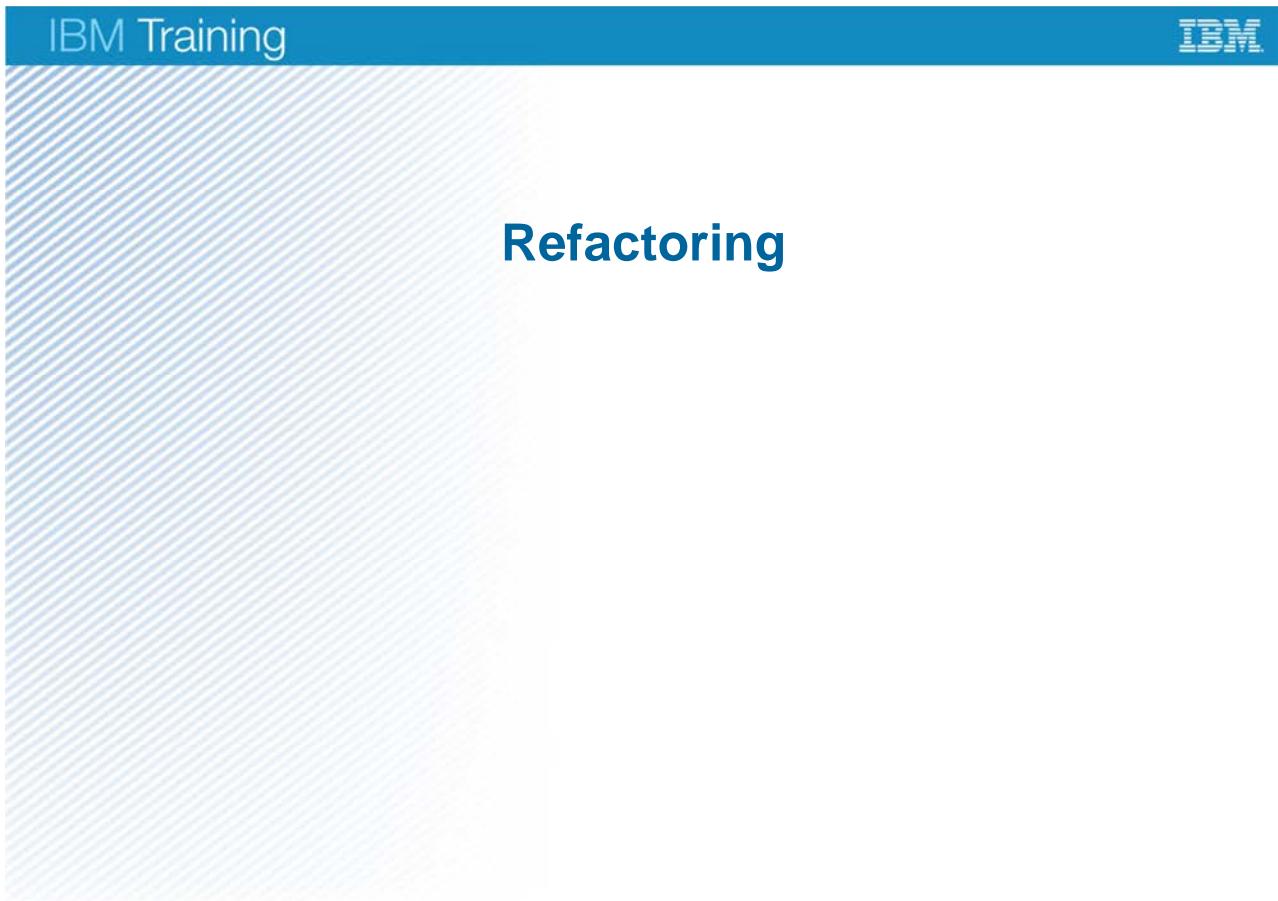
Figure 4-13. Category semantics

By default, the predefined category “any” is set on all business elements, meaning that all business elements can be used in all business rules that are linked to that BOM.

When you use categories, you must set them in three places:

- In the rule project
- In the BOM element
- In the rule

4.3. Refactoring



Unit title

© Copyright IBM Corporation 2017

Figure 4-14. Refactoring

Refactoring

- A mechanism in Rule Designer ensures that changes to the BOM or the vocabulary are reflected in the rules that use those BOM elements or vocabulary
- Triggered after modifying ruleset parameters and variables that are used in rule artifacts

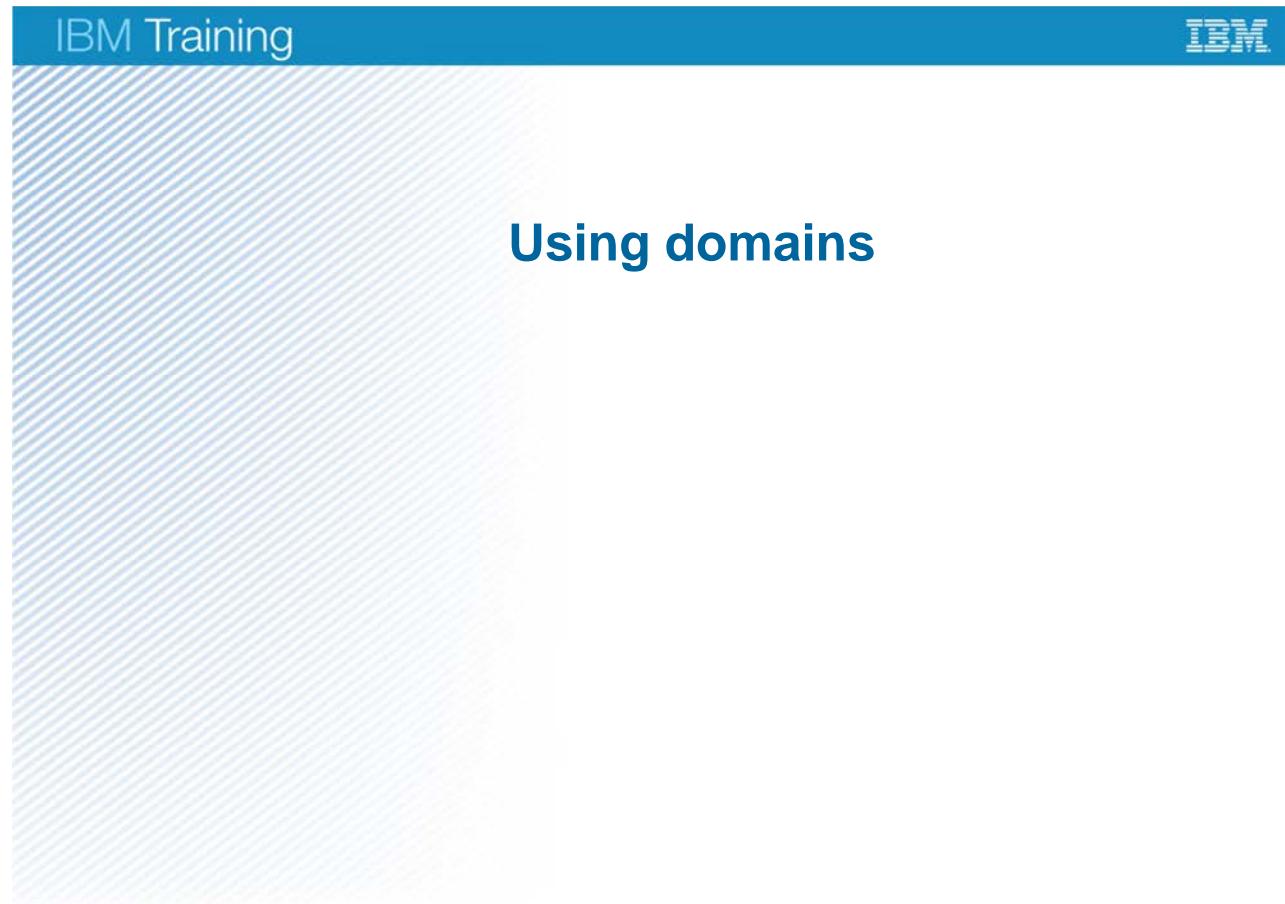
Unit title

© Copyright IBM Corporation 2017

Figure 4-15. Refactoring

Rule Designer has a mechanism for making sure that rules take changes in the BOM or vocabulary into account. After changing the verbalization of a business element that is used in a rule, you can choose to refactor the rules that use the business element so that the rules reflect your changes.

4.4. Using domains



Unit title

© Copyright IBM Corporation 2017

Figure 4-16. Using domains

Domains

- A domain is a way of defining the set of possible values for a type
- Domains provide a set of valid values that you can choose from in the rule editors
 - When you attempt to write rules about BOM elements that use a domain, the rule editor prompts you with the domain values
- You can set a domain on these types of BOM elements:
 - Classes
 - Attribute types
 - Method return types and arguments

Unit title

© Copyright IBM Corporation 2017

Figure 4-17. Domains

Domains help you author and analyze rules. When you write rules that include BOM elements with domains that are attached to them, the rule editors propose valid values. Errors and warnings help validate the values that you specify. When you analyze rules, domain types in the BOM are used to check the consistency of action rule semantics.

Main types of domains

- Static domains:
 - Literals
 - Static references
 - Bounded
 - Collection
 - Other domain types
- Dynamic domains, with values stored outside the BOM
- Enumerated domains:
 - Literals
 - Static references
 - Dynamic domain types
- Complex domains:
 - Bounded
 - Collection
 - Other domain types

Unit title

© Copyright IBM Corporation 2017

Figure 4-18. Main types of domains

The main types of domains are static, dynamic, enumerated, and complex.

Not all kinds of BOM domain are enforced in BAL rules or other business rules. Business rules use only enumerated domains (literal, static reference, or dynamic). A semantic check is done to check that the business rule does not use a value outside its domain, and the Intellirule editor suggests values from the enumerated domains.

However, the semantic check that is done at the business rule level does not detect complex patterns of incorrect usage that involve operators other than **is** or **is not**. Other kinds of domain, such as bounded domains, are not enforced at the business rule level.

Dynamic domains can use an Excel file or be set through the execution of Java code. The next slides provide more details about using Excel.



Cloud

ODM on Cloud supports dynamic domains that are based on Microsoft Excel. Other types of domains are not supported.

Dynamic domains

- You can dynamically populate a domain from a data source, and then synchronize the data source and the domain
- A dynamic domain is an enumerated domain with values from an Excel file or through Java code execution
- You define a dynamic domain in the BOM editor

Unit title

© Copyright IBM Corporation 2017

Figure 4-19. Dynamic domains

The general mechanism that is used to set up a dynamic domain is based on a plug-in that reads the values from the appropriate source. A predefined plug-in is provided to support Microsoft Excel.

Using Excel for dynamic domains

- To define a dynamic domain with an Excel spreadsheet, you include these columns:
 - Value** column: Contains the values of the domain
 - BOM to XOM** column: Contains the BOM-to-XOM mapping of the value
 - Label** column: Contains the verbalization of the value that is used in the rule

	A	B	C
1	Value	BOM to XOM	Label
2	Administrative	return "A01";	Administrative
3	Compensatory	return "B34";	Compensatory
4	Educational	return "B45";	Educational
5	FamilyPersonal	return "C56";	Family/Personal
6	JuryDuty	return "V78";	Jury Duty
7	Military	return "B89";	Military
8	Overtime	return "F23";	Overtime
9	Pregnancy	return "V12";	Pregnancy

- Optional columns can be added for documentation or custom properties
- You add the spreadsheet to the resources folder of the project that defines the BOM

Unit title

© Copyright IBM Corporation 2017

Figure 4-20. Using Excel for dynamic domains

To map the properties correctly, the Microsoft Excel file must have the following structure:

- One row for each value of the dynamic domain
- Three mandatory columns in each row:
 - Value
 - BOM to XOM
 - Label
- No merged cells

Optional columns can be added for each row to include documentation for the value or to support custom properties.

Some properties must be defined on the BOM class to retrieve the information from the Excel file. You use the BOM Editor to map the domain properties to the columns of the spreadsheet.

To make the rule project with the dynamic domain available to rule authors, you publish the project to Decision Center.

Unit summary

- Explain the relationship between the BOM and the rule vocabulary
- Verbalize the BOM
- Create categories to simplify rule authoring
- Define domains

Unit title

© Copyright IBM Corporation 2017

Figure 4-21. Unit summary

Review questions

1. True or False: The vocabulary that is used in rules comes from the BOM.
2. True or False: Verbalizing the BOM involves associating a natural-language term or phrase with BOM members.
3. Refactoring is the Rule Designer mechanism for:
 - A. Attaching natural language terms and phrases to BOM members
 - B. Running simulations on new rules
 - C. Ensuring that changes that are made to the BOM are reflected in the rules that use those BOM elements
 - D. All of the above



Unit title

© Copyright IBM Corporation 2017

Figure 4-22. Review questions

Write your answers here:

- 1.
- 2.
- 3.

Review answers

1. True or False: The vocabulary that is used in rules comes from the BOM.
The answer is True.

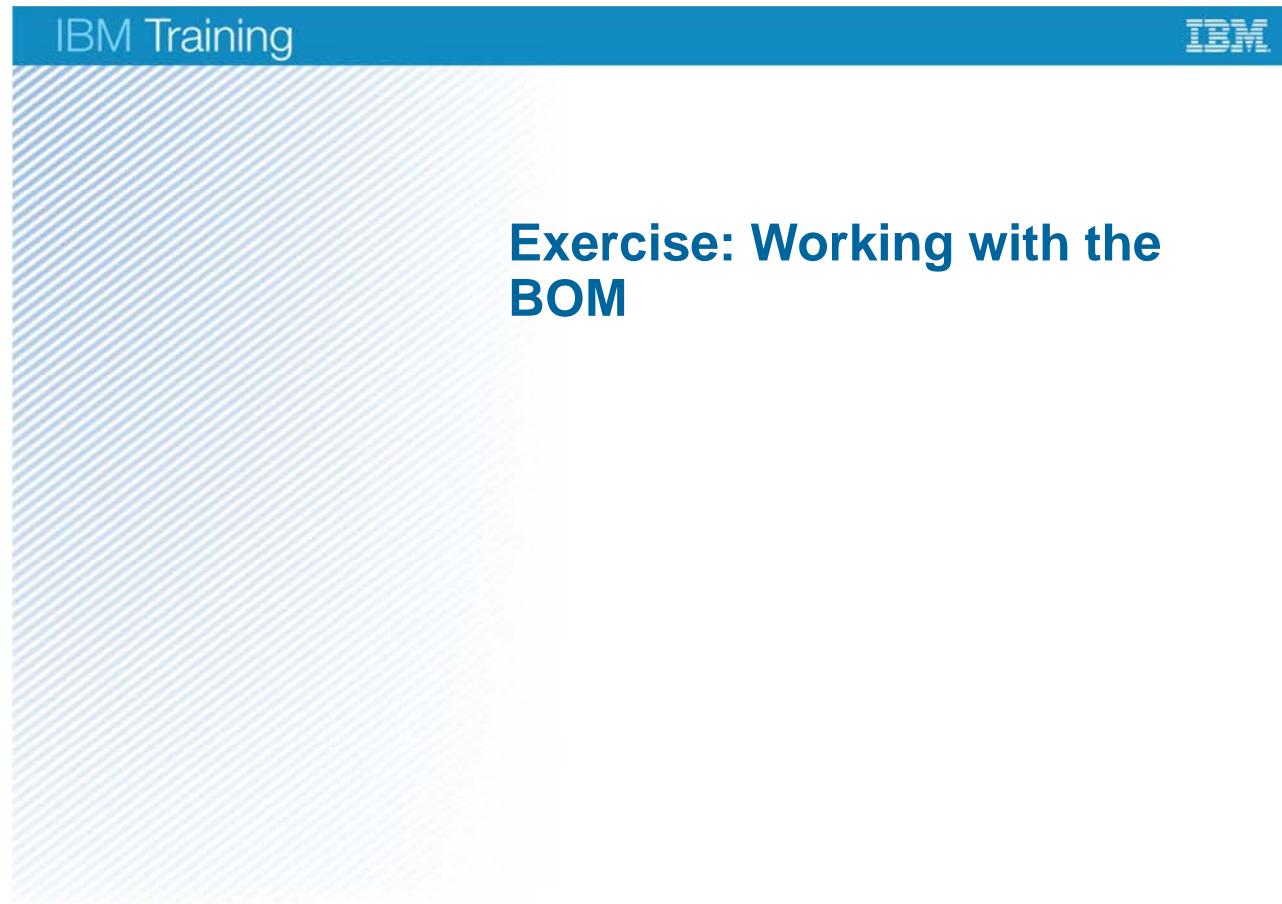
2. True or False: Verbalizing the BOM involves associating a natural-language term or phrase with BOM members.
The answer is True.

3. Refactoring is the Rule Designer mechanism for:
 - A. Attaching natural language terms and phrases to BOM members
 - B. Running simulations on new rules
 - C. Ensuring that changes that are made to the BOM are reflected in the rules that use those BOM elements
 - D. All of the above**The answer is C.**

Unit title

© Copyright IBM Corporation 2017

Figure 4-23. Review answers



Unit title

© Copyright IBM Corporation 2017

Figure 4-24. Exercise: Working with the BOM

Exercise objectives

- Review and correct verbalization of the BOM
- Refactor changes that you make to the vocabulary
- Define domains for customized rule vocabulary



Unit title

© Copyright IBM Corporation 2017

Figure 4-25. Exercise objectives

Unit 5. Introducing Decision Center

Estimated time

01:00

Overview

This unit introduces you to Decision Center and to the Decision Center Business console and Enterprise console.

How you will check your progress

- Checkpoint
- Exercise

Unit objectives

- Describe the Business console and Enterprise console rule authoring environments
- Use the Business console rule editors to create and change rules

Unit title

© Copyright IBM Corporation 2017

Figure 5-1. Unit objectives

Topics

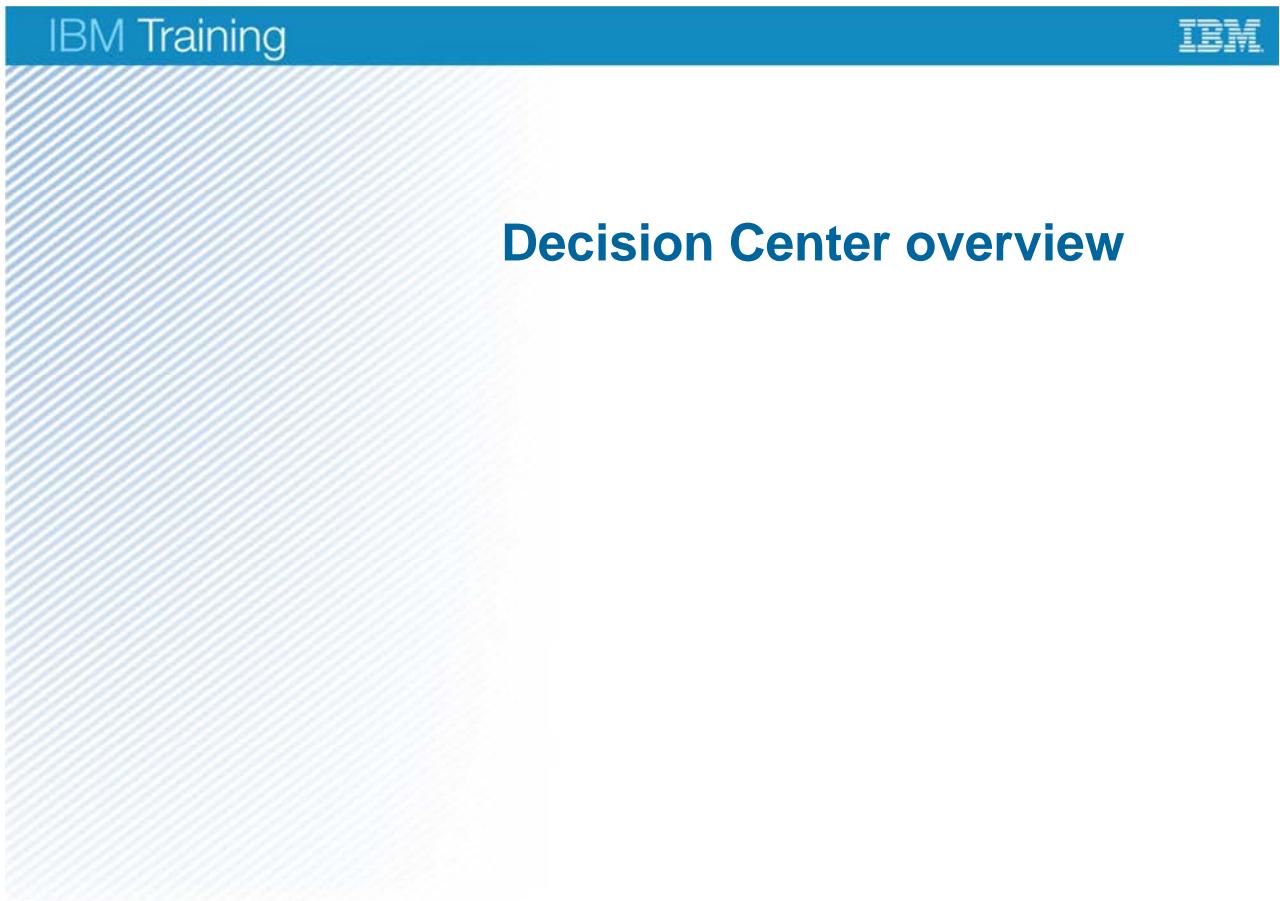
- Decision Center overview
- Tour of the Decision Center Enterprise console
- Tour of the Decision Center Business console

Unit title

© Copyright IBM Corporation 2017

Figure 5-2. Topics

5.1. Decision Center overview



Unit title

© Copyright IBM Corporation 2017

Figure 5-3. Decision Center overview

Decision Center overview

- Primary work environment for business users to access rule projects and work collaboratively on business rules
- Two consoles: Business console and Enterprise console
- Business console:
 - Emphasis on social collaboration
 - Author and manage rules
 - Run tests and simulations
 - Decision governance framework
- Enterprise console:
 - Contains all the elements necessary to author and manage business rules
 - Provides administrative features
- Decision Center is connected to a database called Decision Center Repository

Unit title

© Copyright IBM Corporation 2017

Figure 5-4. Decision Center overview

Decision Center is the primary working environment for business users to author and manage business rules.

Decision Center is a web application that is connected to a server and a database. The database is called the Decision Center Repository.

Decision Center Repository

- Decision Center Repository is the “source of truth”
 - Provides integrated storage of all rule and event artifacts
 - Stores the most comprehensive information about rules over time, including a complete history
 - Copies of rules that are stored outside the repository are considered temporary copies
- Supports rule management features, including:
 - Version control
 - Multiuser access and permissions
 - Baseline management
 - Branching for multiple release management

Unit title

© Copyright IBM Corporation 2017

Figure 5-5. Decision Center Repository

Accessing the Decision Center consoles

- Default user name and password combinations for these roles:

Role	User name	Password
Business user	rtsUser1	rtsUser1
Configuration manager	rtsConfig	rtsConfig
Administrator	rtsAdmin	rtsAdmin

- Business console also has the following users:

Role	User name	Password
Manager	Paul	Paul
Rule author	Bea	Bea

Unit title

© Copyright IBM Corporation 2017

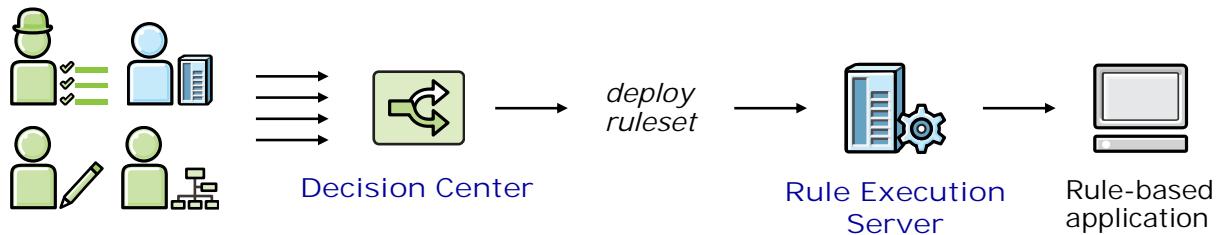
Figure 5-6. Accessing the Decision Center consoles

Users are associated with one or more user groups. The user name and password determine your access rights to different projects, and can be used to specify the permissions that you need to work with rules.

Administrators or developers can create custom groups and define access rights and permissions for them. You learn more about these features in Unit 13, "Working with Decision Center administrative tools" and [Exercise 16, "Working with management features in Decision Center"](#).

Decision Center relationship to rule-based applications

- Changes to rules are not immediately reflected back to the applications that use them
 - After creating or changing rules in Decision Center, they must be deployed
 - Deployment involves sending sets of rules to a server that handles the rule execution and interaction with the rule-based applications



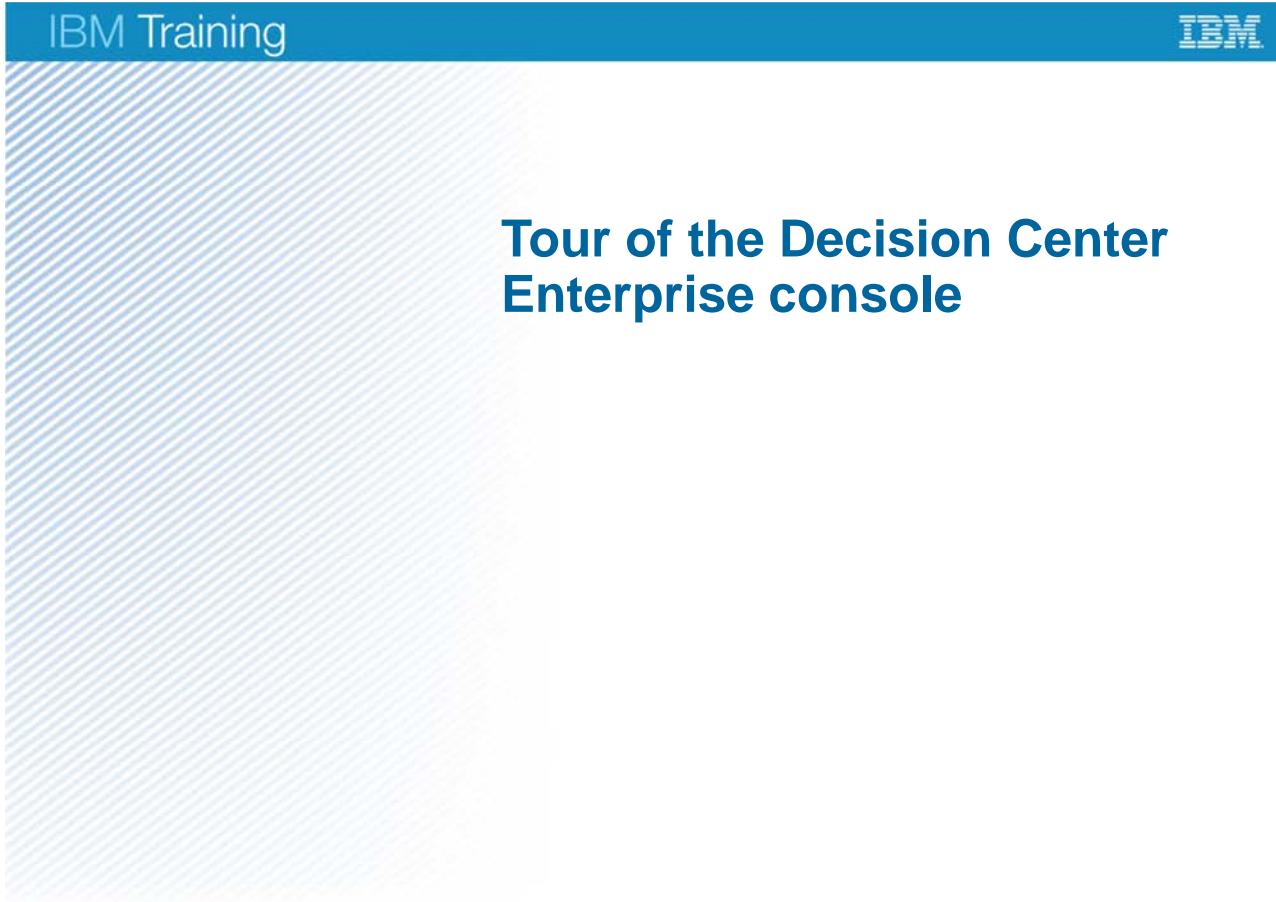
- Rules projects can be deployed from the Enterprise console
 - Only rules in a decision service can be deployed from the Business console

Unit title

© Copyright IBM Corporation 2017

Figure 5-7. Decision Center relationship to rule-based applications

5.2. Tour of the Decision Center Enterprise console



Unit title

© Copyright IBM Corporation 2017

Figure 5-8. Tour of the Decision Center Enterprise console

This topic provides a quick overview of the rule authoring tools in the Decision Center Enterprise console. However, the focus in this unit and in the exercise is on the Decision Center Business console.

Enterprise console: Rule authoring and administration

- Decision Center Enterprise console is a web interface for authoring and managing rules
- In addition to rule editors, the Decision Center Enterprise console includes features for rule management:
 - Importing and exporting of projects and decision services to and from the Decision Center database
 - Erasing projects and decision services from the database

Unit title

© Copyright IBM Corporation 2017

Figure 5-9. Enterprise console: Rule authoring and administration

In addition to rule editors, the Decision Center Enterprise console includes features for rule management.

In Unit 13, "Working with Decision Center administrative tools" and [Exercise 16, "Working with management features in Decision Center"](#), you learn about some of the administrative tools in the Enterprise console.



Enterprise console environment and Home tab

- Default view is the **Home** tab

Welcome to the Decision Center Home Page

Work on a rule project

Project in use:

Branch in use:

Current action:

Work on a decision service

Decision service in use:

Branch in use:

Current action:

Work on a governed decision service

Decision service in use:

Release in use:

Current change activity:

Current action:

Unit title

© Copyright IBM Corporation 2017

Figure 5-10. Enterprise console environment and Home tab

Features are organized into a series of tabs:

- Home:** Select the rule project and baseline
- Explore:** Browse through the current rule project
- Compose:** Create or edit rules and other project elements
- Analyze:** Generate reports on your rule projects and their project elements
- Project:** Perform advanced project-related tasks

Most authoring tasks are done on the **Explore** and **Compose** tabs.

You work with one decision service at a time. Select the one that you want from the decision service selection menu on the **Home** tab. If the decision service has multiple branches, you can choose which branch to work with from this page.

The tabs that are available to users depend on user permissions. This slide shows the tabs that are available with the administrative user, **rtsAdmin**, which includes the **Configure** tab. The **Configure** tab has administrative features that you explore in [Exercise 16, "Working with management features in Decision Center"](#).



Exploring rules

- On the **Explore** tab, you can see how a project is organized
- Rules artifacts are grouped under folders
 - Select a folder to display its contents

Folders map to Rule Designer rule packages

Actions	Name
<input type="checkbox"/>	minimum credit score
<input type="checkbox"/>	minimum income
<input type="checkbox"/>	repayment and score

Rules in the Eligibility folder

Unit title

© Copyright IBM Corporation 2017

Figure 5-11. Exploring rules

Folders contain rules and other elements.

- Default folders:
 - Business Rules
 - Ruleflows
 - Templates
- The “Business Rules” folder lists all the business rules in the project.
- Rules are grouped into folders that map to Rule Designer rule packages.
 - Folders (rule packages) provide organization and structure for your rules.
 - If the rule structure in your organization is not obvious, discuss it with your developers.
 - The folder structure and specific rules that are visible to you depend on the access rights that are associated with your user name.
 - Either the business analyst, administrator, or developer sets up the folder structure to reflect what users see when they sign in.

When you select a folder, the folder is highlighted, and the artifacts in that folder are listed in the table. In this example, the folder contains rules and a decision table. Icons indicate the type of item.



Creating a rule: Using the Compose tab wizard

- To create a rule artifact, start from the **Compose** tab
 - Select the rule artifact type and click **OK** to open the creation wizard



Unit title

© Copyright IBM Corporation 2017

Figure 5-12. Creating a rule: Using the Compose tab wizard

To create an element in the Enterprise console, you can start from the **Compose** tab, and select the type of rule artifact that you want to create.

The default is to create an action rule, which is a general business rule.

When you use the **Compose** tab to create rules, a multistep wizard opens, starting with **Step 1: Properties**.

Basic properties include Name, Folder, and others.

- Name:** Each rule must have a name. The name must be unique within the parent folder.
- Folder:** The folder property specifies where the rule is stored in Decision Center, which is important for you to be able to find it later. In Rule Designer, folders are called *packages*.
 - The slash (/) means the root folder.

After you complete **Step 1: Properties** and click **Next**, the wizard moves to **Step 2: Content**.

You enter the rule statement in **Step 2: Content** of the Compose wizard.

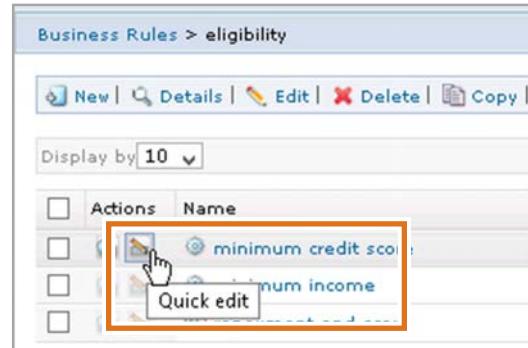
This page of the Compose wizard uses the Guided Editor for rule authoring.

The rule structure is explicit in the editor, so you can easily see where to enter the different parts of the rule:

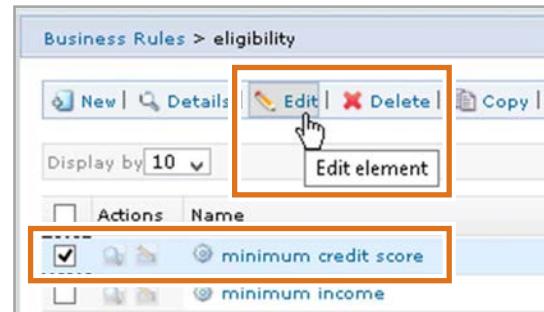
- Definitions
- Conditions (**if**)
- Actions (**then** and **else**)

Editing a rule: Quick edit and the Edit menu

- Quick edit opens an editor on the **Explore** tab
 - Click the **Quick edit** icon in the **Actions** column
 - Editor opens below list of rule artifacts in the current folder



- Full rule editor also available in **Edit** menu
 - Select the rule and click **Edit** on the toolbar
 - Opens the **Compose** tab wizard so that you can edit rule properties, rule content, and other information



Unit title

© Copyright IBM Corporation 2017

Figure 5-13. Editing a rule: Quick edit and the Edit menu

The Enterprise console provides two ways to edit a rule: Quick edit and the **Edit** menu.

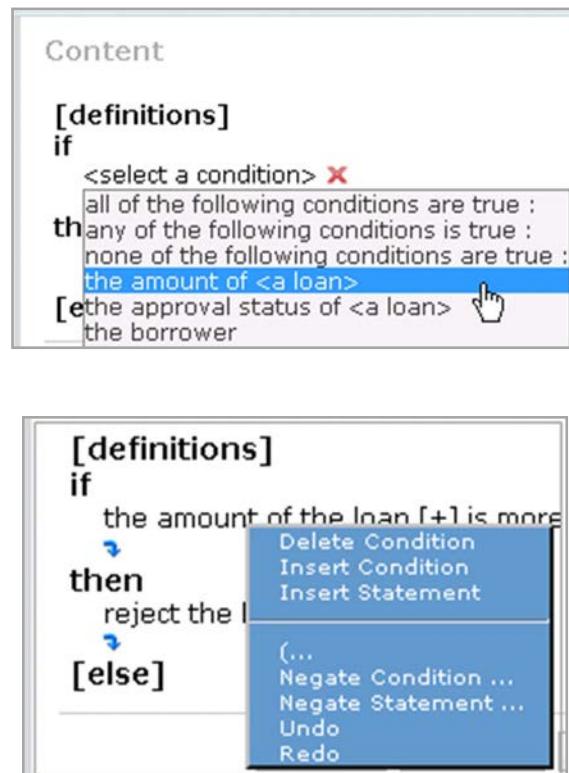
The quick edit feature opens the rule editor in the **Explore** tab. To use quick edit, click the **Quick edit** icon in the **Actions** column. The rule editor opens below the list of rule artifacts in the folder.

For more comprehensive editing, use the **Edit** menu. To open the **Edit** menu, select the rule and click **Edit**. The Edit menu opens the Compose wizard. Rule properties (such as the Name and Status) are edited in **Step 1: Properties**. Rule content, such as condition statements or action statements, is edited in **Step 2: Content**.



Working in the rule editor

- Rule statements are composed from building blocks that are shown in brackets: < > and []
- Example: <select a condition>
- Click a building block to open the vocabulary list
- Vocabulary lists are populated from the BOM
- Can undo or delete rule statements
- Right-click rule statements in rule editor
- Click **Delete** or **Undo** from the menu



Unit title

© Copyright IBM Corporation 2017

Figure 5-14. Working in the rule editor

The Guided Editor is the default editor in the Enterprise console. It is a point-and-click rule editor that makes it easy to enter and edit rules. The structure of the rule is built in to the editor interface.

Building blocks, which are composed of text within angle brackets (< >) and brackets ([]) help you create rule statements by selecting items from lists.

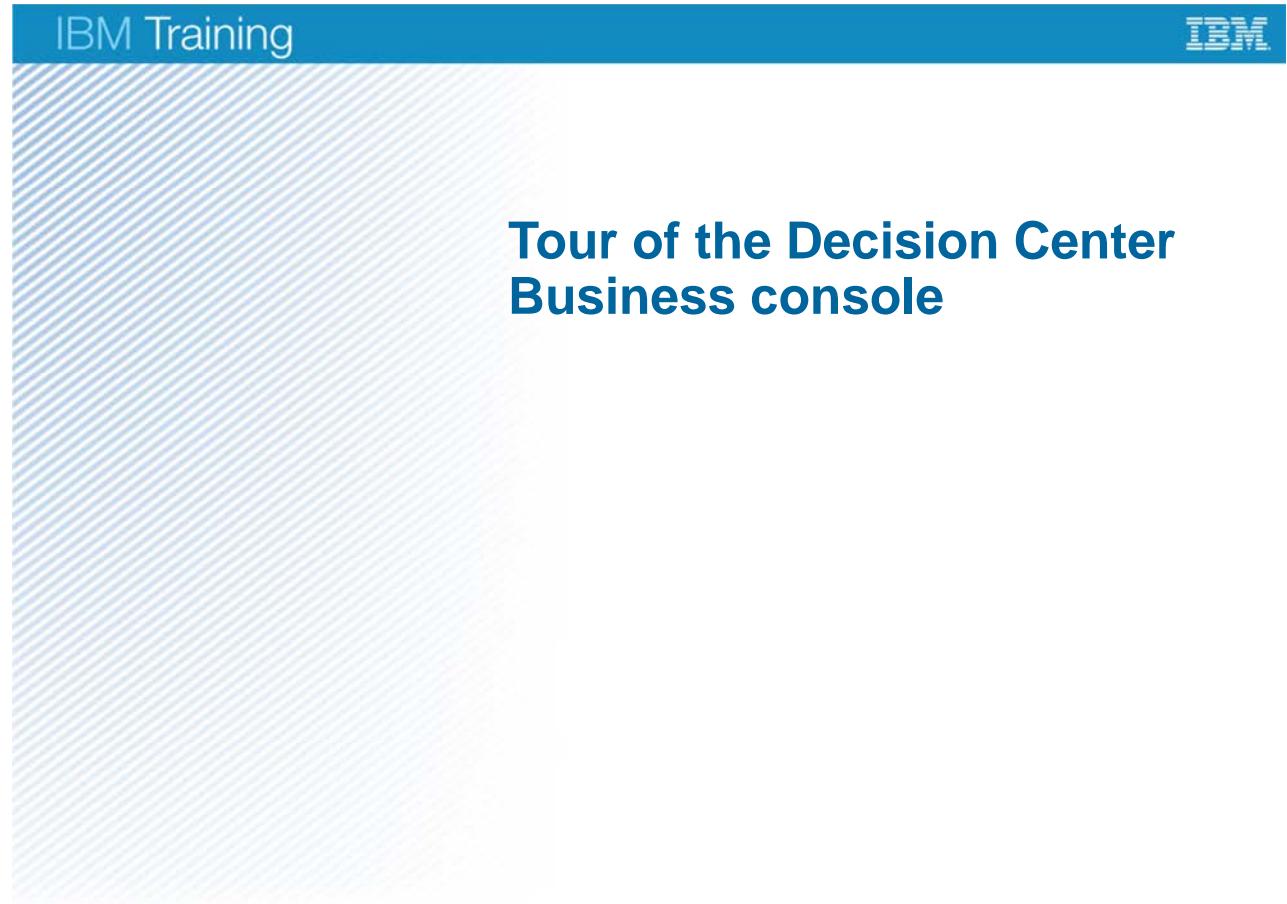
Vocabulary selection menus contain items that you can select to build the rule statement. The lists are populated with vocabulary specific to your business. The vocabulary is defined in the business object model (BOM).

When you write rules, do not be concerned about making errors. As you focus on building rule authoring skills, you also become familiar with the built-in error checking and review features.

For example, to correct possible mistakes, use the **Undo** or **Delete** commands by right-clicking the vocabulary and by using the menu.

When you change or create a rule, it does not immediately change the behavior of the end application. Rule management and governance require a process that includes validation, approval, and testing before deployment. Governance processes can prevent errors in rules from being deployed to production.

5.3. Tour of the Decision Center Business console



Unit title

© Copyright IBM Corporation 2017

Figure 5-15. Tour of the Decision Center Business console

This topic provides a quick overview of the Decision Center Business console.



Decision Center Business console

- Decision Center Business console provides a socially oriented and collaborative rule authoring experience
 - Author, edit, organize, and search for rules in a collaborative environment
- Main features include:
 - Working with rule projects and decision services
 - Managing versions through snapshots
 - Social tools, such as activity streams, comments, and user profiles
 - Governance framework

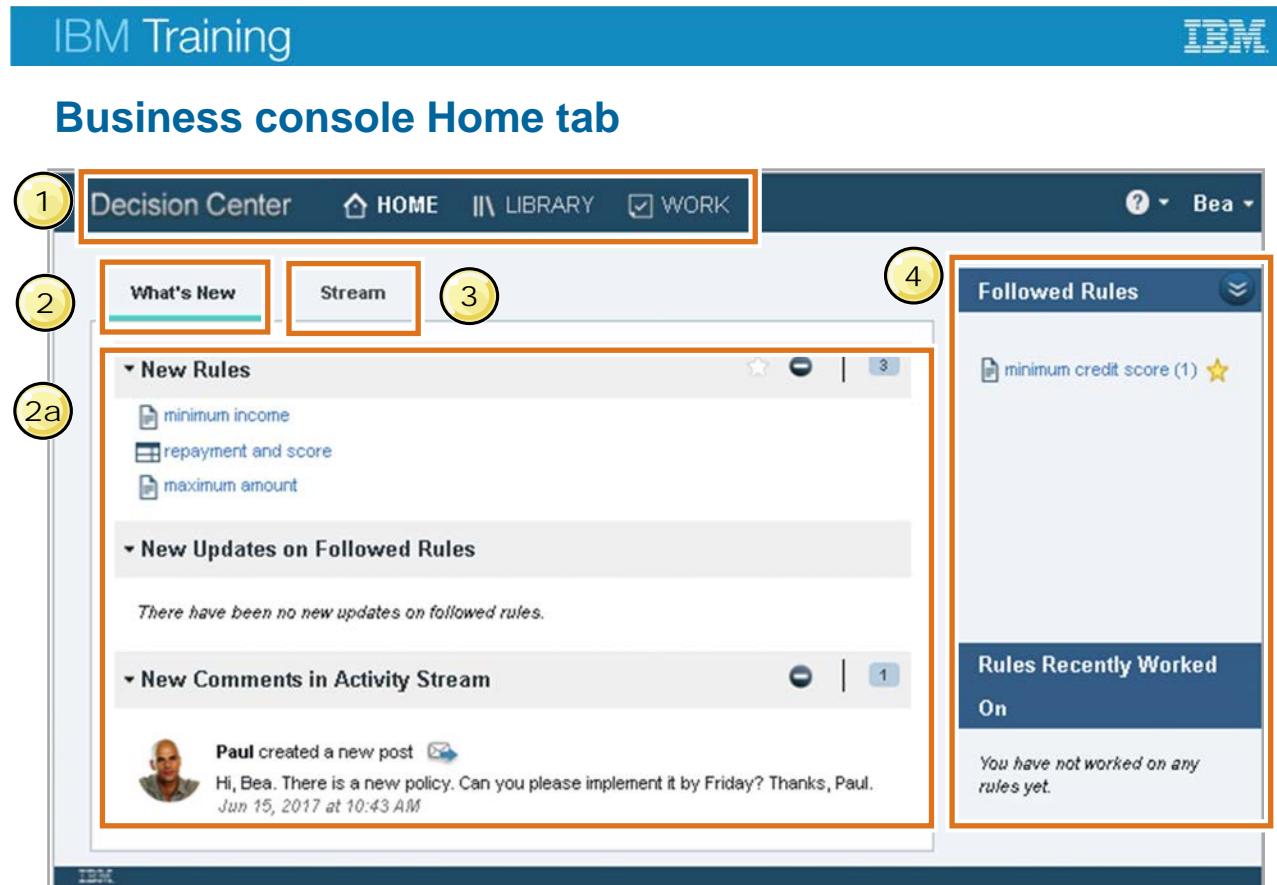
Unit title

© Copyright IBM Corporation 2017

Figure 5-16. Decision Center Business console

The Decision Center Business console is a workspace that business users can use to collaboratively author, edit, organize, and search for business rules.

The decision governance framework in the Business console is covered in Unit 14, "Introducing decision governance" and [Exercise 18, "Working with the decision governance framework"](#).



Unit title

© Copyright IBM Corporation 2017

Figure 5-17. Business console Home tab

The **Home** tab is the default view after you log in to the Business console.

It consists of the following areas:

1. The top banner:

The top banner is always visible.

In addition to displaying your user name, you use the top banner to:

- Switch between main pages
- Log out
- Access the user profile, where you can set preferences such as language and what activities to follow by default
- Access the online help

2. The **What's New** tab:

The **What's New** tab lists activities that happened since you last logged in to the Business console.

- 2a. **What's New** is divided into three sections:

- New Rules
- New Updates on Followed Rules
- New Comments in Activity Stream

3. The **Stream** tab:

The **Stream** tab lists activities that happened on the projects, branches, and rule artifacts that you are following, such as:

- When items are created, edited, or deleted
- Posts and comments
- When snapshots are taken

You can also post comments directly into the stream.

4. The main **Home** tab also includes the following sections, which are on the right side of the browser window:

- Followed Rules
- Rules Recently Worked on

The image shows a screenshot of the IBM Training interface. At the top, there's a blue header bar with the text "IBM Training" on the left and the "IBM" logo on the right. Below the header, the word "Library and Work" is displayed in a large, bold, dark blue font. Underneath this, there's a navigation bar with three items: "Decision Center", "HOME" (with a house icon), "LIBRARY" (with a list icon), and "WORK" (with a checkmark icon). The "LIBRARY" item is highlighted with an orange border.

- The **Library** tab displays all decision services that are available to you

This image is similar to the one above, showing the IBM Training interface. It features the same blue header, "IBM Training" logo, and "Library and Work" title. The navigation bar at the bottom includes "Decision Center", "HOME", "LIBRARY", and "WORK". In this version, the "WORK" item is highlighted with an orange border.

- The **Work** tab lists decision governance framework change or validation activities that you can work on.

Unit title

© Copyright IBM Corporation 2017

Figure 5-18. Library and Work

The **Library** tab shows the decision services that are available to you.

If you work on many decision services, it might be difficult to find the decision service that you need in the Library. You can use the following tools to help you find a project or decision service quickly:

- Sort by date: Click **Date** to arrange projects and decision services by date, with the oldest first.
- Sort by name: Click **Name** to arrange projects and decision services in alphabetical order.
- Use the filter: Enter part of the project or decision service name in the **Filter** field to display the rule projects and decision services that contain your search term.

The **Work** tab lists all the ongoing decision governance framework activities that you can participate in. These activities have shortcuts that you can use to rename, cancel, or change the status of activities.



Working with decision services

- Ungoverned decision services have branches
- Governed decision services have releases
- Click a branch or release to work on a decision service

The screenshot shows the IBM Decision Center interface. On the left, the 'Decision Center' panel displays a navigation bar with 'HOME', 'LIBRARY' (selected), and 'WORK'. Below it, 'All Decision Services' lists 'Miniloan Service'. Under 'Miniloan Service', there are tabs for 'Releases' (selected) and 'Branches'. A large button with a plus sign and the text 'main' is visible. On the right, the 'Loan Validation Service' panel shows a list of 'Releases' and 'Branches'. The 'Releases' tab is selected, showing two entries: 'Initial Release' (Complete, Due Date: Jan 23, 2017, Owner: Paul) and 'Spring Release' (In Progress, Based on: Initial Release, Due Date: Jan 23, 2018, Owner: Paul).

Unit title

© Copyright IBM Corporation 2017

Figure 5-19. Working with decision services

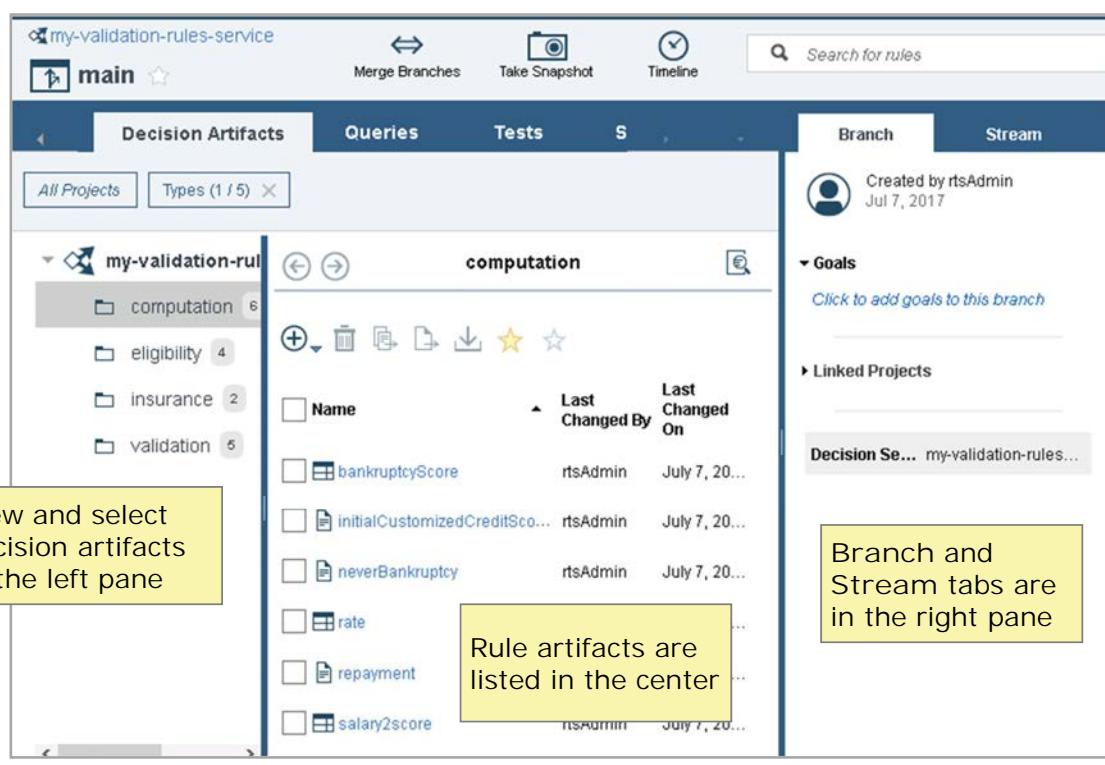
Decision services can be ungoverned or governed. An ungoverned decision service has *branches*, which you can use to manage work on a decision service. A governed decision service uses the decision governance framework to manage changes to a decision service by using *releases* and other tools. Decision governance is covered in detail later in this course.

To work on a branch or a release, click the branch or release to open it in the Business console.

IBM Training



Business console Decision Artifacts tab



View and select decision artifacts in the left pane

Rule artifacts are listed in the center

Branch and Stream tabs are in the right pane

Unit title

© Copyright IBM Corporation 2017

Figure 5-20. Business console Decision Artifacts tab

On the **Decision Artifacts** tab, you can select the decision artifacts that you want to work with. The default view lists only rule artifacts, but you can select the type of decision artifacts that you want listed by clicking the **Types** menu.

The left explorer pane lists the decision artifacts that are in the decision service. You can use this pane to select the various decision artifacts folders (such as rule folders or ruleflow folders).

When you select a folder, its contents are listed in the center pane. You can then create, delete, copy, move, and follow decision artifacts.

The right pane has the **Branch** and **Stream** tabs. The **Branch** tab lists some details about the branch that you are working in, and the **Stream** tab lists the comment and events stream for this branch.



Viewing a rule

- Clicking a rule artifact opens it in the center pane
 - Shows rule artifact content
 - Use the toolbar to access various tools, such as rule view, the rule editor, and rule details

Toolbar with Open Rule View, Edit this rule, and Show Details

definitions

```
set 'minAge' to 0 ;
set 'maxAge' to 150 ;
```

if

it is not true that the age of **'the borrower'** is between **minAge** and **maxAge**

then

in **'the loan report'**, reject the data with the message **"The borrower's age is not valid."**;

Unit title

© Copyright IBM Corporation 2017

Figure 5-21. Viewing a rule

When you click a rule artifact from the center pane list, it opens the rule in the center pane. You can view the content of the rule artifact, and the toolbar lets you open various tools for working on rules:

- Click **Open Rule View** to see a timeline of the rule artifact or open the rule editor
- Click **Edit** to open the rule editor
- Click **Show Details** to see some of the rule artifact details, such as who created the rule, the time and date it was created, and the folder that it is in



Creating rules and decision tables

- In the left explorer pane, make sure that you are in the correct folder.
- In the center pane, click **create an artifact** (the plus [+] sign)
 - Select **New Rule** to create a business rule
 - Select **New Decision Table** to create a decision table



Unit title

© Copyright IBM Corporation 2017

Figure 5-22. Creating rules and decision tables

When you create a rule or decision table, the “Create new Rule” or “Create new Decision Table” window opens.

- Enter a name for the rule or decision table.
- Click **Create**.



Editing an existing rule or decision table

- Two ways to open the rule or decision table editor:
 - Hover the mouse pointer over the rule artifact name in the center pane and click the **Edit this rule** or **Edit this decision table** (pencil) icon



- Click the rule artifact to open it in the center pane, and then clicking the **Edit this rule** or **Edit this decision table** (pencil) icon



- When you are finished with your edits, click **Save**
- In the Create New Version window:
 - Enter a comment to document your changes and click **Create New Version**

Unit title

© Copyright IBM Corporation 2017

Figure 5-23. Editing an existing rule or decision table

If you need to edit an existing rule or decision table, you can open the editors from various locations in the Business console interface.

For example, you can hover the mouse pointer over the rule artifact name when it is listed in the center pane. The hovering action opens a toolbar next to the artifact name, and you can click the **Edit this rule** or **Edit this decision table** (pencil) icon from the toolbar. Alternatively, you can open the rule artifact by clicking the name. In the rule name toolbar, you can click the **Edit this rule** or **Edit this decision table** (pencil) icon to open the editor.



Using the rule editor

- Click the placeholders to select building blocks and vocabulary

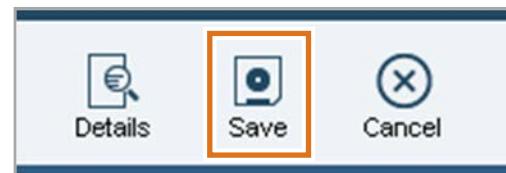
```

definitions
set 'minAge' to 0;
set 'maxAge' to 150;

if
  it is not true that the age of 'the borrower' is between minAge and maxAge
then
  in <a report> . reject the data with the message <a string>
    'the loan report'
  
```

The screenshot shows a code editor window with a toolbar at the top. Below the toolbar, the word 'definitions' is highlighted in blue. The main area contains a rule definition. The 'if' block is present, followed by a condition involving 'minAge' and 'maxAge'. A 'then' block follows, which contains an 'in' clause pointing to a 'report'. The 'reject the data with the message' part is expanded, showing a completion menu with the option 'the loan report' selected.

- When you are finished editing, click **Save**
- In the Create New Version window:
 - Enter a comment that documents your changes
 - Click **Create New Version**



Unit title

© Copyright IBM Corporation 2017

Figure 5-24. Using the rule editor

You can write or edit your rule by:

- Typing directly in the editor.
- Copying text from another editor or application, and pasting it into the editor.
- Clicking placeholders and selecting predefined terms and phrases from a completion menu.



Note

The terms and phrases in the completion menu that are available for you to use are defined in the business vocabulary.

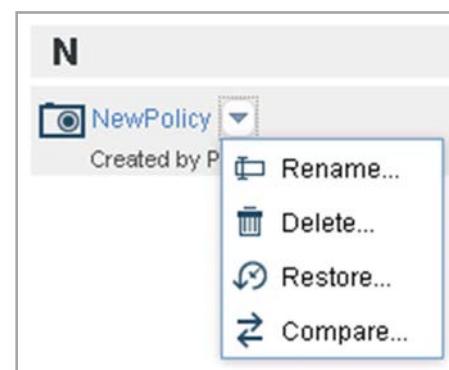


Working with snapshots

- Snapshots capture the state of a project or branch at a specific time
- To take a snapshot, click **Take Snapshot** in the main project view



- To view a snapshot, go to the **Snapshots** tab in the main project view, and click the name of the snapshot from the list
- To work with a snapshot, hover the mouse pointer near the snapshot name, and click the down arrow to open the **Action** menu



Unit title

© Copyright IBM Corporation 2017

Figure 5-25. Working with snapshots

Snapshots are similar to baselines in the Enterprise console. They capture the state of a project or branch at a specific moment in time.

Each branch of a project can contain many snapshots. When you view a snapshot, you cannot edit any of the project elements because you must be in the current state to edit.

Creating a snapshot

In the main project view, click **Take Snapshot**.

In the “Take a Snapshot” window:

- Enter a name for the snapshot.
- Enter a description for the snapshot.
- Click **Create**.

When you click the **Snapshots** tab, snapshots are listed alphabetically by name. You can use the **Filter** field to narrow down the list of snapshots that are shown.

Clicking the name of a snapshot from the list opens a detailed view of the snapshot. You can see the different rules that are contained in the project state that is captured in the snapshot, and you can also restore a snapshot from this view.

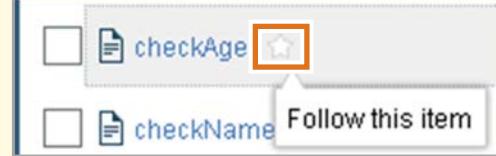
You can do the following tasks with snapshots:

- Rename a snapshot.
- Delete a snapshot.
- Restore a snapshot: Restore a project to the state that is capture in the snapshot. This option is available only to users with administrator privileges.
- Compare snapshots: Select two snapshots to compare differences between project states.

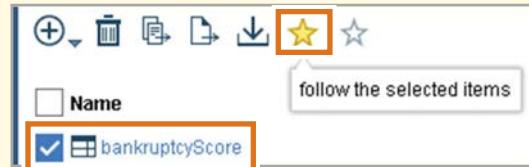
Social features: Following

- Follow releases, activities, and rule artifacts to see notifications about events that happen to the followed item

Hover the mouse pointer near the name of the release, activity, or rule artifact, and click the Follow this item (star) icon.



For rule artifacts, you can also select the rule artifact from the list and in the toolbar, click the follow the selected items (yellow star) icon. The gold star icon is activated for the selected item.



A gold star means that you are subscribed to an item.

To unsubscribe, click the gold star.



[Unit title](#)

© Copyright IBM Corporation 2017

Figure 5-26. Social features: Following

You can subscribe to follow releases, activities, or individual business rules.

- Following events subscribes you to events that occur to the item that you are following, such as changes to rules, status changes, approvals, cancellations, and rejections.
- Subscribed events appear in activity streams.

You can receive notifications about events for the following items:

- Releases: See updates about major changes to the release, such as status changes, approval, cancellation, rejection, and main activity changes
- Activities: See updates about main activity changes, such as changes to status and rule changes
- Individual rule artifacts: Receive updates on the rule artifact across all decision services, releases, or activities that contain the rule

A gold star beside the name of an item means that you are subscribed to it.

To manually subscribe to an item:

- Hover the mouse pointer near the name of the item you want to follow.
- Click the **star** icon.

- The star becomes gold, indicating that you are following its events.
- The item or items you are following are also added to the Followed Rules section on the **Home** tab.

To manually unsubscribe to an item:

- Click a gold star to unsubscribe to the item that you are following.
 - It becomes empty.

You can also set your preferences to automatically follow certain events, such as rules that you edit or releases and activities that you are an approver of. You can enable or disable these options by clicking your user name in the top banner, and then selecting **Profile**.



Social features: Streams and comments

- Use streams to follow work and post comments to team members

The screenshot displays the Stream view of the IBM Business console. At the top, there are tabs for 'What's New' and 'Stream', with 'Stream' being the active tab. Below the tabs is a text input field labeled 'Post a new comment here'. To the right of this field are 'Expand All' and 'Collapse All' buttons. A dropdown menu labeled '▼ Today' is open, showing two items:

- Bea created a version of salary2score rule in my-validation-rules-service project** at 3:13:36 PM. There is a 'Comment' link next to it.
- You created a new post** at 1:15:19 PM. The message reads: 'Hi, Bea. there is a new policy. Can you implement it by Friday? Thanks.' There is a 'Comment' link next to it.

Below these items is another comment from Bea:

- Bea added a comment** at 3:15:11 PM. The message reads: 'Hi, Paul. I completed the changes to the rules according to the new policy.'

At the bottom of the stream area is another 'Post a new comment here' input field.

Unit title

© Copyright IBM Corporation 2017

Figure 5-27. Social features: Streams and comments

Streams

The Business console provides a stream feature, which you can use to follow activities, and interact and collaborate with others through comments.

A Stream view is available beside the **Properties** tab when you view a release or activity, and it lists events that relate to that release or activity.

An event is when someone, including you, does tasks such as:

- Create, edit, or delete a rule or folder.
- Create or change the state of a release, change activity, or validation activity.
- Create or restore a snapshot.
- Post a comment.

Another stream view is available on the **Home** tab. This stream includes only the events that you subscribe to follow. You can follow releases, activities, or individual project elements such as business rules.

Comments

You can post comments of general interest or about a specific release, activity, or rule.

You can post comments in all stream views:

- On the **Home** tab:
 - Comments posted here are visible to all Business console users by default, and can include web links and file attachments.
- When viewing or editing a release, activity, or rule:
 - Comments posted here are visible to users that have access to the release, activity, or rule.

You can reply to any post, from either one of these views, by adding a comment to the post.

Top-level posts

Top-level posts, which you enter in the **Home** tab stream, are visible to all Business console users unless you specify otherwise.

You can post general comments through the top-level stream on the **Home** tab, or you can post comments that are specific to a project.

Unit summary

- Describe the Business console and Enterprise console rule authoring environments
- Use the Business console rule editors to create and change rules

Unit title

© Copyright IBM Corporation 2017

Figure 5-28. Unit summary

Review questions

1. True or False: Rule folders in Decision Center map to Rule Designer rule packages.
2. True or False: A snapshot captures the state of rules in a project or branch at a specific time.
3. True or False: You can follow only rule artifacts in the Business console.



Unit title

© Copyright IBM Corporation 2017

Figure 5-29. Review questions

Write your answers here:

- 1.
- 2.
- 3.

Review answers

1. True or False: Rule folders in Decision Center map to Rule Designer rule packages.

The answer is True.



2. True or False: A snapshot captures the state of rules in a project or branch at a specific time.

The answer is True.

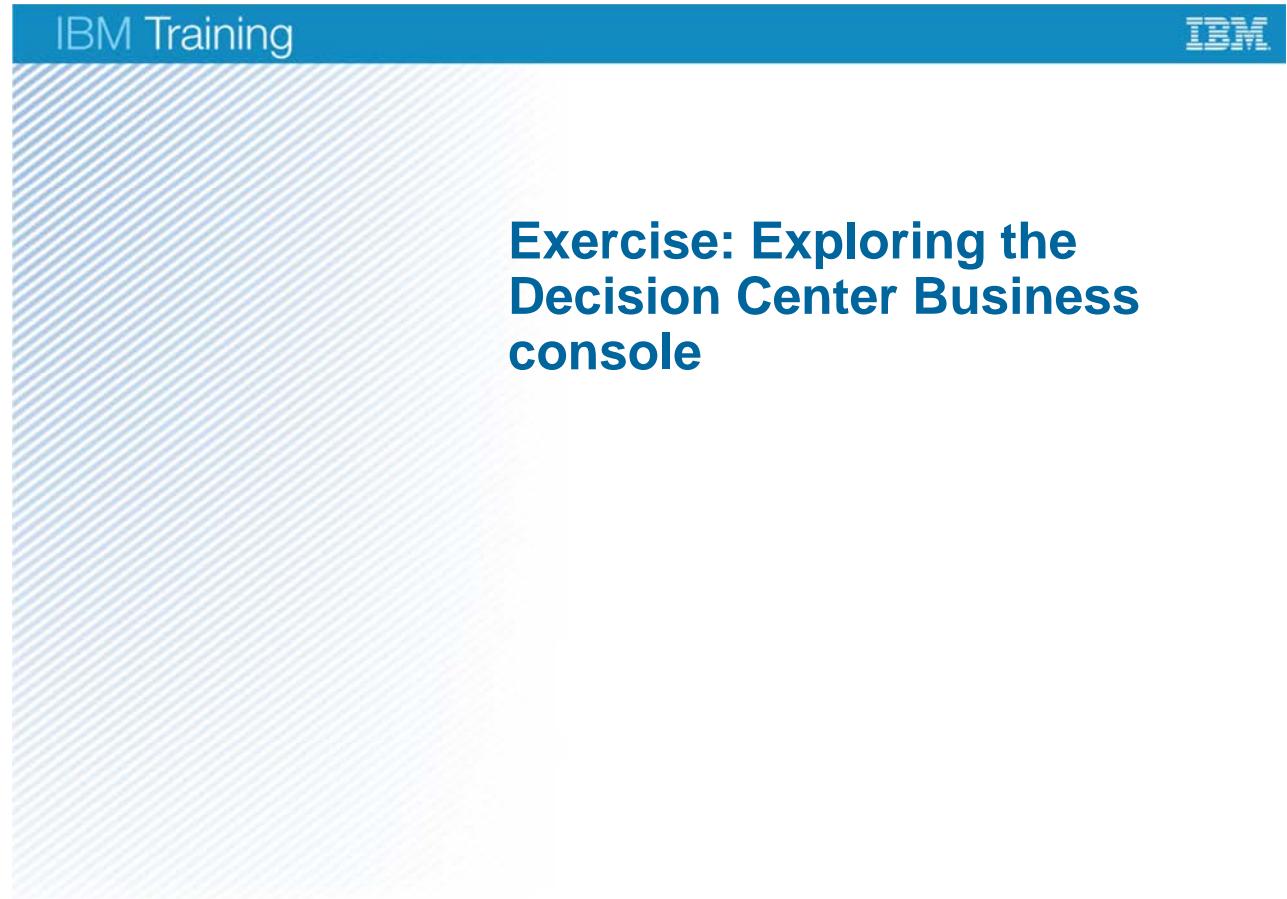
3. True or False: You can follow only business rules in the Business console.

The answer is False. In addition to rules, you can also follow releases and activities to receive notifications about any changes to these decision artifacts.

Unit title

© Copyright IBM Corporation 2017

Figure 5-30. Review answers



Unit title

© Copyright IBM Corporation 2017

Figure 5-31. Exercise: Exploring the Decision Center Business console

Exercise objectives

- Use the Business console rule editors to author rules
- Collaborate with other users to modify rules
- Manage changes by using timelines and snapshots
- Compare versions of rules



Unit title

© Copyright IBM Corporation 2017

Figure 5-32. Exercise objectives

Unit 6. Introducing rule authoring

Estimated time

01:00

Overview

This unit and the next several units focus on various aspects of rule authoring. This unit reviews how the rule language evolves during a business rule project, and introduces the structure of rules.

How you will check your progress

- Checkpoint
- Exercise

Unit objectives

- Describe when rule writing occurs in the implementation process
- Define the typical roles of people who author rules
- Describe how these roles might fit within their organizational environment
- List the Operational Decision Manager tools that are used for rule authoring
- Define the parts of a rule
- Explain the difference between business policy and business rules, and provide an example of each

Unit title

© Copyright IBM Corporation 2017

Figure 6-1. Unit objectives

This unit and the next several units focus on rule authoring techniques and practices.

This unit provides an introduction to rule authoring.

Topics

- From business policy to business rule
- Rule structure
- Rule artifacts

Unit title

© Copyright IBM Corporation 2017

Figure 6-2. Topics

6.1. From business policy to business rule



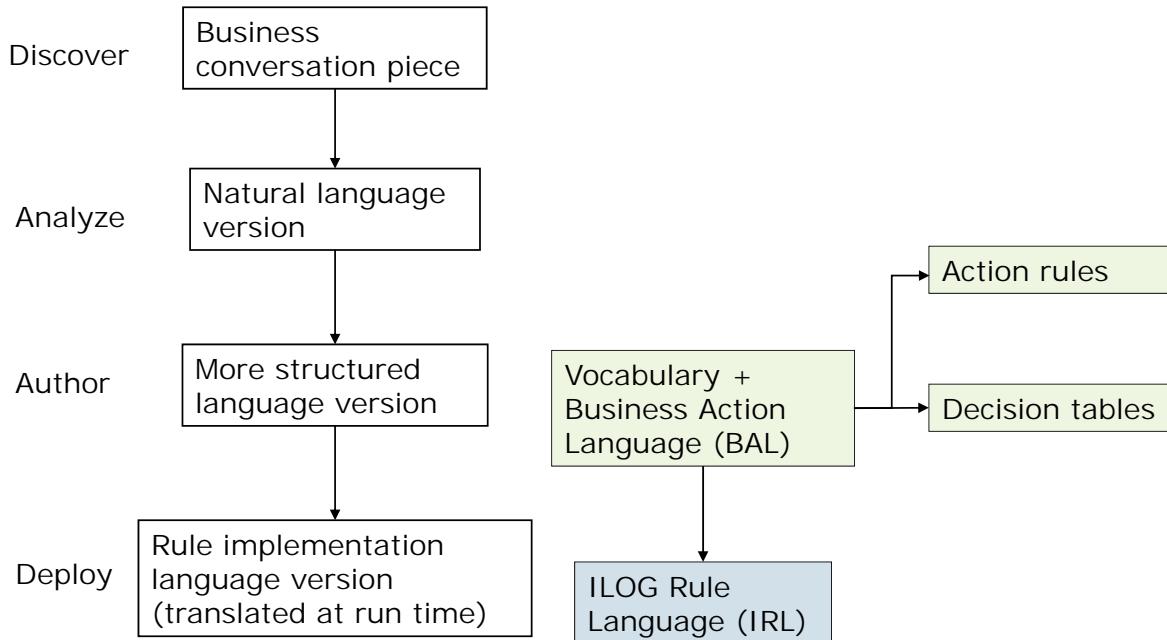
Unit title

© Copyright IBM Corporation 2017

Figure 6-3. From business policy to business rule



Rule language evolves during a project (1 of 2)



Unit title

© Copyright IBM Corporation 2017

Figure 6-4. Rule language evolves during a project (1 of 2)

- **Discover:** During the discovery phase, the basis for rules is captured from interviews, business documents, and discussion.
- **Analyze:** During the analysis phase, business analysts work through what they discovered. Rules become more formalized, but are still on paper and in natural language.
- **Author:** After analysis, business analysts can work with the tools (including Rule Designer and Decision Center rule editors) to create and verbalize the BOM, and write rules. This phase is the **authoring** stage. Rules are written in Business Action Language (BAL), which is a rule language that is used in Operational Decision Manager. BAL is a quasi-natural language that combines rule constructs with terms and phrases that are expressed in the vocabulary. As you learned earlier, the vocabulary is created through verbalization of the BOM, in Rule Designer. Rules can be authored by using Decision Center or Rule Designer. When the rule is authored in one of the rule editors, it uses the vocabulary and BAL constructs.
- **Deploy:** When rules are deployed, the rule statement, which is written in BAL from the BOM vocabulary, is translated into the rule implementation language. The BAL rule statement is what the rule engine sees at run time. This language is called the ILOG Rule Language, or IRL.



Information

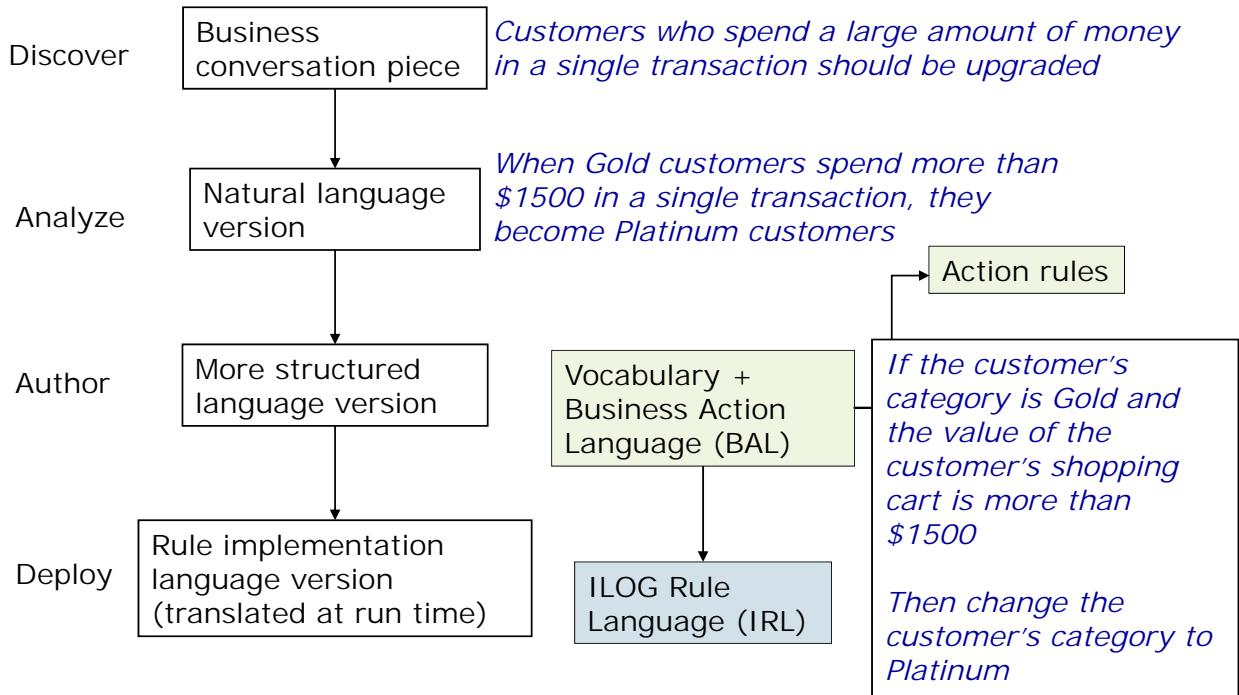
The following notes provide some basic information about IRL:

- IRL operators are a subset of the operators that are provided in the Java programming language.
- A number of keywords are reserved, and naming restrictions apply to package names.
- IRL grammar notation follows the notation that is presented in *The Java Language Specification* by James Gosling, Bill Joy, and Guy Steele (Addison Wesley, Second Edition, 2000).

For more information about IRL, see the product documentation.



Rule language evolves during a project (2 of 2)



Unit title

© Copyright IBM Corporation 2017

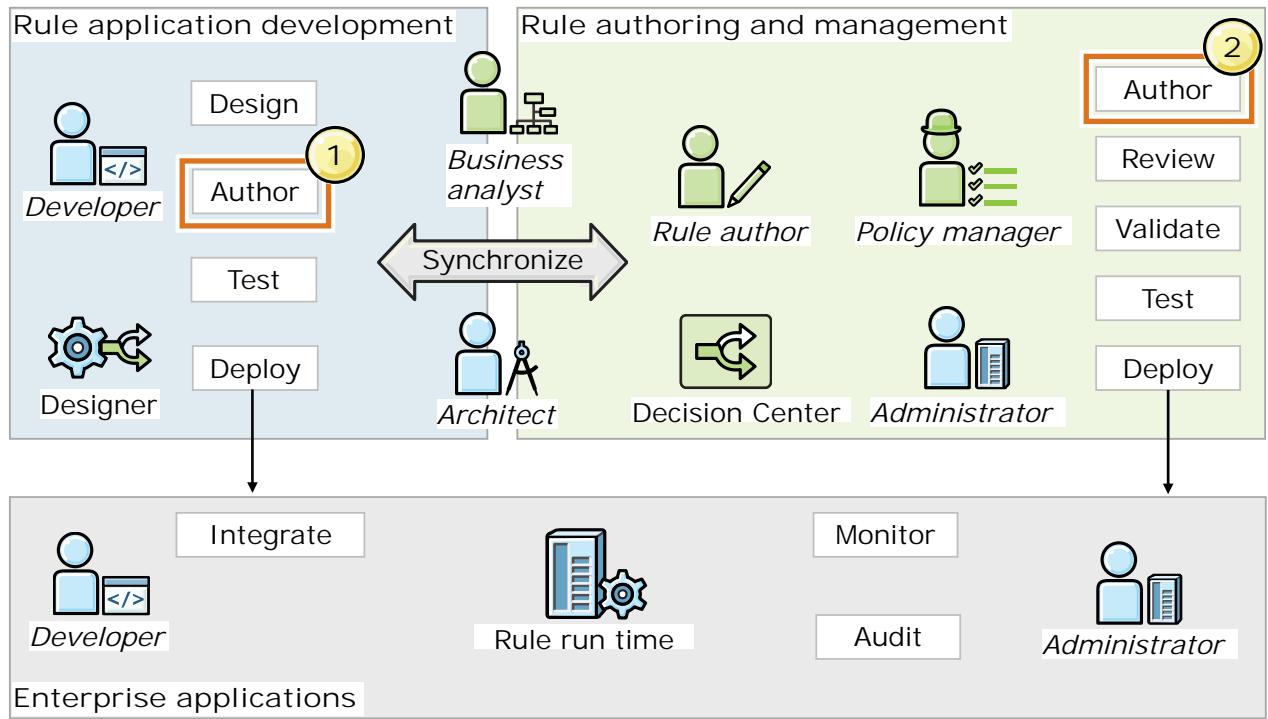
Figure 6-5. Rule language evolves during a project (2 of 2)

- **Discover:** During the discovery phase, a rule might be expressed as:
"Customers who spend a lot of money in a single transaction should be upgraded."
- **Analyze:** During the analysis phase, the previous rule might be reframed as "When a Gold customer spends more than \$1500 in a single transaction, the customer becomes a Platinum customer." In this case, the rule now associates specific vocabulary terms to the customer that are based on purchase amounts.
- **Author:** During the authoring phase, rules are implemented as "if-then" statements in action rules, decision tables, or decision trees.
- **Deploy:** As a business analyst, you do not work directly with IRL.

IBM Training



When rule authoring occurs



Unit title

© Copyright IBM Corporation 2017

Figure 6-6. When rule authoring occurs

Rule authoring can have the following sequence:

- Initial rule development:

During the early stages of a project, a large-scale effort is made to enter the business rules that are identified during the discovery and analysis phases. The rule project is designed with a folder structure to organize the rules in rule packages. As soon as the vocabulary is implemented as a business object model (BOM), rule authoring can begin.

- Ongoing rule maintenance and management:

Management and maintenance continue after the business rule application is rolled out to production. Rule maintenance becomes a regular task for some business users as business policies shift and more rules are identified.

Who writes rules?

- Business analysts
 - Typically involved heavily during initial rule development
 - Might author many rules
- Policy managers and rule authors
 - Act as subject matter experts (SMEs) during early stages
 - Involved in reviewing and validating rules
 - Responsible for maintaining rules (write and edit) after the system is rolled out
- Developers
 - Involved heavily during initial rule development
 - Create technical rules and complex rules

Unit title

© Copyright IBM Corporation 2017

Figure 6-7. Who writes rules?

People in several different roles can write rules.

Business analysts are typically involved heavily during initial rule development, and might author many rules then.

Policy managers and other business users act as subject matter experts during the early stages, and then review and validate rules that are written during development. They are likely to write, edit, and maintain rules after the system is rolled out.

Developers are involved heavily during initial rule development, and create technical and complex rules. At some companies, developers are the primary rule authors.

Where are rules written?

- Decision Center
 - Main rule authoring and management environment for business users
 - Includes features for rule administrators
 - Can use Business console or Enterprise console
- Rule Designer
 - Eclipse-based development environment for developers and technical business analysts
 - Includes advanced rule editors and tools to manage synchronization with Decision Center

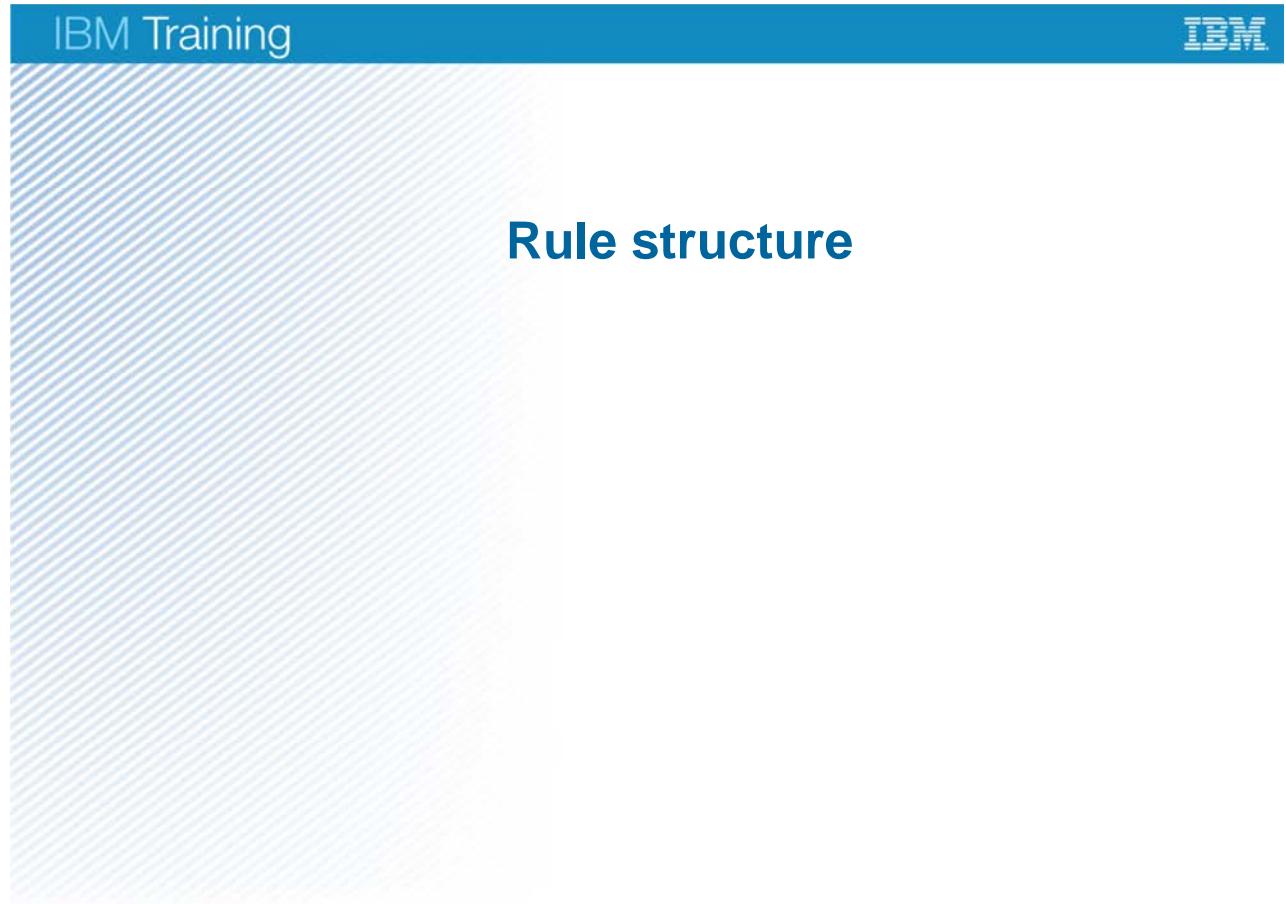
Unit title

© Copyright IBM Corporation 2017

Figure 6-8. Where are rules written?

Rules can be written in the Decision Center consoles and Rule Designer.

6.2. Rule structure



Unit title

© Copyright IBM Corporation 2017

Figure 6-9. Rule structure

Parts of a rule (1 of 3)

Definitions	definitions	Define rule-specific variables (optional)
Conditions	if	Defines the conditions that determine when to execute rule actions <ul style="list-style-type: none"> Condition statements are optional
	then	Defines the actions that are taken when all conditions are met <ul style="list-style-type: none"> All rules must contain at least one action
	else	Defines the action to take when conditions are not met <ul style="list-style-type: none"> The else part is optional when a condition is supplied It cannot be included when the rule does not have conditions

Unit title

© Copyright IBM Corporation 2017

Figure 6-10. Parts of a rule (1 of 3)

A business rule has three main parts: definitions, conditions, and actions.

- Definitions* are used to define any variable that is used within the rule. After you define a variable, you can use it throughout the rule.
- Conditions* specify the criteria that are evaluated to determine whether an action is taken. Conditions are described in the **if** part of a rule.
- Actions* specify the actions to take when the conditions of a rule are met. Actions are described in the **then** part of the rule.

In some cases, the action represents what to do if conditions are not met (when the conditions are “false”), in which case, the action is written as an **else** statement.

The condition part of the rule is evaluated for all objects. When the conditions are satisfied for one or more objects, the action part of the rule is executed for those objects. The execution of the action can add, delete, or modify data.

The definition and condition parts of a rule can be empty. However, a rule must always specify at least one action. If a rule has no definitions and no conditions, it is always executed.

Put together, these parts form the basic structure of every business rule.

The following units, which you cover later in this course, include more detail about the parts of a rule: *Working with conditions in rules*, *Working with definitions in rules*, and *Writing complete rules*.

Parts of a rule (2 of 3)

if

the customer's category is Gold
and the value of the customer's shopping cart is more than 1500,

then

change the customer's category to Platinum

definitions	(none in this example)
if	the customer's category is Gold <i>and</i> the value of the customer's shopping cart is more than 1500
then	change the customer's category to Platinum
else	(none in this example)

Unit title

© Copyright IBM Corporation 2017

Figure 6-11. Parts of a rule (2 of 3)

The example rule here does not have a definition.

The condition (**if** part) has two statements that are evaluated on all the objects (in this case, all the customers).

The action statement says:

"change the customer's category to platinum"

This statement means that the status for all gold customers changes to platinum when they have more than 1500 in their shopping cart.

Parts of a rule (3 of 3)

- If the customers' category is Gold, define them as loyal customers
- If they are loyal customers and they are over 60, give them a 10% discount
- If they are loyal customers and they are under 60, give them a 5% discount

definitions	set “loyal customer” to a customer where the category of this customer is Gold
if	the age of “loyal customer” is greater than 60
then	give a 10% discount to “loyal customer”
else	give a 5% discount to “loyal customer”

Unit title

© Copyright IBM Corporation 2017

Figure 6-12. Parts of a rule (3 of 3)

This example has a definition.

The definition part of the rule is used to set variables that can be used in the rest of the rule.

One use of a variable is to provide a convenient name as a reference for an object, which can simplify the wording of your business rules. After you define the variable, it becomes available in all the parts of the current rule.

Here, “loyal customer” is defined as a variable and is used in the condition and action statements.

Rules must include an action

- Rules must always specify an action
 - Other parts of the rule are optional
- Examples of actions in different rules:

then

```
set the corporate score in 'the loan report' to the  
credit score of 'the borrower' ;
```

then

```
display the message "Congratulations! You have been  
upgraded to Gold membership"
```

Unit title

© Copyright IBM Corporation 2017

Figure 6-13. Rules must include an action

Remember that rules must always specify an action. The other parts of a rule are optional.

Differences between policies and rules

- Rules enforce policies
 - The business policy embodies a business decision
 - The underlying logic to produce that decision is defined in business rules
- Example:
 - A company policy might state:
Managers can charge only \$100.00 per day on the company credit card
 - To transform this statement into a rule statement, you might start with:
if the manager's charge amount > 100.00
then... ?
 - To write a rule that enforces the policy, you must know what action to take

Unit title

© Copyright IBM Corporation 2017

Figure 6-14. Differences between policies and rules

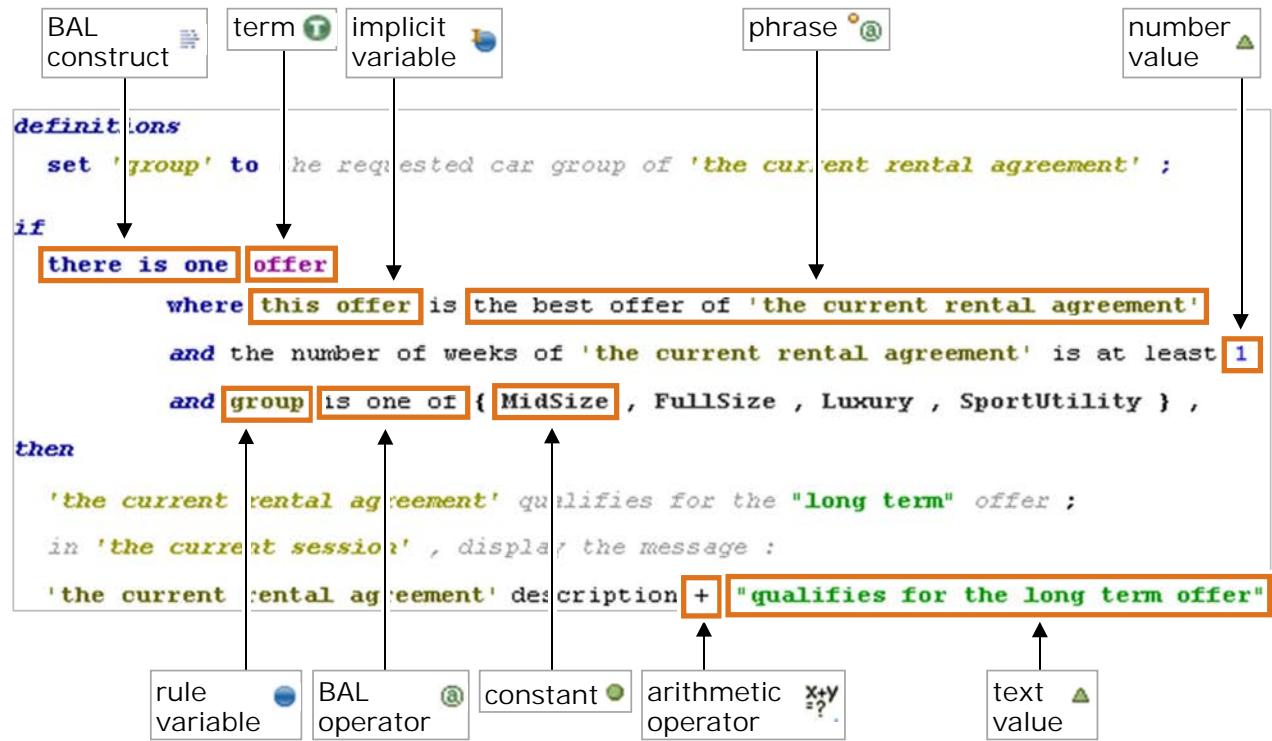
Rules enforce policies. To get from a policy to a rule, you must define an action.

For example, a company policy might state that managers can charge only \$100 per day on a company credit card. Before that policy can become a rule, you must know what action to take when the charge amount is greater than \$100.

Sometimes, a single policy requires several rules to implement it.



Detailed rule structure



Unit title

© Copyright IBM Corporation 2017

Figure 6-15. Detailed rule structure

A complete example of a rule is shown here, highlighting BAL constructs, operators, and variables, which you learn more about in the next units and exercises.

IBM Training

Discussion

Analyzing rules

Unit title

© Copyright IBM Corporation 2017

Figure 6-16. Discussion

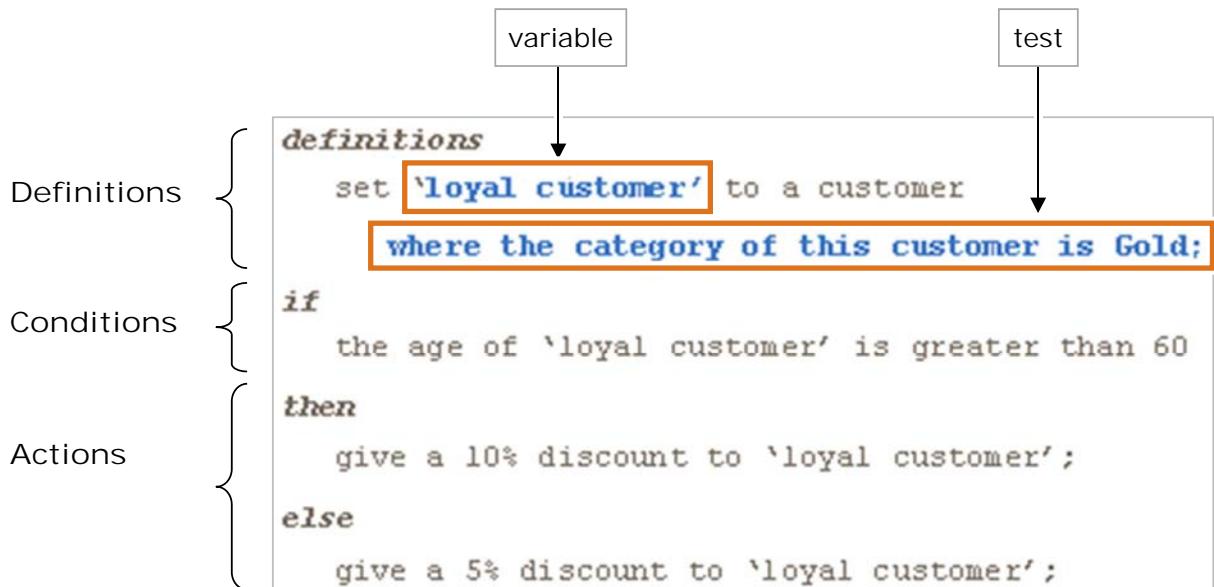
In this discussion, you analyze a rule and determine the underlying policy. You then write down some rules from your own organization.

If you are a classroom or online student, be prepared to share your ideas with the class.

If you are a self-paced student, you can complete this discussion activity as an independent exercise.

IBM Training 

Discussion question: What is the underlying policy for this rule?



Unit title

© Copyright IBM Corporation 2017

Figure 6-17. Discussion question: What is the underlying policy for this rule?

Inside of the rule editor, rules might be different from the underlying policy.

Take time to look at this rule, and consider: what is the underlying business policy?

Discussion: Structuring your rules

Definition	definition	
Condition	If	
Action	Then	
	Else	

Unit title

© Copyright IBM Corporation 2017

Figure 6-18. Discussion: Structuring your rules

Think about the rules that apply in your own business.

1. Take a few minutes and write down one or two rules from your experience.

2. Rewrite your rules, clearly identifying the structure.

Remember that a rule does not always have all parts, but a rule must have an action.

6.3. Rule artifacts



Unit title

© Copyright IBM Corporation 2017

Figure 6-19. Rule artifacts

Types of rule artifacts

- Action rules
 - Sentence-like statements that express a set of conditions and the actions to take when the conditions are found to be true
- Decision tables
 - Represent decision logic as a table
 - Each row in a decision table corresponds to an action rule
- Other artifacts that developers create
 - Ruleflows
 - Variable sets
 - Technical rules

Unit title

© Copyright IBM Corporation 2017

Figure 6-20. Types of rule artifacts

A rule project can contain different types of rule *artifacts*.

Business rules and decision tables are separate types of artifacts.

Business rules are individual rules, also called action rules.

Decision tables are groups of similar rules that are organized in tables as a convenient way of viewing and managing those sets of rules.

Operational Decision Manager also has other types of artifacts besides action rules and decision tables that developers create, such as ruleflows or variable sets.

Each artifact has a different editor.

Depending on whether you are working with the tools or looking at documentation, you might see that the terms “artifact” and “element” are sometimes used to mean the same thing. For example, you might see that “element” is used in Decision Center, and that “artifact” is used in Rule Designer.

Unit review (1 of 2)

- Rule authoring occurs mainly in two stages:
 - Initial rule development at the beginning of a project to implement rules that were identified during discovery and analysis
 - Ongoing maintenance after the business rule application is in production
- Who writes rules?
 - Business analysts, policy managers, and other business users are the most common rule authors over the long term
 - Developers might participate during early stages of the project or when more technical rules are required
- Business rules can be written in Decision Center and Rule Designer

Unit title

© Copyright IBM Corporation 2017

Figure 6-21. Unit review (1 of 2)

This slide is a review of the concepts that are covered in this unit.

Unit review (2 of 2)

- Rules enforce policies
 - Rules are more concrete, and define actions
 - One policy might require several rules to implement it
- Rule structure includes:
 - Definitions (defines variables)
 - Conditions (the “if” part of the rule statement)
 - Actions (the “then” and “else” parts of rule statements)
- Several types of rule artifacts or elements:
 - Business rule
 - Decision table
 - Other elements that developers create (such as ruleflows, variable sets, or technical rules)

Unit title

© Copyright IBM Corporation 2017

Figure 6-22. Unit review (2 of 2)

This slide continues the review of concepts that are covered in this unit.

Unit summary

- Describe when rule writing occurs in the implementation process
- Define the typical roles of people who author rules
- Describe how these roles might fit within their organizational environment
- List the Operational Decision Manager tools that are used for rule authoring
- Define the parts of a rule
- Explain the difference between business policy and business rules, and provide an example of each

Unit title

© Copyright IBM Corporation 2017

Figure 6-23. Unit summary

Review questions (1 of 2)

1. True or False: Rule authoring starts after the rule application is rolled out to production.

2. Which of the following roles typically author rules? Select all that apply.
 - A. Business analysts
 - B. Policy managers
 - C. Administrators
 - D. Developers
 - E. All of the preceding answers



Unit title

© Copyright IBM Corporation 2017

Figure 6-24. Review questions (1 of 2)

Write your answers here:

- 1.

- 2.

Review questions (2 of 2)

3. Which of the following Operational Decision Manager modules include editors for authoring rules?
 - A. Decision Center
 - B. Rule Execution Server
 - C. Rule Designer
 - D. All of the preceding answers
4. True or False: Each business rule must include definitions, conditions, and actions.



Unit title

© Copyright IBM Corporation 2017

Figure 6-25. Review questions (2 of 2)

Write your answers here:

3.

4.

Review answers (1 of 2)



1. True or False: Rule authoring starts after the rule application is rolled out to production.

The answer is False. Rule authoring begins during initial phases of business rule application development to implement the rules that were identified during discovery and analysis. Rule authoring continues as ongoing, regular maintenance after the business rule application is in production.

2. Which of the following roles typically author rules? Select all that apply.

- A. Business analysts
- B. Policy managers
- C. Administrators
- D. Developers
- E. Any of the preceding answers

The answer is A, B, and D. Depending on the company, one person might have all of these roles. However, the administrator role is not involved in rule authoring.

Unit title

© Copyright IBM Corporation 2017

Figure 6-26. Review answers (1 of 2)

Review answers (2 of 2)

3. Which of the following BRMS modules include editors for authoring rules?

- A. [Decision Center](#)
- B. Rule Execution Server
- C. [Rule Designer](#)
- D. All of the preceding answers

The answer is A and C. Decision Center and Rule Designer include editors for authoring rules.

4. True or [False](#): Each business rule must include definitions, conditions, and actions.

The answer is False. Every business rule must have an action. However, a rule is not required to have a definition or a condition statement.



Exercise: Understanding the case study

Unit title

© Copyright IBM Corporation 2017

Figure 6-28. Exercise: Understanding the case study

Exercise objectives

- Describe the business domain
- Describe the business policies and vocabulary requirements for this domain



Unit title

© Copyright IBM Corporation 2017

Figure 6-29. Exercise objectives

Unit 7. Discovering and analyzing rules

Estimated time

01:00

Overview

This unit continues the focus on rule authoring. You learn how to perform rule discovery and analysis tasks by using a realistic case study.

How you will check your progress

- Checkpoint
- Exercises

Unit objectives

- Use good practices to discover rules
- Analyze rules to eliminate problems before implementation

Unit title

© Copyright IBM Corporation 2017

Figure 7-1. Unit objectives

Topics

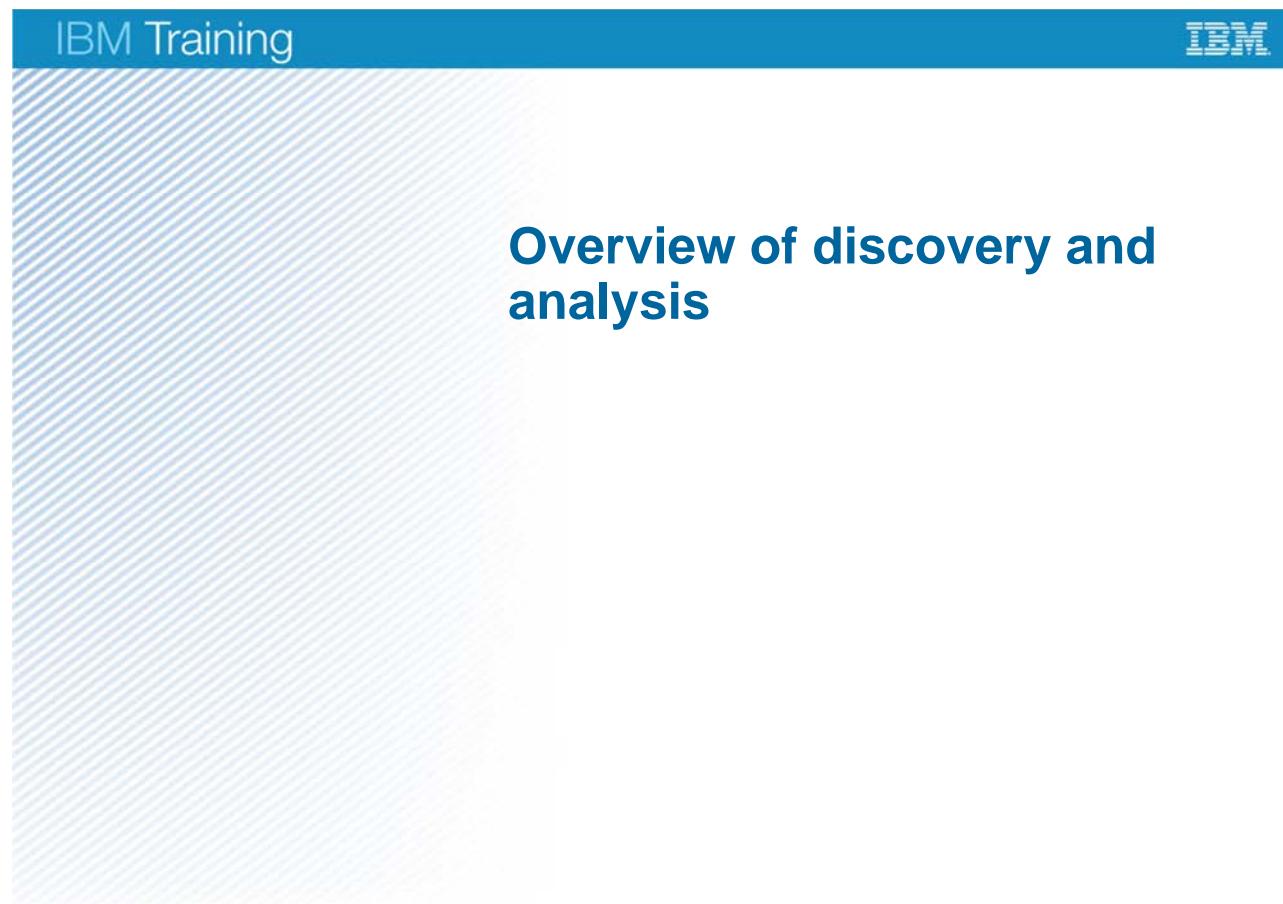
- Overview of discovery and analysis
- Agile Business Rule Development (ABRD)
- Rule discovery
- Rule analysis

Unit title

© Copyright IBM Corporation 2017

Figure 7-2. Topics

7.1. Overview of discovery and analysis



Unit title

© Copyright IBM Corporation 2017

Figure 7-3. Overview of discovery and analysis

Goals of rule discovery and analysis

- Extracting rules as manageable artifacts
- Tracing the rule lifecycle from requirements to deployment
- Linking rules to business context and motivation
- Developing rules that use natural language
- Preparing the object model and rules for implementation and deployment
- Involving the business users in ownership of the rules and gleaning some governance information, such as who modifies the rule, how frequently, and when

Unit title

© Copyright IBM Corporation 2017

Figure 7-4. Goals of rule discovery and analysis

This unit concentrates on the methodology for discovering the rules from use cases, and provides hands-on practice in both capturing and analyzing the rules that you discover. Understanding this methodology helps you to accurately transform your business policies into formal business rules that can be implemented.

This unit walks you through the methodology so that you can be prepared to discover and formalize business rules later so that your rule authors and business users can correctly manage and maintain your rules.

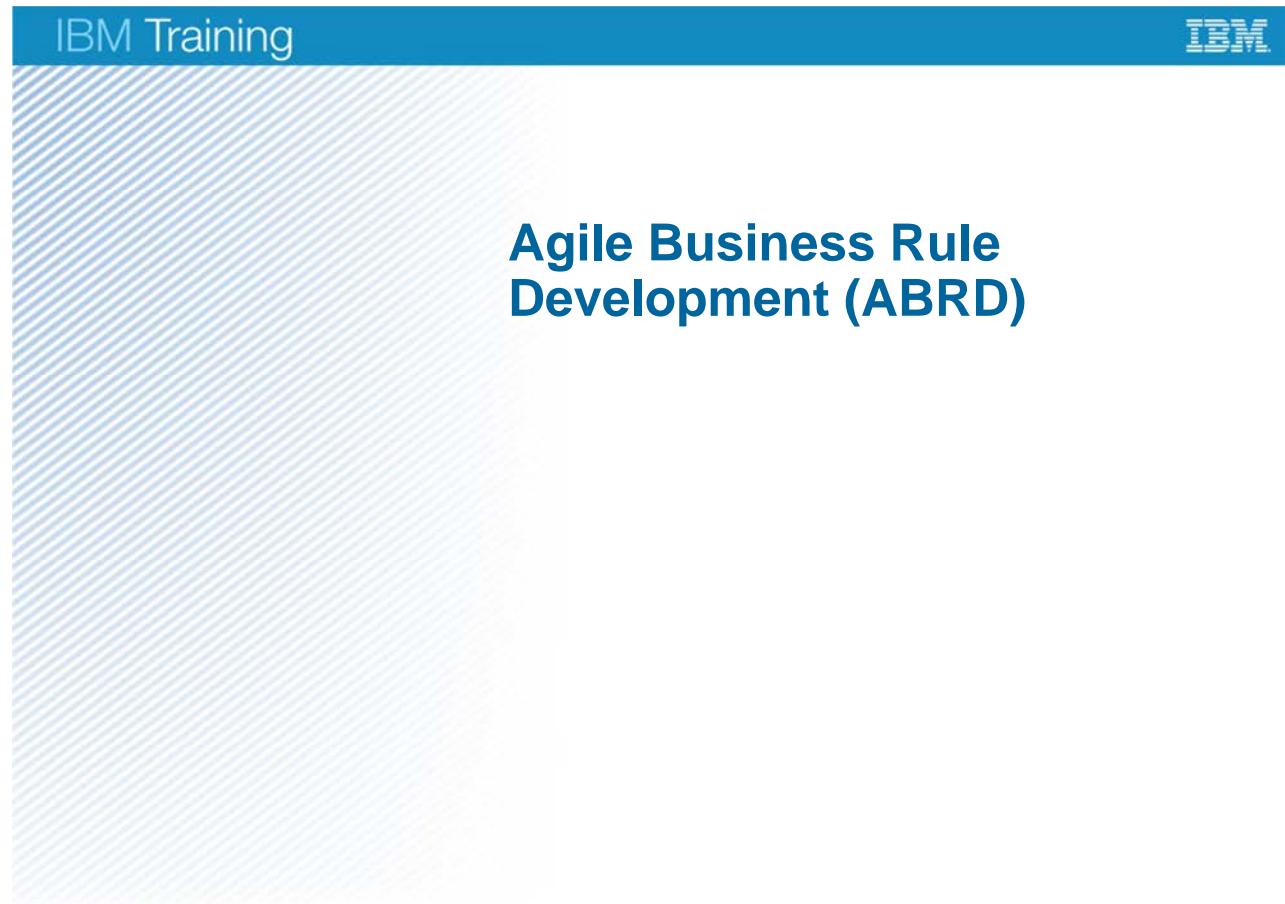
The methodology that is presented in this unit can support your efforts to discover the decisions that must be implemented in your business rule application.

This training presents the Agile Business Rule Development (ABRD) methodology as a set of suggested practices for accomplishing these goals.

Discovery and analysis are iterative processes. They:

- Are done incrementally.
- Are done in parallel with the other tracks.
- Involve extensive interaction between policy managers, rule authors, and business analysts.

7.2. Agile Business Rule Development (ABRD)



Unit title

© Copyright IBM Corporation 2017

Figure 7-5. Agile Business Rule Development (ABRD)

Challenges to implementing business rule applications

- Project goals and requirements are poorly specified
- Decision logic is complex and not easily understood
 - Hidden in heritage code, or only subject matter experts understand it
 - Must rely on multiple SMEs
- Relevant data is difficult to access and use
 - Hidden in desktop repositories such as spreadsheets
 - Inaccurate, not maintained, missing entirely
 - Stranded in difficult-to-connect older systems
- Supporting business processes are nonexistent
- Supporting IT platforms are outdated or nonexistent

Unit title

© Copyright IBM Corporation 2017

Figure 7-6. Challenges to implementing business rule applications

Some of the biggest challenges to implementing a business rule application are revealed during the discovery phase.

- Project goals and requirements might not be adequately specified.
- The decision logic can be complex or hidden in code, or might require multiple subject matter experts to explain it.
- Sometimes the relevant data is difficult to access or inaccurate if it was not maintained. In some cases, it might be missing entirely.
- Support for business processes, or IT platforms, can be out of date or even nonexistent.

With these challenges in mind, be prepared for the discovery phase to take time and several iterations.

Agile Business Rule Development (ABRD)

- Incremental and iterative software development methodology that is based on:
 - Unified process (OpenUP)
 - Agile
 - Extreme programming
- Predefined set of:
 - Phases
 - Activities
 - Techniques
 - Guidelines and suggested practices
 - Deliverables supporting project plan elaboration, governance, and project management tasks

Unit title

© Copyright IBM Corporation 2017

Figure 7-7. Agile Business Rule Development (ABRD)

The ABRD methodology is a set of suggested practices that are based on Unified Process (OpenUP), Agile, and extreme programming (XP) concepts.

ABRD provides a consistent way to build decision management solutions from inception to deployment. The methodology helps you to efficiently accomplish the goals of the discovery and analysis phase, and meet the challenges that arise when an organization implements a business rule application.

ABRD phases of development

- Inception
 - Specify requirements and establish project boundaries
- Elaboration
 - Create use case documents and business models
- Construction
 - Build system incrementally
 - Develop and test rules
- Transition
 - Ensure that rule application is ready for business users

Unit title

© Copyright IBM Corporation 2017

Figure 7-8. ABRD phases of development

ABRD involves four phases of development: inception, elaboration, construction, and transition.

ABRD tracks

Each development phase involves four tracks:

- Architecture track
 - Selection, customization, and support of technology that is used to implement the system
- Decision (rule) track
 - Capture, analysis, automation, and maintenance of the “raw” business rules, which are the business policies behind the decision making
- Process track
 - Interactions between the actors of the system, between actors and the system, and between the system and other automated components
- Data track
 - Implementation of the object model and the database

Unit title

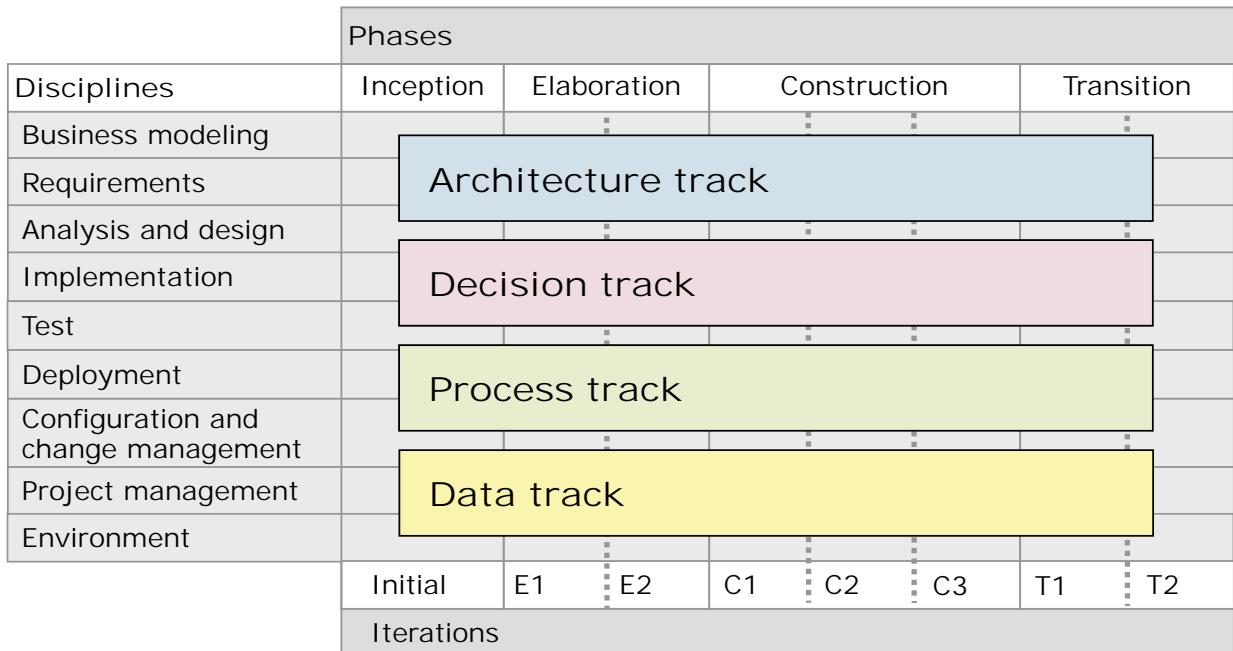
© Copyright IBM Corporation 2017

Figure 7-9. ABRD tracks

Each development phase involves four tracks: the architecture track, the decision (or rule) track, the process track, and the data track.

ABRD methodology

- Each ABRD track is involved at each development phase



Unit title

© Copyright IBM Corporation 2017

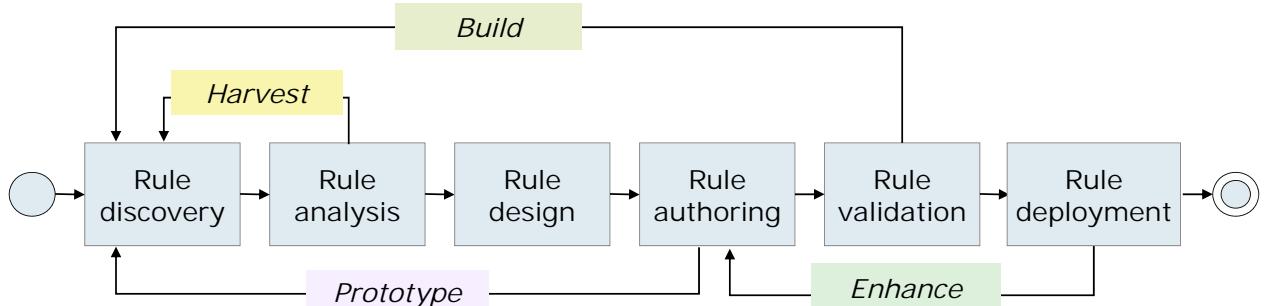
Figure 7-10. ABRD methodology

The tracks reflect that specific activities must be allocated to decisions and processes in each phase of development and in each discipline of the lifecycle for business rule application development.

With the agile approach, iterations of rule discovery and analysis occur in parallel across each of these tracks.

As business analysts, you are involved in all of these tracks. However, your main focus is the decision, or rule, track.

Agile ruleset development



- **Discovery**
 - Harvest rules by using short workshop
- **Analysis**
 - Understand and prepare rules for implementation
- **Design**
 - Define BOM, project structure, and ruleset; prototype rules

- **Authoring**
 - Develop rules and test scenarios
- **Validation**
 - Perform functional tests and involve subject matter experts (SMEs) for feedback
- **Deployment**
 - Use Rule Execution Server staging platform

Unit title

© Copyright IBM Corporation 2017

Figure 7-11. Agile ruleset development

The decision track implementation follows an iterative approach. As shown here, the decision track includes all tasks of the decision lifecycle, starting with discovery and analysis. Rules are developed incrementally, in multiple iterations of short time frames. In ABRD, the entire process lifecycle might not be followed for each iteration. Instead, a number of short cycles between phases might occur, resulting in the gradual evolution of the entire ruleset.

1. After a short discovery workshop, the harvested rules are analyzed and prepared for implementation (discovery, analysis).
2. The rule project structure and sequence of rules are defined, and rules are prototyped (design).
3. Next, the authoring environment is set up so the rules can be entered in the rule editors. Another goal of this phase is creating test scenarios (authoring).
4. The rules are tested and validated (validation).
5. After validation, the rules can be deployed to Rule Execution Server (deployment).

As the ruleset is built, it can be enhanced by adding new facts, entities, or attributes to the object or data model.

7.3. Rule discovery



Unit title

© Copyright IBM Corporation 2017

Figure 7-12. Rule discovery

Rule discovery

- Find business decisions and structure them into rules
- Discovery tasks:
 1. Select a discovery roadmap
 2. Select or confirm rule standards
 3. Discover the rules through the roadmap
 4. Validate that rules correspond to business context
 5. Discover vocabulary
 6. Add concrete scenarios for testing

Unit title

© Copyright IBM Corporation 2017

Figure 7-13. Rule discovery

Rule discovery is the process of finding the business decisions and then structuring those decisions into rules. Unit 2, "Modeling for business rules" demonstrated the path from use cases to decision points to vocabulary.

This unit follows the path from use cases to decision points to rules.

Discovery is an iterative process, so do not expect to find all the rules in one session. The set of rules continues to evolve and grow as you work with policy managers to capture them.

Although the unit describes the discovery tasks one at a time, keep in mind that these tasks are done in parallel during an actual project.

Step 1: Select discovery roadmap

- Defines the starting point and method for discovering rules, including:
 - Rule sources (people, documents, manuals, program code)
 - Participants
 - Scheduling of discovery sessions (workshops)
 - Expected deliverables for discovery sessions
 - Target date for completion of discovery
- Choosing a roadmap depends on factors such as:
 - Available rule sources
 - Type of intermediary deliverable that is most useful to the organization

Unit title

© Copyright IBM Corporation 2017

Figure 7-14. Step 1: Select discovery roadmap

The discovery roadmap is the starting point that defines which method to use for discovering rules, and what type of deliverables you expect to produce as a result.

The type of roadmap that you use determines:

- Where the sources for your rules are
- Who the participants in the discovery phase are
- How to schedule discovery workshops
- What deliverables to generate from the discovery workshops
- The target date for completion of discovery

This training focuses on the use case roadmap. Therefore, the starting point is use cases. Other roadmap types are available, so depending on your application, you might choose to use several types of roadmap.

Roadmaps

- Roadmap methods vary according to the type of modeling that is used in your organization

Roadmap types	Description
Use case	<ul style="list-style-type: none"> • Start with a use case, identify decision points within use cases, and then identify rules that define the user story • Use when the organization or team works with use cases or agile user stories
Business process	<ul style="list-style-type: none"> • Evaluate individual process steps and tasks to find decision points, then rules • Use when the organization or team works with process decomposition for the requirements gathering and analysis phase
Data analysis	<ul style="list-style-type: none"> • Start with the data model (class diagram), identify how data is created or changed, and find the associated rules

Unit title

© Copyright IBM Corporation 2017

Figure 7-15. Roadmaps

You can use different roadmap types, and the methods that are used in each roadmap type depend on the type of modeling that your organization uses.

Roadmap types include:

- Use cases
- Business processes
- Data analysis

This training focuses on the use case roadmap as a suggested practice.

Planning rule discovery time

- Estimating the amount of time that is needed for discovery can be complex
 - Discovery is iterative
 - Estimations can vary depending on the type of roadmap that is chosen
- Discovery time depends on the complexity of the rules and on how you define a rule
 - Examples: Is a decision table a rule? Is each row a rule? How do you create decision tables?
- Best estimations are based on past experiences
- Initial discovery iterations for your first set of rules take longer until you stabilize the BOM
 - After the BOM stabilizes, rule gathering becomes more linear

Unit title

© Copyright IBM Corporation 2017

Figure 7-16. Planning rule discovery time

Planning rule discovery time can be complex. Discovery time depends on several factors, such as which roadmap you choose, the complexity of the rules, and whether the participants in the discovery phase have experience with business rule discovery. The best estimator of time is experience. If discovery is new to you, you might find estimation difficult.

The iterative nature of the process means that discovering the first set of rules takes longer when the BOM is not stable. However, after the BOM stabilizes, rule discovery becomes more linear because there are fewer iterative cycles.

Step 2: Select rule standards

- Rule standards are important for systematic and methodical rule extraction
- To establish rule standards:
 1. Establish a rule classification scheme
 2. Decide where to store rules during the discovery process
 3. Agree on a naming convention for rules
 4. Determine which rules to express in the data model, and which rules to express as natural language rules
 5. Define templates for expressing rules in natural language
 6. Establish rule stewardship procedures (part of rule governance)

Unit title

© Copyright IBM Corporation 2017

Figure 7-17. Step 2: Select rule standards

Rule standards are important for systematic rule extraction.

The main purpose of establishing these standards is so that all members of the discovery team can be consistent.

The following process can help your organization establish rule standards. First, establish how the discovery team is going to classify the rules. The next thing to determine is where rules will be stored throughout the discovery process. The members of the discovery team must also agree on a naming convention for the rules. Next, the discovery team determines which rules are expressed in the data model, and which rules are expressed as natural language rules. They also define any templates for expressing rules in natural language. Finally, the discovery team must establish rule stewardship procedures, which are a part of rule governance.

Business rule classification (1 of 4)

- Ensures consistency in rule capture and design
- Ensures that rules are written and implemented properly
 - Results of rule execution are different depending on how the rule is written
 - Example: Constraints versus guidelines produce different results
- Classification scheme example:
 - Mandatory constraints: Rules use the word “must”
 - Guidelines: Rules use the word “should”
 - Computation: Rules calculate a value
 - Inferences: Rules create a value or new information from existing information
 - Action enablers: Basic “if-then” statements

Unit title

© Copyright IBM Corporation 2017

Figure 7-18. Business rule classification (1 of 4)

Rule classification ensures consistency when you capture and design rules, and ensures that rules are written and implemented correctly.

The results of rule execution can differ according to how the rule is written. For example, a rule that is written as a constraint can cause a transaction to fail when the condition is false. However, if the rule is written as a guideline rather than a constraint, a transaction might be allowed, but with certain side effects.

You can adopt several classification schemes. For example, business rules can be classified as described on the next slide.

Business rule classification (2 of 4)

Classification	Description
Mandatory constraints	<ul style="list-style-type: none"> • Rules that reject the attempted business transaction (use the word “must”) • Grammar: <code><term> MUST HAVE <at least, at most, exactly N of> <term>; <term> MUST BE IN LIST <a,b,c>;</code> • Example: <i>Renters MUST be 25 years old or older AND MUST hold a valid driver license</i>
Guidelines	<ul style="list-style-type: none"> • Rules that do not reject the transaction, but warn about an undesirable circumstance (use “should” or “should not”) • Grammar: <code><term> SHOULD HAVE <at least, at most, exactly n of> <term>; <term> SHOULD BE IN LIST <a,b,c></code> • Example: <i>A branch SHOULD keep at least 5% of free inventory to handle emergencies</i>

Unit title

© Copyright IBM Corporation 2017

Figure 7-19. Business rule classification (2 of 4)

As you classify your rules, try to avoid thinking only in terms of “if-then” statements. Instead, make sure that you extract the complete statement of business policy that describes the rule.

Notice that the grammar that is described in the table is used only for rule documentation purposes, not implementation.

This slide lists the mandatory constraints and guidelines classifications.

Mandatory constraints impose required restrictions on the circumstances of the business transaction. Mandatory constraints must be true.

Guidelines express a warning about undesirable circumstances, but do not evaluate whether these circumstances are true. Guidelines do not reject the business transaction.

Business rule classification (3 of 4)

Classification	Description
Computation	<ul style="list-style-type: none"> Rules that create information from existing information by using mathematical computation Example: <i>Sale price = (original price – discount) + state tax + federal tax</i>
Inferences	<ul style="list-style-type: none"> Rules that create information from existing information The result is a piece of knowledge that is used as a new fact for the rule engine to consider Grammar: <code>IF <term> <operator> <term></code> <code>THEN <term> <operator> <term></code> Example – inferred status value: <i>Status becomes gold if you rent more than four times within 30 days</i>
Action enabler	<ul style="list-style-type: none"> Rules that test conditions and initiate another business event, message, or action when the condition is true Grammar: <code>IF <condition> THEN action</code>

Unit title

© Copyright IBM Corporation 2017

Figure 7-20. Business rule classification (3 of 4)

This slide includes more examples of how you can classify rules. Remember that the grammar that is described in the table is used only for rule documentation, not implementation.

This slide lists the computation, inferences, and action enabler classifications.

Computation rules compute the value of a term from existing information. The result of a computation rule is a new piece of information.

Inferences test conditions, and use existing information to create pieces of information. They establish the truth of a new fact.

Action enablers test conditions. If they find that the conditions are true, they initiate another business event, message, or action.

Business rule classification (4 of 4)

- As you go through the rules, look for the following cues

Circumstances that are <i>not</i> acceptable and reflect a violation of a constraint	Mandatory constraint
Circumstances that yield warnings	Guideline
Circumstances that alter the way the business event is handled and reflect specific values of inferred attributes • Example: <i>If the driver has a frequent renter card, propose an automatic upgrade</i>	Inference
Terms that are calculated values	Computation
Circumstances that trigger another business event while the event is processed • Example: <i>If inventory falls below some threshold, trigger a reorder process</i>	Action enabler

Unit title

© Copyright IBM Corporation 2017

Figure 7-21. Business rule classification (4 of 4)

To classify a rule, first try to determine whether a rule fits into the constraint, guideline, computation, or inference classifications. If not, the rule is probably an action enabler.

Discussion

Classify your rules

Unit title

© Copyright IBM Corporation 2017

Figure 7-22. Discussion

For classroom and online versions of the course, be prepared to discuss your work with the class.

For self-paced versions of this class, you can do this activity as an independent exercise.

Discussion: Classify your rules

- Discussion goal: Find the objects, attributes, and relationships that make up the vocabulary for your business rules
- Write down examples of business rules from your domain and organization
- Classify your example rules

Unit title

© Copyright IBM Corporation 2017

Figure 7-23. Discussion: Classify your rules

For classroom and online versions of the course, discuss your example rules and how you classified them with the class.

For self-paced versions of this course, take time to write down your ideas and notes on the example rules and their classification.

Take at least 10 minutes to provide examples of business rules from your domain and classify them. Write them down here:

Step 3: Discover the rules through the use case roadmap

- Business analysts work with policy managers and rule authors to review each use case
- Look for decision points in use case descriptions
 - Look for decisions, not rules
 - Try to break down decisions into smaller decisions, which are often the rules
- Look for these types of verbs, which are common cues of decision making:
 - Check
 - Qualify
 - Compute, calculate
 - Estimate, evaluate, assess
 - Verify, validate, confirm, decide
 - Compare

Unit title

© Copyright IBM Corporation 2017

Figure 7-24. Step 3: Discover the rules through the use case roadmap

With the use case roadmap, rule discovery begins by reviewing each use case description to find the decision points. At this stage, you are looking for decisions rather than rules. The use case steps describe how a decision is arrived at, and helps you to drill down into smaller decisions to find the underlying rules.

Consider these questions when you review use cases:

- Which use case steps involve decisions?
- Which of the requirements are related to decisions?
- Which use cases or use case steps represent complexity?
 - Complexity often implies decisions.
- Which use cases or use case steps are most subject to change?
 - Change is an indicator of decision making.
- Which verbs are used?

Decision points table example

Decision point name	Description	Source for rule	Current state of automation	Owner
Data validation	Validate the customer and policy data entry	Customer or data	Product data management (PDM)	IT
Customer classification	Assign a customer to its respective group for product delivery, terms of agreement	<ul style="list-style-type: none"> • Customer data analyst • Sales and marketing 	Manual review <ul style="list-style-type: none"> • Marketing database 	Marketing
Claim validation	Validate that the claim and the features that are selected for processing meet requirements	Customer service representative (CSR)	Manual review <ul style="list-style-type: none"> • Meeting with CSR 	Underwriter
Customer history assessment	Evaluation of customer relationship as a current or past customer	Credit scoring policies	Manual review <ul style="list-style-type: none"> • Finance reporting 	Marketing

Unit title

© Copyright IBM Corporation 2017

Figure 7-25. Decision points table example

This slide shows an example of a decision points table.

To ensure that all the decisions are accounted for, you can use a decision points table to list the discovered decisions and the source from which they were identified. The table is a useful intermediate step between the use cases and the rules themselves.

To create a decision points table:

- Name each decision point.
- Describe the decision with the original (“raw”) business language.
- State the source from which the decision was discovered.

Note how the table shown here lists the decision points, the original business policy description, the rule source, and the owner.

Also, as you work through the decisions for each step of the use case, try to determine how to classify the decision.

Decision points table: Classification complexity

Decision point name	Type	Complexity	Description	Source for discovery
Data validation	Action	Simple	Verify that customer, product, and policy data conform	Current application code
Customer eligibility	Inference	Complex	Define whether a requester can become a customer and at what factor	SME interview
Risk rating	Inference computation	Complex	Rate the insurance policy request according to customer profile and product value	SME interview and documentation

Unit title

© Copyright IBM Corporation 2017

Figure 7-26. Decision points table: Classification complexity

This table includes the complexity of classification for each decision.

Formalizing decisions as rules with rule templates

Rule name	Validate Claimant Feature Status		
Rule ID		Creation Date	05/21/2013
Rule Class		Package	IdentityClaimExceptions
Priority	None	Owner	John Smith
Business rule description	There is a claim such that the status is not "OPEN" or "INACTIVE" Create an issue code: "Invalid Claim Status"		
Business motivation	A claim in the wrong state at this step of the business process needs a review by the manager		
Objects involved		Where to express the rule	Decision Center
Who can change rule?	Owner only	When should the change occur?	If new status is defined for Claim
How rule is changed	Using web edition	Message for the action part	

Unit title

© Copyright IBM Corporation 2017

Figure 7-27. Formalizing decisions as rules with rule templates

After identifying the “raw” decisions, you can formalize them as rule statements in rule templates.

The rule template that is shown here describes the rule, and provides enough metadata to trace the rule back to its source and owner.

Rule documents

- Are used to maintain a full list of rules
- Provide explanation about the way the rules are documented, including:
 - **Rule classification:** Explains the classification scheme that is used for your rules
 - **Naming conventions:** Ensures that the rule names are meaningful to readers
 - **Document structure:** Helps readers find their way through the documentation (such as a table of contents)
 - **Rule templates**

Unit title

© Copyright IBM Corporation 2017

Figure 7-28. Rule documents

You can maintain your full list of rules in a rule document.

A rule document must explain how the rules are presented, including what rule classification was used, what the naming convention is, and how the document is structured.

Step 4: Authenticate the rules

- Validate that rules match business context
- Verify that rules guide the business behavior correctly
- Make sure that appropriate stakeholders are involved in approving or changing rules and that they are all in agreement
- Ensure traceability by connecting each rule to the policy that it implements

Unit title

© Copyright IBM Corporation 2017

Figure 7-29. Step 4: Authenticate the rules

Authentication is an aspect of governance. To authenticate a rule, you must be able to show the motivation that led to creating the rule. You must also be able to prove that the rule accurately corresponds to the business context.

How can you authenticate a rule?

- Make sure that the rules guide the business behavior correctly.
- Verify that the appropriate stakeholders are involved and in agreement about approving or changing rules.
- Ensure traceability by connecting each rule to the business policy it implements so that future policy updates can be reflected in the rules.

Step 5: Discover vocabulary

- Discovering vocabulary includes:
 - Identifying the objects that are referenced in the rules
 - Identifying the relationships between objects
 - Building the object model
- Discovery of data and rules is done in parallel, iteratively
- Object model evolves as rules are written

Unit title

© Copyright IBM Corporation 2017

Figure 7-30. Step 5: Discover vocabulary

As shown in Unit 2, "Modeling for business rules", discovering the vocabulary involves creating an object model that identifies the objects that are used in your rules, and the relationships between them. The object model evolves iteratively as more rules are written.

Step 6: Create test scenarios

- Create scenarios that help test the rules
 - Include use case scenarios from which rules were identified
 - Make sure to add missing scenarios
- Business analysts can design scenarios and provide them to IT for implementation
 - Establishing test conditions gives developers a good idea of what to code for testing

Unit title

© Copyright IBM Corporation 2017

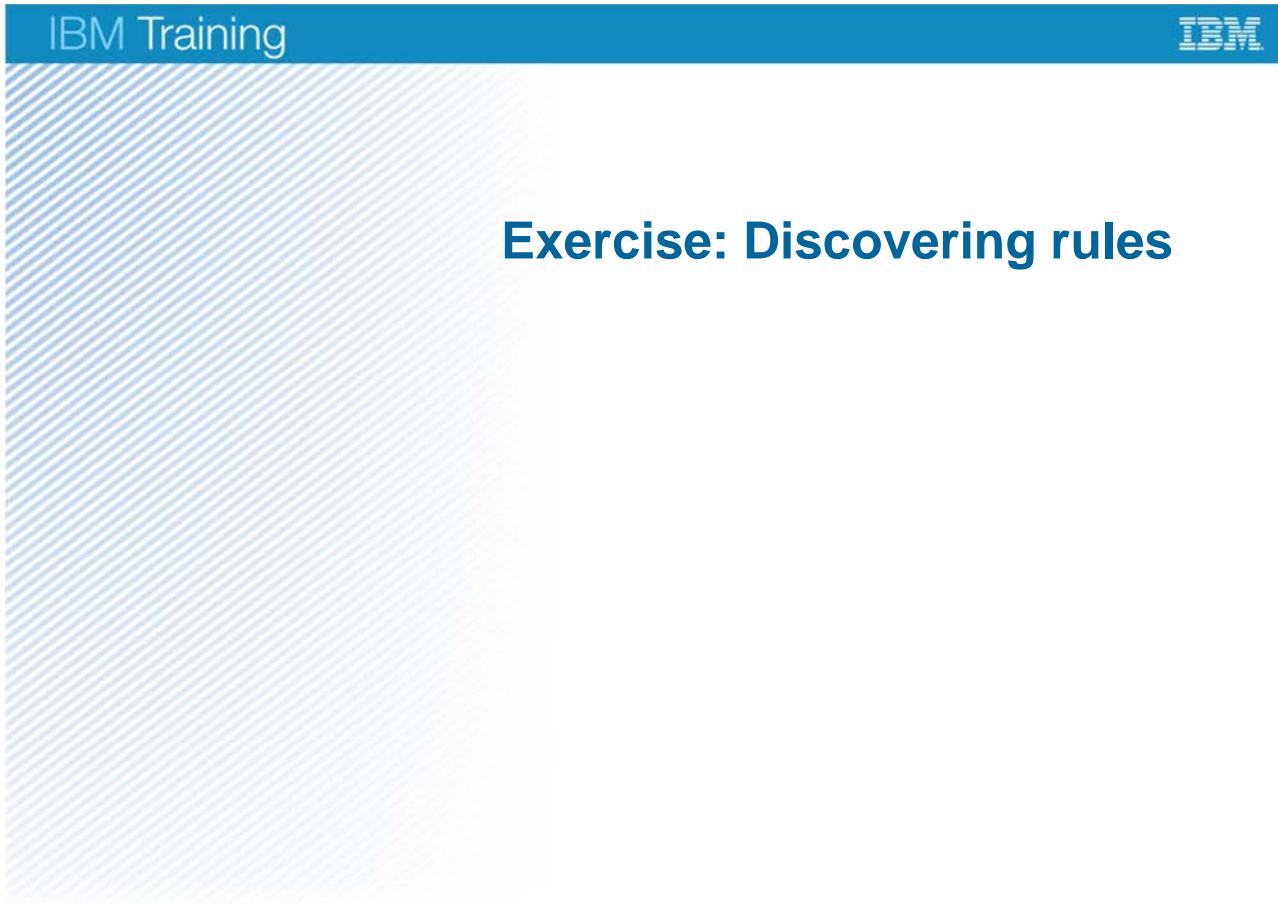
Figure 7-31. Step 6: Create test scenarios

After you know which rules are used at each use case step, you can create scenarios that test these rules. Scenarios are useful for validating the system and all of the rules.

As you analyze the use cases and inherent decisions, you might uncover new use cases that help you to identify what is missing, and design appropriate scenarios.

As an example, consider the problems that arise from borrowers who are granted loans that they cannot afford to repay. How can you create or change rules to ensure that loans are granted only to borrowers who can afford to make the payments? One way to test that your rules work properly is to use the information about borrowers who defaulted on their loans in the previous year as test data. Your tests can verify whether the new rules reject those borrowers.

Establishing test conditions up front allows developers to have a good idea of what to code for testing. The testing environment can be set up so that you can test your own rules.



Unit title

© Copyright IBM Corporation 2017

Figure 7-32. Exercise: Discovering rules

This exercise is based on the car insurance domain, and the scenario is provided for you in the *Understanding the Case Study* book.

The supporting files that you use during these exercises are available on the computer lab environment that is provided for this course. You might not have time to complete the exercise entirely, so after trying the exercises on your own first, see the solution files for a complete answer.

Exercise objectives

- Elicit decisions from use cases
- Transform business policy into formal rule statements

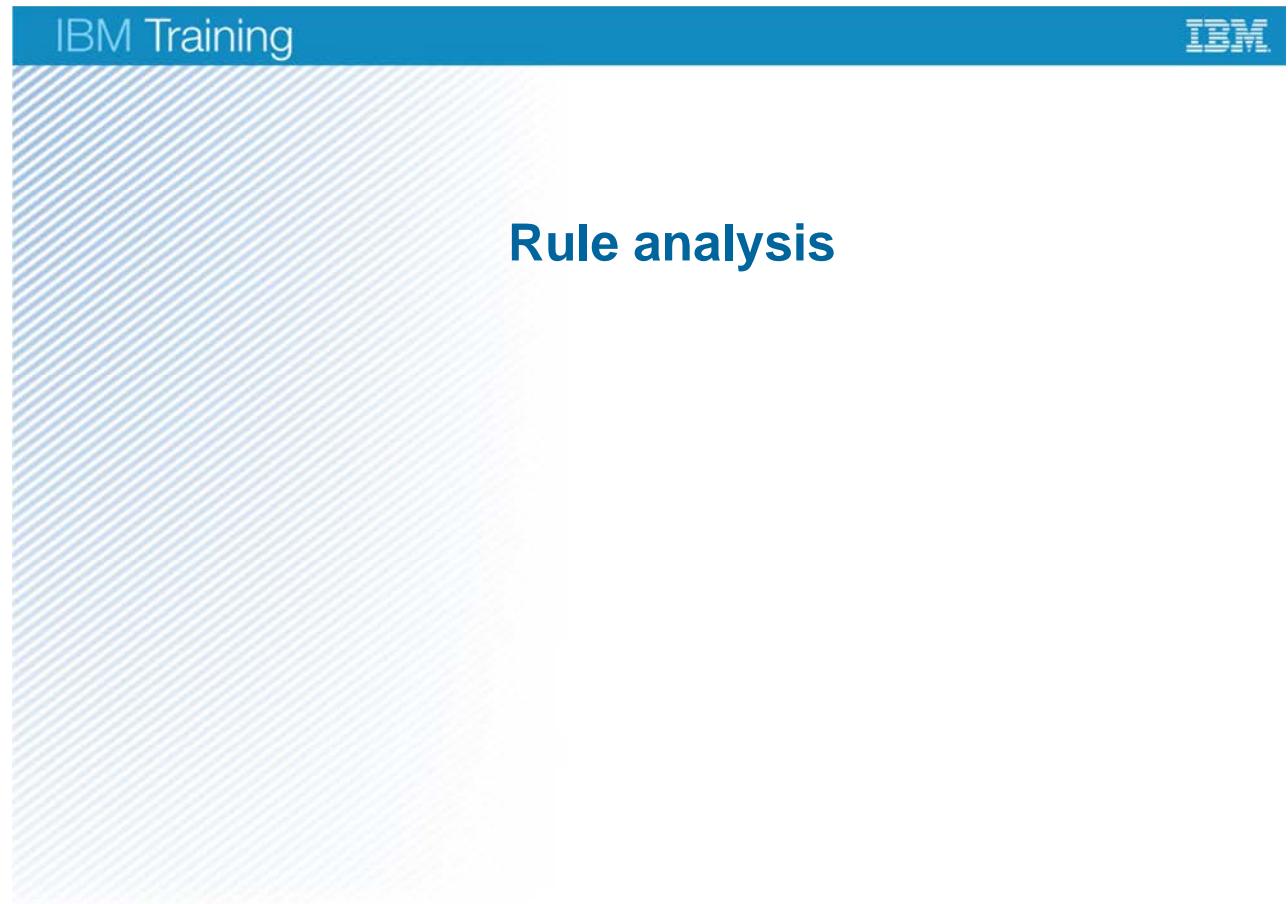


Unit title

© Copyright IBM Corporation 2017

Figure 7-33. Exercise objectives

7.4. Rule analysis



Unit title

© Copyright IBM Corporation 2017

Figure 7-34. Rule analysis

Rule analysis

- Combine the rules into a complete and coherent set
 - Does *not* involve validation, which is a separate process that involves business experts
 - Does *not* involve optimization or implementation, which are design issues
- Analysis tasks:
 1. Make each rule atomic
 2. Identify rule patterns
 3. Remove redundancy and overlaps
 4. Resolve inconsistency
 5. Ensure completeness
 6. Identify dependencies
 7. Refine the process
 8. Optimize the rules

Unit title

© Copyright IBM Corporation 2017

Figure 7-35. Rule analysis

After discovering the rules, the next goal is to make sure that the rules form a complete and coherent set of rules. Each rule must be analyzed both individually and as part of the set of rules.

Step 1: Make rules atomic

- Make sure that each rule has one result
 - Determine the smallest set of conditions you need to accomplish one rule
- Example: Business decision

The credit card that is used to guarantee a rental must belong to one of the authorized drivers, and this driver must sign the rental agreement. Other drivers must sign an “additional drivers” authorization form.

 - **Rule 1:** The credit card that is used to guarantee a rental must belong to one of the authorized drivers
 - **Rule 2:** And this driver must sign the rental agreement
 - **Rule 3:** Other drivers must sign an “additional drivers” authorization form

Unit title

© Copyright IBM Corporation 2017

Figure 7-36. Step 1: Make rules atomic

Making rules atomic means ensuring that each rule produces only one result. A business policy might encompass more than one statement of logic, and can be implemented that way in the rule editors. It is easier to analyze rules when they are in an atomic state.

The example business policy that is shown here can be broken into three rules. Although these three rules are part of a single business policy, breaking them up helps the analysis process.

To make a rule atomic, look for the smallest set of conditions to accomplish one result.

Step 1: Make rules atomic: Action enablers and inference rules

- Do not allow ANDs on the right side of the rule

- Break the rule into two rules

- IF A THEN do(B) AND do(C)

- => IF A THEN do(B)

- => IF A THEN do(C)

- Do not allow ORs on the left side of the rule

- Break the rule into two rules

- IF applicant is employed OR applicant is retired with indexed income, THEN approve loan

- => IF applicant is employed, THEN approve loan

- => IF applicant is retired with indexed income, THEN approve loan

Unit title

© Copyright IBM Corporation 2017

Figure 7-37. Step 1: Make rules atomic: Action enablers and inference rules

To make action enablers and inference rules atomic, look for multiple action statements as a result of the same condition. In other words, look for “AND” on the right side of the rule (or after the “then”). As written here, **IF A THEN do(B) AND do(C)** must become **IF A THEN do(B)** and **IF A THEN do(C)**.

Another clue that a rule is not atomic is when the rule has multiple conditions joined with “OR.” For example, **IF A OR B THEN do(C)** would become **IF A THEN do(B)** and **IF A THEN do(C)**.

Dividing the rule into two rules does not create a dependency or a contradiction, and in each example that is shown here, the new rules produce a single result.

Step 1: Make rules atomic: Constraints and guidelines

- Make sure that each rule contains only necessary conditions
- Do not allow multiple conditions (AND)
 - A driver must be 25 years old or older AND must have good credit rating
 - => A driver must be 25 years old or older
 - => A driver must have good credit rating
- When the condition contains an OR, do not break the rule
 - “A OR B must be true” is not equivalent to “A must be true” AND “B must be true”

Unit title

© Copyright IBM Corporation 2017

Figure 7-38. Step 1: Make rules atomic: Constraints and guidelines

Do not include AND in rules that are categorized as constraints or guidelines.

However, if the condition contains an OR, leave the rule as is. The statement “A or B must be true” is already atomic and is not the same as having two separate constraints: “A must be true” AND “B must be true.”

Step 2: Identify rule patterns

- Take advantage of emerging regularities to design simpler and domain-specific rule patterns
- Use tables to identify rules that are based on the same properties or reach the same conclusions
 - Include the rule ID, the property, and the conclusion
 - Example:

Rule ID	Driver.age	Rental status
R23	< 25	REJECT
R34	> 75	REJECT

Unit title

© Copyright IBM Corporation 2017

Figure 7-39. Step 2: Identify rule patterns

During discovery, the rules are captured from various sources and might seem disjointed. However, as you start analysis, patterns among the rules can emerge.

To capture rules that are based on the same properties or reach the same conclusions, you can organize them in tables and implement them as decision tables.



Information

The case study includes patterns among the rules. Later, when you start working on the exercises, take note of the patterns that are found among the eligibility rules that are captured as decision tables.

Step 3: Remove redundancy and overlaps

- Focus on rules of the same “family”
- A rule family is a group of rules that work with the same objects or values:
 - Constrain the same object (constraints)
 - Infer the same knowledge (inference rules)
 - Compute the same values (computation rules)
 - Enable the same action (action enablers)
- This step is simpler when rules are atomic

Unit title

© Copyright IBM Corporation 2017

Figure 7-40. Step 3: Remove redundancy and overlaps

When you look for redundancies, overlaps, and inconsistencies, focus on rules of the same family. A rule family is a group of rules that constrain the same object, infer the same knowledge, compute the same quantities, or enable the same action.

Step 3a: Remove redundancy

- Redundancies create a common data value, a common truth value, or initiate a common action
- Redundancy can occur when you have identical rules or when rules become equivalent through renaming or rewording
 - Changing the order of conditions:
IF A AND B THEN C is equivalent to **IF B AND A THEN C**
 - Using a negative form of the same rule:
IF A AND B THEN C is equivalent to **IF (NOT C) THEN (NOT A) OR (NOT B)**
- Duplications through transformation, such as:
 - Inversion of conditions
 - Other changes

Unit title

© Copyright IBM Corporation 2017

Figure 7-41. Step 3a: Remove redundancy

Redundancy is when you have identical rules. Watch for more subtle forms of redundancy that occur when rules become equivalent through renaming or rewording.

Step 3b: Remove overlaps

- Rule overlap occurs when rules share a common set of conditions or actions
- Overlaps or similarities that result in contradictions or other inconsistencies can be the result of:
 - Incomplete rules
 - Rules that are taken out of context
 - Misunderstanding of business rules
 - Contradictions in business rules
- To resolve overlaps between rules, look for what can be eliminated
 - Example:
 - Rule 1:** IF A AND B THEN C
 - Rule 2:** IF A THEN C
 - Since A results in C, B is not required as a condition
 - Eliminate **Rule 1**

Unit title

© Copyright IBM Corporation 2017

Figure 7-42. Step 3b: Remove overlaps

As you saw earlier, when you make rules atomic, you can end up with rules that share conditions or actions. To resolve overlaps between rules, try to recognize what can be eliminated.

To illustrate, consider this example.

A financial institution issues loans, and decides that an acceptable debt-to-income ratio on a loan application must be less than 33%. To submit a valid loan application, a potential borrower must have some income.



Note

A borrower can still have an income without being employed. If a borrower has no income, then the debt-to-income ratio does not apply, and their loan application data is rejected.

The following business rules cover the debt-to-income ratio for loan applications:

- IF the borrower is employed AND has a debt-to-income ratio < 0.33 THEN approve loan
- IF the borrower has a debt-to-income ratio < 0.33 THEN approve loan

In this case, you can ignore the employed condition, since debt-to-income ratio determines the result in both rules.

Step 4: Resolve inconsistency

- Semantically equivalent conditions with contradictory conclusions
 - Example:
Rule 1: IF A THEN B
Rule 2: IF A THEN NOT(B)
 - Conflicting rules, probably coming from two sources
 - Necessary conditions in either rule are missing
- Equivalent conclusions that are derived from contradictory conditions
 - Example:
Rule 1: IF A THEN B
Rule 2: IF NOT (A) THEN B
 - The condition is not relevant to the conclusion

Unit title

© Copyright IBM Corporation 2017

Figure 7-43. Step 4: Resolve inconsistency

Inconsistencies occur when you have semantically equivalent conditions with contradictory conclusions, or when you have rules with semantically equivalent conclusions that result from contradictory conditions.

To resolve inconsistencies, look for:

- Rules that have semantically equivalent conditions but lead to contradictory conclusions
 - You might find contradictions among rules that are discovered from two different sources, or that are based on incomplete information.
- Rules that have semantically equivalent conclusions but result from contradictory conditions
 - When you find rules that lead to the same results, make sure that the conditions are accurate. You might find that some conditions are no longer relevant.

Step 5: Ensure completeness among rules

- Make sure that rules cover all possible situations, with no gaps or loopholes
- Make sure all derived or calculated properties in the object model have corresponding computation or inference rules
- Use concrete scenarios to identify missing or unhandled cases
 - Example:
Loans for a value greater than \$10,000 should be approved by the branch manager
 - Who approves loans of value less than \$10,000?

Unit title

© Copyright IBM Corporation 2017

Figure 7-44. Step 5: Ensure completeness among rules

To ensure completeness, make sure that the rules cover all situations with no gaps or loopholes.

All derived or calculated properties in the object model must have corresponding computation or inference rules. For example, if a “status” property is inferred from the amount that a customer spends, make sure that there is a rule to assign a value to “status.”

Completeness is easier to verify when the rules follow simple patterns, like decision tables.

Otherwise, you must ask what-if questions to ensure that no business events are overlooked. For example, consider a rule that says:

Loan amounts over \$10,000 should be approved by the branch manager

This rule does not specify who can approve loans of value less than \$10,000. Concrete scenarios can help identify missing or unhandled cases.

Step 6: Identify dependencies

- Dependency exists between rules when one rule triggers another by creating a condition that makes the other rule applicable
- Recognizing dependencies helps determine the rule execution sequence
- Example:
 - R1 creates data
 - R2 tests the data that R1 created
 - R2 depends on R1, so the execution sequence is:
 1. Run R1
 2. Run R2

Unit title

© Copyright IBM Corporation 2017

Figure 7-45. Step 6: Identify dependencies

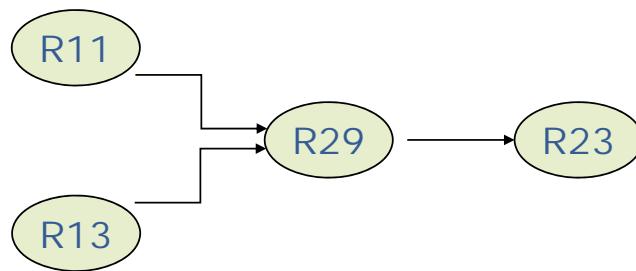
Dependency exists between rules when one rule triggers another by creating a condition that makes the other rule applicable. For example, if Rule A computes a price and Rule B states that the price must be below some threshold, then Rule B depends on Rule A.

To look for rule dependencies, consider all of the rules together, focusing on rules that define and reference the same objects or properties. After you identify rule dependencies, you must validate the process within which the rules are invoked. If Rule B depends on Rule A, then Rule A must be evaluated before Rule B.

Identifying dependencies helps you with the next step of refining the process.

Example: Rule dependency table

Rule ID	Classification	Description	Depends on
R23	Computation	Computes rental cost	R29
R29	Inference	Sets the appropriate discount status	R11, R13
R32	Constraint	Rejects drivers younger than 25	



Unit title

© Copyright IBM Corporation 2017

Figure 7-46. Example: Rule dependency table

The example here shows how you can capture dependencies in a table or graphically.

In this example, Rule 23, which computes rental cost, depends on Rule 29, which determines the discount. Therefore, the discount rule (R29) must be evaluated before computing the cost (R23).

Step 7: Refine process

- To validate rule dependencies against the process:
 - Ensure execution flows according to dependencies
 - Ensure that the execution sequence does not contradict the process
- Contradictions might require redesigning the rules or the ruleflow

Unit title

© Copyright IBM Corporation 2017

Figure 7-47. Step 7: Refine process

After rule dependencies are identified, the next step is to compare those dependencies with the process flow to see whether there are any inconsistencies.

If you find contradictions between the rule execution and the process, you might need to redesign the rules or change the ruleflow.

Step 8: Optimize the rules

- Review policies and rule parameters to assess whether they help attain your business objectives
- If changes to the rules are required:
 - Analyze the effect of such changes on the business
 - Publish both the change and its effect on the business
- Design governance processes to support this step

Unit title

© Copyright IBM Corporation 2017

Figure 7-48. Step 8: Optimize the rules

The goal of optimizing rules is to review policies and rule parameters to assess whether they help attain your business objectives.

Business policies and threshold values require regular assessment for their business relevance, and are likely to change or be updated to meet current business or market needs.

Make sure that some governance processes are in place to support this step to ensure that both policy or rule changes, and the effect of those changes, are documented.

Governance is explained in the last unit and exercise of this course.

Unit summary

- Use good practices to discover rules
- Analyze rules to eliminate problems before implementation

Unit title

© Copyright IBM Corporation 2017

Figure 7-49. Unit summary

Review questions

1. True or False: Rule discovery involves identifying decisions and formalizing those decisions as rule statements.
2. Analyzing the business rules that are documented during rule discovery involves which of the following tasks?
Select all that apply:
 - A. Ensuring that rules are atomic
 - B. Removing redundancies
 - C. Identifying dependencies among rules
 - D. Synchronizing with Rule Designer



Unit title

© Copyright IBM Corporation 2017

Figure 7-50. Review questions

Write your answers here:

1.

2.

Review answers

1. True or False: Rule discovery involves identifying decisions and formalizing those decisions as rule statements.

The answer is True.

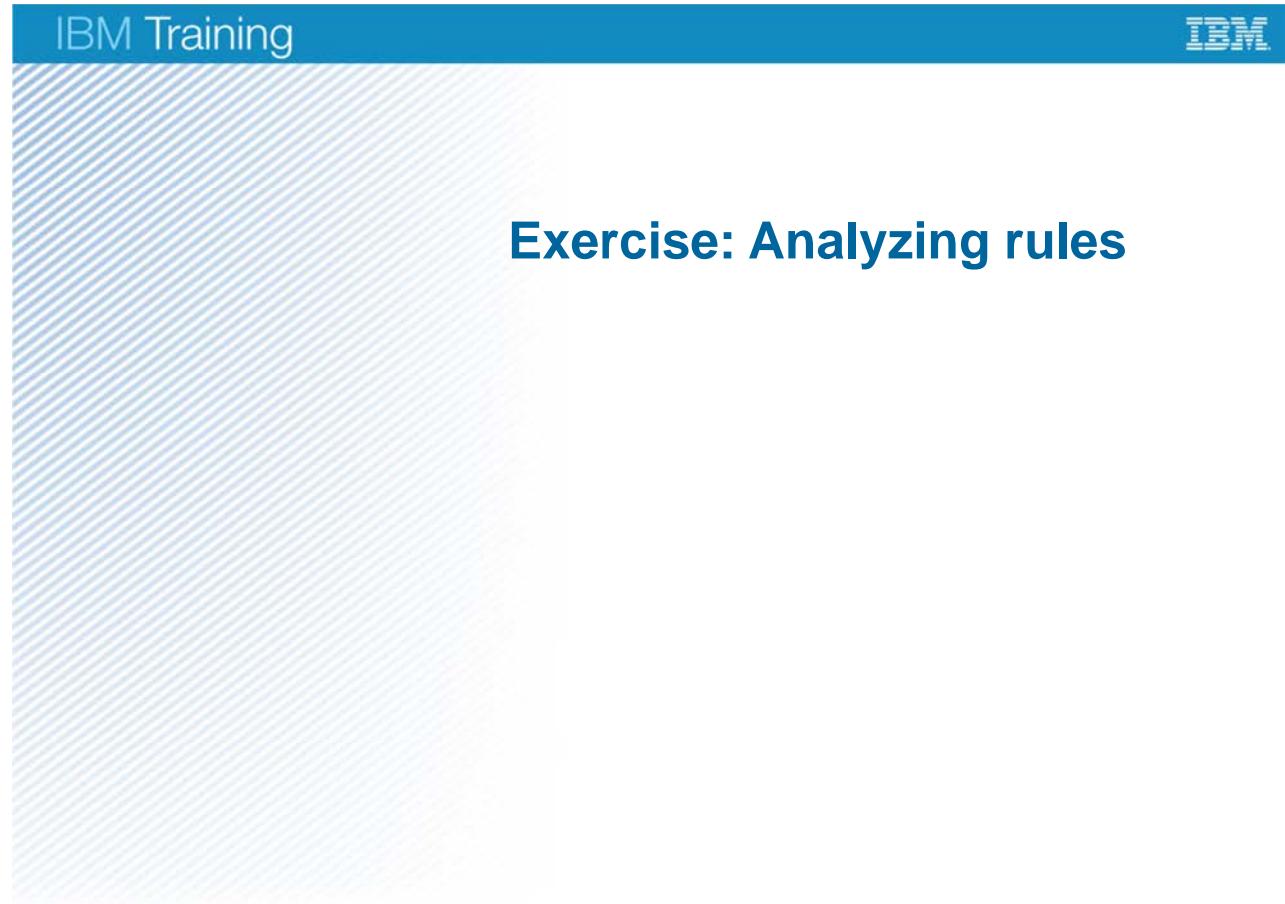


2. Analyzing the business rules that are documented during rule discovery involves which of the following tasks?

Select all that apply:

- A. Ensuring that rules are atomic
- B. Removing redundancies
- C. Identifying dependencies among rules
- D. Synchronizing with Rule Designer

The answer is A, B, and C. During the discovery and analysis phase, rules are still on paper, not authored in the tool, so synchronization is not possible.



Unit title

© Copyright IBM Corporation 2017

Figure 7-52. Exercise: Analyzing rules

These exercises continue with the car insurance case study, and you now do analysis on the rules that are discovered in the first set of exercises.

Supporting files for these exercises are found in the <labfilesDir>\CaseStudyExercises directory.

Exercise objectives

- Make rules atomic
- Eliminate overlap, redundancy, and inconsistency
- Identify dependencies between individual rules and rule families



Unit title

© Copyright IBM Corporation 2017

Figure 7-53. Exercise objectives

Unit 8. Working with conditions in rules

Estimated time

01:00

Overview

This unit continues the focus on rule authoring. You learn about various types of condition statements and the Business Action Language (BAL) constructs that help build them. You also learn about the relationship between the BOM and the vocabulary available in the rule editors.

How you will check your progress

- Checkpoint
- Exercise

Unit objectives

- Identify basic types of conditions and give examples
- Determine which BAL constructs to use for various types of condition statements
- Describe how to use Boolean logic in condition statements
- Describe the relationship between rule vocabulary in the rule editors and the BOM

Unit title

© Copyright IBM Corporation 2017

Figure 8-1. Unit objectives

This unit covers different types of conditions statements and the Business Action Language (BAL) constructs that help you build them. It helps you see the relationship between the contents of vocabulary lists and the business object model.

Topics

- Rule structure review
- Rule condition examples
- Working with BAL constructs to build condition statements
- Using Boolean logic in rules
- Writing condition statements in the Business console

Unit title

© Copyright IBM Corporation 2017

Figure 8-2. Topics

8.1. Rule structure review

Rule structure review

Unit title

© Copyright IBM Corporation 2017

Figure 8-3. Rule structure review

Parts of a rule (rule structure)

Definitions	definitions	Define rule-specific variables (optional)
Conditions	if	Define the conditions that determine when a rule is executed • Condition statements are optional
Actions	then	Define the actions that are performed when all conditions are met • All rules must contain at least one action
	else	Defines an alternative action when conditions are not met • The else part is optional when a condition is supplied • It cannot be included when the rule does not have conditions

Unit title

© Copyright IBM Corporation 2017

Figure 8-4. Parts of a rule (rule structure)

This unit focuses on writing one of the central parts of a business rule: conditions. However, it is important to think of condition statements in the context of the complete rule, so this topic provides a review of rule structure.

As you learned earlier, rules have three main parts: definitions, conditions, and actions. They can also have an **else** statement.

You use the **definitions** part of the rule to define variables that you can use in other parts of the rule. Definitions are optional. Variables are shown in vocabulary lists in the editor, where they can be selected. They can simplify the phrasing of the rule statements.

Conditions start with **if** and determine when a rule is executed.

Conditions always have an answer of true or false. In other words, a condition (or set of conditions) is either met, or is not met. Conditions are not required. When the rule does not include conditions, the actions always apply.

Actions start with **then** and specify what should happen, or what actions to take when conditions are met. Every rule must have at least one action.

An **else** statement can be used to define an alternative action when conditions are *not* met. The **else** statement is optional when you have a rule condition. If the rule does not have conditions, you cannot have an **else** statement.

8.2. Rule condition examples

Rule condition examples

Unit title

© Copyright IBM Corporation 2017

Figure 8-5. Rule condition examples

Rule condition examples (1 of 3)

What does a condition do?

- Defines the test criteria that determine when a rule is executed
- Results in a “true” or “false” response
 - The rule actions are performed when the condition test result is “true”
- Example:


```
if
    the category of the customer is GOLD
then
    set the discount rate to 10%
```

 - A discount is given only when the condition test is true
- When defining conditions, ask yourself:
 - Under what circumstances should this action be executed?
 - What set of objects does this rule evaluate?

Unit title

© Copyright IBM Corporation 2017

Figure 8-6. Rule condition examples (1 of 3)

Conditions define tests that determine when a rule is executed. The condition test is either met or not met, so the condition that results is either true or false.

Conditions determine whether rule actions are triggered. When the condition test result is true, the action, or the **then** part of the rule, is executed. When the condition is false, the action is ignored. Or, if the action includes an **else** statement, then a false condition triggers the **else** part of the rule.

When identifying the condition, you can ask yourself questions such as:

- What set of objects will this rule act on?
- Under what circumstances should this action be executed?

In the simple example here, the rule evaluates the customer object. The circumstances in which the action should be executed occur when the category of a customer object is gold.

Rule condition examples (2 of 3)

- A condition can include logical combinations of simple conditions by using Boolean expressions:
 - AND
 - OR
- Example that uses AND

```

if
  the ownership of 'the vehicle' is one of { Financed by credit
  , Leased }
  and the type of 'the policy' is ThirdParty
then
  refuse the application because "Collision insurance is
  compulsory if the vehicle is not owned (financed or leased)"
;
```

Unit title

© Copyright IBM Corporation 2017

Figure 8-7. Rule condition examples (2 of 3)

A condition can include logical combinations of simple conditions by using ANDs and ORs, which is also called Boolean logic. Boolean terms are integral to rule authoring.

The example on this slide uses the Boolean term AND to build a condition statement. This condition statement tests the following conditions:

- Ownership of the vehicle: Is the vehicle financed, leased, or owned?
- Policy type: Is the policy type `ThirdParty`?

You learn more about Boolean logic later in this unit.

Rule condition examples (3 of 3)

- For rules that have multiple conditions, you can use:

- any of the following** (compound OR)
- all of the following** (compound AND)
- none of the following** (negative compound AND)

- Example: Compound AND

```

if
    all of the following conditions are true
        the ownership of 'the vehicle' is one of { Financed      by
            credit, Leased }
        the type of 'the policy' is ThirdParty
then
    refuse the application because "Collision insurance is
    compulsory if the vehicle is not owned (financed or leased)"

```

Unit title

© Copyright IBM Corporation 2017

Figure 8-8. Rule condition examples (3 of 3)

Rules with multiple conditions can be written with these BAL constructs:

- any of the following
 - Equivalent to a compound OR
- all of the following
 - Equivalent to a compound AND
- none of the following
 - Equivalent to a negative compound AND

8.3. Working with BAL constructs to build condition statements

Working with BAL constructs to build condition statements

Unit title

© Copyright IBM Corporation 2017

Figure 8-9. Working with BAL constructs to build condition statements

Business Action Language (BAL) constructs and operators

- Business Action Language (BAL) constructs and operators
 - Special, predefined words and phrases available in the rule editors
 - Combine with your business vocabulary

Unit title

© Copyright IBM Corporation 2017

Figure 8-10. Business Action Language (BAL) constructs and operators

Operational Decision Manager uses Business Action Language (BAL), which includes special, predefined words and phrases to formulate rules in the rule editors. These BAL constructs are like building blocks that you combine with the vocabulary for your business, which is built into the business object model. Together, the BAL constructs and your vocabulary can express rules as natural-language statements.

The complete reference for BAL constructs is available in the product documentation.



Reminder

When you work with Rule Designer, you might see references to IRL, or ILOG Rule Language. As you might remember from the discussion of IRL in Unit 6, "Introducing rule authoring", IRL contains a set of keywords, and has its own syntax to structure each part of the rule.

As a business analyst, you author rules primarily in BAL, which is automatically translated into IRL.

For more information about BAL constructs, see the product documentation.

Types of conditions

- The most basic conditions can:
 - Make a comparison
 - if the age of the driver is at least 18
 - Count
 - if the number of drivers in the request is more than 6
 - Check the existence of an object (special type of count)
 - if there is at least one driver
 - Evaluate set membership (test whether an object belongs to a set)
 - if the ownership of the vehicle is one of { Financed by credit, Leased }

Unit title

© Copyright IBM Corporation 2017

Figure 8-11. Types of conditions

There are a few basic conditions, which include:

- Making a comparison.
- Counting.
- Checking the existence of an object.
- Evaluating whether an object belongs to a set.

The following slides provide more details about condition types.

Comparison conditions

- Compares or establishes relationships between different terms that are found in rule statements
- Example: Compare the age of the driver with the value “18”
if the age of the driver is at least 18
- What other examples can you think of?
- For reference:
 - In your Exercise Guide, turn to the “Writing condition statements” exercise
 - Make notes in the section: BAL condition constructs

Unit title

© Copyright IBM Corporation 2017

Figure 8-12. Comparison conditions

Comparison conditions are used to compare or establish relationships between the different terms in rule statements. For example:

if the value of the shopping cart is more than 100

This condition statement compares the purchase value (“the value of the shopping cart”) to a specific value (100). The statement uses a “is more than” as the BAL comparison operator. BAL includes various operators to compare text, numbers, dates, and lists.



Important

Some BAL constructs are listed in your Exercise Guide.

- Open [Section 2, "BAL condition constructs,"](#) on page 10-5 of [Exercise 10, "Working with conditions in rules".](#)
- Take some time to think about examples that you might use in your business rules, and use the space that is provided to make notes.

Count conditions

- Counts the number of occurrences of something
- Example: Count the number of occurrences of car and compares it to the value “6”
if the number of drivers in the request is more than 6
- What other examples can you think of?
- For reference:
 - In your Exercise Guide, turn to the “Writing condition statements” exercise
 - Make notes in the section: BAL condition constructs

Unit title

© Copyright IBM Corporation 2017

Figure 8-13. Count conditions

Count conditions count the number of occurrences of something.

For example:

if the number of drivers in the request is more than 6

This statement counts the number of objects (in this case, “drivers”) and compares that to a value (6).

Notice that there is both a count (“the number of”) and a comparison (“is more than”) in this statement.

Existence conditions

- Tests whether an object is present among the set of objects that are passed to the rule engine
- Examples:
 - if there is at least one customer
 - if there are at least 3 items in the items of the shopping cart
 - if there is at least one item in the list of items in the shopping cart
 - if there is at least one item in the list of items in the shopping cart
where the price of this item is more than 1000
- For reference:
 - In your Exercise Guide, turn to the “Writing condition statements” exercise
 - Make notes in the section: BAL condition constructs

Unit title

© Copyright IBM Corporation 2017

Figure 8-14. Existence conditions

Condition statements that test for existence are a subset of count conditions. You use them to test for the existence of one or more items (a subset) within a set of objects (the count). You test whether something “is there” or is included in the set. For example:

Is there a customer of a particular type?

In the condition *if there is at least one customer*, you are testing whether there is at least one customer among all the customers that matches the condition. The construct “there is at least one” is a building block that tests for existence.

The shopping-cart existence tests evaluate the set of objects that are provided to the rule engine to see whether the shopping cart has three items. You might use this type of rule to give a discount on orders that include more than three items. In this case, the condition can be written this way:

```
if there are at least 3 items in the items of the shopping cart
```

The phrase “items of the shopping cart” is a reference to a collection.

If you are writing rules for an application that validates orders, the application is not interested in processing an order that has no items in the shopping cart. So you can write a condition to test that uses “there is at least one item” to filter orders, as shown in the example on this slide:

```
if there is at least one item in the list of items in the shopping cart
```

Sometimes, existence tests use a **where** clause:

if there is at least one item in the list of items in the shopping cart where
the price of this item is more than 1000

The **where** clause acts as a filter and tests whether a specific object exists in the set of objects.

Evaluate set membership conditions

- Tests whether an object belongs to a set
- Example
 - When the customer is in the Silver, Gold, or Platinum category, but not in the Normal category:

```
if the classification of the vehicle is one of
{ High-end luxury , Ultra luxury }

if the ownership of the vehicle is one of
{ Financed by credit, Leased}
```

- What other examples can you think of?
- For reference:
 - In your Exercise Guide, turn to the “Writing condition statements” exercise
 - Make notes in the section: BAL condition constructs

Unit title

© Copyright IBM Corporation 2017

Figure 8-15. Evaluate set membership conditions

When you want to test whether an object belongs to a set, you use an “evaluate set membership” building block.

For example, you might have a set of customer categories: Normal, Silver, Gold, or Platinum. You can test to see whether a customer is assigned to one of these categories so that you can apply discounts or upgrades for that category.

A condition statement for that test can be written this way:

```
if the customer category is one of {Silver, Gold, Platinum}
```

The “is one of” building block can be used to test whether an object is a member of a set.

Negative conditions

- Conditions can be negated by using the “if it is not true” statement
- Example: *The driver is not eligible because of age*
 if it is not true that the age (in years) of the driver
 at the start date of the policy is between 18 and 80
 - In this case, negation simplifies the condition statement
 - Otherwise, this statement would be either a disjunction (or two rules) with these conditions:
 the age of the driver is more than 18
 AND
 the age of the driver is less than 80
- What other examples can you think of?

Unit title

© Copyright IBM Corporation 2017

Figure 8-16. Negative conditions

Many Boolean (true or false) conditions are written with a positive scenario in mind. When the vocabulary has no phrase to express the opposite of a condition, you can use the “if it is not true” construct to negate the condition.

In general, the logic of a negative expression is more difficult to interpret, so it is advisable to avoid them where possible.

BAL number operators

BAL construct	Operator
Is more than	>
Is at least	\geq
Is less than	<
Is at most	\leq
Equals	=
Does not equal	\neq
Is between <number> and <number> • Includes endpoints	[,]
Is strictly between <number> and <number> • Excludes endpoints] , [
Is at least <number> and less than <number>	[, [
Is more than <number> and at most <number>] ,]

Unit title

© Copyright IBM Corporation 2017

Figure 8-17. BAL number operators

Number operators specify numerical comparison relationships. You can also use the plus sign (+) to concatenate strings. For example:

```
refuse insurance because "Driver " + the first name of driver + " " + the last
name of driver + " has had too many accidents";
```

In addition to the standard arithmetic operators, such as +, -, *, and /, a range of number operators are expressed in BAL constructs, as listed here.

BAL arithmetic operators

BAL construct	Operator
<number> + <number>	Adds a number to another number
<number> - <number>	Subtracts a number from another number
<number> / <number>	Divides a number by another number
<number> * <number>	Multiplies a number by another number
<text> + <text>	Concatenates two strings Example: set 'full name' to 'first name' + ' ' + 'last name'

Unit title

© Copyright IBM Corporation 2017

Figure 8-18. BAL arithmetic operators

This table lists BAL arithmetic operators and how they are used.

Discussion

Working with conditions in rules: Identifying conditions

Unit title

© Copyright IBM Corporation 2017

Figure 8-19. Discussion

This discussion is based on [Exercise 10, "Working with conditions in rules"](#).

The next slide contains instructions for completing this activity.

For classroom and online students, be prepared to discuss your work with the rest of the class.

For self-paced students, you can complete this activity as an independent exercise by following the instructions.

Discussion: Working with conditions in rules

- This discussion is based on Exercise 10, “Working with conditions in rules”
- In the exercise guide, follow the instructions in Section 2, “Identifying conditions”

Unit title

© Copyright IBM Corporation 2017

Figure 8-20. Discussion: Working with conditions in rules

For classroom or online students, discuss your results with the class after you finish working through the instructions.

If you are a self-paced student, take time to complete the exercise instructions, and write down your ideas.

1. In your Exercise Guide, turn to the exercise called [Exercise 10, "Working with conditions in rules"](#).
2. Follow the instructions in [Section 3, "Identifying conditions,"](#) on page 10-11.

8.4. Using Boolean logic in rules

Using Boolean logic in rules

Unit title

© Copyright IBM Corporation 2017

Figure 8-21. Using Boolean logic in rules

Boolean logic in rules: ANDs, ORs, and NOTs

- Central to rule authoring
- Remember: Conditions are either *true* or *not true*
- Can help phrase rules in a way that makes sense to reviewers
- The next few slides review basics
 - The meaning of OR
 - The meaning of AND
 - Mixing ANDs and ORs
 - Negating conditions
 - Stating that something is true for all Xs

Unit title

© Copyright IBM Corporation 2017

Figure 8-22. Boolean logic in rules: ANDs, ORs, and NOTs

Boolean logic, which involves the use of ANDs, ORs, and NOTs, is central to writing rules. It is important to remember that conditions are either true or *not true*. Keeping that in mind can help you to phrase rules in a way that makes sense to people who see them later.

Using Boolean operators helps you to create statements that resolve logically.

The meaning of OR

- Mutually exclusive: Only one option can be true
 - Example: A driver can be EITHER male OR female
- Inclusive: More than one option can be true
 - In BAL, the OR is always inclusive
- Example: EITHER the primary driver had an accident OR one of the secondary drivers had an accident
 - This condition is “true” if the primary driver OR any of the secondary drivers had an accident

Primary driver had an accident	Secondary driver had an accident	(primary driver had an accident) OR (secondary driver had an accident)
True	True	True
True	False	True
False	True	True
False	False	False

Unit title

© Copyright IBM Corporation 2017

Figure 8-23. The meaning of OR

It is possible to use the Boolean OR in two ways:

- Mutually exclusive, where only one option can be true.
- Inclusive, where more than one option can be true.

In BAL, the OR is always used inclusively.

The meaning of AND

- Example policy:

To be eligible for coverage, a vehicle must meet these criteria:

- The age of the vehicle must be less than 40 years
AND
- The vehicle must be for personal use only

- Under what conditions would a car be eligible?

Individual			Age < 40	Personal use only	Eligible
Name	Age	Usage			
Jane	2	Personal	True	True	True
Martin	1	Personal and commercial	True	False	False
Melissa	60	Personal	False	True	False
Mark	40	Commercial	False	False	False

Unit title

© Copyright IBM Corporation 2017

Figure 8-24. The meaning of AND

In BAL, the Boolean operator AND is always additive. Both conditions must test true for the condition to be met.

Mixing ANDs and ORs

- Equivalent expressions

- A vehicle must be less than 40 years old **AND** (**either** be used for personal use **OR** be used for business travel)

Is the same as:

A vehicle must be either (less than 40 **AND** used for personal use) **OR** (less than 40 **AND** be used for business travel)

- A AND (B OR C)**

Is the same as:

(A AND B) OR (A AND C)

Unit title

© Copyright IBM Corporation 2017

Figure 8-25. Mixing ANDs and ORs

You can combine ANDs and ORs in statements. Such statements might require that you insert parentheses to clarify your meaning.

The two statements on this slide are both equivalent expressions.



Note

Business travel includes commuting to work sites other than your regular place of work, and it can be covered on a personal insurance policy.

Business travel does not include commercial use of the vehicle for delivery or selling from door to door. Commercial use of a vehicle requires a commercial insurance policy.

Negating conditions

- Consider this condition:
 - the age (in years) of the driver at the start date of the policy **is at least 18**

Negating this condition means:

- the age (in years) of the driver at the start date of the policy **is less than 18**
- When no convenient operator exists to express the opposite of a condition, use ***it is not true that*** to express the Boolean NOT
- Example: *Test whether the driver is a secondary or co-driver*
 - No vocabulary term is directly available to represent “co-driver” so you use a negation with “primary driver”
 - Negative condition:
it is not true that the driver is a primary driver

Unit title

© Copyright IBM Corporation 2017

Figure 8-26. Negating conditions

Some conditions must be written as negative statements to evaluate whether the condition is true.

For example, consider this condition:

the driver's age is at least 18

The opposite or negative of that condition can be written as:

the driver's age is less than 18

When no operator or phrase exists to express the negative or opposite of a condition, you can use “it is not true that” statements in front of the condition.

Negating AND (1 of 2)

- Consider these conditions that are combined with an AND operator:


```
if
        the age ( in years ) of the vehicle at the start date of
        the policy is less than 40
      AND the usage of the vehicle is Personal
```
- Failing this condition means:


```
it is not true that (the age ( in years ) of the vehicle
        at the start date of the policy is less than 40
      AND the usage of the vehicle is Personal )
```
- This statement means:
 - Either the age of the vehicle is NOT less than 40
 - OR the usage of the vehicle is NOT personal
 - OR both
- NOT (A AND B) is the same as NOT (A) OR NOT (B)

Unit title

© Copyright IBM Corporation 2017

Figure 8-27. Negating AND (1 of 2)

In the example on the slide, the two conditions, or the age of the vehicle in years and the usage category of the vehicle as Personal, are combined with an AND operator.

In this example:

- If either of the conditions are false, the conditions fail.
- If both of the conditions are false, the conditions also fail.

The next slide provides further examples of these conditions.

Negating AND (2 of 2)

- Not (A AND B) is the same as NOT (A) OR NOT (B)

Individual			Age < 40	Personal use only	Eligible	Not eligible
Name	Age	Usage				
Jane	2	Personal	True	True	True	False
Martin	1	Personal and commercial	True	False	False	True
Melissa	60	Personal	False	True	False	True
Mark	40	Commercial	False	False	False	True

Unit title

© Copyright IBM Corporation 2017

Figure 8-28. Negating AND (2 of 2)

This table demonstrates the previous negation example.

- If the vehicle age condition or the Personal usage category condition is false, the driver is ineligible: NOT (A) OR NOT (B).
- If both the vehicle age condition and the Personal usage category condition are false, the driver is ineligible: NOT (A AND B).

Negating OR

- Consider this policy:
 - *Either the primary driver had an accident OR one of the secondary drivers had an accident*
- The negation of the policy becomes:
 - *Neither the primary driver NOR the secondary drivers had an accident*
 - *The primary driver did NOT have an accident AND the secondary drivers did NOT have an accident*
- NOT (A OR B) is the same as NOT (A) AND NOT (B)

Unit title

© Copyright IBM Corporation 2017

Figure 8-29. Negating OR

Negating “there is at least one”

- Consider this condition:
 - Business policy: *At least one of the drivers is a primary driver*
 - Condition statement:
there is at least one driver where this driver is a primary driver
- To negate the policy, you rephrase it as:
 - *None of the drivers needs to be a primary driver*
- The negative condition can be expressed in two ways:
 - it is not true that (there is at least one driver where this driver is a primary driver)
 - there is no driver where this driver is a primary driver
- In general, the negation of “there is at least one X where Y is true” is expressed as:
There is no X where Y is true

Unit title

© Copyright IBM Corporation 2017

Figure 8-30. Negating “there is at least one”

When you must negate a condition that includes “there is at least one,” you do it differently.

If you have a business policy that says:

at least one of the drivers needs to be a primary driver

The condition is phrased as:

there is at least one driver where this driver is a primary driver

To negate the original business policy, you might say: *None of the drivers must be a primary driver*.

You can write that condition in two ways:

it is not true that (there is at least one driver where this driver is a primary driver)

there is no driver where this driver is a primary driver

The second statement is easier to read.

Stating that something is true for all Xs

- Consider this business policy:
ALL drivers must have less than three accidents
- This policy is equivalent to saying:
NONE of the drivers had 3 or more accidents
Or
NONE of the drivers have NOT had fewer than 3 accidents
- As a formal rule statement, you can formulate this rule with BAL as:
there is no driver where it is not true that the number accidents of this driver is less than 3
- In general:
ALL the Xs are such that <condition>
Is equivalent to:
NONE of the Xs is such that NOT <condition>
Formulated in BAL as:
there is no X where it is not true that <condition>

Unit title

© Copyright IBM Corporation 2017

Figure 8-31. Stating that something is true for all Xs

Creating a statement that says that something is true for all occurrences can be tricky.

Generally, you would say:

All the Xs are such that a condition applies.

That statement has the same meaning as:

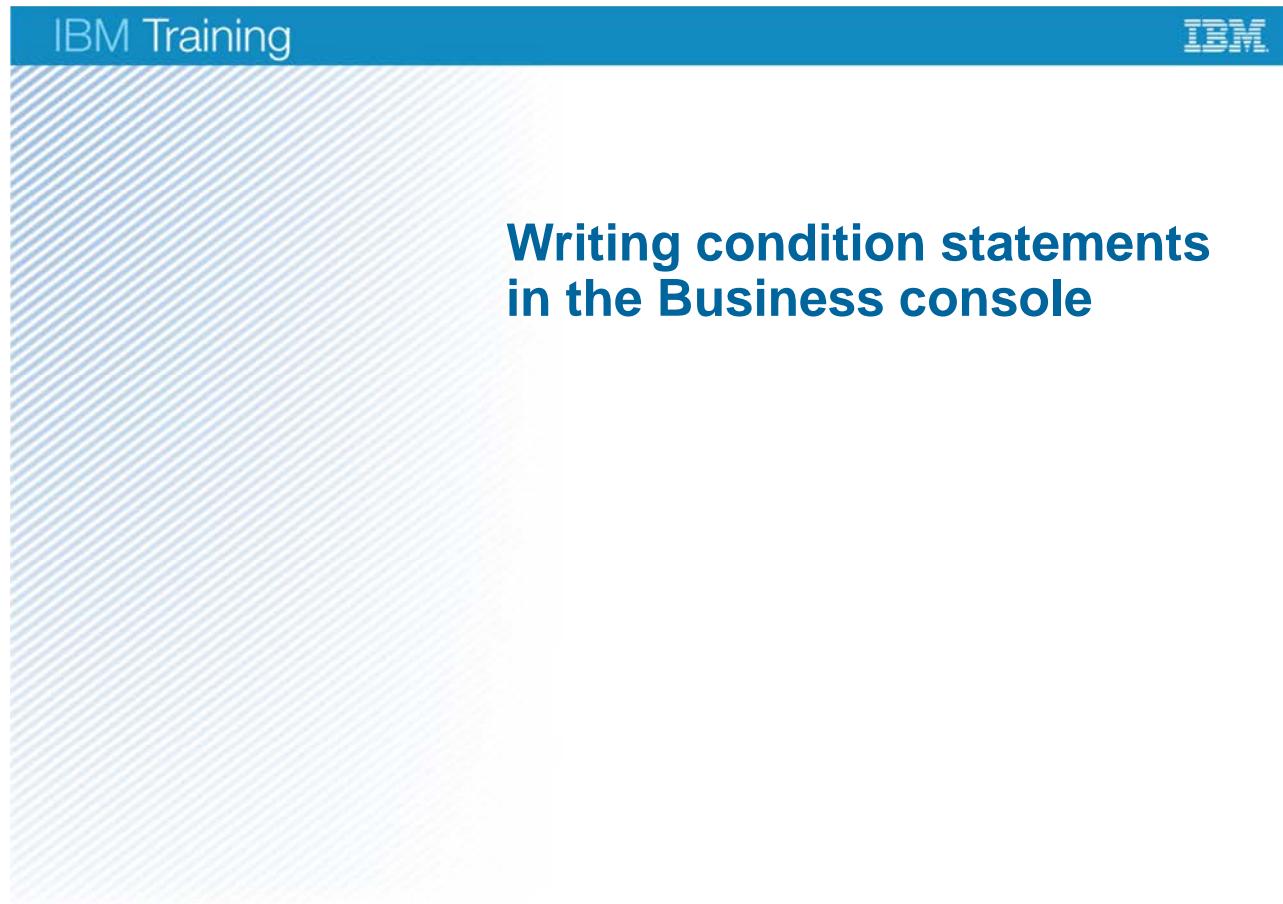
NONE of the Xs is such that the condition does not apply.

In a rule statement, you would say:

there is no X where it is not true that the condition applies

The keys are to use “there is no” before the name of the object, and “it is not true that” before the condition.

8.5. Writing condition statements in the Business console



Unit title

© Copyright IBM Corporation 2017

Figure 8-32. Writing condition statements in the Business console



Review: BOM relationships in vocabulary lists (1 of 2)

- The ruleset parameters in the case study are `request` and `response`

Input Parameters	
Define the parameters required to call the execution.	
Parameter name request	Verbalization the request

Output Parameters	
Define the parameters that are initialized and returned by the execution.	
Parameter name response	Verbalization the response

- `request`
 - Provides input data
- `response`
 - Returns output data

Unit title

© Copyright IBM Corporation 2017

Figure 8-33. Review: BOM relationships in vocabulary lists (1 of 2)

As you select vocabulary during rule authoring, many of the items that you select access information that was sent to the rule engine through the input ruleset parameter. Recall that ruleset variables are defined in the Decision Operation for the decision service.

Variables can also provide input to a rule, which you learn more about later.

Review: BOM relationships in vocabulary lists (2 of 2)

- Vocabulary options indicate relationships between BOM members
- The **of <object_placeholder>** structure indicates an *attribute* of the object
 - In this case, **the vehicle of <a request>** indicates that the vehicle data is passed in through the Request object
- To simplify readability, developers created a ruleset variable: **the vehicle**
 - This variable is equivalent to **the vehicle of <a request>**
 - You can choose this variable when you write rules about vehicles



Unit title

© Copyright IBM Corporation 2017

Figure 8-34. Review: BOM relationships in vocabulary lists (2 of 2)

The relationships that are available in this vocabulary list illustrate the connections within the business object model.

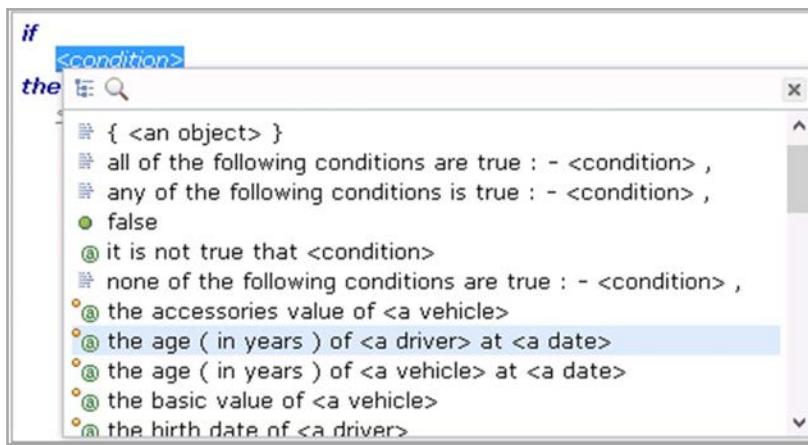
When you see multiple selections as in the example, how do you know which one to pick?

You must think about the rule you are writing, and what data you have.

In the car insurance application process, the data that you have is the input data, which the request submitter enters when applying for car insurance. A ruleset parameter provides this data for the car insurance application. Thus, the phrase you want to select is: **the vehicle of <a request>**

Choosing the correct vocabulary

- Recurring question for rule authors: How do you choose the correct vocabulary from the lists?
- When you click <condition>, how do you know what to look for and which term or phrase to pick?
 - For example, to implement a business policy about the age of the driver, do you search for “driver” or “age”?



Unit title

© Copyright IBM Corporation 2017

Figure 8-35. Choosing the correct vocabulary

A recurring question that occurs when you author rules is how to know what to look for in the vocabulary lists.

To create a rule about the age of the driver, you might search for “the driver” in the list. However, the vocabulary you need is phrased as: **age (in years) of <a driver>...**

This phrase might not be what you expect because it starts with the verbalization of the attribute, instead of starting with the class.

In fact, for most rules, the starting point is the verbalization of the attribute or value that you want. In this case, that is “age.” Most vocabulary terms or phrases are in the form “the <specific value> of <object>.”

There are some exceptions, which are covered later. You might find this type of phrase awkward at first, but you get used to it as you become familiar with rule authoring.

Where to find terms and phrases in the vocabulary menu

- Where to find terms and phrases
 - Clicking <*placeholders*> opens vocabulary lists in the menu
 - Open the completion menu by pressing the Space bar
- More BAL constructs, Boolean operators, and vocabulary
 - Boolean operators: Click ***is***
 - Arithmetic operators: Enter them by typing them in the rule statement
 - Variables and other phrases: Click the placeholders or use the menu
- Become familiar with the BOM and the vocabulary
 - Ask for documentation about the BOM, classes, attributes, relationships, and vocabulary

Unit title

© Copyright IBM Corporation 2017

Figure 8-36. Where to find terms and phrases in the vocabulary menu

During the demonstration, you saw that some vocabulary items were not immediately obvious in the rule editor. However, they can be opened by clicking certain words or icons.

- Methods that are associated with an object can be displayed by clicking verbs like ***is***.
- Enter mathematical operators by typing them in the rule statement.
- Select variables and phrases by clicking placeholders (if any) or by using the completion menu.

To understand what to click and where to look for the right selections, you must become familiar with the objects in the business object model and its associated vocabulary. If the BOM was developed without your involvement, work with your developers to understand it. Ask for documentation about the BOM, classes, attributes, relationships, and vocabulary.

Unit summary

- Identify basic types of conditions and give examples
- Determine which BAL constructs to use for various types of condition statements
- Describe how to use Boolean logic in condition statements
- Describe the relationship between rule vocabulary in the rule editors and the BOM

Unit title

© Copyright IBM Corporation 2017

Figure 8-37. Unit summary

Review questions (1 of 2)

1. Which of the following statements about rule conditions are correct?
 - A. Conditions determine when a rule is executed
 - B. At least one condition statement must always be included in a rule
 - C. Conditions always have an answer of true or false: a condition is either met, or not met
 - D. A rule can include multiple condition statements
2. True or False: Being familiar with the business object model is not a requirement for working effectively with the vocabulary in the rule editor.



Unit title

© Copyright IBM Corporation 2017

Figure 8-38. Review questions (1 of 2)

Write your answers here.

1.

2.

Review questions (2 of 2)

3. Which of the following tests can a condition statement check for?
 - A. Compare one value to another
 - B. Count the number of occurrences of something
 - C. Check to see whether an object exists
 - D. Check to see whether an object belongs to a set

4. Which of the following statements about using Boolean logic in rules are true?
 - A. In BAL, the Boolean OR is always mutually exclusive
 - B. The Boolean operator AND is always additive
 - C. You can use either ANDs or ORs in condition statements, but not both



Unit title

© Copyright IBM Corporation 2017

Figure 8-39. Review questions (2 of 2)

Write your answers here.

3.

4.

Review answers (1 of 3)



1. Which of the following statements about rule conditions are correct?
 - A. Conditions determine when a rule is executed
 - B. At least one condition statement must always be included in a rule
 - C. Conditions always have an answer of true or false: a condition is either met, or not met
 - D. A rule can include multiple condition statements

The answer is A, C, and D. Conditions can sometimes be omitted, in which case, the actions always apply, if an applicable object is present.

2. True or False: Being familiar with the business object model is not a requirement for working effectively with the vocabulary in the rule editor.

The answer is False. It is important to become familiar with the business object model because it helps you understand the contents of the vocabulary lists. If you were not involved in development of the business object model, you should plan to work with your developers to gain an understanding of it.

Unit title

© Copyright IBM Corporation 2017

Figure 8-40. Review answers (1 of 3)

Review answers (2 of 3)

3. Which of the following tests can a condition statement check for?

A. Compare one value to another

This type of statement is called a comparison condition

B. Count the number of occurrences of something

This type of statement is called a count condition

C. Check to see whether an object exists

This type of statement is called an existence check

D. Check to see whether an object belongs to a set

This type of statement is called a set membership condition

The answer is A, B, C, and D.



Unit title

© Copyright IBM Corporation 2017

Figure 8-41. Review answers (2 of 3)

Review answers (3 of 3)

4. Which of the following statements about using Boolean logic in rules are true?
 - A. In BAL, the Boolean OR is always mutually exclusive
 - B. The Boolean operator AND is always additive
 - C. You can use either ANDs or ORs in condition statements, but not both

The answer is B. In BAL, the Boolean OR is always inclusive. You can combine ANDs and ORs in condition statements. However, you might find it helpful to use parentheses to clarify your meaning.



Figure 8-42. Review answers (3 of 3)

Exercise: Working with conditions in rules

Unit title

© Copyright IBM Corporation 2017

Figure 8-43. Exercise: Working with conditions in rules

Exercise objectives

- Identify BAL constructs that apply to specific types of conditions
- Use the BAL reference documentation to look up constructs and syntax
- Use the correct BAL constructs to define condition statements in rules



Unit title

© Copyright IBM Corporation 2017

Figure 8-44. Exercise objectives

Exercise review

- Each member of a class in the BOM represents an attribute or method
- Each attribute can be either:
 - A value, like a number or text
 - A reference to another type of object that has its own members
- BOM members can be *exposed* or accessible by verbalizing them
 - Result: Verbalized BOM elements are included in the vocabulary lists when you author rules for that type of BOM element

Unit title

© Copyright IBM Corporation 2017

Figure 8-45. Exercise review

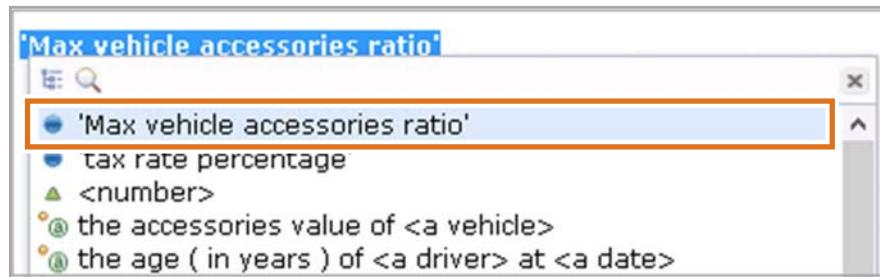
Now that you are more familiar with rules, take some time to review the relationship of the BOM to what is visible in the rule editors.

Each member of a class in the BOM represents an attribute or a method, or as was stated earlier, a specific term that is used in the conditions. You can compare these attributes, members, or terms with either a value (like a number or text) or a reference to another member, attribute, or term of the same type.

By verbalizing the BOM classes and members, the verbalized BOM members become available in the rule editor vocabulary lists when you author rules for that type of BOM element.

Exercise review: Using a constant versus using a variable

- Some policy updates can result in changes to a value such as the accessories-to-base value ratio of *0.8* that you used during the exercise
- To facilitate such changes, you can use a ruleset variable that is defined at the project level
 - You reworked the “Accessories value” rule to use a ruleset variable that is predefined at the project level:



- **Consider:** When might it be useful to use ruleset variables in your rules?

Unit title

© Copyright IBM Corporation 2017

Figure 8-46. Exercise review: Using a constant versus using a variable

As you saw during the exercise, it can be useful to define some values through a variable, instead of entering the actual value in the rule itself.

During the exercise, you changed the rule to use a ruleset variable that was defined at the project level. By defining a ruleset variable, the same value can be used in multiple rules. If the business policy that uses that value changes, the change can be entered in one place and automatically updated in all rules that reference the ruleset variable.

Exercise review: Advanced practice

- Were you able to find the appropriate vocabulary?
- Did you have any problems with building the rule statement?
- Did your rule match the rule statement in the solution?

Unit title

© Copyright IBM Corporation 2017

Figure 8-47. Exercise review: Advanced practice

Unit 9. Working with definitions in rules

Estimated time

01:00

Overview

This unit continues the focus on rule authoring. It teaches you how to define and use variables in rules. You learn which situations require certain types of variables, and which BAL constructs to use when you define those variables.

How you will check your progress

- Checkpoint
- Exercise

Unit objectives

- Describe why you might need to use rule definitions
- Explain how to define various types of variables and which BAL constructs to use

Unit title

© Copyright IBM Corporation 2017

Figure 9-1. Unit objectives

Topics

- Understanding definitions and variables
- Types of variables
- Working with definitions
- Types of variable definitions
- Using BAL constructs in variables
- Working with collections of objects
- Mechanics of defining variables

Unit title

© Copyright IBM Corporation 2017

Figure 9-2. Topics

9.1. Understanding definitions and variables

Understanding definitions and variables

Unit title

© Copyright IBM Corporation 2017

Figure 9-3. Understanding definitions and variables

What is a definition?

Definitions	definitions	Define rule variables (optional)
Conditions	if	Defines the conditions that determine when a rule is executed <ul style="list-style-type: none"> • Condition statements are optional
	then	Defines the actions that are performed when all conditions are met <ul style="list-style-type: none"> • All rules must contain at least one action
	else	Defines an alternative action when conditions are not met <ul style="list-style-type: none"> • The else part is optional when a condition is supplied • It cannot be included when the rule does not have conditions

Unit title

© Copyright IBM Corporation 2017

Figure 9-4. What is a definition?

A definition is a statement in a rule where you define and set rule variables. A variable is a convenient name that is assigned to an object or other data. You define variables at the beginning of a rule so that they can be used in later parts of a rule statement.

The definition statement can also be used to modify the rule behavior by specifying preconditions.

What is a variable? (1 of 2)

- A variable is a representation of an object or other value
- To define a variable, you assign:
 - A name
 - A type
 - A value
- Objects and other data that is assigned as variable values can be passed to the rule by using:
 - Ruleset parameters
 - Ruleset variables
 - Objects in working memory

Unit title

© Copyright IBM Corporation 2017

Figure 9-5. What is a variable? (1 of 2)

Variables represent an object or other data by using a convenient name.

Variables are created in the definitions part of the rule, and can be used throughout the rule. The value of the variable can be modified in the action part of the same rule.

This unit describes several different kinds of variables, starting with the ones that are created in rule definitions.

What is a variable? (2 of 2)

- The actual value of the variable can be set to one of the following values:

A constant	<ul style="list-style-type: none"> A number: 12 A literal character string: "Gold customer" A Boolean, an object domain element: STATUS_APPROVED
An expression	<ul style="list-style-type: none"> An arithmetic expression whose result is a number: <i>the price of the product * 1.20</i> A text concatenation or formatting expression: <i>"The loan request for " + the name of the applicant + " is rejected"</i> A Boolean expression: <i>the customer is married</i> A navigation expression to an object: <i>the last transaction of the checking account of the customer</i>
An object	For example, a reference to a predefined business term: <i>the driver</i>
A collection of objects	<ul style="list-style-type: none"> A list of numbers { 1, 3, 5 } A list of objects { CHINA, FRANCE, UK, US }

[Unit title](#)

© Copyright IBM Corporation 2017

Figure 9-6. What is a variable? (2 of 2)

The actual value of a variable can be a constant, an expression, an object, or a collection of objects.

Defining variables in a rule (1 of 3)

- You can use variables to identify, and then reference, an occurrence of an object or other BOM member by a convenient name
- Examples of variables:

BOM	Variable
Vehicle	the vehicle
Driver	the driver
CollisionInsurance	'collision policy'

Unit title

© Copyright IBM Corporation 2017

Figure 9-7. Defining variables in a rule (1 of 3)

In the case study, the BOM includes objects such as `Driver`, `Vehicle`, and `Request`. When you want to reference a particular occurrence of one of these objects, you can do so through a variable, such as `the driver` or `the vehicle`.

In the rule editor, if the name of a variable is composed of several words, you enclose them in single quotation marks (' '), such as '`high risk driver`'.

Defining variables in a rule (2 of 3)

- Make rules easier to read by creating a variable that is more concise than the vocabulary phrases available from the BOM
- To more easily distinguish between two instances of an object or draw a correlation between similar objects
 - For example, if you want to refer to a driver and the spouse of the driver:
If 'driver 1' is married to 'driver 2' and 'driver 2' is insured
- To define constants, or collections of objects:
 - *Max = 3*
 - *All customers*
- Variables that are defined in the definitions part of the rule become available in the vocabulary lists in the condition and action parts of the rule

Unit title

© Copyright IBM Corporation 2017

Figure 9-8. Defining variables in a rule (2 of 3)

You can use variables to:

- Distinguish between two instances of an object.
- Draw a correlation between similar objects.
- Define constants.
- Define collections of objects.

Defining variables in a rule (3 of 3)

- Variables that are defined in the definitions part of the rule are *local* to that rule
 - Cannot be used outside the rule
 - To reuse this type of variable in other rules, you must redefine it in those rules
- If you have a set of rules that use the same definitions, you can create a rule template with the definitions included
- To create variables that can be used across rules or across rule packages, you must use other types of variables

Unit title

© Copyright IBM Corporation 2017

Figure 9-9. Defining variables in a rule (3 of 3)

Definitions are limited to the rule in which they are defined. They show up in the vocabulary lists only for that rule. If you want to use the same definition in more than one rule, you must write that definition in each rule that uses it.

It is fairly common to have a set of rules that use the same definitions. If you have this situation, you can create a rule template with the definitions and use that to create the rules.

Not all variables are defined in definitions for a specific rule. However, some can be used across rules, such as the ruleset variables that you used in an earlier exercise.

9.2. Types of variables

Types of variables

Unit title

© Copyright IBM Corporation 2017

Figure 9-10. Types of variables

Types of variables

- Ruleset variables:
 - Must first create a variable set to contain the ruleset variables
 - Can also be used with ruleset parameters to exchange data at the application or rule-project level
 - Define in Rule Designer
 - Create ruleset variable sets
 - Define ruleset parameters
 - Define in Decision Center
 - Create ruleset variable sets
- Automatic variables:
 - Declared at the BOM level, accessible to all rules that use that BOM class
 - Define in Rule Designer
- Rule variables
 - Declared in a rule
 - Define in Decision Center rule editor

Unit title

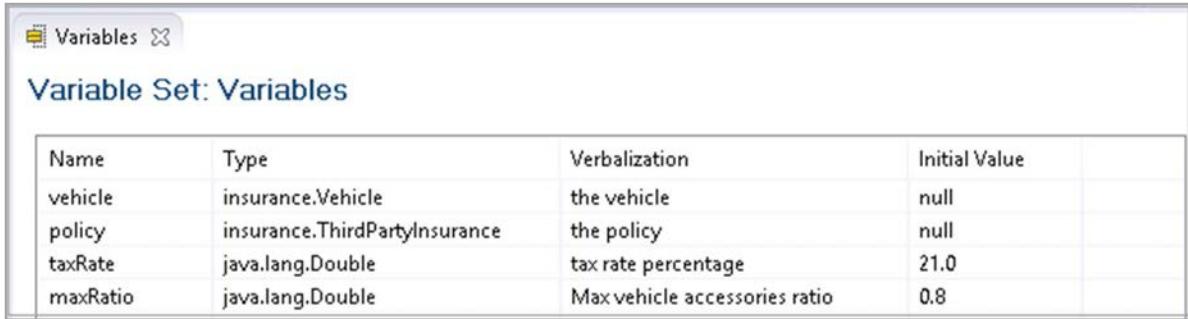
© Copyright IBM Corporation 2017

Figure 9-11. Types of variables

In addition to local variables (specific to the rule that they are defined in), variables are also defined as ruleset variables and automatic variables.

Ruleset variables

- Variables that you can use in all business rules of the ruleset
- Must create a variable set to hold ruleset variables
- Can also use ruleset variables with ruleset parameters to exchange data between the tasks in a ruleflow



The screenshot shows a software interface titled "Variables" with a sub-section titled "Variable Set: Variables". A table lists four variables:

Name	Type	Verbalization	Initial Value
vehicle	insurance.Vehicle	the vehicle	null
policy	insurance.ThirdPartyInsurance	the policy	null
taxRate	java.lang.Double	tax rate percentage	21.0
maxRatio	java.lang.Double	Max vehicle accessories ratio	0.8

Unit title

© Copyright IBM Corporation 2017

Figure 9-12. Ruleset variables

You can use ruleset variables in all of the rule artifacts in the ruleset. It is important to remember that ruleset variables belong to the entire ruleset. To define a ruleset variable, you first need to create a variable set, which is used to group the ruleset variables together. Ruleset variables are accessible from business rules only if you verbalize them.

You can also use ruleset variables with ruleset parameters to exchange data between ruleflow tasks.

Developers can create ruleset variables for you in Rule Designer, or you can create them yourself in Decision Center.

Using ruleset variables with ruleset parameters

- Developers create these ruleset variables and parameters in collaboration with business analysts
 - Defined in a decision operation for a decision service
- Three directions: IN, OUT, and IN_OUT
- IN (input parameters):
 - Parameter values are provided as input to the ruleset on execution
- OUT (output parameters):
 - Parameter values are set by the execution of the ruleset
 - Parameter values are provided as output when ruleset execution is complete
- IN_OUT (input-output parameters):
 - Parameter values are provided as input to the ruleset on execution
 - These values can be modified by the ruleset and provided as output when ruleset execution is complete

Unit title

© Copyright IBM Corporation 2017

Figure 9-13. Using ruleset variables with ruleset parameters

Developers collaborate with business analysts to create ruleset parameters, which are used to pass data back and forth between the application and the rule engine. Recall from Unit 2, "Modeling for business rules" that ruleset variables that are used for ruleset parameters are defined in the decision operation for a decision service. They can carry objects as input to the rule engine. The rule engine evaluates each of the rules in the ruleset against those objects. The rule engine decision is returned to the application through an output ruleset parameter or an input/output parameter.

Only ruleset variables that are defined at the top-level package are eligible to be used as ruleset parameters.

Ruleset variables with ruleset parameters: Example

- In the case study:
 - An input ruleset parameter passes the insurance request object to the rule engine
 - The rule engine evaluates the underwriting rules against the insurance request object, and then returns the request object as an output ruleset parameter

The screenshot shows two panels: 'Input Parameters' and 'Output Parameters'. Both panels have a header with a blue background and white text, followed by a table with three columns: 'Parameter name', 'Verbalization', and 'Type'. The 'Input Parameters' panel has one row with a highlighted row for 'request'. The 'Output Parameters' panel also has one row with a highlighted row for 'response'.

Input Parameters		
Define the parameters required to call the execution.		
Parameter name	Verbalization	Type
request	the request	insurance.Request

Output Parameters		
Define the parameters that are initialized and returned by the execution.		
Parameter name	Verbalization	Type
response	the response	insurance.Response

Unit title

© Copyright IBM Corporation 2017

Figure 9-14. Ruleset variables with ruleset parameters: Example

The following notes describe how ruleset parameters work in the case study.

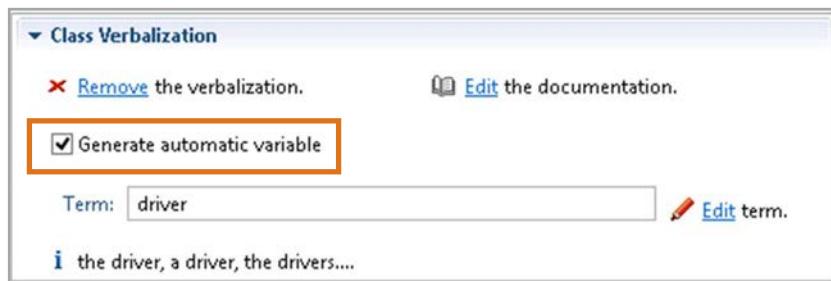
- A decision service is implemented to determine whether the insurance company should accept to underwrite an insurance policy for a particular customer.
 - The input for this decision is an insurance request that includes at least one driver, a vehicle, and the type of insurance specified.
 - The expected output of this decision is one of these outcomes:
 - Approval, along with a proposed policy and price.
 - Refusal, along with an explanation.
 - Referral to an underwriter for further review, along with an explanation.
- The underwriting rules are defined in a decision service, along with the ruleset parameters, and the decision service is deployed as a ruleset to the Rule Execution Server.
 - Rule Execution Server provides a managed execution environment for the rule engine.
 - The technical term for making the rules accessible to the rule engine is *deployment*.
- An insurance request object is passed to the rule engine by an *input* ruleset parameter.

- The rule engine evaluates the underwriting rules against the request object to determine which rules to execute. Each condition of each rule in the ruleset is tested against the object.
- After rule execution is complete, the rule engine returns the decision within the insurance request object through an *output* ruleset parameter.



Automatic variables

- Automatic variables are declared at the BOM level in Rule Designer



- The variable becomes available in any rules that use that BOM class
 - Cannot declare a ruleset variable with the same name as an automatic variable
 - Identified with the article “the”
- Useful when groups of objects are passed to the rule engine through working memory

[Unit title](#)

© Copyright IBM Corporation 2017

Figure 9-15. Automatic variables

Automatic variables are variables that you declare as an instance of a specific BOM class. Automatic variables must be declared in the BOM editor in Rule Designer. When automatic variables are defined on a BOM class, they are available for use in any rules that use the associated BOM class, without needing to explicitly declare the variable in a rule definition.

In the rule editors, automatic variables are identified with the article: *the*

You use automatic variables when groups of objects are passed to the rule engine through working memory, which is a logical storage space for objects that are going to be evaluated with the rules. Automatic variables enable you to access each object in working memory.

For example, assume that the working memory contains a group of customers and a rule uses the automatic variable *the driver*. During execution, the automatic variable value is set to each driver in the working memory so that the rule tests each driver in the working memory.



Important

You cannot define one variable to be the same as another variable of the same type. For example, if your rule requires you to refer to more than one occurrence of `driver`, you must define the other variables explicitly, in the definitions part of the rule. The automatic variable would no longer be available in this rule. Also, if an automatic variable is declared for a BOM class, you cannot declare another variable in the definitions part of the rule with the same name as the automatic variable.



Rule variables

- Rule variables are created and defined at the rule level
- Created in a **definitions** section for the rule
- Can be used only in the rule where it is declared
 - If a variable must be used in different rules, consider working with development to create an automatic or ruleset variable

```

definitions
  set 'driver' to a driver in the drivers of 'the request';

if
  it is not true that the age (in years) of driver at the start date of 'the policy' is between 16 and 80
then
  refuse the application because
  "At least one driver does not meet the age constraints" ;

```

Unit title

© Copyright IBM Corporation 2017

Figure 9-16. Rule variables

Rule variables are variables that are local to the rule: it can be used only in the rule where it is declared. You use the definitions section of the rule editor to create rule variables.

Where you use variables

- All verbalized variables are available in condition and action vocabulary menus in the rule editors
 - If you are unable to find a term in the vocabulary menus for your rule, and you cannot define a variable to provide the terms, work with developers to create the missing vocabulary
- Work with developers to create ruleset parameters and automatic variables because they can affect rule execution performance

Unit title

© Copyright IBM Corporation 2017

Figure 9-17. Where you use variables

Because ruleset variables and automatic variables can affect rule execution performance, developers usually control the creation of those types of artifacts.

When you cannot find an appropriate vocabulary item in the vocabulary lists for your rule, you should work with the developers to update the vocabulary. The update might require that they make specific data available through a new variable or through code.

Initializing variables

- Variables must be assigned a value (initialized) before a rule can be evaluated or executed

- Definitions can include **where** constructs (also called *predicates*)
 - The **where** clause constrains when a variable is assigned a value

- Example:

```
set 'gold customer' to a customer  
    where the customer category is Gold
```

- The variable is assigned a value only when the object is a Gold customer
 - If no Gold customer is found, the variable cannot be initialized (or assigned a value), and the rest of the rule is irrelevant

- **Important:** The rule engine cannot evaluate a rule when the variables do not have values

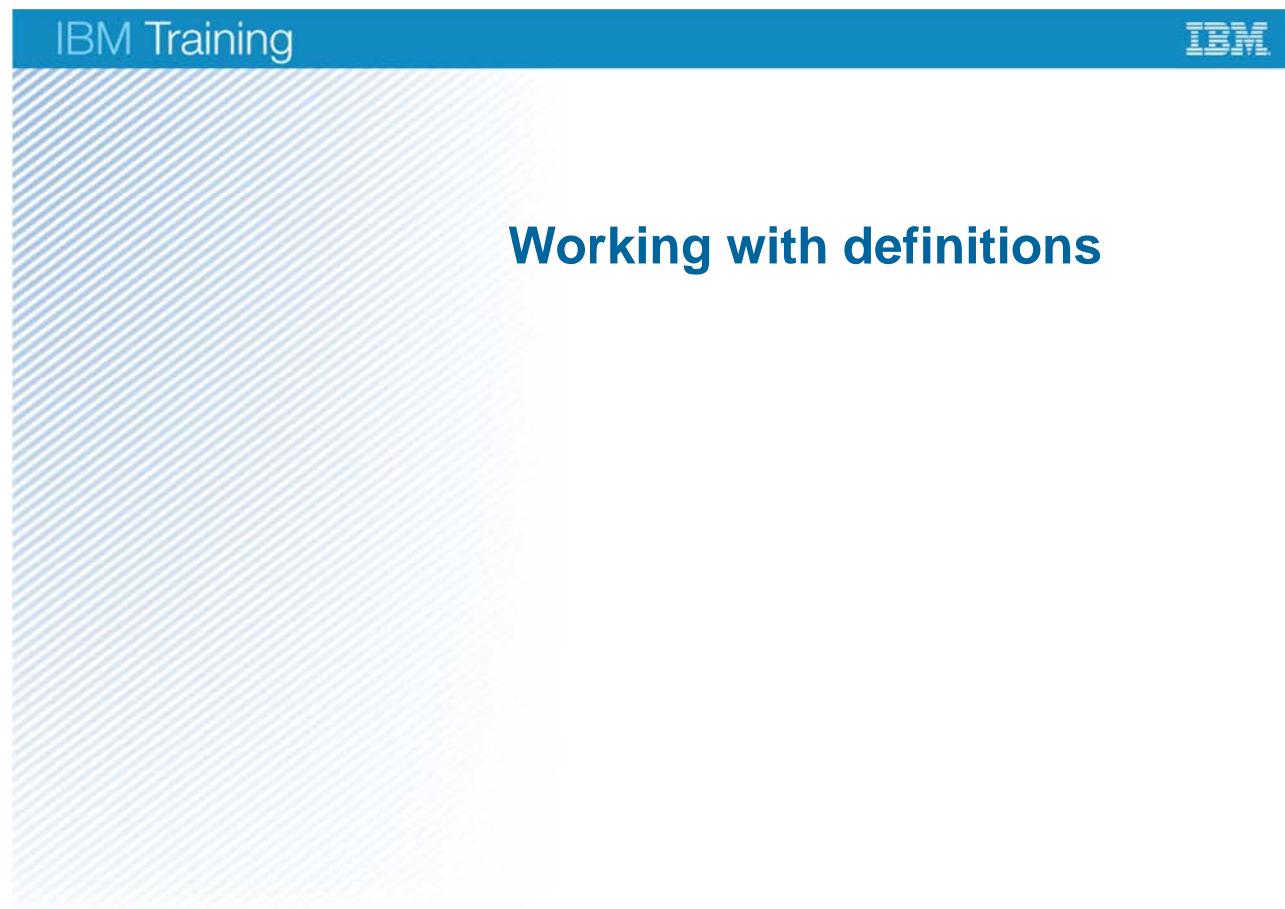
Unit title

© Copyright IBM Corporation 2017

Figure 9-18. Initializing variables

Variables must be assigned a value, or initialized, before a rule can be evaluated or executed.

9.3. Working with definitions



Unit title

© Copyright IBM Corporation 2017

Figure 9-19. Working with definitions

Definitions and rule variables overview

- Rule variables are created and defined in the **definitions** part of a rule
- Create the **definitions** section in the Business console rule editor
 - Must go before the **conditions** section (the **if** part of the rule)
- Scope of variable is the rule that declares the variable
 - Variables that are defined in a rule are available only in that rule

Unit title

© Copyright IBM Corporation 2017

Figure 9-20. Definitions and rule variables overview

This slide summarizes some of the main aspects of rule variables and rule definitions.

Defining a rule variable

- To define a variable in the **definitions** part of the rule:
 - Specify the name of your variable
 - Assign a value to the variable
 - Scope of variable is the rule in which the variable is declared
- Examples:
 - set '**the customer**' to 'a customer'
 - set '**the cart**' to 'the shopping cart of the customer'
 - set '**Smith**' to 'a customer...'
 - set '**MAX**' to '3'
- These variables become available in vocabulary lists when you create more definitions, condition statements, and action statements
 - the customer
 - the cart
 - Smith
 - max

Unit title

© Copyright IBM Corporation 2017

Figure 9-21. Defining a rule variable

To define a variable in the **definitions** section, you first specify the name of your variable, and then you set the variable to a value.

When you assign a value to the variable, the value type determines the variable type. For example, a variable that is set to “a customer” can accept only customer objects.

Defining preconditions

- A definition can also specify a precondition, which helps qualify the action carried out in an `else` statement

- **Example:** In this rule, discounts apply only to Gold customers:

```
definitions
  set applicant to a customer
  where the category of this customer is Gold
  if
    the value of the applicant's shopping cart is more than $100
  then
    apply a 15% discount
  else
    apply a 5% discount
```

- **Question:** How much discount does a Silver customer who spends \$80 receive?

Unit title

© Copyright IBM Corporation 2017

Figure 9-22. Defining preconditions

A definition can also specify a precondition, which helps qualify the action that is carried out in an `else` statement. Some preconditions use the `where` clause as part of the definition statement.

Remember that in order for a rule to be a candidate to execute, all of the variables in the definitions and the preconditions must have a value.

In the rule on this slide, discounts apply only to Gold customers, as specified in the precondition, which says:

```
set applicant to a customer
where the category of this customer is Gold
```

In this rule, if you have a customer with Gold status and the customer purchases more than \$100, the customer is given a 15% discount. If the customer purchases less than \$100, the customer is given a 5% discount.

Question: How much of a discount does a Silver customer who spends \$80 receive?

Answer: None. If the customer does not have Gold status, the customer does not meet the precondition, so the customer is not given a discount.

Comparing rules with and without preconditions

<pre> definitions set 'good customer' to a customer <u>where</u> the category of this customer is Gold if the value of the 'good customer' shopping cart is more than \$100 then apply a 15% discount else apply a 5% discount </pre>	<pre> if all of the following conditions are true: -the value of the customer's shopping cart is more than \$100 -the category of the customer is Gold then apply a 15% discount else apply a 5% discount </pre>
---	--

- **Question:** When the customer status is specified as a condition rather than a precondition, how much discount does a Silver customer who spends \$80 receive?

Unit title

© Copyright IBM Corporation 2017

Figure 9-23. Comparing rules with and without preconditions

This slide compares two rules: one that uses a precondition, and one that does not.

Question: When the customer status is specified as a condition rather than a precondition, how much of a discount does a Silver customer who spends \$80 receive?

Answer: 5%. The business meaning of this second rule is that if you have a customer with a Gold status and the customer purchases \$100, the customer is given a 15% discount. For all other customers, apply a 5% discount.

Notice that using the precondition gives different results from using conditions in the condition statement.

While it is important that you understand the subtle ways that the `else` clause can work, do not worry if these concepts seem somewhat challenging.

When you write rules, consider carefully whether your rules would be easier to understand if they were written as two rules, instead of one rule with an `else` clause. Two atomic rules are almost always clearer and easier to maintain.

When to define variables

- To create a vocabulary item that is not available in the vocabulary list
- To use wording for a BOM member that is simpler than the verbalization provided in the vocabulary phrases of the rule editor vocabulary lists
- To refer to two different objects of the same type
 - Example: Two drivers, the primary driver and the spouse of the primary driver
- To define a precondition
 - Example:

```
set 'gold customer' to a customer
where the category of this customer is Gold;
```
- Developers often create complex variables, which can be made available in templates

Unit title

© Copyright IBM Corporation 2017

Figure 9-24. When to define variables

9.4. Types of variable definitions

Types of variable definitions

Unit title

© Copyright IBM Corporation 2017

Figure 9-25. Types of variable definitions

Defining variables: Constants

- Constants are values that do not change, or are expected to change rarely
 - Example: The value of **Pi** never changes
- You can use uppercase as a naming convention to distinguish constants from other variables that might change
 - Example:

```
set 'MAX' to '3'
```
- Expressions (formula)
 - Example:

```
set 'monthly premium' to the total price of 'the
policy' / 12 ;
```

Unit title

© Copyright IBM Corporation 2017

Figure 9-26. Defining variables: Constants

These slides summarize the types of values that can be assigned to variables in the definitions part of the rule.

Defining variables: Expressions

- Expressions or formulas can be captured in a simple phrase
- Example:

```
set 'monthly premium' to the total price of 'the  
policy' / 12 ;
```

Unit title

© Copyright IBM Corporation 2017

Figure 9-27. Defining variables: Expressions

Defining variables: Objects (1 of 2)

- When the value of a variable is an object, the variable makes the object and all its attributes available

- Example:

```
set MyVehicle to the vehicle of 'the request'
```

- You can now use the `MyVehicle` variable to access the vehicle and all its attributes
- In the rule editor, all the attributes of the vehicle object also become available in the vocabulary list when you write a rule with the `MyVehicle` variable

Unit title

© Copyright IBM Corporation 2017

Figure 9-28. Defining variables: Objects (1 of 2)

Defining variables: Objects (2 of 2)

- You can define multiple objects as variables
- Some business rules involve several objects of the same type that must be expressed in terms of variables

```
definitions
    set driver1 to a driver;
    set driver2 to a driver;

if
    driver1 is married to driver2
    and driver2 is insured

then
    ...
    ...
```

- This function is useful when you have a situation where you must draw a correlation between similar objects

Unit title

© Copyright IBM Corporation 2017

Figure 9-29. Defining variables: Objects (2 of 2)

Some business rules involve several objects of the same type, and each must be referenced with variables, such as when you have two drivers who are married. By specifically defining two different driver variables, you can draw a correlation between them.

Defining variables: Collections of objects

- Some objects are passed to the rule engine as a collection, such as an array or a list
- Example:
 - In the case study, the request object can include up to six drivers
 - The `request.drivers` object is a collection of drivers, not just a single driver
- By defining a collection variable, all of the values or terms for the entire collection, both attributes and methods, become available in the rule editor vocabulary lists

Unit title

© Copyright IBM Corporation 2017

Figure 9-30. Defining variables: Collections of objects

Remember that a variable is like a box, and assigning a value to the variable determines the “shape” of the box so that only objects of the same shape can fit. If a variable is designed to hold a constant, you cannot put an object or a collection into that box.

9.5. Using BAL constructs in variables

Using BAL constructs in variables

Unit title

© Copyright IBM Corporation 2017

Figure 9-31. Using BAL constructs in variables

Using BAL constructs in variables

- Some BAL constructs that are useful when creating definitions include:
 - `in <collection>`
 - `where <test>`
 - `from <relation>`

Unit title

© Copyright IBM Corporation 2017

Figure 9-32. Using BAL constructs in variables

Using the `in <collection>` construct

- Expands the set of data that rules can use
- By using this construct, you can access a single object in a collection of objects
- The `<collection>` parameter must be an expression that denotes a collection

```
definitions
    set driver to a driver in the drivers of 'the request' ;
if
    the number of accidents of driver is more than 4
then
    refuse the application because "Driver " + the first name
    of driver + " " + the last name of driver + " has had too
    many accidents";
```

Unit title

© Copyright IBM Corporation 2017

Figure 9-33. Using the `in <collection>` construct

The `in <collection>` construct is used to retrieve a single object in a collection of objects that are related to an existing object. The `<collection>` parameter must be an expression that denotes a collection. This collection must already be defined, either in an earlier `set` statement in the definitions part of the rule, in another variable or parameter, in the object model, or elsewhere in the rule project.

In the example that is shown here, `driver` is set to find a driver “in” the “collection” of the `drivers` of the request. You can use this construct to access an individual driver from a collection of driver objects in an insurance request. During execution, all drivers available in the request are assigned to the `driver` variable. Each driver is tested in the `if` part to check whether the number of accidents of the driver is more than 4. If the condition tests true, then the action is executed.

Using the where <test> construct

- Matches objects against tests to get an object that fulfills a particular test condition

definitions

```
set driver to a driver in the drivers of 'the request'
  where the number of license withdrawals of this driver is
    more than 1;
then
  refuse the application because "Driver " + the first name
  of driver + the last name of driver + " has had too many
  license withdrawals";
```

Unit title

© Copyright IBM Corporation 2017

Figure 9-34. Using the where <test> construct

The `where <test>` construct matches objects against test conditions to get an object that fulfills a particular test condition.

This example accesses a driver object from the collection of driver objects in the insurance request, and uses the `where` test to check the driving history of that driver object. For any driver object that meets the test (`number of license withdrawals of this driver is more than 1`), that driver object is assigned to the `driver` variable and the rule executes.

Using the from <relation> construct

- Use the `from <relation>` construct to retrieve one distinct object from another object

- Expands the set of data that you can access by enabling you to access objects through their relationship

```
set 'preferred CD' to a CD from the preferred item of the
customer
```

```
set 'collision policy' to a collision insurance from 'the
policy' ;
```

- Useful only with objects (classes) that are related

- For example, the following rule declares a variable that is based on the inheritance relationship between vehicle and the specialized class motorcycle:

definitions

```
set 'the motorcycle' to a motorcycle from the vehicle of the
customer;
```

if

```
the price of 'the motorcycle' is more than ...
```

Unit title

© Copyright IBM Corporation 2017

Figure 9-35. Using the from <relation> construct

The `from <relation>` construct is not used often, but it does allow you to retrieve one distinct object from another one based on the relationship between those objects.

In the example on this slide, the variable `preferred CD` is bound to: the preferred item of the customer.

When this construct is used, its purpose is typically to simplify the navigation to an object.

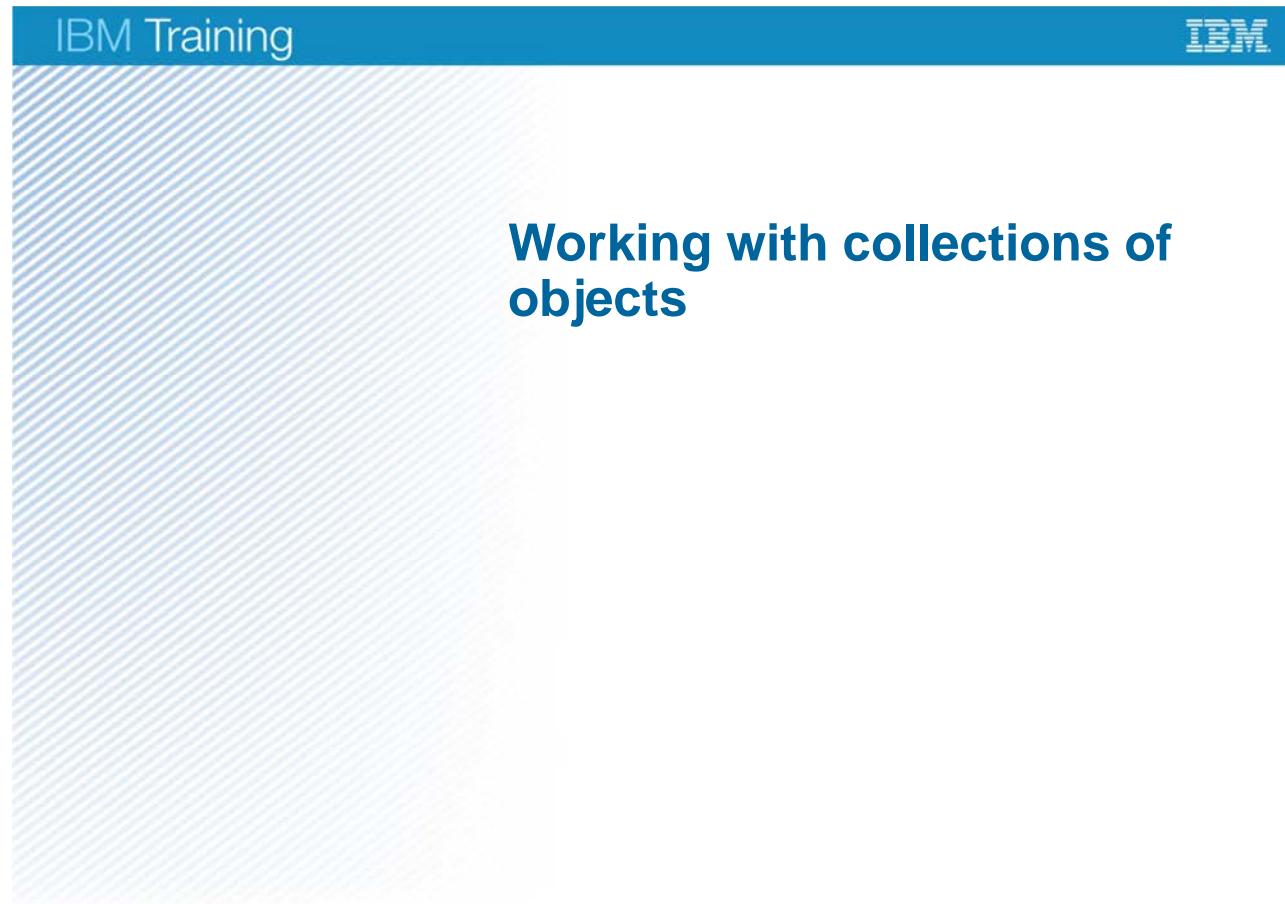
Consider a Vehicle class with a specialized Motorcycle class. If an object of type Vehicle is passed to the rule engine, you can define a variable of type Motorcycle and use the `from <relation>` construct to access only Motorcycle objects.

```
set 'variable1' to a motorcycle from the vehicle of applicant ;
```

If the vehicle object is not a motorcycle, the rule does not execute.

The `from <relation>` construct is useful mainly for casting the value of an object that is accessed through an inheritance relationship, but it can be used for any relationship type.

9.6. Working with collections of objects



Unit title

© Copyright IBM Corporation 2017

Figure 9-36. Working with collections of objects

BAL constructs to use with collections

- When working with a collection of objects, use the following constructs

- The `in <collection>` construct to retrieve individual objects in a collection of objects:

```
set 'risky drivers' to all drivers in the drivers of 'the request'
```

- The `where <test>` construct to distinguish between objects in a collection:

```
set 'risky drivers' to all drivers in the drivers of 'the request'
```

```
where each driver is not the primary driver of 'the request'  
and ...
```

Unit title

© Copyright IBM Corporation 2017

Figure 9-37. BAL constructs to use with collections

Defining a collection of objects

- You can define a variable that refers to an existing collection of objects to:
 - Perform tests on the size of the collection (for example, to determine how many objects are present)
 - Perform an action on all elements in a collection
- You can also define a variable that filters a collection of objects that satisfy a particular condition
 - Example:


```
set 'risky drivers' to all drivers in the drivers of 'the request'
where each driver is not the primary driver of 'the request'
and the number of at-fault accidents of each driver in the
last 5 years is at least 1;
```
 - The where <test> construct acts as a filter that is applied against the collection of all drivers in the drivers of 'the request'

Unit title

© Copyright IBM Corporation 2017

Figure 9-38. Defining a collection of objects

Using a defined collection in conditions

- A condition can perform different tests on a collection:
 - Size of the collection
 - Size of a subset of the collection
 - The presence or absence of a particular object (either in a collection, or in working memory)

- Example: Size of a subset

definitions

```
set 'risky drivers' to all drivers in the drivers of 'the
request'
where each driver is not the primary driver of 'the
request' and (the number of at-fault accidents of each
driver in the last 5 years is at least 1 or the number
of license withdrawals of each driver is at least 1 );
if
  the number of drivers in 'risky drivers' is more than 3
then
  refuse the application because "Too many risky drivers";
```

Unit title

© Copyright IBM Corporation 2017

Figure 9-39. Using a defined collection in conditions

Conditions can test various aspects of a collection that are specified in a definition. For example, you can test the size of the collection, the size of a subset of a collection, or the presence or absence of a particular object, either in a collection or in working memory.

In this rule, the definition part defines the `risky drivers` subset from the collection of `all drivers` in the `request`, where the driving history of each driver is used to determine eligibility. The condition tests the size of the subset.

The variable, `risky drivers`, also simplifies the readability of the condition statement.

Using a collection in action statements

- A rule can perform one or several actions on each object in a collection

```
definitions
set 'risky drivers' to all drivers in the drivers of 'the
request'
where each driver is not the primary driver of 'the
request' and (the number of at-fault accidents of each
driver in the last 5 years is at least 1 or the number
of license withdrawals of each driver is at least 1 );
then
for each driver in 'risky drivers' :
- set the total price before tax of 'the policy' to the
third party price before tax of 'the policy' + 20 ;
```

Unit title

© Copyright IBM Corporation 2017

Figure 9-40. Using a collection in action statements

By defining a collection in the definitions part of the rule, you can use that collection in the action statement so that the same action is executed on each element in the collection.

In this rule, the definition part defines the `risky drivers` subset from the collection of drivers, where the driving history of each driver is used to determine the price of the policy. The action in this rule iterates through each driver in the subset, and adds an extra amount to the total price.

Notice that this rule does not have conditions, which means the rule always executes when the `risky drivers` variable is assigned a value.

Discussion

Working with definitions in rules: Identifying variables for definitions

Unit title

© Copyright IBM Corporation 2017

Figure 9-41. Discussion

This discussion activity is based on [Exercise 11, "Working with definitions in rules"](#). The next slide contains instructions for completing this activity.

If you are a classroom or online student, be prepared to discuss your answers with the rest of the class when you complete the activity.

If you are a self-paced student, you can complete this activity as an independent exercise by following the instructions.

Discussion: Working with definitions in rules.

- This discussion is based on Exercise 11, "Working with definitions in rules"
- Follow the instructions in Section 1, "Identifying variables for definitions".

Unit title

© Copyright IBM Corporation 2017

Figure 9-42. Discussion: Working with definitions in rules.

If you are a classroom or online student, discuss your results with the class after you finish working through the activity.

If you are a self-paced student, take some time to go through the instructions and write down your ideas.

1. In your Exercise Guide, turn to [Exercise 11, "Working with definitions in rules"](#).
2. Follow the instructions in [Section 1, "Identifying variables for definitions,"](#) on page 11-2.

9.7. Mechanics of defining variables

Mechanics of defining variables

Unit title

© Copyright IBM Corporation 2017

Figure 9-43. Mechanics of defining variables

Defining variables in the Business console (1 of 3)

1. Create a definitions section before the if part of the rule.

```
definitions  
if  
  <condition>  
then  
  <action>;
```

Note:

When you first create the definitions section, you see a warning about the word “definitions” because you have not defined the variable. After you select the variable definition phrase from the menu, the rule editor changes the word “definitions” into a section header, and the warning goes away.

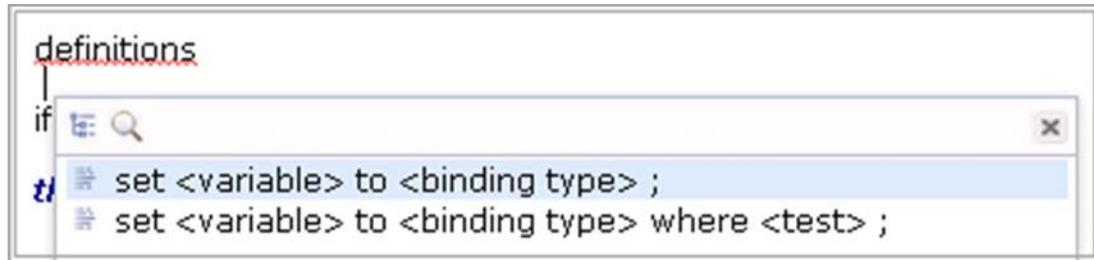
Figure 9-44. Defining variables in the Business console (1 of 3)

The mechanics of creating a definition are straightforward. In the rule editor, you create a **definitions** section, which must be before the **if** section. You can ignore the warning about the word “definitions” that appears in the rule editor; it will go away after you start defining the variable.



Defining variables in the Business console (2 of 3)

2. Use the completion menu to select the variable definition phrase.



3. Use the completion menu to select the placeholder string for the variable name.



Unit title

© Copyright IBM Corporation 2017

Figure 9-45. Defining variables in the Business console (2 of 3)

The next steps for defining a variable include selecting the variable definition phrase from the menu, and assigning a name to the variable.

Defining variables in the Business console (3 of 3)

4. Assign a name to the variable by typing it into the editor.

```
definitions
set 'collision policy' to <binding type>;
```

5. Click the **<binding type>** placeholder and use the menu to complete the variable definition.

```
definitions
set 'collision policy' to <binding type>;
if
<condition>
then
<action>;
```

Completion menu:

- a channel type
- a character
- a classification type
- a collision insurance**
- a date

Unit title

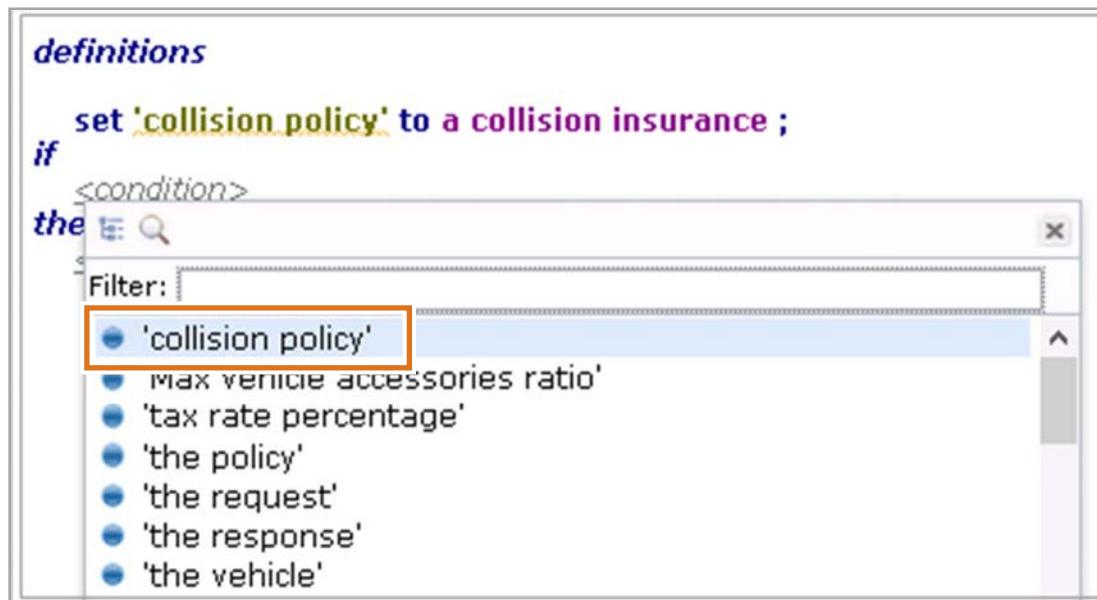
© Copyright IBM Corporation 2017

Figure 9-46. Defining variables in the Business console (3 of 3)

After you assign a name to the variable, use the completion menu to finish building your variable definition.

Using a variable

- After you define a variable, it becomes available in vocabulary lists



Unit title

© Copyright IBM Corporation 2017

Figure 9-47. Using a variable

After you define a variable, it becomes available in vocabulary lists.

Variables are usually, but not always, at the top of the list and in alphabetical order. However, the rule editor predicts which variables you are most likely to choose. Variables that you define and the ones that are of the correct type for your statement are the variables that are shown at the top of the list.

If you do not see a variable that you expect to see in the vocabulary list, make sure to scroll through the whole list.

Unit summary

- Describe why you might need to use rule definitions
- Explain how to define various types of variables and which BAL constructs to use

Unit title

© Copyright IBM Corporation 2017

Figure 9-48. Unit summary

Review questions (1 of 2)

1. True or False: Definitions are important because they are used to create all variables.
2. Which of the following examples are ways that definitions are used? Select all that apply.
 - A. To provide a term that can be reused throughout the rest of the rule
 - B. To specify a precondition that must be met
 - C. To indicate what should happen when a condition is met
 - D. To indicate that a rule must look at each value within a list or collection



Unit title

© Copyright IBM Corporation 2017

Figure 9-49. Review questions (1 of 2)

Write your answers here:

1.

2.

Review questions (2 of 2)

3. True or False: Determining and defining ruleset parameters is an important task that all rule authors should know how to do.
4. True or False: Rule variables can be used in any rule artifact in the ruleset.
5. True or False: Simplifying the language of rule statements is an important reason to create definitions.



Unit title

© Copyright IBM Corporation 2017

Figure 9-50. Review questions (2 of 2)

Write your answers here:

3.

4.

5.

Review answers (1 of 2)

- True or False: Definitions are important because they are used to create all variables.

The answer is False. Definitions are used to create rule-specific variables that are available only for statements within that rule. Other types of variables include ruleset parameters, ruleset variables, and automatic variables.

- Which of the following are ways that definitions are used? Select all that apply.

- A. To provide a term that can be reused throughout the rest of the rule
- B. To specify a precondition that must be met
- C. To indicate what should happen when a condition is met
- D. To indicate that a rule must look at each value within a list, or collection

The answer is A, B, and D. Actions, not definitions, are used to indicate what should happen.



Unit title

© Copyright IBM Corporation 2017

Figure 9-51. Review answers (1 of 2)

Review answers (2 of 2)

3. True or False: Determining and defining ruleset parameters is an important task that all rule authors should know how to do.

The answer is False. Developers always create ruleset parameters in Rule Designer.



4. True or False: Rule variables can be used in any rule artifact in the ruleset.

The answer is False. A rule variable applies only to the rule where it is defined. A rule variable is defined in the definitions statement of the rule.

5. True or False: Simplifying the language of rule statements is an important reason to create definitions.

The answer is True.

Unit title

© Copyright IBM Corporation 2017

Figure 9-52. Review answers (2 of 2)

Exercise: Working with definitions in rules

Unit title

© Copyright IBM Corporation 2017

Figure 9-53. Exercise: Working with definitions in rules

Exercise objectives

- Identify and use common BAL constructs to write definition statements, including *in*, *from*, and *where*
- Use variables to make a rule easier to read
- Write definition statements that use objects and collections of objects

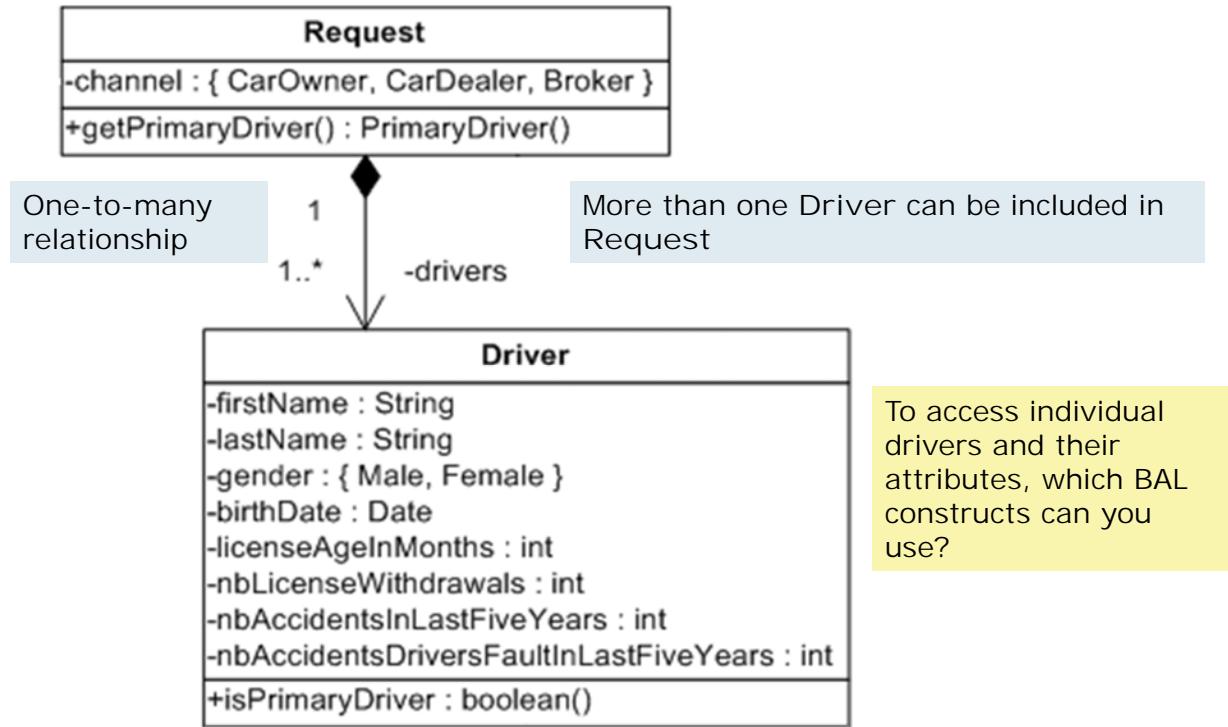


Unit title

© Copyright IBM Corporation 2017

Figure 9-54. Exercise objectives

Exercise overview: Reviewing the BOM before you start



Unit title

© Copyright IBM Corporation 2017

Figure 9-55. Exercise overview: Reviewing the BOM before you start

The BOM diagram illustrates a one-to-many relationship between the insurance request and the Driver class, which indicates that there can be more than one driver.

Because the request can include a collection of drivers, you can use the `in <list>` construct to access each driver.

During this exercise, you see how to use a **where** test instead of a condition statement. You also learn how to access each driver by defining a collection variable.

Exercise review: Solution 1

```

definitions
  set driver to a driver in the drivers of 'the request' ;
  if
    it is not true that the age ( in years ) of driver at the
    start date of 'the policy' is between 18 and 80
  then
    refuse the application because "At least one driver does not
    meet the age constraints";

```

- The variable gets any driver included in the request
- The age of the driver is tested in the condition statement
- Note that the *driver* variable in the definition is set to something “in” a collection, as opposed to being a collection variable
 - in <list> construct

Unit title

© Copyright IBM Corporation 2017

Figure 9-56. Exercise review: Solution 1

This task has three possible solutions. These solutions represent different patterns. Each solution uses definitions in a different way, all of which are valid.

The task was to create the rule “Driver age” that tests the ages of the drivers to determine whether any of the drivers are disqualified because of age constraints.

As with many rules, there is more than one way to approach writing this rule.

Exercise review: Solution 2

definitions

```
set driver to a driver in the drivers of 'the request'
  where it is not true that the age ( in years ) of driver at
  the start date of 'the policy' is between 18 and 80
then
  refuse the application because "At least one driver does not
  meet the age constraints";
```

- This solution uses a precondition to filter drivers according to age
- The definition that is used in the example gets any driver, and filters the drivers by using the `where <test>` clause
- Logically, this solution is equivalent to Solution 1

Unit title

© Copyright IBM Corporation 2017

Figure 9-57. Exercise review: Solution 2

Solution 2 uses a precondition to filter the drivers.

Exercise review: Solution 3

definitions

```
set 'disqualified drivers' to all drivers in the drivers of
'the request'
  where it is not true that the age ( in years ) of each
  driver at the start date of 'the policy' is between 18 and 80
then
  for each driver in 'disqualified drivers' :
    - refuse the application because "At least one driver does
      not meet the age constraints";
```

- The variable 'disqualified drivers' defines a collection variable (a set of drivers that are filtered by age)

Unit title

© Copyright IBM Corporation 2017

Figure 9-58. Exercise review: Solution 3

In Solution 3, the variable 'disqualified drivers' defines a collection variable: a set of drivers whose age disqualifies them for insurance coverage.

Exercise review: Comparison of solutions 1 and 2

<p>#1</p> <p><i>definitions</i></p> <p> set 'myVariable' to a driver</p> <p> <i>if</i></p> <p> 'myVariable' satisfies COND</p> <p> <i>then</i></p> <p> do ACTION</p>	<p>#2</p> <p><i>definitions</i></p> <p> set 'myVariable' to a driver</p> <p> <i>where</i> this driver satisfies TEST</p> <p> <i>then</i></p> <p> do ACTION</p>
<ul style="list-style-type: none"> • These rules are logically equivalent: the actions execute under the same circumstances • Their behavior would differ if they had an else statement <ul style="list-style-type: none"> ▪ If no driver satisfies the if statement, an else statement would execute ▪ If no driver satisfies the where clause, a then or an else statement cannot execute • Solution 2 is preferred because performance improves when you can filter early 	

Unit title

© Copyright IBM Corporation 2017

Figure 9-59. Exercise review: Comparison of solutions 1 and 2

Comparing solutions:

- In Solution 2, the definition acts as a precondition that eliminates the need for a condition statement.
- These rules are logically equivalent and execute under the same circumstances.

Exercise review: Comparison of solutions 2 and 3

<p>#2</p> <p><i>definitions</i></p> <pre>set 'myVariable' to a driver where this driver satisfies TEST then do ACTION</pre>	<p>#3</p> <p><i>definitions</i></p> <pre>set 'myCollection' to all drivers where each driver satisfies TEST then for each driver in 'myCollection': - do ACTION on this driver</pre>
<ul style="list-style-type: none"> • If n drivers satisfy TEST, the rule on the left would execute n times • Solution 3 executes only one time to handle the n drivers • Solution 2 is simpler, cleaner, and generally preferred 	

Unit title

© Copyright IBM Corporation 2017

Figure 9-60. Exercise review: Comparison of solutions 2 and 3

Comparing solutions:

- In Solution 2, the variable is set to something *in* a collection, as opposed to *being* a collection.
- In Solution 3, the variable *disqualified drivers* *is* a collection.
- Solution 2 might execute multiple times.
- Solution 3 would execute once for all drivers that are filtered into the variable.

Using the rule in Solution 3 results in a slight performance advantage, but remember that the rule engine has built-in, efficient ways of iterating over the objects in working memory. You do not need to worry about trying to write rules that are most efficient. Instead, focus on the simplicity and modularity of the rules. The rule engine can handle the mechanics.

Unit 10. Writing complete rules

Estimated time

01:00

Overview

This unit continues the focus on rule authoring. It covers common rule action patterns, the use of multiple actions in a single rule, and when to use the `else` statement. You learn how to resolve errors that appear in the rule editor, and how to avoid ambiguity in rule statements.

How you will check your progress

- Checkpoint
- Exercise

Unit objectives

- Explain the purpose of action statements in rules
- Give examples of rule actions
- Explain how to resolve error messages that are commonly encountered in the rule editors
- Explain the use of an else statement

Unit title

© Copyright IBM Corporation 2017

Figure 10-1. Unit objectives

Topics

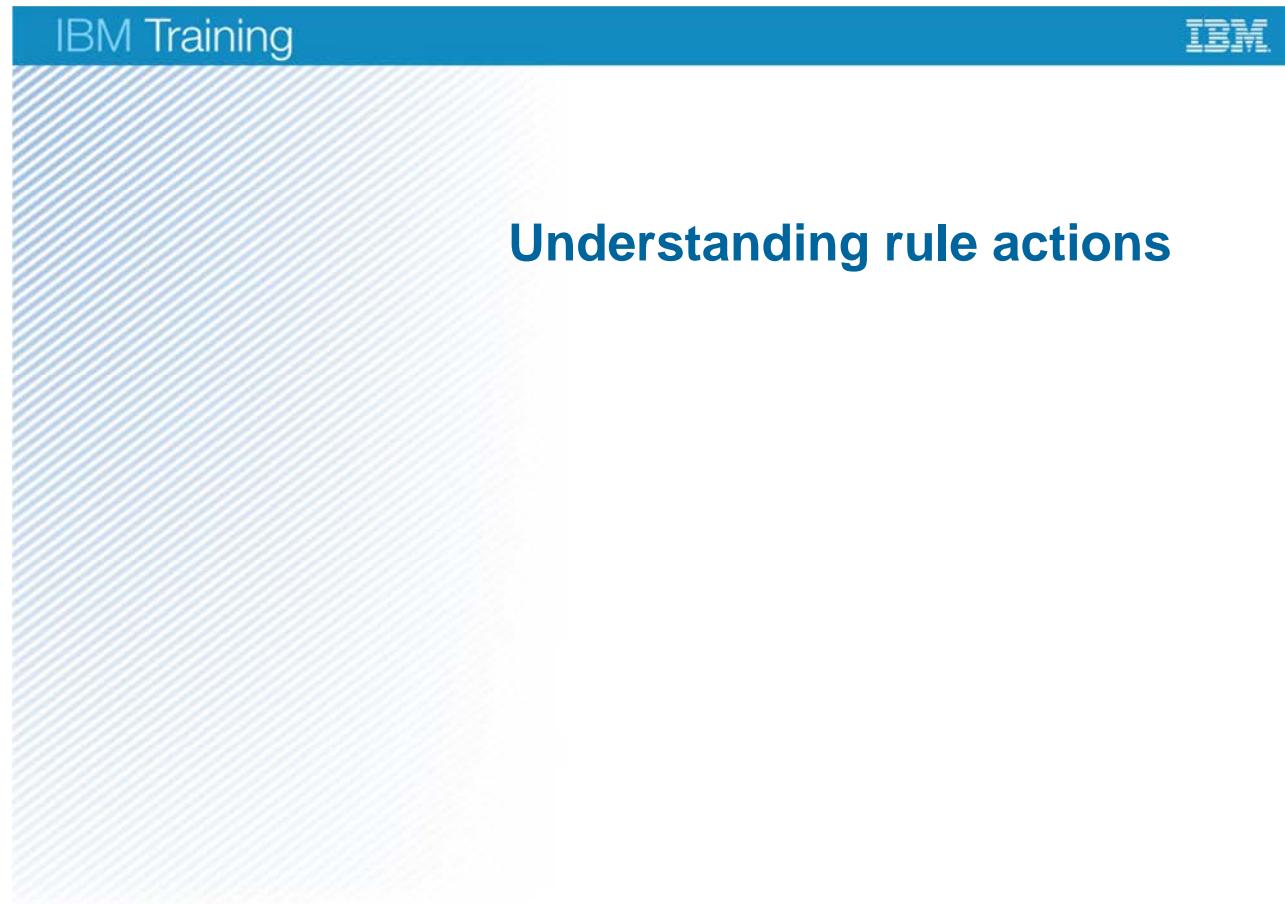
- Understanding rule actions
- Rules that perform computations
- Handling errors

Unit title

© Copyright IBM Corporation 2017

Figure 10-2. Topics

10.1. Understanding rule actions



Unit title

© Copyright IBM Corporation 2017

Figure 10-3. Understanding rule actions

What is an action?

Definitions	definitions	Define rule-specific variables (optional)
Conditions	if	Defines the conditions that determine when a rule is executed <ul style="list-style-type: none"> • Condition statements are optional
Actions	then	Defines the actions that are performed when all conditions are met <ul style="list-style-type: none"> • All rules must contain at least one action
	else	Defines an alternative action when conditions are not met <ul style="list-style-type: none"> • The else part is optional when a condition is supplied • It cannot be included when the rule does not have conditions

Unit title

© Copyright IBM Corporation 2017

Figure 10-4. What is an action?

Actions are the final part of a rule statement.

The action part of a rule describes what to do when the conditions of the rule are met, and in some cases, what to do when the conditions are not met.

Rule actions

- Actions state what the rule does
 - Actions are the executable part of the rule
- A rule must specify at least one action statement
- Actions are executed when all of the conditions and preconditions are met
 - If the rule does not have conditions or preconditions, the rule actions are executed if an object that matches is found
 - Example:

```
set the amount of 'the account' to 95
```

If 'the account' does not exist, the action is not executed
- A rule can be composed of an action statement only
 - When rules do not include definition or condition statements, the actions are always executed

Unit title

© Copyright IBM Corporation 2017

Figure 10-5. Rule actions

A rule must specify at least one action statement. An action statement is required.

Examples of rule actions

- Rule actions can modify objects, attributes, or variables by using “set” statements

```
    set the membership of the customer to GOLD
```

- Actions can execute a method or a function, such as “apply a 10% discount”:

```
if
    the value of the shopping cart is more than $100
then
    apply a 15% discount on the shopping cart
```

- Action that sets Boolean values:

- make it true that
- make it false that

Unit title

© Copyright IBM Corporation 2017

Figure 10-6. Examples of rule actions

Rules that include only actions

- Action rules with no definitions or conditions always execute:

```
then
```

```
  set the total price of 'the policy'  
  to (100 + 'tax rate percentage') * the total price before tax  
  of 'the policy' / 100 ;
```

Unit title

© Copyright IBM Corporation 2017

Figure 10-7. Rules that include only actions

Recall that rules must have at least one action: definitions and conditions are optional.

Rules that do not have definitions or conditions always execute.

Multiple actions

- If there is more than one action in a rule statement, the actions are carried out in the order in which they appear
- Boolean operators (AND, OR) are not required in action statements
 - Multiple actions can be listed


```
then
            apply a 5% discount;
            display the message: "A five percent discount has been
            applied";
```
 - Boolean OR is addressed by using an **else** statement
- When you have multiple actions in a rule statement, make sure that the rule is atomic

[Unit title](#)

© Copyright IBM Corporation 2017

Figure 10-8. Multiple actions

Remember that each rule should be atomic, and that each rule has only one name. Think about maintaining rules that might add a second action, and make sure that the name reflects the result of both actions.

In other words, if you have two actions, then they should be multiple steps of a single “logical” action. For example, you might set a value *and* print a message to say that the value was set. Or, you might do a calculation and want to separate the steps of the calculation so that the formula is clear.

Just because two rules have the same condition does not mean they should be the same rule. Indeed, if they do two different actions, then they should be separate rules.

Else statements

- An **else** statement specifies rule behavior when conditions are not met

```

if
  the accessories value of 'the vehicle' / the basic value
  of 'the vehicle' is more than 'Max vehicle accessories
  ratio'
then
  set the max insured value of 'the vehicle' to the basic
  value of 'the vehicle' * (1.0 + 'Max vehicle accessories
  ratio') ;
else
  set the max insured value of 'the vehicle' to the basic
  value of 'the vehicle' + the accessories value of 'the
  vehicle' ;

```

Unit title

© Copyright IBM Corporation 2017

Figure 10-9. Else statements

An **else** statement says what should happen when conditions are not met. The example on this slide illustrates how the use of **else** works.

In the rule on this slide, the value of vehicle accessories is divided by the basic value of the vehicle. The result is compared to the ruleset variable: `Max vehicle accessories ratio`

If the result is more than `Max vehicle accessories ratio`, then the maximum insurable value for the vehicle is calculated.

However, if the condition is *not* met, the **else** statement comes into play. In other words, if the result is not more than `Max vehicle accessories ratio`, then the maximum insurable value is calculated according to the current accessories value + the basic value of the vehicle.

When to use else statements

- Rules with `else` are error-prone and hard to maintain
 - Except in simple cases, such as computing a value, it is often difficult to determine which action to take in the `else` part of the rule


```
if
    the borrower is a US citizen
then
    The borrower is eligible
else ...
```

 - If you later decide to *also* accept permanent residents, you must modify the existing rule
 - The use of `else` here ties the rule about US citizens to every case when the borrower is not a US citizen
 - If a borrower later modifies immigration status to become a permanent resident, how does the rule application know to look for the rule about US citizens?
- **Note:** Nested `if/else` pairs are not allowed

Unit title

© Copyright IBM Corporation 2017

Figure 10-10. When to use else statements

The use of `else` is discouraged, for two main reasons:

- Rules with `else` are error-prone.
 - In the example here, what should the `else` statement be?
 - `Else` statements are difficult to understand, which is why they are error-prone.
- Rules with `else` are also hard to maintain.
 - In the example here, if the business decides to accept permanent residents in addition to US citizens, you must then modify the existing rule.
 - In this case, the use of `else` would mean that the rule about US citizens is tied to the way you handle every case where the borrower is not a US citizen. Such a rule would make it difficult for someone who returns later with a modified immigration status. How would they know to look at the rule for US citizens?

Usually, it is much clearer to create two rules instead of using an `else` statement:

- One rule for the positive condition test.
- One rule for the negative condition test.

The only valid use of `else` in good rule design is for a truly Boolean condition that never changes. For example, you might say: if a customer is married, then do something, else for unmarried, do something else. Even this rule becomes problematic if the logic suddenly must also handle unmarried partners as if they are married. For that reason, it is better to start with two rules.

10.2. Rules that perform computations

Rules that perform computations

Unit title

© Copyright IBM Corporation 2017

Figure 10-11. Rules that perform computations

Rules that perform computations

- Some rules calculate a value that you use in a separate rule
- By creating a computation rule that calculates a certain value, you can reuse that computation in many rules, instead of rewriting the computation in each rule
 - If there is a change in how the formula is computed, you can update it in one place and automatically reuse it in the rules that reference it

Unit title

© Copyright IBM Corporation 2017

Figure 10-12. Rules that perform computations

Rules can also be used to perform calculations that are used in other rules. You see how this process works in the next exercise, when you create rules that compute premiums.

By using this approach, you reduce the maintenance work that is required on rules. If the calculation or threshold changes, you can change the computation in just one place rather than in many rules.



Computation examples (1 of 2)

- After specific values are calculated, other rules can use these values
 - Example: Base price for minimum third-party coverage is calculated in the **Compute base price table**

The screenshot shows the IBM Decision Table interface. On the left, there is a navigation tree for a project named "carinsurance-dt-start". Under "Price Third-Party", the "Base Price" rule is selected. To the right, the main area displays the "Base Price" table with the following data:

	Third-party max		Base price	Variable rate
	min	max		
1	5,000	10,000	50	0
2	10,000	20,000	0	0.005
3	$\geq 20,000$		80	0.001

Unit title

© Copyright IBM Corporation 2017

Figure 10-13. Computation examples (1 of 2)

This slide shows an example of a computation rule. In the exercises, you work with a table that calculates the base price for minimum third-party coverage. You can then create other rules that refer to the values that are calculated in the table.



Computation examples (2 of 2)

- The **Calculate Global Premium** folder includes only computation rules that calculate the total price
 - Example: The Calculate third party global premium before tax rule uses the value that is calculated in the **Compute base price table**

The screenshot shows the IBM TRIRIGA application interface. On the left, there is a navigation tree under the 'carinsurance-actions-start' node, which includes a 'buffer' node and several computation rule nodes: 'Calculate Global Premium', 'Check Eligibility', 'Price Collision', 'Price Options', and 'Price Third-Party'. The 'Calculate Global Premium' node is highlighted with a blue selection bar. On the right, there is a list of computation rules with columns for Name, Last Changed By, and Last Changed On. Two rules are visible: 'Calculate collision global premium before t...' (changed by rtsUser1 on May 10, 2017) and 'Calculate third party global premium before...' (changed by tsAdmin on April 21, 2017). The rule name 'Calculate third party global premium before...' is also highlighted with an orange box. Below the list, there is a 'definitions' section containing a set of rules:

```

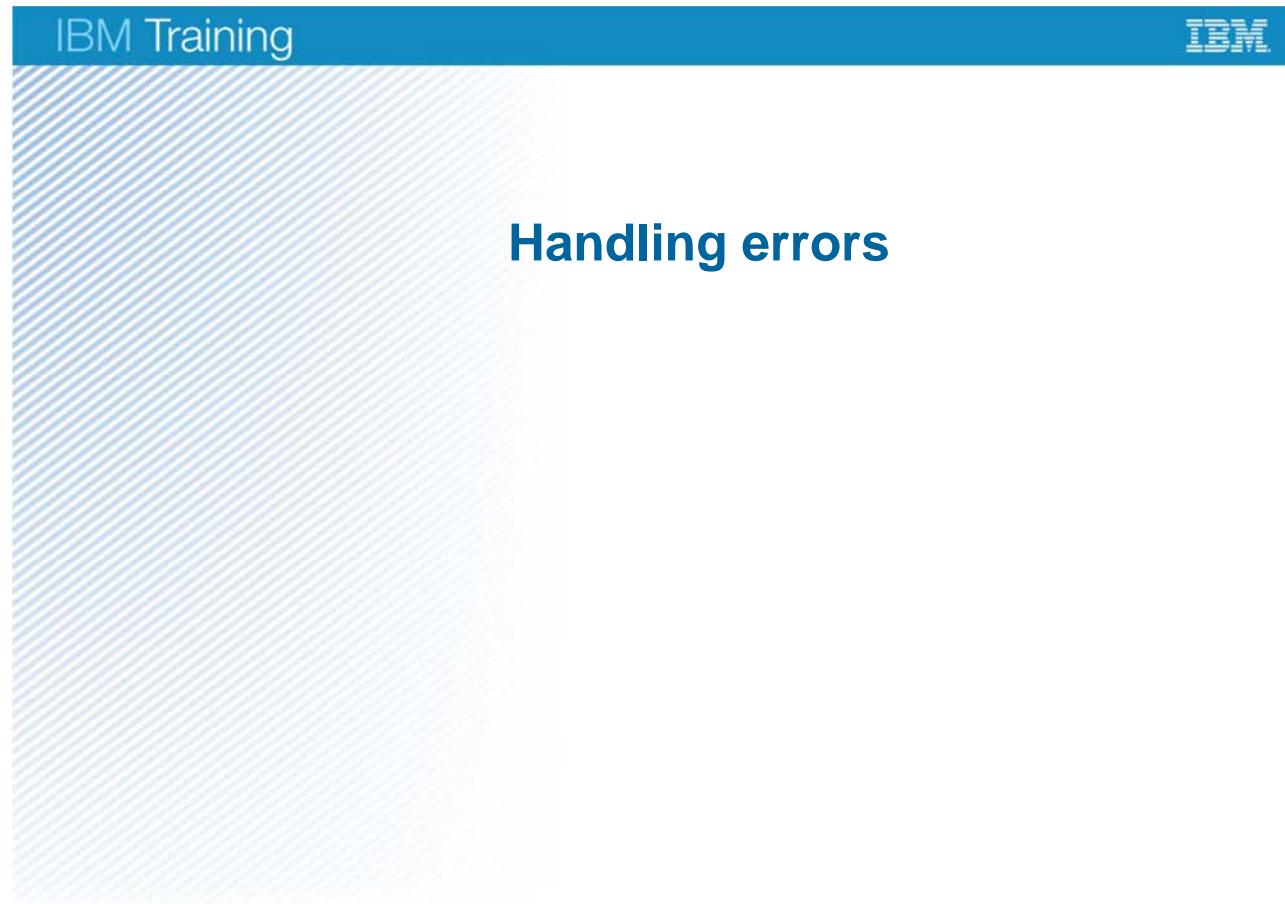
definitions
  set 'third party policy' to 'the policy'
    where the type of this third party insurance is ThirdParty ;
  then
    set the total price before tax of 'third party policy' to the third party price before tax of 'third party policy';
  
```

Unit title © Copyright IBM Corporation 2017

Figure 10-14. Computation examples (2 of 2)

This slide shows an example of a computation rule that uses the values that are calculated in the Compute base price table.

10.3. Handling errors



Unit title

© Copyright IBM Corporation 2017

Figure 10-15. Handling errors

Identifying errors

- As you create rules, you might see error icons or messages beside lines of text in your rule
- Errors:
 - Red icons
 - Prevent compilation
- Warnings:
 - Yellow icons
 - Allow compilation
- These basic syntax errors are rule-specific

Unit title

© Copyright IBM Corporation 2017

Figure 10-16. Identifying errors

Icons indicate whether errors are *errors* or *warnings*. Errors indicate that a rule cannot be compiled. Warnings indicate that a rule can compile, but it has some aspect that seems problematic.

These types of basic syntax errors are rule-specific, and are shown for individual rules only. It is also possible to check for other types of errors across rules. You learn about those types of checks in a later unit.

Effect of errors

- Rules with basic syntax errors are not included when the ruleset is generated
- Examples of basic syntax errors:
 - A rule statement uses a business term that is not recognized or no longer valid
 - A rule statement is ambiguous, meaning that there is more than one correct interpretation
 - A variable in the definitions part is already declared or of the wrong type
 - A rule is incomplete, that is, some of the placeholders were not completed
 - A rule has an **else** part with no **if** part

Unit title

© Copyright IBM Corporation 2017

Figure 10-17. Effect of errors

Avoiding ambiguity with parentheses (1 of 2)

- Parentheses can be used to clarify situations that might otherwise be ambiguous

Example:

if

all of the following conditions are true :

- the category of the customer is Gold
- the age of the customer is at least 18
- or the salary of the customer is more than 10000

- Is the final “or” statement part of the previous phrase “the age of the customer is at least 15,” or a separate item?
- Parentheses can be used for clarity:

if

all of the following conditions are true :

- the category of the customer is Gold
- (the age of the customer is at least 18
or the salary of the customer is more than 10000)

Unit title

© Copyright IBM Corporation 2017

Figure 10-18. Avoiding ambiguity with parentheses (1 of 2)

You can use parentheses to clarify situations that might otherwise be ambiguous. For example, in this rule, it is hard to tell whether the final **or** statement is part of the previous phrase (**the age of the customer is at least 18**) or is a separate item.

Parentheses can clarify this statement. In the second version, notice that the parentheses enclose the two clauses.

Avoiding ambiguity with parentheses (2 of 2)

- Although parentheses can clarify complex rules, evaluate whether the rule makes more sense as two rules
- Consider: Do you want to tie two policies together?
 - Example: In the previous rule example, is the age of the customer related to the salary, or are they included together only because they share an action?
- Use ANDs and ORs to group things that are related
- Do not use OR just because the actions are the same, and you want to reuse the actions

Unit title

© Copyright IBM Corporation 2017

Figure 10-19. Avoiding ambiguity with parentheses (2 of 2)

Although parentheses can clarify complex rules, it is a good idea to evaluate whether the rule would make more sense as two rules. Ask yourself: *Do you want to tie two policies together?*

Do not use OR just because the actions are the same. Remember that using an OR in the conditions violates the guideline to make rules atomic. Even when you consider the two conditions to be equivalent, ask yourself whether the rules can be more easily understood and maintained as separate rules.

Use of parenthesis () in rule action

- Parentheses are also used to clarify the order of operations in statements

then

```
set the total price of 'the policy'  
to (100 + 'tax rate percentage') * the total price before tax  
of 'the policy' / 100 ;
```

- To include parentheses in a rule action, type them around the phrase that you want to enclose

Unit title

© Copyright IBM Corporation 2017

Figure 10-20. Use of parenthesis () in rule action

Parentheses can also be used to clarify the order of operations in statements like the one shown on this slide.

If the parentheses are removed, you get a different result.

Reviewing written rules: Verifying accuracy

- After you author rules, it is important to verify them
- Compare rule to actual business intent
- Check for accuracy and completeness
- Adjust the rules, as needed

Unit title

© Copyright IBM Corporation 2017

Figure 10-21. Reviewing written rules: Verifying accuracy

After you author rules, it is important to verify that they meet the intended need. Be prepared to compare the rule to the actual business intent, check for accuracy and completeness, and adjust it as needed.

Before rules are deployed, it is also important to test them. You can run tests and simulations by using the Business console testing and simulation features.

Unit summary

- Explain the purpose of action statements in rules
- Give examples of rule actions
- Explain how to resolve error messages that are commonly encountered in the rule editors
- Explain the use of an else statement

Unit title

© Copyright IBM Corporation 2017

Figure 10-22. Unit summary

Review questions (1 of 2)

1. True or False: If a rule contains a condition statement, an action statement is optional.
2. Which of the following items are common rule action patterns?
 - A. Set the value of X to Y
 - B. Make it true that
 - C. Make it false that
 - D. Display a message XYZ
 - E. All of the above
3. True or False: If a rule has multiple action statements, you must use the Boolean AND/OR operators.



Unit title

© Copyright IBM Corporation 2017

Figure 10-23. Review questions (1 of 2)

Write your answers here:

- 1.
- 2.
- 3.

Review questions (2 of 2)

4. True or False: “Else” statements define what should happen when conditions are not met; they are an essential part of rules.
5. True or False: You can tell when a rule has a basic syntax error because a message is shown in the Rule editor.
6. True or False: It does not matter whether rules have basic syntax errors. You can safely ignore them because they represent only syntax.



Unit title

© Copyright IBM Corporation 2017

Figure 10-24. Review questions (2 of 2)

Write your answers here:

4.

5.

6.

Review answers (1 of 2)

1. True or False: If a rule contains a condition statement, an action statement is optional.
The answer is False. A rule must specify at least one action statement.
2. Which of the following items are common rule action patterns?
 - A. Set the value of X to Y
 - B. Make it true that
 - C. Make it false that
 - D. Display a message XYZ
 - E. All of the above**The answer is E.**
3. True or False: If a rule has multiple action statements, you must use the Boolean AND/OR operators.
The answer is False. Multiple action statements can be listed, without an AND operator. If an OR needs to be indicated, it is done by creating an “else” clause.

Unit title

© Copyright IBM Corporation 2017

Figure 10-25. Review answers (1 of 2)



Review answers (2 of 2)



4. True or False: “Else” statements define what should happen when conditions are not met; they are an essential part of rules.

The answer is False. The use of “else” is discouraged. When you are tempted to include an “else” statement, consider carefully whether the rule would be clearer if defined as multiple rules.

5. True or False: You can tell when a rule has a basic syntax error because a message is shown in the Rule editor.

The answer is True. Rule-specific errors are displayed in the rule editors.

6. True or False: It does not matter whether rules have basic syntax errors. You can safely ignore them because they represent only syntax.

The answer is False. Rules with basic syntax errors are not included when the ruleset is generated.

Unit title

© Copyright IBM Corporation 2017

Figure 10-26. Review answers (2 of 2)

Exercise: Writing complete rules

Unit title

© Copyright IBM Corporation 2017

Figure 10-27. Exercise: Writing complete rules

Exercise objectives

- Write clear and unambiguous action statements in rules
- Apply the concepts of building definition, condition, and action statements to write complete rules



Unit title

© Copyright IBM Corporation 2017

Figure 10-28. Exercise objectives

Unit 11. Authoring decision tables

Estimated time

00:30

Overview

This unit continues the focus on rule authoring. You learn how to work with decision tables.

How you will check your progress

- Checkpoint
- Exercises

Unit objectives

- Describe how to express policy as decision tables
- Explain when to create decision tables versus individual action rules

Unit title

© Copyright IBM Corporation 2017

Figure 11-1. Unit objectives

This unit covers decision tables and trees, with a focus on working with decision tables. It also covers the benefits of using decision tables, and how to work with the editor.

Topics

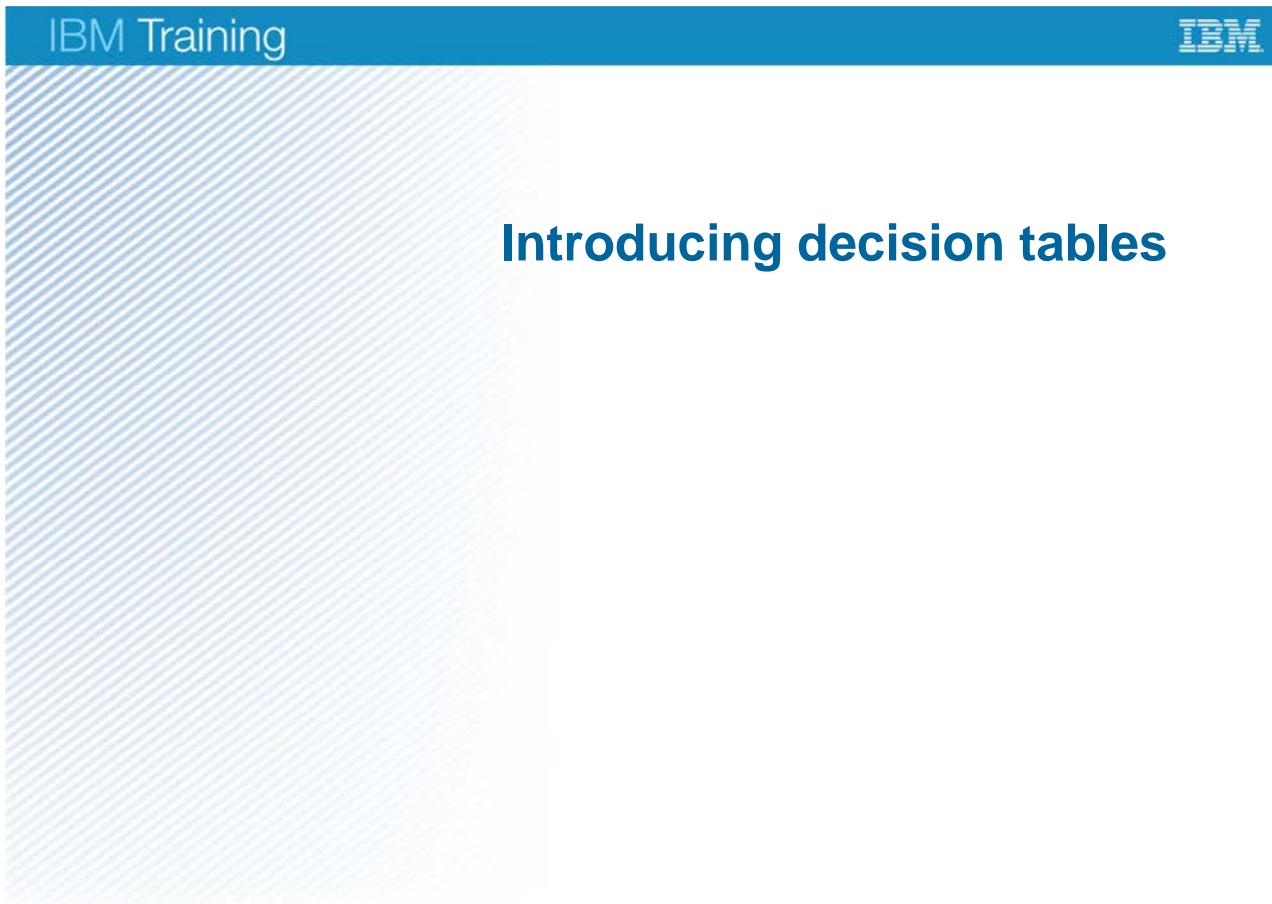
- Introducing decision tables
- Decision tables overview
- Working with decision tables

Unit title

© Copyright IBM Corporation 2017

Figure 11-2. Topics

11.1. Introducing decision tables



Unit title

© Copyright IBM Corporation 2017

Figure 11-3. Introducing decision tables

Introducing decision tables

- Decision tables provide a concise way of viewing a set of business rules in the form of a spreadsheet
- Decision tables are suitable for representing sets of business rules that have a uniform structure
- Advantages of decision tables
 - Preconditions that apply to an entire decision table
 - Built-in error checking that verifies the structure of your data for gaps and overlaps

Unit title

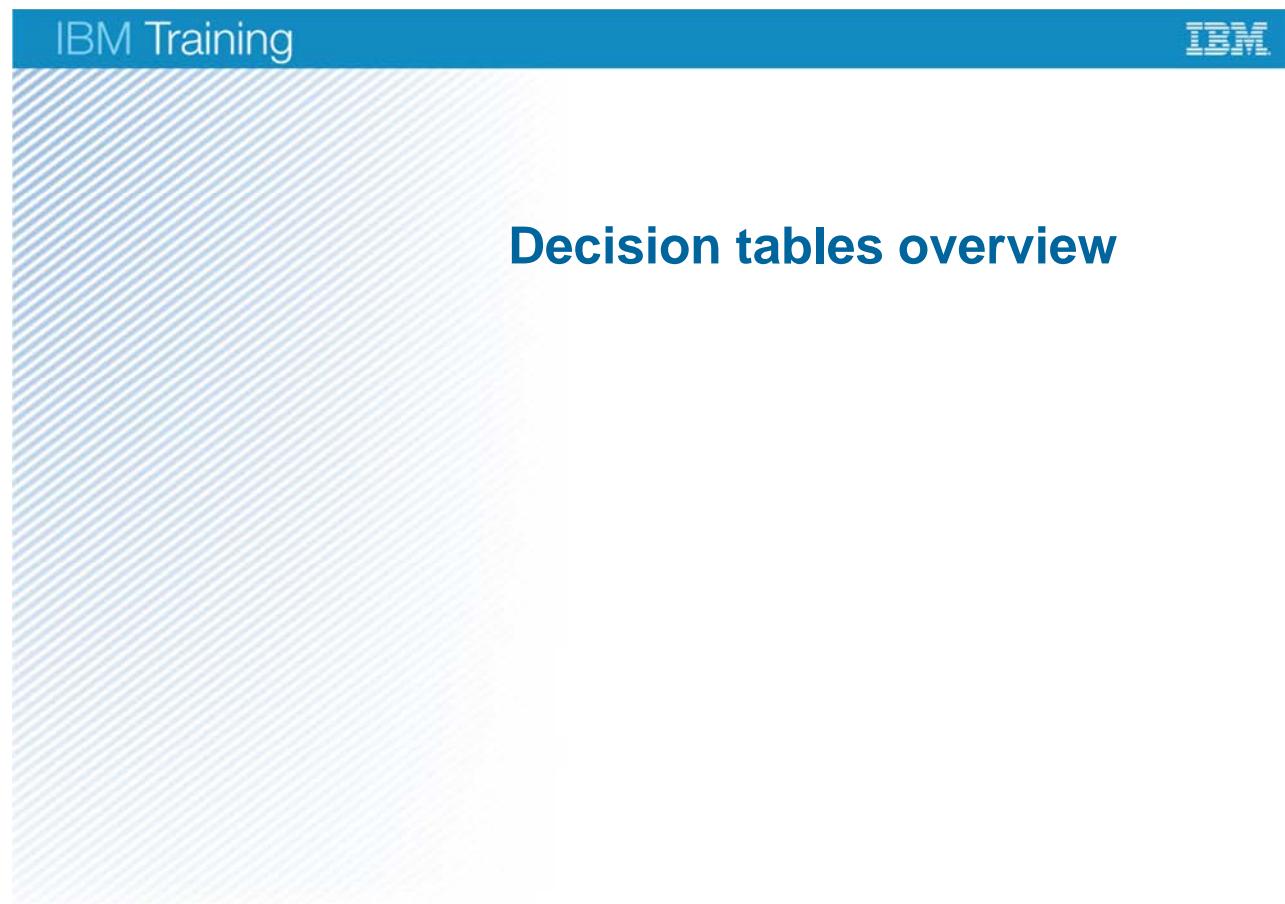
© Copyright IBM Corporation 2017

Figure 11-4. Introducing decision tables

Decision tables provide a concise way of viewing a set of business rules in the form of a spreadsheet.

Decision tables are suitable for representing sets of business rules that have a uniform structure. A decision table groups rules that share a rule statement but have different conditions and actions.

11.2. Decision tables overview



Unit title

© Copyright IBM Corporation 2017

Figure 11-5. Decision tables overview

About decision tables

- Convenient way to view and manage large sets of similar business rules
- Composed of rows and columns:
 - Each row corresponds to a single rule
 - Columns define the conditions and actions
- Other benefits:
 - Can apply preconditions to all rules
 - Easy to see overlaps or gaps
 - Consistency checking features (static rule analysis)
- When to use decision tables:
 - Common condition tests
 - Similar actions

Unit title

© Copyright IBM Corporation 2017

Figure 11-6. About decision tables

Decision tables provide an alternative and convenient way to view and manage large sets of similar business rules.

They are composed of rows and columns that work together to form rules. Each row corresponds to a single rule, while columns are used to define the conditions and actions.

To create a rule, you add a row to the decision table and enter values in the new cells. When an application calls the table, it runs the rules. If the conditions in a row are met, the rule that is formed by the row does the actions in the row.

Some benefits of creating decision tables include:

- Preconditions can be applied to all rules.
- You can easily see overlaps and gaps between rules.
- Decision tables have built-in consistency checking features.

Decision tables are generally used when rules share common condition terms, and when they have similar actions.

Example: Symmetrical rules

- Patterns are a good indicator that a decision table might be useful
 - If **the miles per year of the vehicle** is less than 4,000, then **set the third party price before tax of the policy** to the third party price before tax of the policy * 0.6
 - If **the miles per year of the vehicle** is at least 4,000 and less than 8,000, then **set the third party price before tax of the policy** to the third party price before tax of the policy * 0.8
 - If **the miles per year of the vehicle** is at least 8,000 and less than 15,000, then **set the third party price before tax of the policy** to the third party price before tax of the policy * 1
 - If **the miles per year of the vehicle** is at least 15,000 and less than 25,000, then **set the third party price before tax of the policy** to the third party price before tax of the policy * 1.2
 - If **the miles per year of the vehicle** is at least 25,000, then **set the third party price before tax of the policy** to the third party price before tax of the policy * 1.3
- Each of these rules can become a row in a decision table

Unit title

© Copyright IBM Corporation 2017

Figure 11-7. Example: Symmetrical rules

Patterns in rules are a good indicator that a decision table might be the best way to implement those rules.

Each of the rules on this slide can become a row in a decision table.

Notice that each rule uses the same information.

- All the conditions test the same attribute: **the miles per year of the vehicle**
- Each action sets the same attribute: **the third party price before tax of the policy**

Look for this type of a pattern to determine whether the business logic belongs in a decision table.

Example: Symmetrical rules in a decision table

- As a decision table, the same set of rules is much easier to read

	Vehicle miles per year	MPY Factor
1	< 4,000	0.6
2	[4,000 8,000[0.8
3	[8,000 15,000[1
4	[15,000 25,000[1.2
5	≥ 25,000	1.3

- Each row represents a rule
 - When conditions for a row are met, actions in that row are executed

	Vehicle miles per year	MPY Factor
1	< 4,000	0.6

if
all of the following conditions are true :
 - (the miles per year of **'the vehicle'** is less than **4000**),
then
 set the third party price before tax of **'the policy'** to the third party price before tax of **'the policy' * 0.6** ;

Unit title

© Copyright IBM Corporation 2017

Figure 11-8. Example: Symmetrical rules in a decision table

On this slide, you see the same list of rules now presented as an easy-to-read decision table. Each rule becomes a row.

Condition columns are on the left, and show the number of miles per year that a vehicle uses. The action column is on the right, and represents the calculation to set the third-party price before tax of the policy.

Because each row is a distinct rule, only one row of a decision table can execute. When conditions for one of the rows are met, the actions in that row are executed.

Structure of a decision table

	Vehicle miles per year		MPY Factor
1	< 4,000		0.6
2	[4,000		0.8
3	[8,000		1
4	[15,000		1.2
5	≥ 25,000		1.3

Unit title

© Copyright IBM Corporation 2017

Figure 11-9. Structure of a decision table

This slide shows the different parts of a decision table.

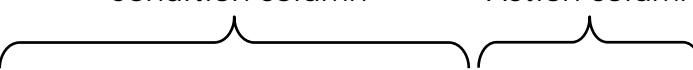
A decision table has condition columns and action columns. Each row represents a rule.

Column headers are labels that you choose to identify the condition or action. You can click a column header to see the condition test or action for that column.

Row headers are the row numbers. You can click a row header to see the full rule for that row.

Columns for conditions and actions

- Condition columns are on the left and have a gray column header
- Action columns are on the right and have a blue column header



	Vehicle miles per year	MPY Factor
1	< 4,000	0.6
2	[4,000 8,000[0.8
3	[8,000 15,000[1
4	[15,000 25,000[1.2
5	≥ 25,000	1.3

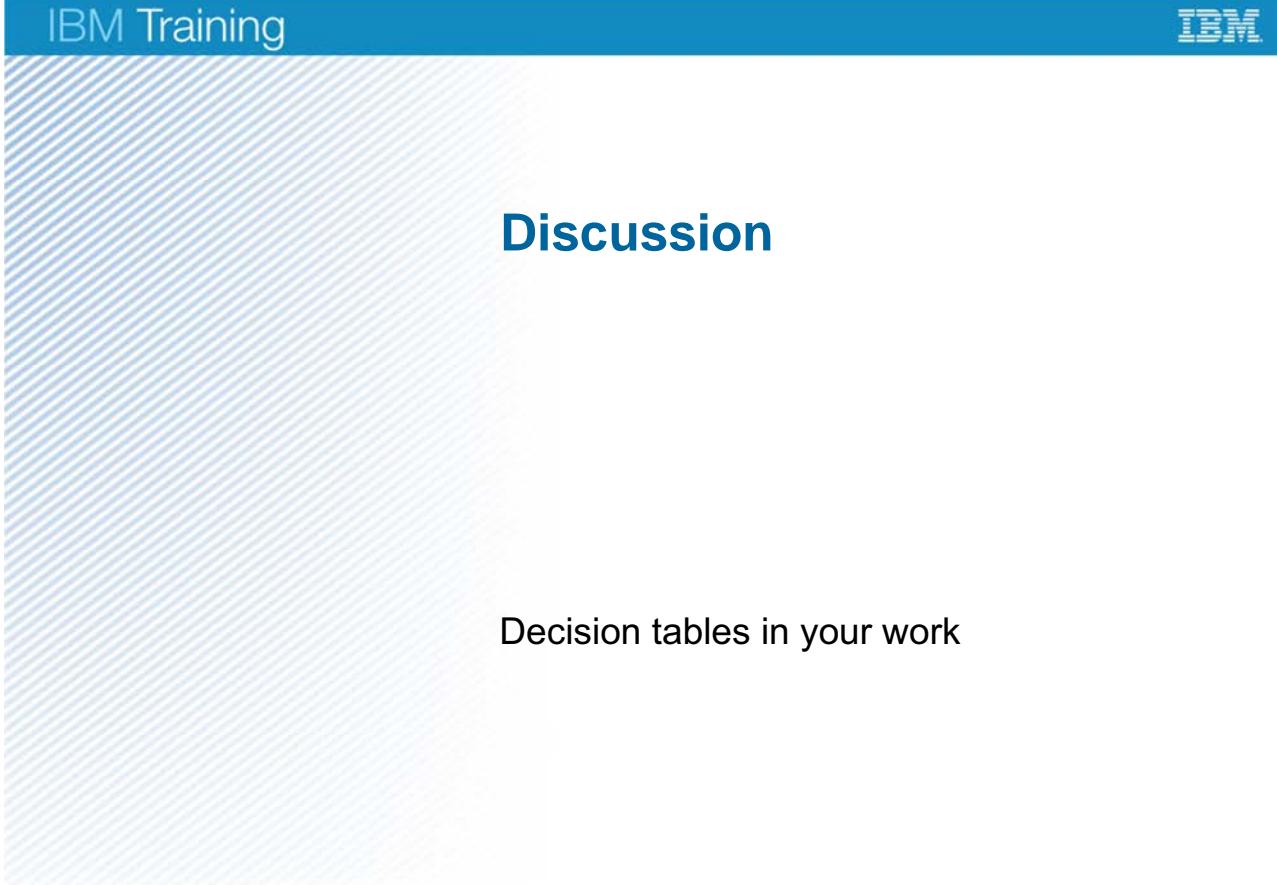
Unit title

© Copyright IBM Corporation 2017

Figure 11-10. Columns for conditions and actions

In the decision table editor, condition columns are always on the left, and action columns are always on the right.

Notice that the condition and action columns have headers with different colors, which helps to visually distinguish them. Condition columns have a gray column header, and action columns have a blue column header.



The slide features a blue header bar with 'IBM Training' on the left and the IBM logo on the right. The main content area has a light blue diagonal striped background. A large blue title 'Discussion' is centered at the top. Below it, the text 'Decision tables in your work' is displayed. The bottom of the slide contains copyright information and a figure caption.

Discussion

Decision tables in your work

Unit title

© Copyright IBM Corporation 2017

Figure 11-11. Discussion

For this discussion activity, take a few minutes to consider the questions on the next slide.

For classroom and online versions of the course, be prepared to discuss your ideas with the other students and your instructor.

For self-paced versions of this class, you can do this activity as an independent exercise.



Discussion: Decision tables in your work

1. What uses do you see for decision tables at your work?
 2. What types of rules do you work with that might be reorganized as decision tables?

Unit title

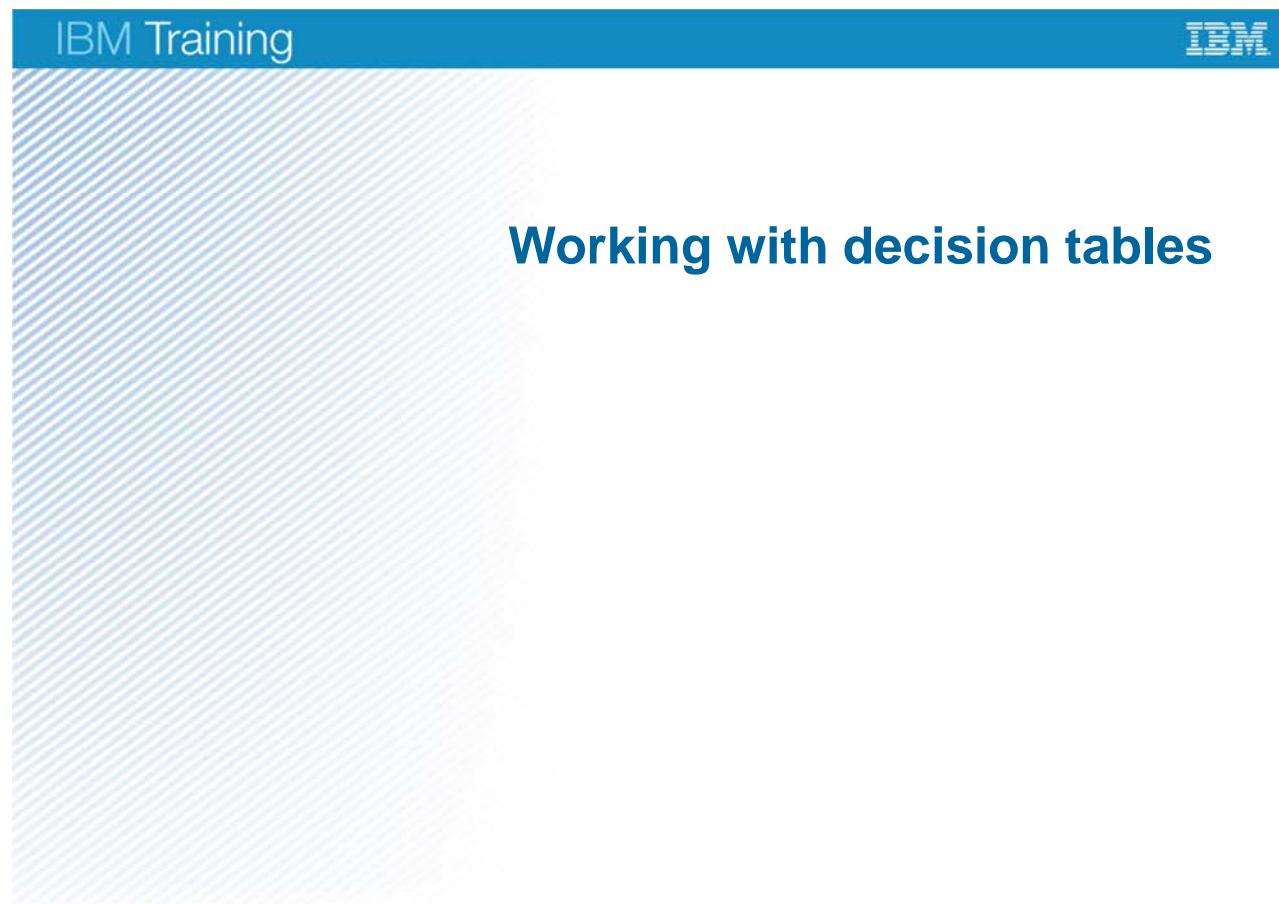
© Copyright IBM Corporation 2017

Figure 11-12. Discussion: Decision tables in your work

For classroom and online versions of the course, discuss with your instructor how you might use decision tables in your work.

For self-paced versions of this class, take some time to make notes in answer to these questions.

11.3. Working with decision tables



Unit title

© Copyright IBM Corporation 2017

Figure 11-13. Working with decision tables

Working with decision tables: Overview

- Two things to learn:
 - How to structure decision tables
 - Mechanics of using the editor
- Build tables by defining the condition and action statements for each column
 - Complete the table by manually entering values in each cell
 - If needed, edit conditions for each cell after you enter condition cell values

Unit title

© Copyright IBM Corporation 2017

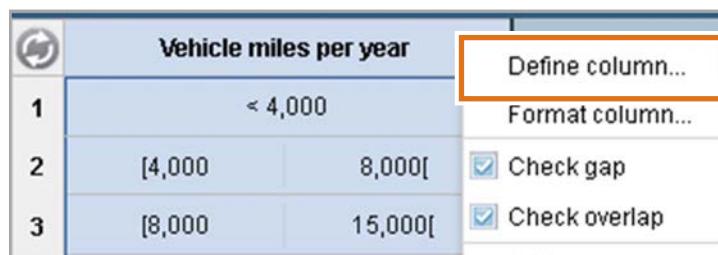
Figure 11-14. Working with decision tables: Overview

Much as with rules, the most important part of learning to work with decision tables is developing a sense of how to structure the content you put into the decision table.

Try not to make your decision tables too large. A few hundred rows and 10 – 20 columns are an acceptable size for efficient performance. If a decision table becomes too large, the time it takes to edit, save, and compile the data becomes too long. Compilation is when the IRL of the table is created so that the rule engine can evaluate it.

Defining conditions (1 of 2)

- Define conditions in a Condition column
- In a Condition column, right-click the column header and click **Define column**
 - For new conditions: Opens the Define Update Column window
 - For existing conditions: Opens the Update Condition Column window



	Vehicle miles per year	
1	< 4,000	
2	[4,000	8,000[
3	[8,000	15,000[

Unit title

© Copyright IBM Corporation 2017

Figure 11-15. Defining conditions (1 of 2)

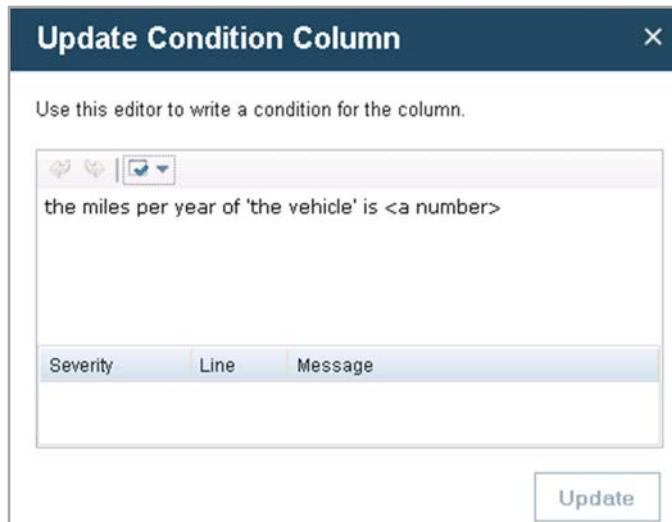
To write the condition test for a column, you right-click the condition header and click **Define column** to open the condition editor. For a new condition, the editor opens in the Define Condition Column window. For an existing condition, the editor opens in the Update Condition Column window.

This example shows a condition column with both a single value and two values. When condition or action columns have more than one value, the editor splits the column automatically into subcolumns.



Defining conditions (2 of 2)

- Author or edit the condition in the condition editor
 - One condition test per column



- After authoring or editing the condition, click **Define** (for a new condition) or **Update** (for an existing condition) to save your changes

[Unit title](#)

© Copyright IBM Corporation 2017

Figure 11-16. Defining conditions (2 of 2)

In the condition editor, you can build the condition test in a similar manner as in the rule editor. You can type the condition into the editor directly, or use the completion menu.

After you define the overall condition for the column, you can modify the condition tests for each row.

Splitting conditions

- A condition is split across two columns when two placeholders are left incomplete in the condition test
 - For example, the BAL operators ***is between*** and ***is at least and less than*** include <min> and <max> placeholders that can be completed in the table

	Vehicle miles per year		MPY Factor
	< 4,000	≥ 4,000	
1	< 4,000		0.6
2	[4,000	8,000[0.8
3	[8,000	15,000[1
4	[15,000	25,000[1.2
5	≥ 25,000		1.3

Unit title

© Copyright IBM Corporation 2017

Figure 11-17. Splitting conditions

Conditions can be split as subcolumns in the table when the rule statement they represent contains more than one empty placeholder.

In this example, a condition is split across two subcolumns in row 2 - 4 to indicate minimum and maximum values.



Defining actions (1 of 2)

- Define actions in the **Action** column
 - You must have a defined condition or set of conditions for the table before you can create an action
- Defining actions is similar to defining conditions
 - After rule conditions are defined, right-click the Action column header and click **Define column**

	Vehicle miles per year	MPY Factor	
1	< 4,000	0.6	Define column...
2	[4,000 8,000[0.8	Format column... Cut

Unit title

© Copyright IBM Corporation 2017

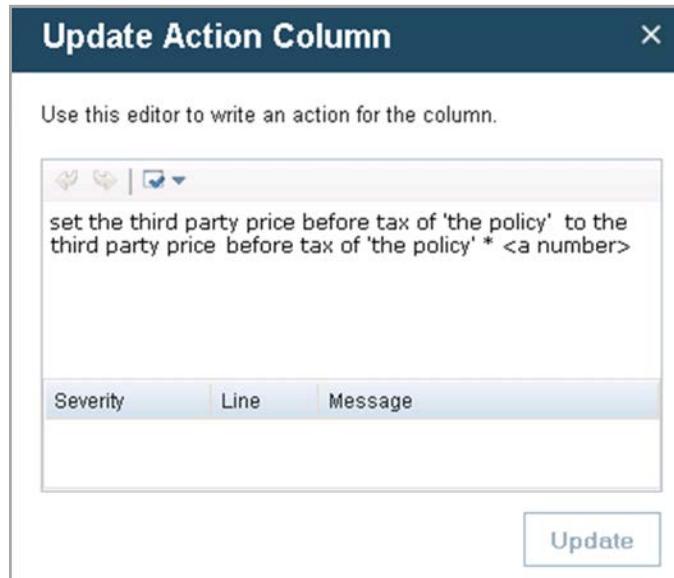
Figure 11-18. Defining actions (1 of 2)

You define action columns in the same manner as condition columns. You right-click the action column header and then click **Define column** to open the action editor.



Defining actions (2 of 2)

- Action editor opens in the Define Action Column window (new actions) or the Update Action Column window (existing actions)
 - For new actions, click **Define** and for existing actions, click **Update**



Unit title

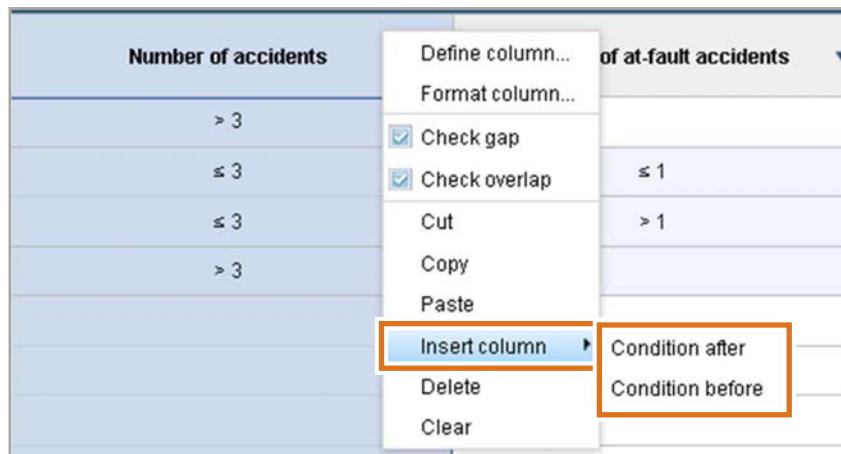
© Copyright IBM Corporation 2017

Figure 11-19. Defining actions (2 of 2)

In the action editor, you can build the action statement in the same manner as in the condition editor or rule editor. You can type the action into the editor directly, or use the completion menu.

Building decision tables by adding columns (1 of 2)

- You build tables by adding columns
 - Use column menu to insert additional condition or action columns
- Condition column options
 - Right-click condition column header and click **Insert column**
 - Insert condition to *right* of current column: Click **Condition after**
 - Insert condition to *left* of current column: Click **Condition before**



Unit title

© Copyright IBM Corporation 2017

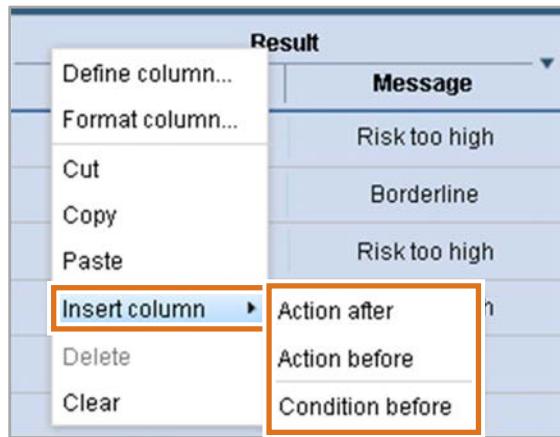
Figure 11-20. Building decision tables by adding columns (1 of 2)

You can add condition or action columns to the table by using the **Insert column** option on the column menu.

Based on which column type you select (either condition or action), the **Insert column** options change to reflect available features for that selection. For example, when you select **Insert column** on a condition column, you see options to insert a **Condition before** (to the left of the column) or a **Condition after** (to the right of the column).

Building decision tables by adding columns (2 of 2)

- Action column options
 - Right-click action column header and click **Insert column**
 - Insert action to *right* of current column: Click **Action after**
 - Insert action to *left* of current column: Click **Action before**
 - Insert condition to left of first action column: Click **Condition before**



Unit title

© Copyright IBM Corporation 2017

Figure 11-21. Building decision tables by adding columns (2 of 2)

Similar to adding condition columns, when you select **Insert column** on an action column, you see options to insert **Action before** (to the left of the column), or **Action after** (to the right of the column). If the action column is the first or only action column in the table, you can also insert a **Condition before** (to the left) of the action column.



Viewing the meaning of individual rules in decision tables

- Hover over the row header to see the complete rule for any row

	Driver age		Number of accidents	Number of claims
	min	max		
1	[65	70]	> 3	
	70	80	3	1

if
all of the following conditions are true :
 - (the age (in years) of the primary driver of **'the request'** at the start date of **'the policy'** is between **65** and **70**)
 - (the number accidents of the primary driver of **'the request'** is more than **3**),
then
 set response status to Refused and add message "**Risk too high**";

Unit title

© Copyright IBM Corporation 2017

Figure 11-22. Viewing the meaning of individual rules in decision tables

You can see the rule statement for any rule by hovering over the numbered row header for that rule. Here, you see the rule statement for row 1 of this decision table.



Editing tools and options

- You can use various tools in the decision table editor toolbar



- Editing:
 - Undo last action
 - Redo last action
- Preconditions
 - Apply a condition to all rules in table
 - Define variables
- Details
 - Decision table properties
 - Gap and overlap checking

Unit title

© Copyright IBM Corporation 2017

Figure 11-23. Editing tools and options

The decision table editor has a toolbar with several features that you can use while authoring and updating decision tables.

For example, you can:

- Undo or redo editing actions.
- Apply preconditions to a decision table.
- Change options for the decision table, such automatic checking for gaps and overlaps in cell values.

Preconditions and table properties are covered in upcoming slides.

Editing cell values

- Can edit cell values directly by editing the value in place
- Can use the **Change operator** option to change the operator in a condition or action
- Can also define a custom condition or action in a cell
 - Example: Decision table contains a condition statement in a rule that has a different structure from the rest of the condition statements in that column

Unit title

© Copyright IBM Corporation 2017

Figure 11-24. Editing cell values

You can edit cell values in a decision table in different ways. For example, you can directly edit the value in the cell.

You can also change the operator for a cell, or create a cell that has a custom condition or action.



Editing in place

- To enter values into a decision table, you can type directly in cells

	Vehicle miles per year	MPY Factor
1	4000	0.6
2	[4,000 8,000[0.8
3	[8,000 15,000[1
4	[15,000 25,000[1.2
5	≥ 25,000	1.3

Unit title

© Copyright IBM Corporation 2017

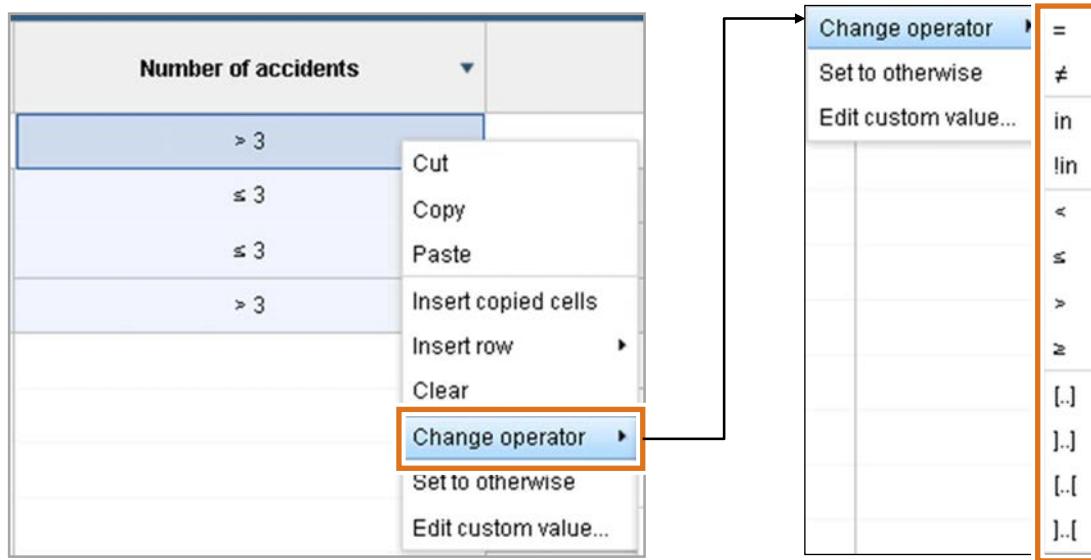
Figure 11-25. Editing in place

You can type directly in cells, one at a time. After you finish typing, click another cell to complete the edit.



Changing the operator

- To change the operator in the condition statement for a cell:
 - Right-click the cell to select **Change operator**
 - Find the operator that you need, and click to select it



Unit title

© Copyright IBM Corporation 2017

Figure 11-26. Changing the operator

If the condition statement in a cell requires a different operator than the rest of the conditions in that column, you can use the **Change operator** feature to select the operator that you need.



Custom values for cells

• Right-click the cell and click **Edit custom value**

• Define statement in the Edit Custom Value window

• Click **OK**

A screenshot of a decision table titled 'Driver age'. The table has columns 'min' and 'max'. Row 1 contains the value '65' in the 'min' cell. A context menu is open over this cell, with the option 'Edit custom value...' highlighted and surrounded by a red box.

The 'Edit Custom Value' dialog box is shown. It contains a text area with the expression: 'the age (in years) of the primary driver of 'the request' at the start date of 'the policy' is between 65 and 70'. Below the text area are tabs for Severity, Line, and Message, and a large 'OK' button at the bottom right.

Unit title

© Copyright IBM Corporation 2017

Figure 11-27. Custom values for cells

If the cell has a condition or action statement that has a different structure from the other cells in that column, you can create a custom condition or action for that cell. To define a custom value for the cell, right-click the cell and click **Edit custom value**. The Edit Custom Value window opens, where you can write the custom statement for the cell. Click **OK** to close the window.



Entering domain values

- In the column definition:
 - Select the **is one of <objects>** construct
 - Click **<objects>** and select the appropriate vocabulary

Define Condition Column

Use this editor to write a condition for the column.

the horse power of 'the vehicle' is one of <objects>

<horse power types>

Car power

Class D
<input type="checkbox"/> Class A
<input type="checkbox"/> Class B
<input checked="" type="checkbox"/> Class C
<input checked="" type="checkbox"/> Class D
<input type="checkbox"/> Class E

- In the column cell:
 - Click the column cell for the rule (or table row)
 - Select the appropriate options for the rule from the menu

Unit title

© Copyright IBM Corporation 2017

Figure 11-28. Entering domain values

In a decision table, you can enter domain values by including a domain in the column definition. You then can select the appropriate domain values for the rule (table row) by clicking the column cell and by using the selection menu.

Empty cells

- When a condition column is empty, the condition is always satisfied
- When an action column is empty, the action is ignored

	Third-party max liability amount		Base price	Variable rate
	min	max		
1	5,000	10,000	50	
2	10,000	20,000		0.005
3	≥ 20,000		80	0.001

Unit title

© Copyright IBM Corporation 2017

Figure 11-29. Empty cells

Empty cells in decision tables have different impacts, depending on the type of cell.

- When a condition column is empty, the condition is always satisfied.
- When an action column is empty, the action is ignored.

In rule 1 and rule 2 on this slide, the action column is empty. The action statements in those columns do not affect the price of the premium.

Disabling action cells

- Action phrases cannot be changed for a specific cell, but action cells can be disabled
 - When the cell is disabled, the action is ignored for that particular row
 - Equivalent to leaving the cell empty

	Third-party max liability amount		Base price	Variable rate
	min	max		
1	5,000	10,000	50	0
2	10,000	20,000	0	0.005
3	≥ 20,000		80	0.001

- To disable an action cell, right-click the cell and click **Disable**
- To enable a disabled cell, right-click the cell and click **Enable**

Unit title

© Copyright IBM Corporation 2017

Figure 11-30. Disabling action cells

When you define an action column, you cannot change the action phrase for a specific cell.

However, you can disable action cells so that the action is ignored for that particular row. Disabling a cell is equivalent to clearing the contents of the cell, except that you keep the information it contains so that it can be reenabled later if required.

Action cells can be disabled or enabled by right-clicking the cell and clicking **Disable** (for an enabled cell) or **Enable** (for a disabled cell).



Preconditions for a decision table (1 of 2)

- A precondition can:
 - Limit the scope of the rules in a decision, by applying a condition to all the rules in the table
 - Define variables that can be used in all the rules of the decision table
- All rules of the decision table are prefixed with the precondition
- If precondition is not satisfied, none of the rules can be evaluated

	Primary driver	Primary driver garage	GAL Factor
1	< 21	Suburb	2.8
2	< 21	CentralCity	2.85
3	< 21	Rural	2.4

Unit title

© Copyright IBM Corporation 2017

Figure 11-31. Preconditions for a decision table (1 of 2)

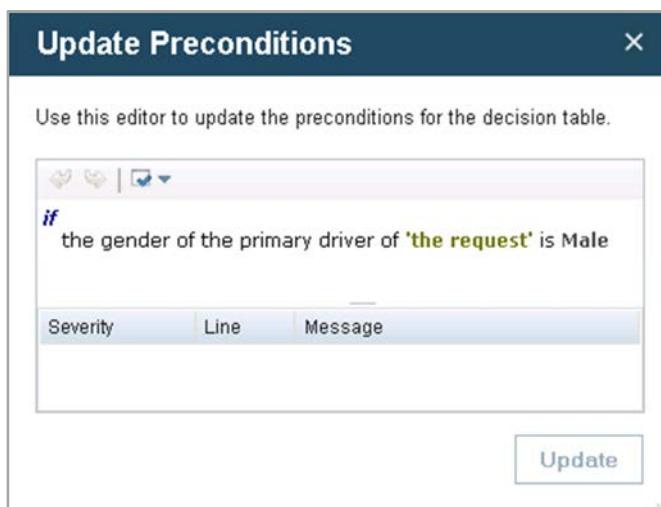
Preconditions in a decision table can be used to define variables that can be used in all the rules of the decision table. Preconditions can also limit the scope of the rules in a decision, by applying a condition to all the rules in the table.

When a precondition exists, all rules in the decision table are prefixed with the precondition. If the precondition is not satisfied, the table cannot be evaluated.



Preconditions for a decision table (2 of 2)

- Define preconditions in the decision table editor
 - Click the **Preconditions** icon in the toolbar
 - Define the decision table preconditions in the preconditions editor window



Unit title

© Copyright IBM Corporation 2017

Figure 11-32. Preconditions for a decision table (2 of 2)

To create or update table preconditions, click the **Preconditions** icon in the decision table editor toolbar to open the editor.

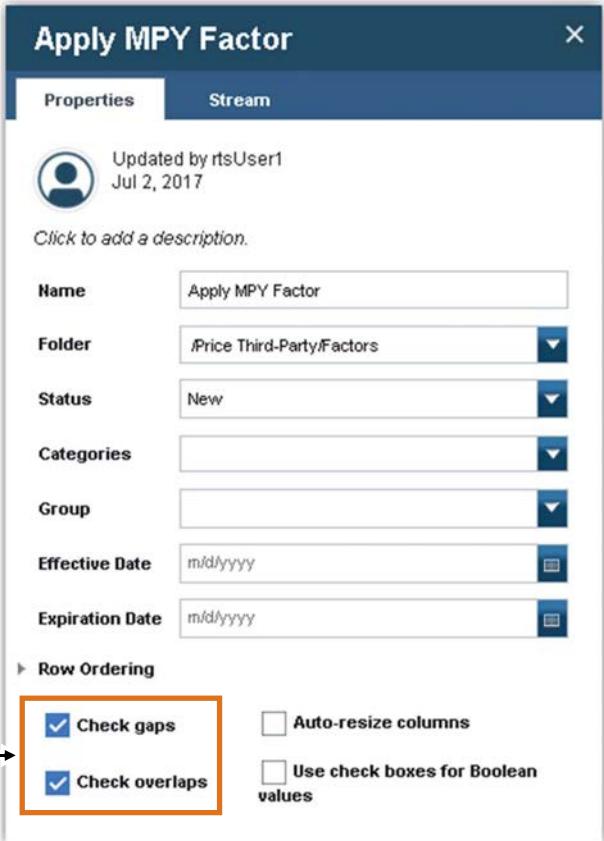



Decision table properties

- Open by clicking the **Details** icon



- Specify table options and error checking options



Unit title

© Copyright IBM Corporation 2017

Figure 11-33. Decision table properties

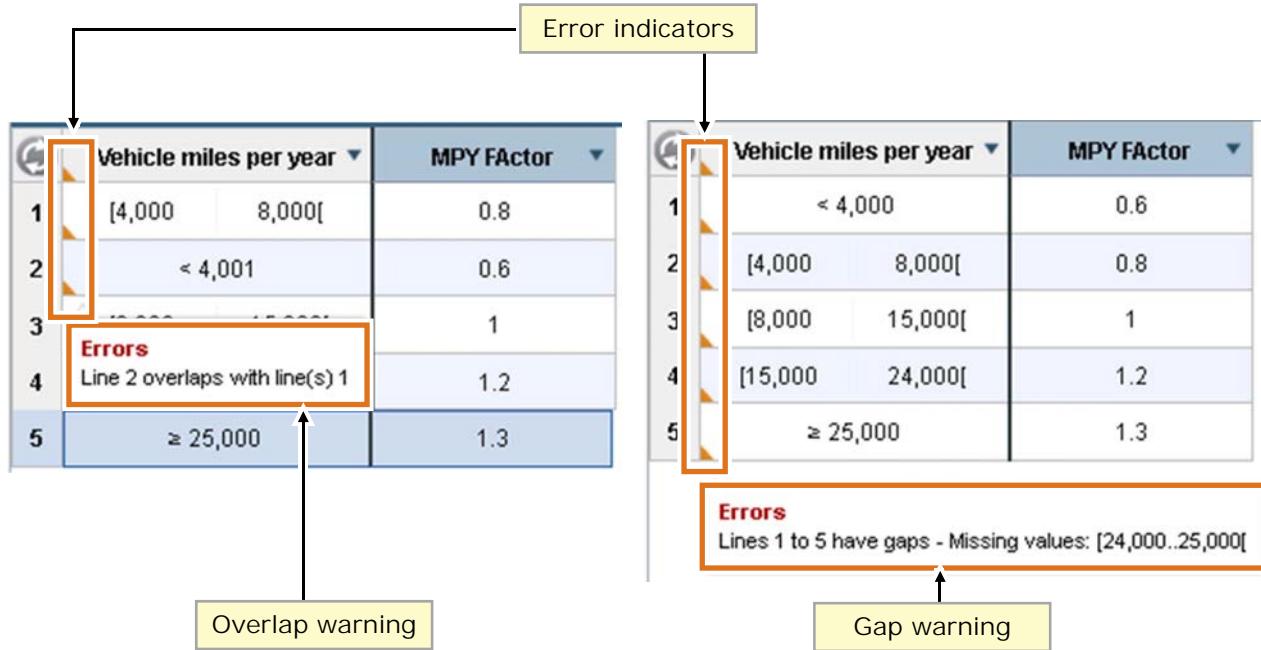
Various decision table properties can be viewed and modified through the **Properties** tab of the decision table details window. You can open the details window in the decision table editor by clicking the **Details** icon in the toolbar. Properties include the name of the decision table, the rule folder it is in, and table row ordering options.

You can also specify whether you want gap and overlap checks to be done automatically on the **Properties** tab.



Overlap and gap checks

- Automatic checks for overlaps and gaps can be enabled or disabled for the table



Unit title

© Copyright IBM Corporation 2017

Figure 11-34. Overlap and gap checks

Decision tables include checks to help you confirm that you included all the information that is required.

- Overlap checks:
 - Verify that the values you enter for a condition do not overlap and are not identical throughout the different rows.
 - Check for hierarchical overlaps (that is, overlaps involving more than one column) and distinguish between real overlaps (duplications in multiple columns) and local overlaps (duplications of a value in one column).
- Gap checks:
 - Verify that the cells in a condition column consider all possible cases to ensure that the values in the table do not have any gaps.

You can turn on automatic checks for overlaps and gaps. When the checks are enabled, you can see visual indicators of errors in the table, as shown on this slide.

By default, gap and overlap are turned on for condition columns when you create a decision table. You can control which checks are done through the **Properties** tab of the decision table details window.

Decision tables and Excel

- Can copy the contents of an Excel spreadsheet into a decision table
 - Use this feature when the spreadsheet and decision table structure are the same
- Can also export decision table to an Excel spreadsheet

Unit title

© Copyright IBM Corporation 2017

Figure 11-35. Decision tables and Excel

Other features that you might want to explore in the product documentation include decision table support for Excel spreadsheets. If the structures of the Excel spreadsheet and the decision table are the same, you can copy the contents of an Excel spreadsheet into a decision table. You can also export decision tables to Excel.

For more information about these features, see the product documentation.

Unit summary

- Describe how to express policy as decision tables
- Explain when to create decision tables versus individual action rules

Unit title

© Copyright IBM Corporation 2017

Figure 11-36. Unit summary

Review questions

1. Answer True or False for each of the following statements. Empty cells in decision tables have different impacts, depending on the type of cell:
 - A. When a condition column is empty, the condition is always satisfied.
 - B. When an action column is empty, the action is ignored.

2. Answer True or False for each of the following statements. Some of the advantages of using decision tables include the ability to:
 - A. Set preconditions on all rules in a table
 - B. Define variables that can be used in all of the decision tables in a decision service



Unit title

© Copyright IBM Corporation 2017

Figure 11-37. Review questions

Write your answers here:

1.

2.

Review answers

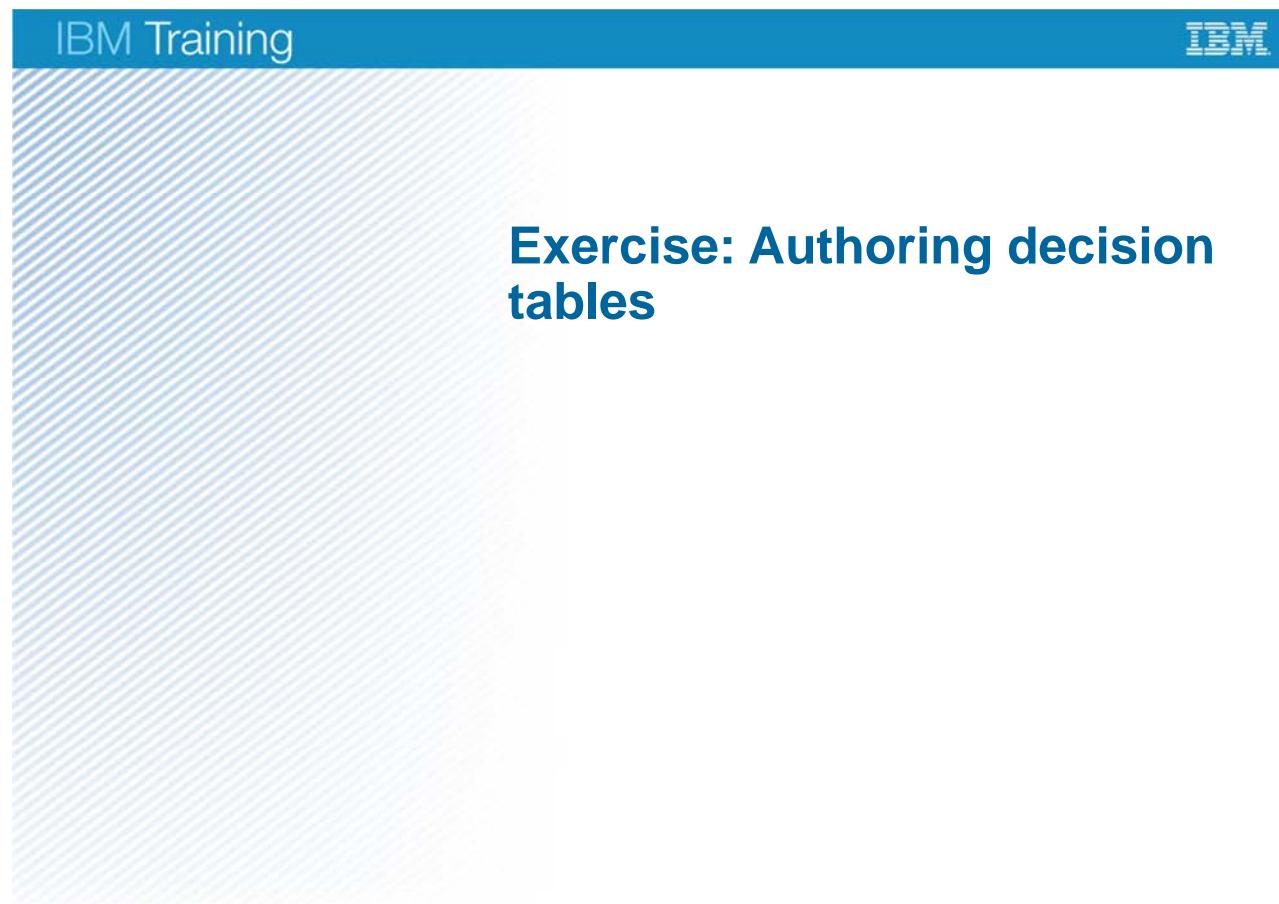
1. Answer True or False for each of the following statements. Empty cells in decision tables have different impacts, depending on the type of cell:
 - A. When a condition column is empty, the condition is always satisfied.
[The answer is True.](#)
 - B. When an action column is empty, the action is ignored.
[The answer is True.](#)
2. Answer True or False for each of the following statements. Some of the advantages of using decision tables include the ability to:
 - A. Set preconditions on all rules in a table
[The answer is True.](#)
 - B. Define variables that can be used in all of the decision tables in a decision service
[The answer is False. The variables that you define in the preconditions for a decision table apply only to the decision table in which they are defined.](#)



Unit title

© Copyright IBM Corporation 2017

Figure 11-38. Review answers



Unit title

© Copyright IBM Corporation 2017

Figure 11-39. Exercise: Authoring decision tables

Exercise objectives

- Use the decision table editor to create a decision table
- Change rule conditions in a condition cell



Unit title

© Copyright IBM Corporation 2017

Figure 11-40. Exercise objectives



Unit title

© Copyright IBM Corporation 2017

Figure 11-41. Exercise: Authoring rules: Putting it all together

Exercise introduction

- Use the rule editors to transform business policy into formal rule statements



Unit title

© Copyright IBM Corporation 2017

Figure 11-42. Exercise introduction

Unit 12. Running tests and simulations

Estimated time

01:00

Overview

This unit and the following units focus on various Decision Center tools for managing rules and decision services. In this unit, you learn how to run tests and simulations to test the effects of rule changes and validate decision logic.

How you will check your progress

- Checkpoint
- Exercise

Unit objectives

- Describe the basic features of testing and simulation
- Collaborate with developers to set up testing and simulation
- Identify the required input from the BOM and define the expected results of rule execution in scenario files to generate scenario file templates

Unit title

© Copyright IBM Corporation 2017

Figure 12-1. Unit objectives

Topics

- Overview of testing and simulation
- Working with scenarios
- Defining tests
- Defining simulations
- Collaborating with developers

Unit title

© Copyright IBM Corporation 2017

Figure 12-2. Topics

12.1. Overview of testing and simulation

Overview of testing and simulation

Unit title

© Copyright IBM Corporation 2017

Figure 12-3. Overview of testing and simulation

What is testing and simulation?

- Testing
 - Validate the correctness of a ruleset
 - Run set of rules on scenarios
 - Compare expected results with the actual results of ruleset execution
 - Define and run test suites in Business console
- Simulations
 - Evaluate the potential impact of changes to rules (“what-if” analysis)
 - Run rules on scenarios
 - Results are used to assess and refine rule behavior
 - Define and run simulation configurations in Business console
- Scenarios
 - Provide data for tests and simulations in Microsoft Excel files
 - Use real or fictitious data

Unit title

© Copyright IBM Corporation 2017

Figure 12-4. What is testing and simulation?

Decision Center provides testing and simulation capabilities that you can use to validate the behavior of rulesets.

You can use testing to validate the behavior of a ruleset by comparing the results that you expect with the actual results that are obtained during execution.

Simulations help you evaluate the potential effect of changes you might want to make to your rules, by analyzing what-if scenarios against realistic data.

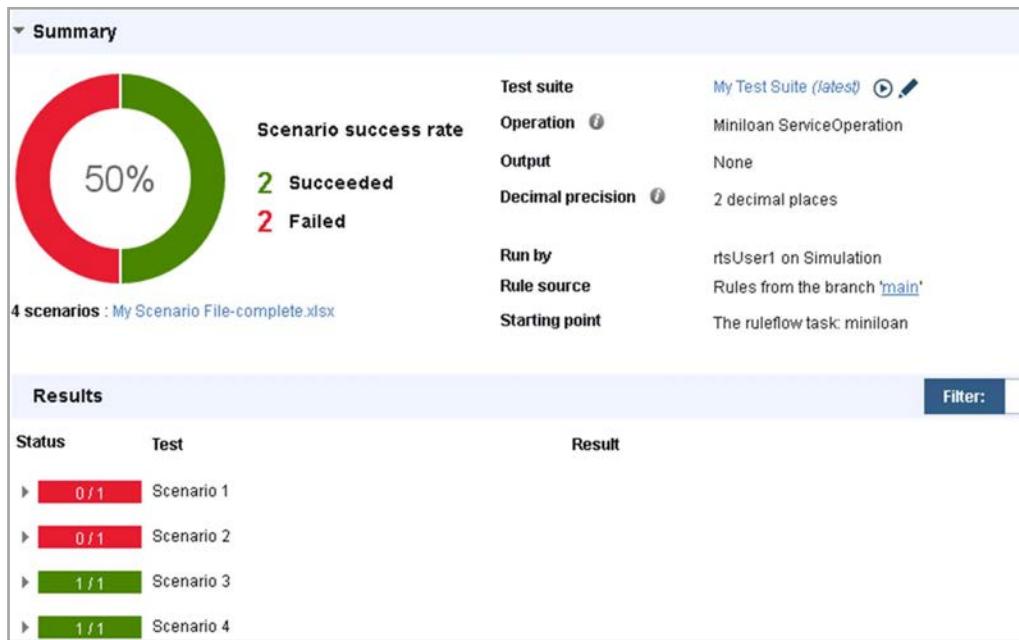
Simulation reports provide some business-relevant interpretation of the results, which is based on predefined key performance indicators (KPIs).

Both tests and simulations use *scenarios* as input, and are executed by using a test and simulation engine.



Reports

- The report summarizes the information that is generated during execution



Unit title

© Copyright IBM Corporation 2017

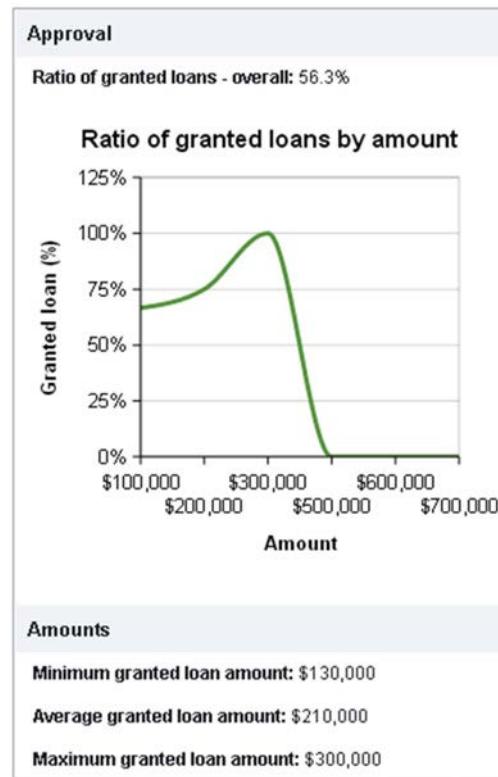
Figure 12-5. Reports

The report plays an important role in understanding your results. Running a test suite or simulation generates a large amount of information. The report organizes that information into a readable summary that you can view in Decision Center, as shown in the sample test report in the slide.

You can customize what information is included in the report when you define the test suite or simulation configuration.

KPI results

- Key performance indicators (KPIs) define:
 - How the performance of a simulation is calculated
 - How results are presented in the simulation report
- Multiple KPIs can be used for each simulation
- KPIs can be defined in the Decision Center Business console



Unit title

© Copyright IBM Corporation 2017

Figure 12-6. KPI results

To evaluate the results of simulations, the **Key Performance Indicators** section provides some business-relevant interpretation of simulation results.

Key performance indicators (KPIs) define:

- How the performance of a simulation is calculated.
- How results are presented in the simulation report.

Multiple KPIs can be used for each simulation.

The example report on this slide shows KPIs in two sections:

1. The Approval section with a chart for the ratio of loans that are granted by loan amounts.
2. The Amounts section, with KPI calculations for maximum, average and minimum amounts.

12.2. Working with scenarios

Working with scenarios

Unit title

© Copyright IBM Corporation 2017

Figure 12-7. Working with scenarios

What are scenarios?

- Provide all the necessary information that is required to validate behavior of rules
- Input values can be:
 - Use case scenarios
 - Historical databases that contain real customer information
- Default format: Microsoft Excel spreadsheet
 - Each named row represents a scenario
 - Columns indicate what data is included for each scenario
- Excel scenario files are designed for:
 - Projects with relatively simple and small object models
 - Testing that uses up to a few thousand scenarios

Unit title

© Copyright IBM Corporation 2017

Figure 12-8. What are scenarios?

Scenarios represent fictitious or actual business data that you use as input for testing or simulation. You can generate and edit scenario files in Excel 2007 – 2010. Rows in the spreadsheet represent scenarios. The Excel format is appropriate for projects with relatively simple and small object models, or testing that uses thousands of scenarios.

You can generate Excel scenario file templates from Decision Center and manually populate them with values. For more complex templates, you can ask developers to generate the templates in Rule Designer and populate them for you.

Scenario file data

- Excel scenario files hold scenario data that is used to validate rules by running tests or simulations
 - Never modify column structure of the scenario file sheets
- Contain the following sheets:
 - Scenarios:** Input data for scenarios
 - Expected Results:** Stores results that you expect from tests
- Can also contain:
 - Data entry sheets: Groups information that is used in other sheets
 - Expected Execution Details:** Stores execution details that you expect from tests

		the request			
Scenario ID	description	channel	drivers	policy	vehicle
Eligibility Scenario 1	Eligibility (should be eligible)	CarOwner	driver 1	third party insurance 1	vehicle 1
Eligibility Scenario 2	Eligibility (Car make is Ferrari)	CarOwner	driver 1	third party insurance 1	vehicle 2
Eligibility Scenario 3	Eligibility (Age of car is 41)	CarOwner	driver 1	third party insurance 1	vehicle 3

Unit title

© Copyright IBM Corporation 2017

Figure 12-9. Scenario file data

Excel scenario files contain data that you can use to validate rules in tests and simulations. These scenarios are used during tests and simulations to see whether the rules in the decision service provide the result that you expect.

The scenario files include sheets for scenario data (**Scenarios**) and expected results from test execution (**Expected Results**). They can also include:

- Data entry sheets that define data that is used in other scenario file sheets
- Expected execution details from test execution (**Expected Execution Details**)



Important

Do not modify the column structure of the scenario file sheets.

Excel scenario files and the BOM

- Excel scenario files mirror your BOM
 - When you define the scenario file template, you use the BOM class and attribute names
 - Information from the BOM is used to generate the **Scenarios** and **Expected Results** sheets of the Excel file
 - When the template file is generated, column headings use the verbalization
- Be familiar with both the BOM and its verbalization
- Understand the relationship between the Excel scenario files and classes in the BOM
- You must understand your object model and the main objects you are sending as scenario data

Unit title

© Copyright IBM Corporation 2017

Figure 12-10. Excel scenario files and the BOM

Excel scenario files mirror your BOM, so to recognize the relationship between the Excel files and the BOM, you must understand your object model, and be familiar with the BOM members *and* their verbalization.

When you create the scenario file template, you select which BOM members to include, and the template is generated with the verbalized names of the BOM members. The structure of the scenario file reflects which objects you send as input.



Relationships with the BOM

- **Scenarios** sheet

- Based on BOM classes that define ruleset input parameters
- Column headings use the verbalized names for BOM classes and attributes

		the borrower			the loan				
5	6	Scenario ID	description	name	credit score	yearly income	amount	duration	yearly interest rate
9	Scenario 1			John	600	100000	100000	240	0.05
10	Scenario 2			Jane	780	67,500	275,000	240	0.05
11	Scenario 3			Joan	600	80000	500000	240	0.05
12	Scenario 4			Joe	600	80000	250000	240	0.05

Scenarios Expected Results HELP

- **Expected Results** sheet

- Based on ruleset output parameters and BOM classes
- Column headings are defined by the verbalization of BOM class attributes

5	6	Scenario ID	the loan is approved equals
9	Scenario 1		FALSE
10	Scenario 2		TRUE
11	Scenario 3		FALSE
12	Scenario 4		TRUE

Scenarios Expected Results HELP

Unit title

© Copyright IBM Corporation 2017

Figure 12-11. Relationships with the BOM

Some columns in the scenario file are based on information from the BOM. For example, the **Scenarios** and **Expected Results** sheets of the scenario file are generated by using data from the BOM.

In the example on the slide, the **Scenarios** sheet has four different scenarios. Each scenario has an input parameter that is called **the borrower**, which has the **name**, **credit score**, and **yearly income** attributes. Each scenario also has an input parameter that is called **the loan**, which has the following attributes: **amount**, **duration**, and **yearly interest rate**.

The example **Expected Results** sheet that the result is based on the response output parameter, which includes the Boolean attribute: **the loan is approved equals**.

Scenarios

The top-level sheet is always called **Scenarios**

- The main objects that are provided as input to rule execution are shown on the **Scenarios** sheet
- Each row corresponds to a scenario

Example: Loan application

- Columns show the borrower and the loan classes and their attributes
- Two scenarios:
 - **Big Loan:** What happens when John asks for a large loan of 500000?
 - **Small Loan:** What happens when John asks for a small loan of 25000?

5	the borrower				the loan			
	Scenario ID	first name	last name	credit score	yearly income	duration	amount	rate
6	Big Loan	John	Smith	600	80000	24	500000	5
9	Small Loan	John	Smith	600	80000	24	25000	5

Scenarios HELP

Unit title

© Copyright IBM Corporation 2017

Figure 12-12. Scenarios

The top sheet of the scenario file in Excel is always called “Scenarios,” and lists the main objects that are provided as input.

The scenario file template is generated with column headings that display each input object. The headings use the verbalized name of the BOM member.

Each row in the **Scenarios** sheet represents one scenario, and each scenario must contain all of the information that is needed to process a transaction.

You complete the columns of the **Scenarios** sheet with values for each scenario in your test suite or simulation. The columns are:

- **Scenario ID:** The name that identifies the scenario. It must be unique.
- **Description:** This cell is a free-text cell that you use to describe the scenario.
- **Business terms:** Values for the scenarios, which are based on input parameters. The bold columns or subcolumns are mandatory.

In the example that is shown here, the Excel file contains two scenarios that have the descriptive scenario IDs **Big Loan** and **Small Loan**.

This example has two input objects: **the borrower** and **the loan**. The subheadings identify which object attributes to test.



Expected results (1 of 2)

Define expected results according to how you expect the rules to behave when evaluated with your scenarios

- Compare expected results to the actual results obtained from execution to see whether they match

Example: Loan application

- Shows the expected results for a loan application, based on **Big Loan** (500000) and **Small Loan** (25000) scenarios
- The three columns represent the expected results for three tests:
 - the yearly repayment equals
 - the loan report is approved
 - the message equals

5	Scenario ID	the yearly repayment equals	the loan report is approved equals	the message equals
9	Big Loan	39597	FALSE	Too big Debt-To-Income ratio
10	Small Loan	1979	TRUE	Loan approved
11				

Scenarios Expected Results HELP

Unit title

© Copyright IBM Corporation 2017

Figure 12-13. Expected results (1 of 2)

To evaluate the results after running tests, you define **expected results** for each scenario according to how you expect the rules to behave.

After test execution, a report is returned based on a comparison of the expected values, as compared to the actual results, to see whether they match.

When the expected values do not match the actual results, you can investigate whether the rule is incorrect or if the scenario values must change.

Expected Results are on the last tabbed sheet of the scenario file template.

The columns in the **Expected Results** sheet are generated according to the BOM classes that you use as ruleset output parameters, and you select which attributes you want returned to test the decision results.

In this example, you can see the expected results for the **Big Loan** and **Small Loan** scenarios. The **Big Loan** scenario results in the loan request being rejected, as it is not approved. However, the **Small Loan** scenario results in an approved loan request.

Expected results (2 of 2)

- Developers can provide a populated scenario file
 - Requires more programming time from the developers
 - Collaborate with developers to identify default values
- You can replace prepopulated results with the actual results after an initial run, if they are correct
- To populate the expected results yourself, use:
 - Default values
 - True expected values
 - Actual results that were obtained during previous rule execution

Unit title

© Copyright IBM Corporation 2017

Figure 12-14. Expected results (2 of 2)

If you need developers to provide a prepopulated scenario file, keep in mind that it does require more programming time from them. You also must help them identify which values to include for the expected results.

If you are replacing an existing system, developers might be able to populate the expected results with data from your old system. However, if the behavior of the old system is changed with the new implementation, output data might not match previous results. So expect to analyze how the logic is implemented to be sure that implementation produces the expected behavior.

You can manually edit a prepopulated scenario file, and particularly after running some tests, you might need to experiment with the values until both the rules and the results are correct.

If you are populating the expected results sheet yourself, make sure to use the following types of values:

- Default values.
- True expected values or actual results that are obtained during the previous rule execution.

Understanding results

- All reports contain these sections:
 - Summary
 - Run
- The **Summary** section shows the number of scenarios that ran and the success rate, which is the percentage of scenarios that were run successfully
- The **Run** section shows the results of each test on each scenario:
 - **Success:** Expected results match the actual results
 - **Unsuccessful:** Expected results are different from the actual results
 - **Error:** Scenarios cannot be run because of improper formatting or other errors
- When the expected values do not match the actual results
 - Investigate to determine whether the rule is incorrect
 - Check whether scenario values must change

Unit title

© Copyright IBM Corporation 2017

Figure 12-15. Understanding results

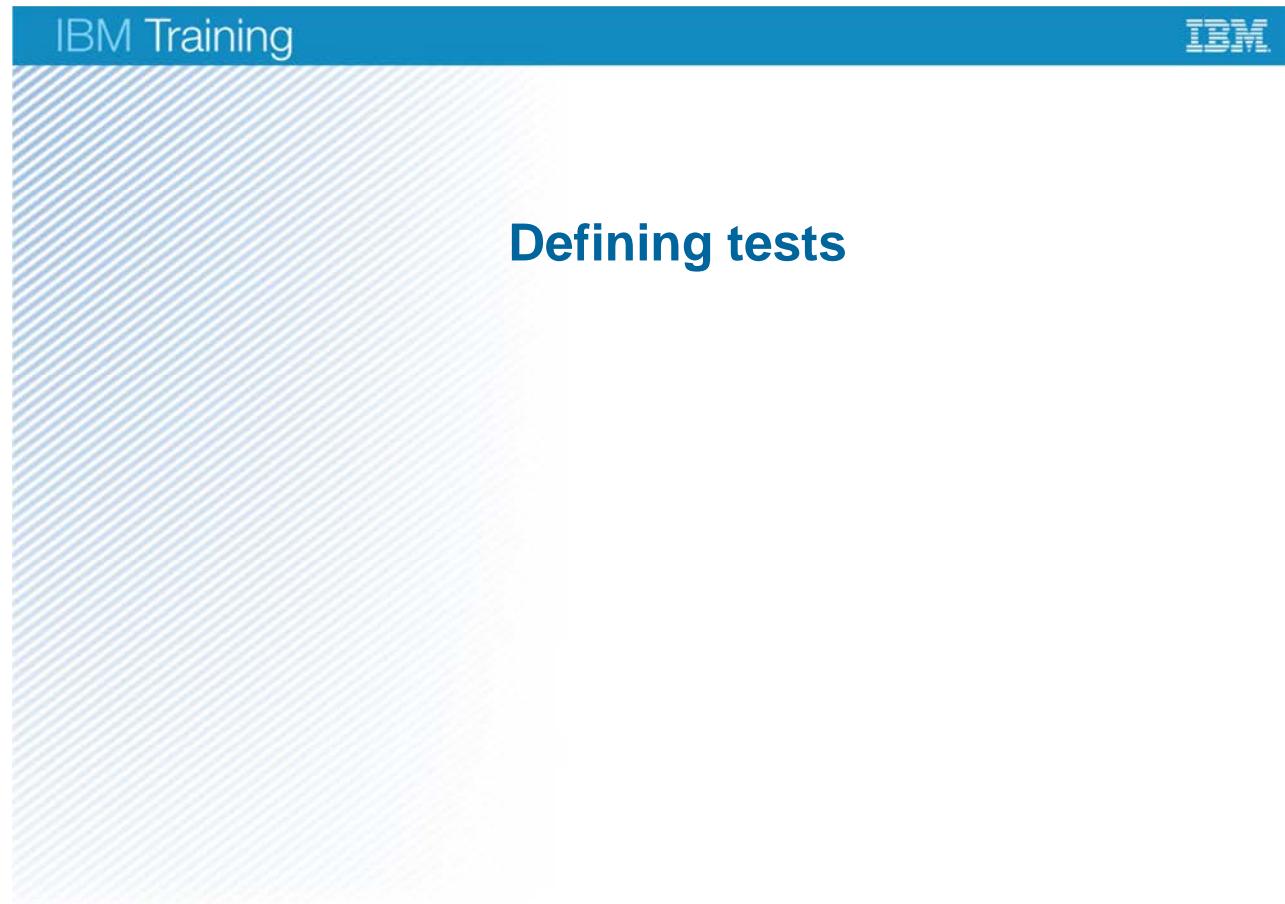
After you run a test, you use the report to evaluate results. The **Summary** section describes the results for each test. The **Run** section describes the individual results for each scenario, along with the execution details that you choose to include in the report.



Information

If the results are not what you expect, you can send your development team a file that contains everything that they require to reproduce your test environment.

12.3. Defining tests



Unit title

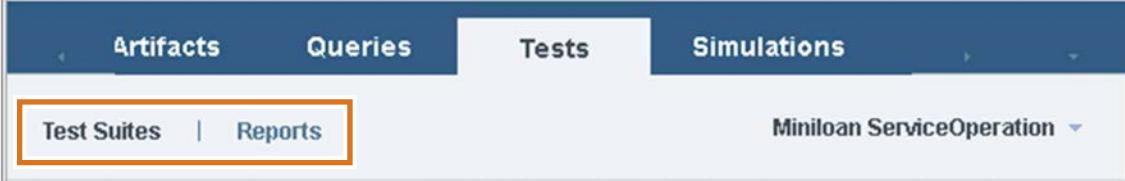
© Copyright IBM Corporation 2017

Figure 12-16. Defining tests

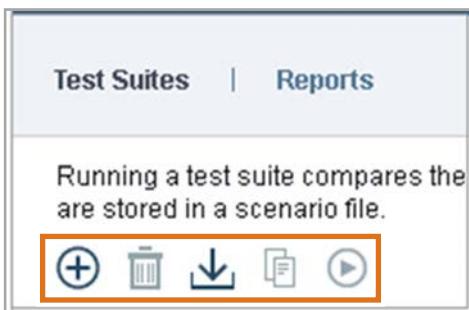
IBM Training 

Defining tests in the Business console

- **Tests** tab includes **Test Suites** and **Reports**



- On the **Tests Suites** subtab, you can use the icons to create a new test suite or to first generate a new scenario file to use in the test suite



Unit title

© Copyright IBM Corporation 2017

Figure 12-17. Defining tests in the Business console

You define test suites and view the results of test execution on the **Tests** tab.



Generating a scenario file for testing (1 of 3)

- On the **Tests** tab of the toolbar, on the **Test Suites** page, use the **Generate Scenario File** icon



- Enter a scenario file name and details in the template form

Miniloan Service > main

Generate Scenario File

Scenario files let you enter the scenario data and expected results. Generate and download the empty scenario file, define and enter scenario values using Excel, and add the completed scenario file to a test suite.

* **Filename:**
My Scenario File - 2017-07-03_03-32-26

Scenario file format:
Excel 2007 (tabbed)

Download

Unit title

© Copyright IBM Corporation 2017

Figure 12-18. Generating a scenario file for testing (1 of 3)

You can generate an Excel scenario file template on the **Test Suites** subtab of the **Tests** tab.



Generating a scenario file for testing (2 of 3)

- Choose which BOM fields to include in the **Expected Results** sheet

Operation:
Miniloan ServiceOperation

Select the tests to include in the scenario file.

* Tests:

Field	Operator
<input type="checkbox"/> the loan	equals
<input type="checkbox"/> amount	equals
<input type="checkbox"/> approval status	equals
<input type="checkbox"/> approved	equals
<input type="checkbox"/> duration	equals
<input type="checkbox"/> messages	equals (unordered)
<input type="checkbox"/> yearly interest rate	equals
<input type="checkbox"/> yearly repayment	equals

Unit title

© Copyright IBM Corporation 2017

Figure 12-19. Generating a scenario file for testing (2 of 3)

In the Tests section, select the tests that you want to include in the **Expected Results** sheet. These tests are based on the BOM.



Generating a scenario file for testing (3 of 3)

- If needed, you can choose the execution details to include in the report

Expected execution details to include in scenario file:

Field	Operator
<input type="checkbox"/> The number of rules fired	equals
<input type="checkbox"/> The number of executed ruleflow task	equals
<input type="checkbox"/> The number of rules not fired	equals
<input type="checkbox"/> The number of not executed ruleflow	equals
<input type="checkbox"/> The list of rules not fired	equals (unordered)
<input type="checkbox"/> The list of not executed ruleflow task	equals (unordered)
<input type="checkbox"/> The list of rules fired	equals (unordered)
<input type="checkbox"/> The list of executed ruleflow tasks	equals (unordered)
<input type="checkbox"/> The duration (in ms) of execution	is lower than or equals
<input type="checkbox"/> The list of rules	equals (unordered)
<input type="checkbox"/> The list of ruleflow tasks	equals (unordered)

Unit title

© Copyright IBM Corporation 2017

Figure 12-20. Generating a scenario file for testing (3 of 3)

You can also select test execution details to include in the report.



Reports for test results

- Results for each scenario include execution details that you chose in the test suite

Results			Filter:
Status	Test	Result	
▼ 0 / 1	Scenario 1		
▼	Execution Details		
	The list of rules fired	eligibility,repayment and score 2	
	✗ the loan is approved equals	false	true
▶ 0 / 1	Scenario 2		
▼ 1 / 1	Scenario 3		
▶	Execution Details		
	✓ the loan is approved equals	false	false
▶ 1 / 1	Scenario 4		

Unit title

© Copyright IBM Corporation 2017

Figure 12-21. Reports for test results

After running tests, the results for each scenario are listed. You can review the report summary and details to assess the test results.

12.4. Defining simulations

Defining simulations

Unit title

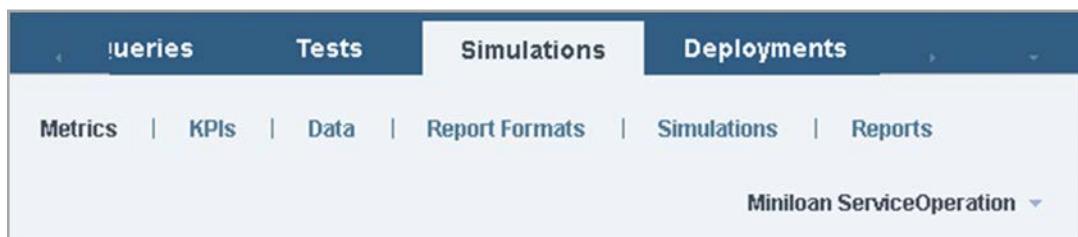
© Copyright IBM Corporation 2017

Figure 12-22. Defining simulations



Defining simulations

- Business console includes a defined workflow for assembling and running a simulation
- Workflow for setting up simulations includes defining:
 - Metrics
 - KPIs
 - Data
 - Report formats
 - Simulation configurations



Unit title

© Copyright IBM Corporation 2017

Figure 12-23. Defining simulations

Defining simulations involves several steps. The Business console takes you through the process for defining and running a simulation.

First, you create metrics that define the values that are used in KPIs. The information in the KPIs is calculated according to metrics that you define. You must define the metrics before you define the KPIs. Next, you specify which data source to use. You also define the layout for the report format to make the results readable.

The simulation, like the test suite, saves the data and report format information to be run. You can review the results in the report.

Metrics

- Define the values that are used in the KPIs
- Metric types:
 - Numeric: Number values
 - DateTime: Date and time values
 - Boolean: Checks whether a value is true or false
 - String: Checks for text values
 - Domain: Predefined values in metric expressions
- Must use a phrase to define the metric
- Use “when” to define conditions in metrics
 - Define conditions in metrics because conditions cannot be defined in KPIs
 - Example:
the amount of 'the loan' when 'the loan' is approved

Unit title

© Copyright IBM Corporation 2017

Figure 12-24. Metrics

Metric values define the values that are used in KPIs.

Metrics can be one of the following types:

- Numeric.
 - Example: the amount of 'the loan'
- DateTime.
 - Example: the start date of a third party insurance
- Boolean.
 - Example: the corporate score of 'the loan' is at least 15
- String.
 - Example: the address of 'the primary driver'
- Domain.
 - Example: In the metric expression the category of a customer, a domain metric might limit category values to Platinum, Gold, and Silver.

- In this case, the simulation would use only the input data that matches the predefined values.

You can specify a default value unless you want the KPI to determine the value.

Because conditions cannot be specified in KPIs, any conditions that you want to use in the simulation must be defined in the metric that you use in the KPI.



Important

When you create a metric or a KPI, you cannot use a one-word verbalization such as `age`. You must use a phrase such as `age of the borrower`. A short, implicit verbalization generates an error.

KPIs

- KPIs show the results from running a simulation
- KPIs pair a metric with a KPI value
- Types of KPIs
 - Scalar: Shows a single value that was taken from a group of values
 - Grouped: Shows a set of values
- KPIs can use one or two metrics
 - When you use two metrics, the second defines a distribution of values

Unit title

© Copyright IBM Corporation 2017

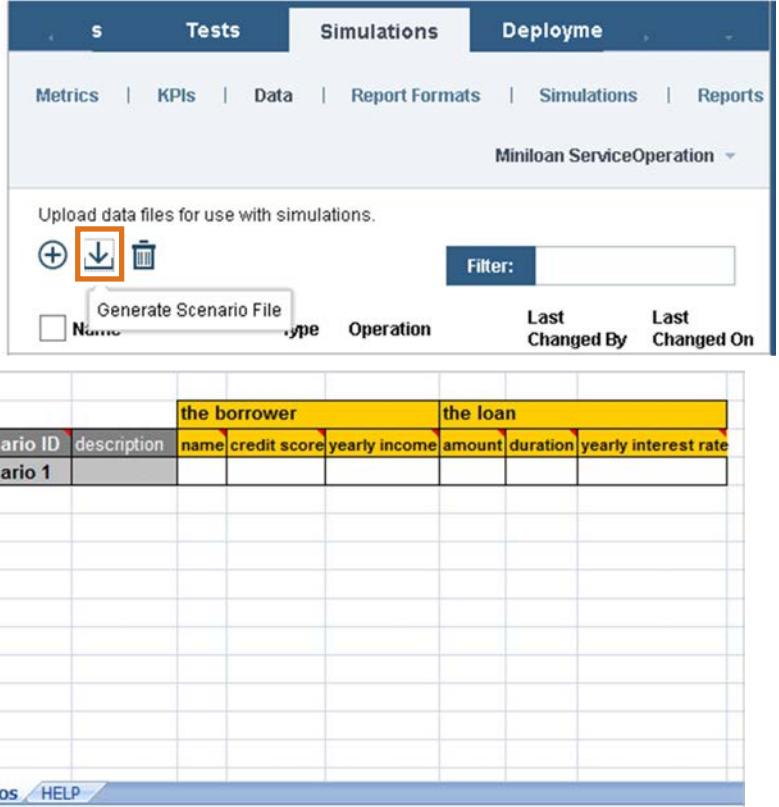
Figure 12-25. KPIs

KPIs show the results from running a simulation. When you define a KPI, you pair a metric with a KPI value.

IBM Training 

Data

- Simulation data can be provided in an Excel scenario file
- You can generate an Excel scenario file from the **Data** subtab



Scenario ID	description	name	credit score	yearly income	amount	duration	yearly interest rate
Scenario 1							

Unit title © Copyright IBM Corporation 2017

Figure 12-26. Data

Simulations use business information that is provided as input data. The data can be either real or fictitious. You can use an Excel scenario file to provide the data. For large data sets, a custom data provider can also be used as a data source. However, this unit and the accompanying exercise do not cover custom data providers.

Report formats

- As with tests, when you run a simulation, it displays its results in a report
 - Can define various reports for a simulation to generate different reports
- Define report layout by adding sections and KPIs
- Report information can be formatted as text or as a graph
- Text: For KPIs that produce a single value from a set of data
 - Can format text, including fonts and colors
- Graph: For KPIs that produce a set of comparable values
 - Graph types include bar, line, area, and pie charts

Unit title

© Copyright IBM Corporation 2017

Figure 12-27. Report formats

You define the layout for the report by choosing which KPIs to include and specifying where they should appear. You drag the KPI from the list of available KPIs on to the report template. You can also format report information as text or as a graph, depending on the values that are produced by the KPIs during the simulation.

12.5. Collaborating with developers



Unit title

© Copyright IBM Corporation 2017

Figure 12-28. Collaborating with developers

Collaborating with developers

- Varying levels of collaboration might be required for some tasks:
 - Defining the decision service with the correct signature
 - Validating the BOM and generating complex scenario file templates
 - Populating scenario files with test data from a database or other source
 - Setting up a custom data provider

- Developers also use Rule Designer for these tasks:
 - Run tests to validate the remote conditions
 - Publish the validated rule project to Decision Center

[Unit title](#)

© Copyright IBM Corporation 2017

Figure 12-29. Collaborating with developers

To enable you to run tests and simulations from the Business console, developers work in Rule Designer to:

- Define the decision service with the correct signature.
- Validate that the BOM is properly configured for testing.
- Generate an Excel scenario file template and populate the file with scenario data, if required.



Note

You can generate scenario file templates yourself in the Business console, or you can collaborate with developers for more complex scenario files. If the object model is complex, collaborating with developers can save you much time and effort.

-
- Publish the validated rule project to Decision Center.

Developers or administrators ensure that the test server is accessible to the Business console.

Unit summary

- Describe the basic features of testing and simulation
- Collaborate with developers to set up testing and simulation
- Identify the required input from the BOM and define the expected results of rule execution in scenario files to generate scenario file templates

Unit title

© Copyright IBM Corporation 2017

Figure 12-30. Unit summary

Review questions

1. True or False: You can use testing to validate the behavior of a ruleset by comparing the results that you expect with key performance indicators (KPIs).
2. True or False: Scenarios represent fictitious or actual business data that you use as input for both testing and simulation, and can be stored in Excel format.
3. True or False: Detailed reports of test or simulation results are returned to Rule Designer for analysis by developers.



Unit title

© Copyright IBM Corporation 2017

Figure 12-31. Review questions

Write your answers here:

- 1.
- 2.
- 3.

Review answers

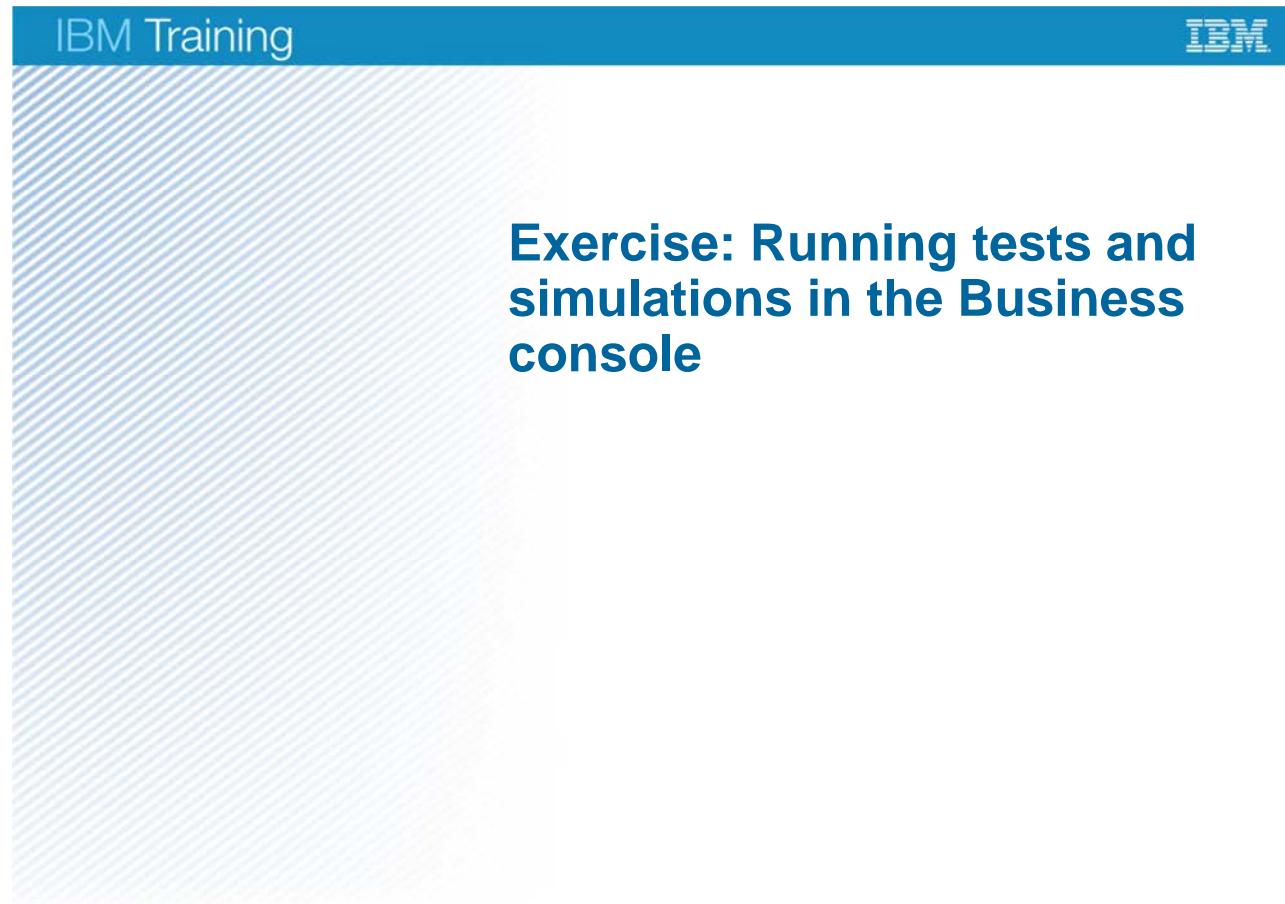
1. True or False: You can use testing to validate the behavior of a ruleset by comparing the results that you expect with key performance indicators (KPIs).
The answer is False: You can use testing to validate the behavior of a ruleset by comparing the results that you expect with the actual results that are obtained during execution.
2. True or False: Scenarios represent fictitious or actual business data that you use as input for both testing and simulation and can be stored in Excel format.
The answer is True.
3. True or False: Detailed reports of test or simulation results are returned to Rule Designer for analysis by developers.
The answer is False: HTML reports are returned directly to the Business console for you to analyze the results of your tests or simulations.



Unit title

© Copyright IBM Corporation 2017

Figure 12-32. Review answers



Unit title

© Copyright IBM Corporation 2017

Figure 12-33. Exercise: Running tests and simulations in the Business console

Exercise objectives

- Test rules by using the Microsoft Excel format to store scenarios and their expected results
- Run simulations based on scenario data



Unit title

© Copyright IBM Corporation 2017

Figure 12-34. Exercise objectives

Unit 13. Working with Decision Center administrative tools

Estimated time

01:00

Overview

This unit continues the focus on managing rules and decision services. In this unit, you learn about some administrative tools in Decision Center that you can use to manage decision service projects. You also learn how to manage user access to Decision Center.

How you will check your progress

- Checkpoint
- Exercises

Unit objectives

- Work with decision service administrative tools in the Enterprise console and the Business console
- Manage groups and users in the Business console

Unit title

© Copyright IBM Corporation 2017

Figure 13-1. Unit objectives

Topics

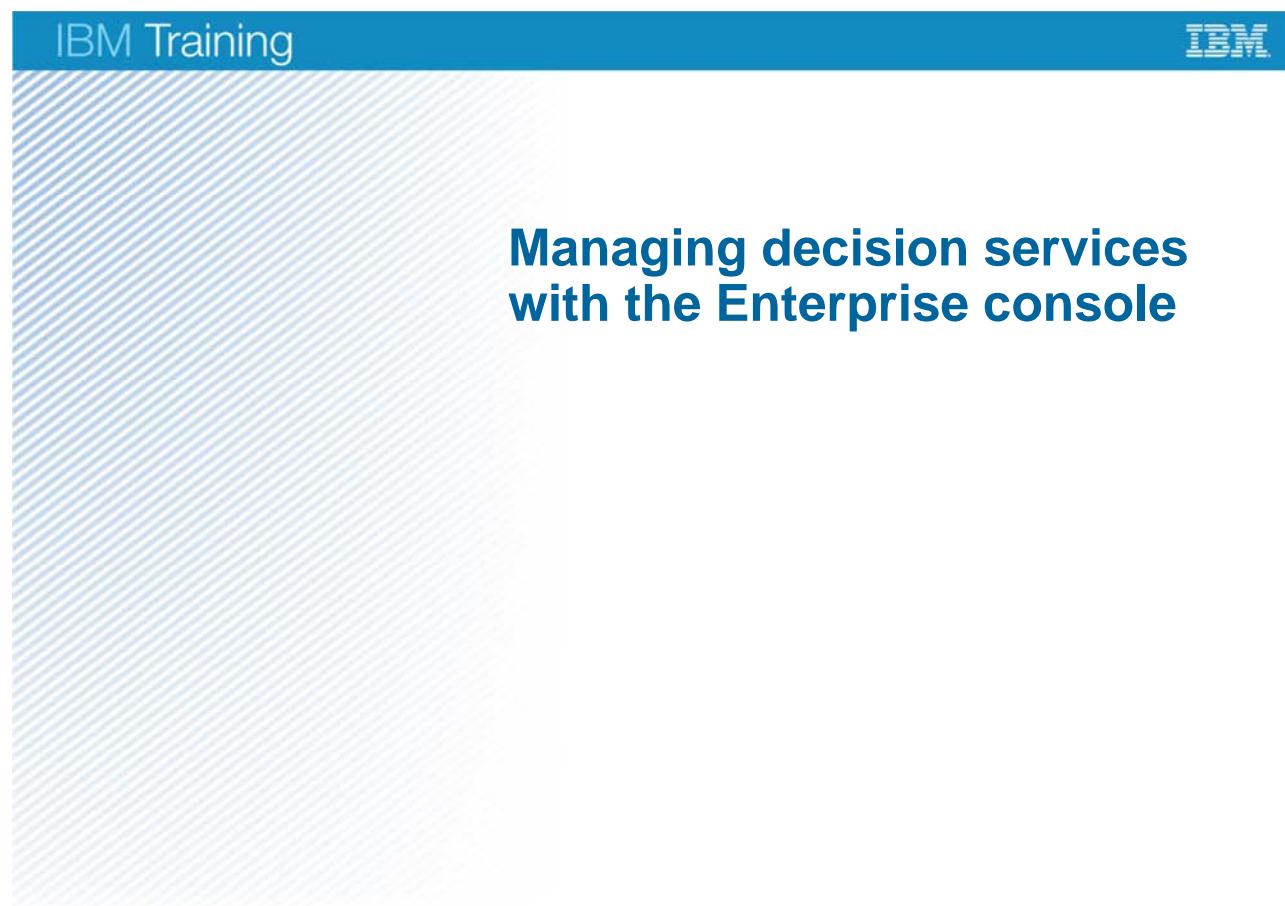
- Managing decision services with the Enterprise console
- Branch management
- Managing users and permissions

Unit title

© Copyright IBM Corporation 2017

Figure 13-2. Topics

13.1. Managing decision services with the Enterprise console



Unit title

© Copyright IBM Corporation 2017

Figure 13-3. Managing decision services with the Enterprise console



Administration tools in the Enterprise console

- Enterprise console provides administration tools to manage decision services in the Decision Center database
 - Import and export decision services
 - Erase decision services
- Administration tools available on the **Configure** tab
 - Access to these tools is limited to users with administrative privileges

The screenshot shows the IBM Decision Center Enterprise console interface. At the top, there is a navigation bar with tabs: Home, Explore, Compose, Analyze, Project, and Configure. The 'Configure' tab is highlighted with a blue background and a white border. Below the navigation bar, there is a sidebar titled 'Administration'. Inside the 'Administration' sidebar, there are several options: 'Installation Settings Wizard' (with a sub-note 'Modify an existing installation of Decision Center'), 'Diagnostics' (with a sub-note 'Run diagnostics to check the Decision Center system'), 'Clean Decision Center Cache' (with a sub-note 'Cleans the cache generated by the ruleset generation'), 'Import Projects' (with a sub-note 'Import a .zip file containing one or more projects'), 'Export Current Project State' (with a sub-note 'Export and download the current project for the selected branch or baseline'), and 'Erase Current Decision Service' (with a sub-note 'Erase the decision service, its branches, and its history. This operation cannot be undone'). An orange callout box surrounds the 'Import Projects' option.

Unit title

© Copyright IBM Corporation 2017

Figure 13-4. Administration tools in the Enterprise console

The Enterprise console offers various administrative tools that can help you manage decision services. For example, you can use the Enterprise console to import or export a decision service. You can also use the Enterprise console to erase a decision service from the Decision Center database. These tools are on the **Configure** tab.



Opening projects in Enterprise console

- On the **Home** tab, choose the project or decision service

A screenshot of the Decision Center Home Page. At the top, there is a navigation bar with tabs: Home, Explore, Compose, Query, Analyze, Project, and Configure. The Home tab is highlighted with a red box. Below the navigation bar, the page title is "Welcome to the Decision Center Home Page". There are three main sections for selecting work types:

- Work on a rule project:** Project in use: loanvalidation-rules, Branch in use: <none>, Current action: Work on branch.
- Work on a decision service (selected):** Decision service in use: Miniloan Service, Branch in use: main, Current action: Work on branch.
- Work on a governed decision service:** Decision service in use: Loan Validation Service, Release in use: Initial Release, Current change activity: <none>, Current action: Work on release.

Unit title

© Copyright IBM Corporation 2017

Figure 13-5. Opening projects in Enterprise console

In the Enterprise console, you use the **Home** tab to select a project or decision service to work on.

Importing and exporting projects in Decision Center (1 of 2)

- To share projects between separate instances of Decision Center or with the development team, you can use import and export tools
 - Projects and decision services are imported and exported as compressed files
- If the project exists in the database when you import a project, Decision Center can synchronize the projects
 - To avoid synchronizing, you can erase the existing project before importing
- Exported projects can be imported into a Rule Designer workspace or into another instance of Decision Center

Unit title

© Copyright IBM Corporation 2017

Figure 13-6. Importing and exporting projects in Decision Center (1 of 2)

You can use the import and export features to share decision services with the development team, or to share decision services between separate Decision Center instances.



Importing and exporting projects in Decision Center (2 of 2)

- To import a project or decision service:
 - Open the **Configure** tab and select **Import Projects**

- To export a project:
 - On the **Home** tab, select the project or decision service to export
 - On the **Configure** tab, choose **Export Current Project State**

Administration

[Installation Settings Wizard](#)
Modify an existing installation of Decision Center

[Diagnostics](#)
Run diagnostics to check the Decision Center system

[Clean Decision Center Cache](#)
Cleans the cache generated by the ruleset generation

Import Projects
Import a .zip file containing one or more projects

Export Current Project State
Export and download the current project for the selected branch or baseline

[Erase Current Decision Service](#)
Erase the decision service, its branches, and its history. This operation cannot be undone

Unit title

© Copyright IBM Corporation 2017

Figure 13-7. Importing and exporting projects in Decision Center (2 of 2)

To import or export decision services by using the Enterprise console, you use the **Import Projects** or **Export Current Project State** features on the **Configure** tab.



Erasing projects in Decision Center

- You can erase a project or decision service from the database
 - Open the project on the **Home** tab
 - On the **Configure** tab, select **Erase Current Decision Service**

The screenshot shows the 'Administration' section of the Decision Center interface. It includes links for 'Installation Settings Wizard', 'Diagnostics', 'Clean Decision Center Cache', 'Import Projects', 'Export Current Project State', and 'Erase Current Decision Service'. The 'Erase Current Decision Service' link is highlighted with a red border, indicating it is the focus of the figure.

Administration

[Installation Settings Wizard](#)
Modify an existing installation of Decision Center

[Diagnostics](#)
Run diagnostics to check the Decision Center system

[Clean Decision Center Cache](#)
Cleans the cache generated by the ruleset generation

[Import Projects](#)
Import a .zip file containing one or more projects

[Export Current Project State](#)
Export and download the current project for the selected branch or baseline

[Erase Current Decision Service](#)
Erase the decision service, its branches, and its history. This operation cannot be undone

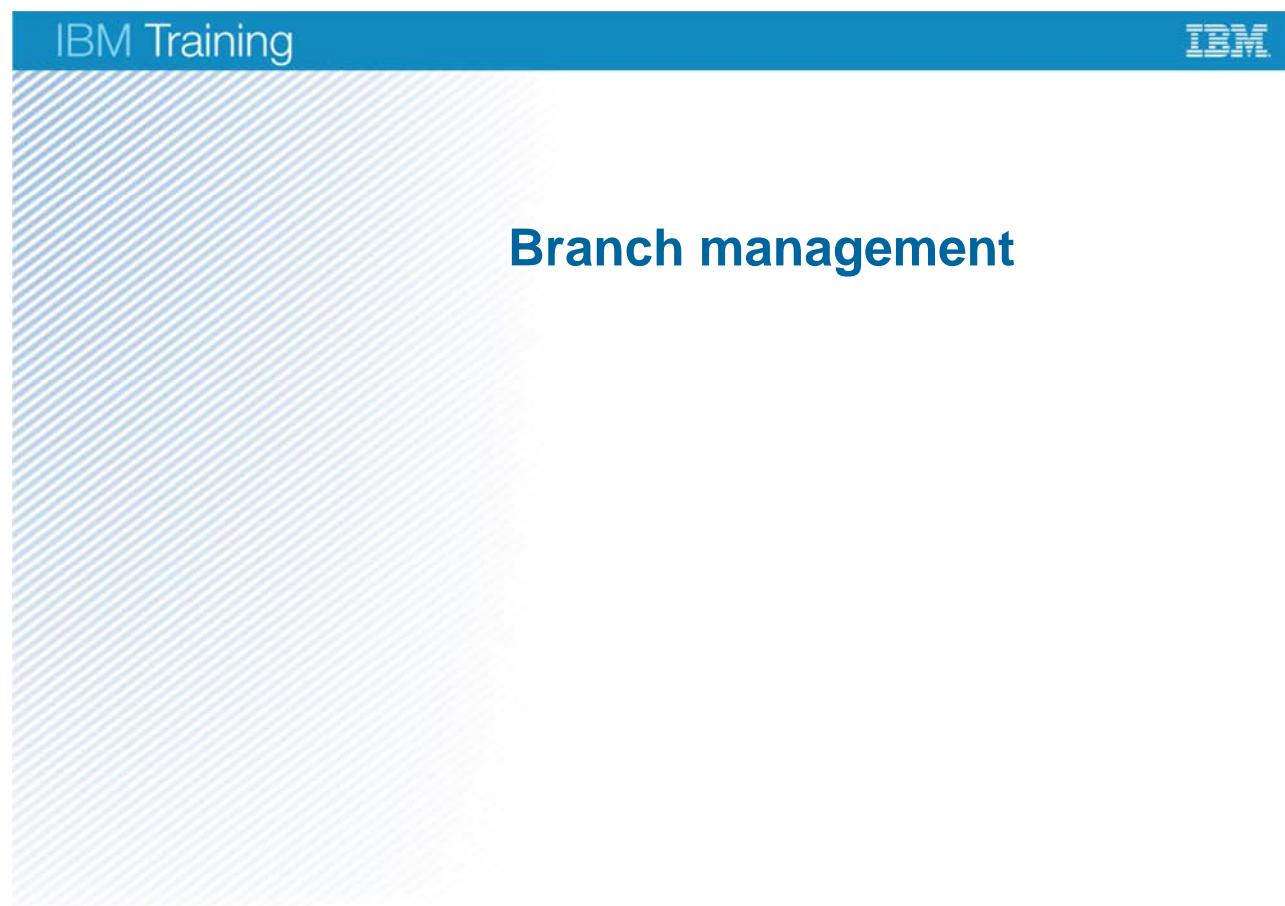
Unit title

© Copyright IBM Corporation 2017

Figure 13-8. Erasing projects in Decision Center

You can also use the Enterprise console to erase a decision service from the Decision Center database. The **Erase Current Decision Service** feature is also on the **Configure** tab.

13.2. Branch management



Unit title

© Copyright IBM Corporation 2017

Figure 13-9. Branch management

Branches

- Branches help manage project changes over time
- When a branch is created, it contains an exact replica of every project element that is contained in its parent branch
 - Work on subbranches does not affect the contents of the parent
- Branches can be:
 - Releases and change activities of a decision service
 - A regular branch of a decision service, stemming from the main branch
 - Snapshots
- When a governed decision service is published to the Decision Center database, both a **main** branch and an Initial Release are created
- When an ungoverned decision service is published, a **main** branch is created

Unit title

© Copyright IBM Corporation 2017

Figure 13-10. Branches

In the Business console, you can:

- Create branches based on the main branch.
- Rename branches.
- Merge branches.
- Delete branches.
 - However, you cannot delete the main branch.

Snapshots

- Snapshots capture the state of a branch at a specific moment in time
- You can create snapshots of the different kinds of branches, including releases and change activities
 - Snapshots can be consulted, compared, renamed, or deleted
 - Snapshot contents cannot be edited
- You can restore a snapshot so that it becomes the current state of a branch
 - You must be the owner of the release or an administrator
 - The release must be in progress, and all the activities of the release must be complete or canceled
- Automatic snapshots
 - When you create a new change activity
 - Before merging contents of a completed change activity
 - At the moment of deployment

Unit title

© Copyright IBM Corporation 2017

Figure 13-11. Snapshots

Snapshots of any branch can also be considered branches because they represent a read-only state of the rules of the branch at a past moment in time.

Managing branches in the Business console

Create new branch from the **Branches** tab

- Click **New Branch** (plus [+]) icon
- In the “Create a Branch” window, define:
 - Branch name
 - Parent branch
 - Goals for the branch

Can switch between branches

- Click the menu next to branch name to select another branch to work in



Merge branches after you complete changes to the new branch

- In the toolbar, click **Merge Branches**
- Select the branch that you want to merge with to open the merge table
 - Merge table lists all of the differences between the two branches
- Select merge options, such as merge direction and merge action
- Apply the merge

Unit title

© Copyright IBM Corporation 2017

Figure 13-12. Managing branches in the Business console

To work with branches, you must create a branch from the **Branches** tab of the decision service. To create a branch, you must define a name for the branch and designate the parent branch (that is, the branch that you are basing the new branch on). You can also describe the goals for the branch.

You can switch between decision service branches by clicking the menu next to the branch name, and selecting a different branch to work on.

After you are finished working with the new branch, you can merge it into another branch. In the toolbar, click **Merge Branches** to select the branch that you want to merge with. After you click **Merge**, you can choose various merge options, such as:

- Set individual merge actions on each branch difference in the merge table.
- Merge direction: Can merge changes to both branches, or to only one branch.
- Reset all actions: Resets the actions in the merge table to their default setting.
- Ignore all changes: Sets the action for all branch differences in the merge table to **do not modify**.

After you are finished with the options, you can then apply the merge.

13.3. Managing users and permissions



Unit title

© Copyright IBM Corporation 2017

Figure 13-13. Managing users and permissions

Mandatory role definitions to access Decision Center

- Default role definitions to access to Decision Center through the Business and Enterprise consoles:
 - rtsAdministrator
 - rtsConfigManager
 - rtsInstaller
 - rtsUser
- Every user of Decision Center, including new custom users, must belong to at least one of the mandatory roles
- You map your users and groups to these mandatory roles on the application server
- Role definitions are declared on the application server

Unit title

© Copyright IBM Corporation 2017

Figure 13-14. Mandatory role definitions to access Decision Center

Permissions are defined through roles and roles are mapped to groups. When a user logs in, Decision Center evaluates what the user can do based on the permissions that are granted to the group or groups to which the user belongs.

Custom users and groups

- Custom users and groups must be defined in the application server
- Define custom groups
 - Directly on the application server
 - By importing from an LDAP repository
- To define custom groups and users directly on the application server:
 1. Define a new group
 2. Map a mandatory Decision Center role to the group
 3. Define the new users and specify to which group or groups they belong
 4. Verify the addition of the custom users by signing in to Decision Center consoles with their credentials
- Other user and permission management tools are available in Business console

Unit title

© Copyright IBM Corporation 2017

Figure 13-15. Custom users and groups

Authentication is the process of verifying the identity of a user. You can employ any custom authentication mechanism, if the user (or user's group, as defined in the user registry) is mapped to at least one of the mandatory roles that are defined for the ODM applications.

It is not necessary for the user or group names to be identical to the role names. You can define your own custom user roles or groups if you make sure that they are mapped to the mandatory roles.

When you define custom group, you define it the application server user registry.

When a user signs in, Decision Center relies on the application server to authenticate the user. This authentication is accomplished through the communication between the server and the user registry, according to the user credentials that are defined in the user registry.



Note

For testing purposes, create a default user and password for each of your custom groups.

Authorization is the process of assigning permissions to a user. Within Decision Center, you must also do the last step that is listed here, which enables you to use the Decision Center permission

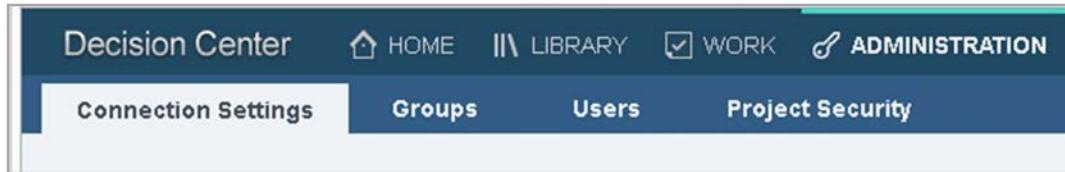
mechanism and specify fine-grained permissions on individual artifacts within a project. The authenticated user can access only artifacts and operations for which permission is authorized according to their role.

For more information, see the product documentation.



User management in Business console

- Administrator tools for user management available on the **Administration** tab



- Manage users and groups
- Assign permissions
- Create LDAP connections
- Import users and groups from LDAP

Unit title

© Copyright IBM Corporation 2017

Figure 13-16. User management in Business console

Only users who have administrator permissions can access the **Administration** tab.

Working with LDAP: Overview

- LDAP server authenticates users
 - Create users and user groups in LDAP
 - LDAP checks user credentials
- You must authorize authenticated users from LDAP and delegate authorization to Decision Center
- Requires modifying security realm of the application server
- Users and groups that are managed in your LDAP repository become visible in the application server
 - Decision Center security roles are mapped to LDAP users
 - Authorized users can sign in to Business console

Unit title

© Copyright IBM Corporation 2017

Figure 13-17. Working with LDAP: Overview

LDAP stands for Lightweight Directory Access Protocol. LDAP can be used to store user names and passwords for authentication.

In ODM, after an authenticated user is logged in, that user must be authorized to access the application. ODM user authorization is based on the groups to which the user belongs. ODM groups are based on user roles, such as rule author or administrator.

With ODM, you can configure the application server so that it uses LDAP to store user names and passwords and handle authentication. However, authorization occurs in ODM. Administrator users can use the Business console to authorize an LDAP user by assigning permissions to the group or groups to which the users belong.

Working with LDAP: Process

- Set up:
 - Configure LDAP server and create users and groups
 - Configure application server to include LDAP, including connection settings and user and group data
- In the Business console:
 - Create LDAP connection
 - Import LDAP users and groups
 - Assign roles and permissions to LDAP groups
- Must be a user with administrator permissions to work with user management in the Business console

Unit title

© Copyright IBM Corporation 2017

Figure 13-18. Working with LDAP: Process

This slide describes the process for using an LDAP server with Decision Center.

In [Exercise 17, "Managing user access in Decision Center"](#), you set up an LDAP server and connect it to it in the Business console.



Creating an LDAP connection in Business console

- Enter connection settings to a running LDAP server

Create Connection

Connection name	myLDAP
LDAP URL	ldap://localhost:10389
Search connection DN	uid=admin,ou=system
Search connection password	*****
Group search base	dc=loanserviceco,dc=com
Group search filter	member
Group name attribute	cn
User name attribute	uid
Email attribute	

Create

Unit title

© Copyright IBM Corporation 2017

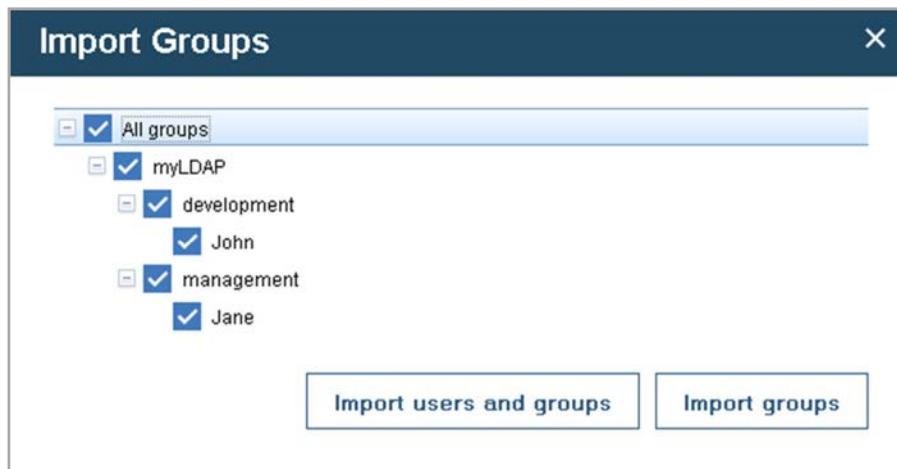
Figure 13-19. Creating an LDAP connection in Business console

You can create a connection to a running LDAP server through the Business console **Administration** tab. On the **Connection Settings** tab, click **New Connection** (the plus [+] sign icon), and enter the LDAP connection details in the form.



Importing groups and users from LDAP

- Import the LDAP users and groups into Decision Center
 - Click the **Groups** tab, and then click **Import users from LDAP** (the download icon)
 - Select the groups and users to import



Unit title

© Copyright IBM Corporation 2017

Figure 13-20. Importing groups and users from LDAP

After you create a connection to the LDAP server, you can import the LDAP users and groups into Decision Center.



Assigning permissions to imported groups

- In the **Groups** tab, edit the imported LDAP group
 - Assign roles and permissions
 - Add members

The screenshot shows the 'Edit Group' dialog box. At the top, it says 'Edit Group' and has a close button 'X'. Below that, the 'Group Name' is set to 'management'. There are four expandable sections: 'Description' (with a placeholder 'Click to add a description to this group'), 'Roles' (listing 'rtsUser' and 'rtsAdministrator'), 'Permissions' (listing 'Full Authoring'), and 'Members of this group' (which contains a plus sign icon and a 'All groups' button). At the bottom right is a 'Done' button.

Unit title

© Copyright IBM Corporation 2017

Figure 13-21. Assigning permissions to imported groups

To assign roles and permissions to an imported LDAP group, you must edit the group. Then, you can use the Roles section to add other roles to the group, and the Permissions section to designate the permissions for the group. You can also add users as members of the group.



Assigning users to groups

- In the **Users** tab, you can edit a user to add the user to the groups to which the user should belong

The screenshot shows the 'Edit a User' dialog box. At the top, there are fields for 'User name' (containing 'John') and 'Email' (with placeholder text 'Enter an email address'). Below these is a section titled 'Define the groups that the user belongs to.' Underneath is a 'Group' section with a tree view. The 'All groups' node is expanded, showing several groups: 'development' (which is checked), 'management', 'rmtUserGroup', 'rtsAdministrator', 'rtsConfigManager', 'rtsInstaller', and 'rtsUser'. A 'Done' button is located at the bottom right of the dialog.

Unit title

© Copyright IBM Corporation 2017

Figure 13-22. Assigning users to groups

You can also add a user to a group by going through the **Users** tab. Edit a user in the list to see the groups that you can add them to.

Unit summary

- Work with decision service administrative tools in the Enterprise console and the Business console
- Manage groups and users in the Business console

Unit title

© Copyright IBM Corporation 2017

Figure 13-23. Unit summary

Review questions

1. True or False: Business users with administrative privileges can manage user access and permissions in the Business console.
2. True or False: After a decision service is published to Decision Center, you cannot delete it from the database by using the Business console.



Unit title

© Copyright IBM Corporation 2017

Figure 13-24. Review questions

Write your answers here:

1.

2.

Review answers

1. True or False: Business users with administrative privileges can manage user access and permissions in the Business console.

The answer is True.



2. True or False: After a decision service is published to Decision Center, you cannot delete it from the database by using the Business console.

The answer is True. You must use the Enterprise console to delete decision services from the database.

Unit title

© Copyright IBM Corporation 2017

Figure 13-25. Review answers

Exercise: Working with management features in Decision Center

Unit title

© Copyright IBM Corporation 2017

Figure 13-26. Exercise: Working with management features in Decision Center

Exercise objectives

- Import, export, and delete decision services by using the Enterprise console
- Work with ungoverned decision service branches in the Business console



Unit title

© Copyright IBM Corporation 2017

Figure 13-27. Exercise objectives

Exercise: Managing user access in Decision Center

Unit title

© Copyright IBM Corporation 2017

Figure 13-28. Exercise: Managing user access in Decision Center

Exercise introduction

- Define custom groups and users on the application server
- Manage groups and users from an LDAP repository



Unit title

© Copyright IBM Corporation 2017

Figure 13-29. Exercise introduction

Unit 14. Introducing decision governance

Estimated time

01:45

Overview

This unit continues the focus on rule and decision service management. In this unit, you learn how to identify governance issues and how to use Operational Decision Manager features to support decision governance.

How you will check your progress

- Checkpoint
- Exercise

Unit objectives

- Explain governance issues and good practices
- Identify Operational Decision Manager features that support decision governance
- Describe the decision governance framework

Unit title

© Copyright IBM Corporation 2017

Figure 14-1. Unit objectives

Topics

- What is decision governance?
- Operational Decision Manager support for governance
- Introducing the decision governance framework

Unit title

© Copyright IBM Corporation 2017

Figure 14-2. Topics

14.1. What is decision governance?

What is decision governance?

Unit title

© Copyright IBM Corporation 2017

Figure 14-3. What is decision governance?

What is decision governance?

- Management of the lifecycle of decision logic, from initial development through to deployment and maintenance
- Provides an organizational framework that instills confidence in all stakeholders
 - Development team might hesitate to hand over control of decisions to business users
 - Business users might hesitate to accept control for fear of breaking something
- Goal
 - Ensure that business and development teams collaborate effectively
 - Ensure that project outcome meets expectations

Unit title

© Copyright IBM Corporation 2017

Figure 14-4. What is decision governance?

Governance is a broad term that involves not only defining processes, but also maintaining those processes and being able to audit them.

Decision governance is management of the decision logic lifecycle, from initial development through to deployment and maintenance.

By using the BRMS approach, business users can bypass the IT team when changing business policies so that updates can be implemented, tested, and deployed to production with minimal dependence on IT. However, when implementing a decision management solution, the development team might hesitate to hand over control of decisions to business users, and business users might also hesitate to accept control for fear of breaking something. Governance provides an organizational framework that instills confidence in all stakeholders.

The goal of governance is to ensure that business and development teams collaborate effectively, and that the project outcome meets expectations.

Why decision governance is required

- Rules and event artifacts play a dual role within an organization
 - From the business perspective, they represent critical business logic
 - From the development perspective, they are part of the actual software that runs on enterprise data
- Decision management must span business and IT teams

Unit title

© Copyright IBM Corporation 2017

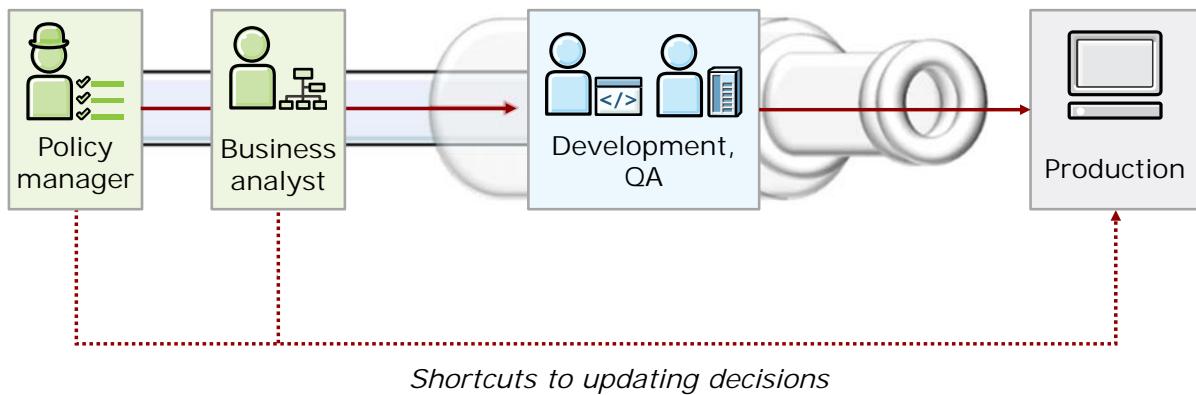
Figure 14-5. Why decision governance is required

Decision governance must span both business and IT teams so that they can work together seamlessly and efficiently from their different perspectives and tools.

IBM Training



Traditional approach to maintenance



Unit title

© Copyright IBM Corporation 2017

Figure 14-6. Traditional approach to maintenance

In the traditional approach, the IT development-QA pair can represent a bottleneck. Business policy updates, which are the new requirements that come from the business team, must go through that bottleneck.

For some short-term changes, such as promotions and specials, the cycle from requirements to production is so long that updates cannot be implemented, as promotions expire before they can be deployed.

One of the main benefits of using the BRMS approach is to avoid the bottleneck by proposing these shortcuts:

- Bypass the IT group by starting with the business analyst, and push a business policy update to production.
- The business policy manager pushes updates directly to production.
 - This direct route requires that business users have enough control of the application so that they can perform requirements gathering, analysis, implementation, testing, and deployment of the changes.

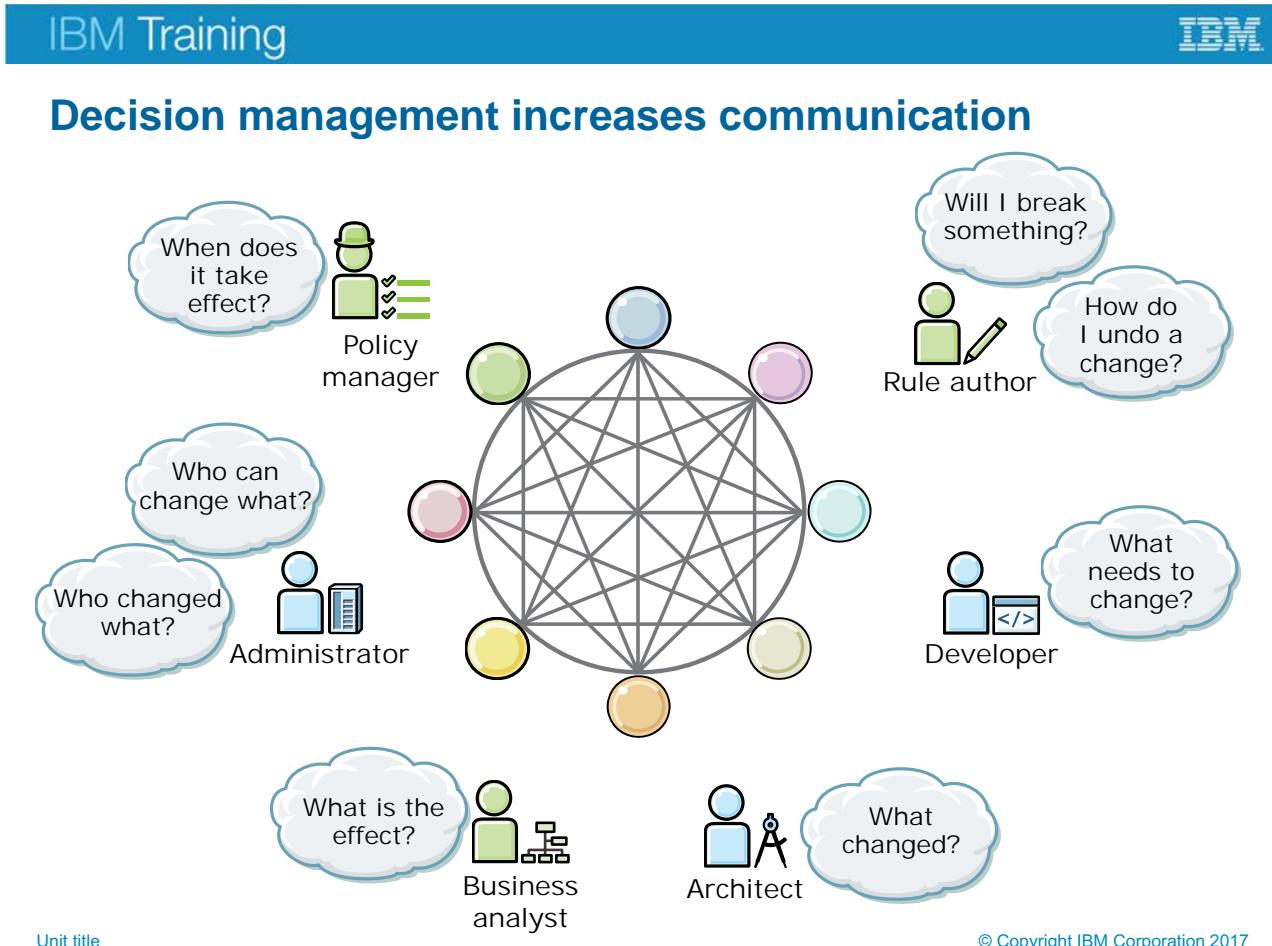


Figure 14-7. Decision management increases communication

The agile nature of the BRMS approach brings business and technical teams together regularly. The rules provide a common vocabulary for both stakeholders, resulting in increased visibility and flow of information to more people. Increased communication encourages stakeholders to take more ownership of problems and be willing to solve them.

- Business users can consult and update the business policies.
- IT support can monitor the execution of rules, and investigate when users flag problems.
- The development team can enhance the applications with new rulesets and customizations as the application grows.
- The QA team can test the rule and event artifacts and debug the application.
- Business analysts can review the vocabulary, create rules, and simulate the effect of changes.

This approach, which facilitates concurrent enhancements and updates, puts all stakeholders close to the application and requires that they work closely together. The iterative nature of Agile Business Rule Development requires frequent short questions that must be quickly answered and implemented.

Examples:

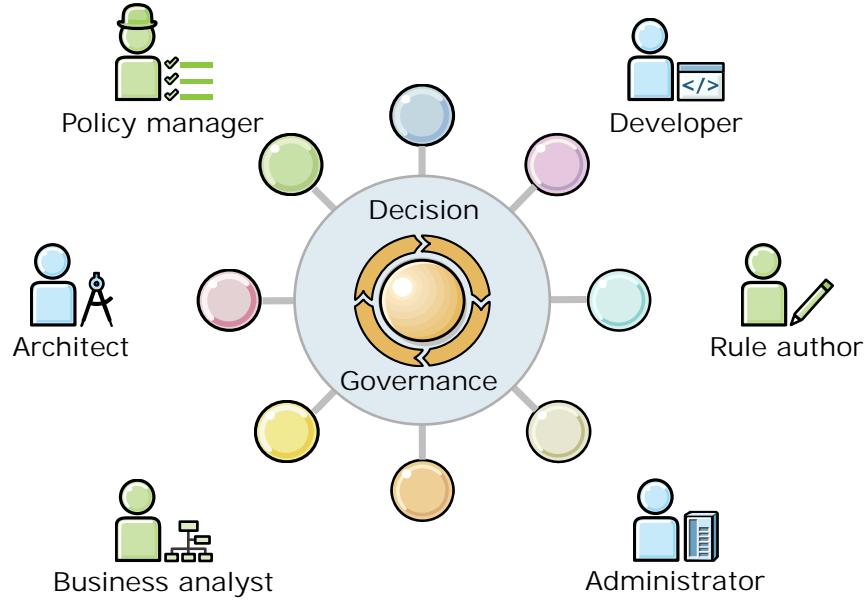
- Application lifecycle: How is this rule linked to the application?

- Permission management and security: Who is authorized to view, modify, or deploy this rule?
- Business policy applicability: Is this rule applicable in this version of the policy?
- Business safety: Are the rules that are deployed to production under control?

However, the advantages of increased agility can be lost if it is not properly governed. Conflicting interests, partial information, and office politics can lead to careless decisions if there is no process or authority with a holistic view of the system to manage change.

Decision governance goal

- Governance adds a layer of discipline to communication and change management



Unit title

© Copyright IBM Corporation 2017

Figure 14-8. Decision governance goal

Decision governance prevents the escalation of problems by providing a discipline layer to communication and change management for application maintenance.

Governance imposes a structured interaction through a set of well-defined processes.

Definition of project governance

- A purpose
 - Charter and goals clearly stated
- A definition of stakeholders
 - With their roles and responsibilities
- A process and a set of activities
- An assignment of the roles
- An entity to manage (govern) the process
- A demonstration that the process is consistently executed

Unit title

© Copyright IBM Corporation 2017

Figure 14-9. Definition of project governance

While governance implementation might vary from one organization to another and from one project to another, a basic definition of governance for a project includes the elements that are listed here.

Business Decision Management group

- Stewards of the governance processes:
 - Ensure that governance processes are properly defined and enforced
 - Address any issue that affects the project
 - Provide overall direction and advice to project managers
- Formed from among the stakeholders:
 - Business analysts and policy managers who can contribute to the governance objectives
 - Represent various stakeholders and facilitate communication between these entities (project management office, quality management, subject matter expert, and others)
- Other responsibilities:
 - Identifying business decision requirements within the organization
 - Ensuring consistency of rules across departments, functions, locations, and applications
 - Training and mentoring
 - Becoming a Center of Excellence

Unit title

© Copyright IBM Corporation 2017

Figure 14-10. Business Decision Management group

As already noted, one of the first steps to implementing governance is to define roles that are based on your organizational structure, and to set up a dedicated team to manage the business rules.

A **business rule management group** acts as a steward of the governance processes, ensuring that governance processes are properly defined and enforced. This group can be designated as a *Rule Governance Center of Excellence*.

Members must be able to address any issue that affects the project, and provide overall direction and advice to project managers.

The group is formed from core team members who are involved in initial development of the application, including business analysts and policy managers, who can contribute to the governance objectives. Include members that represent various entities and facilitate communication between these entities to ensure that expectations are met.

This group might take on more rule-related responsibilities, including:

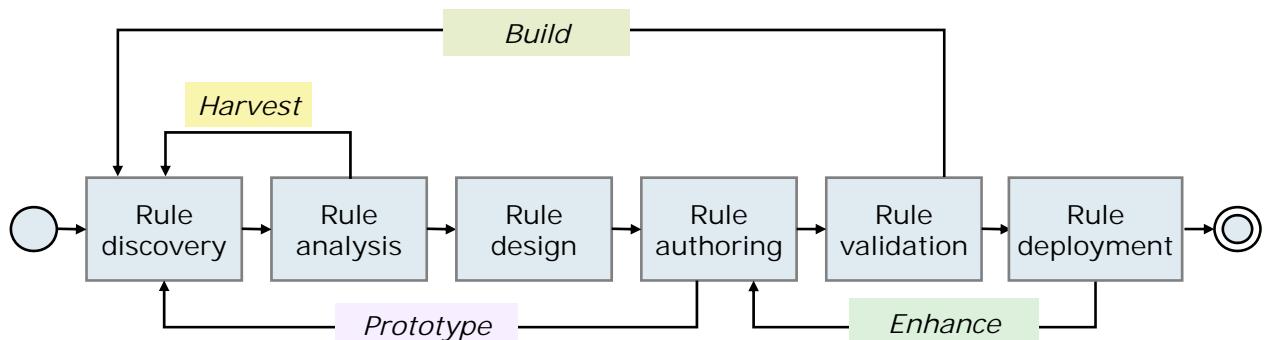
- Identifying business rule needs within the company.
- Ensuring consistency of business rules across departments, functions, locations, and applications.

- Training and mentoring.

You can avoid potential issues by establishing balanced representation of the stakeholders, and ensuring that they have a clear understanding of the purpose and motivation for such a group.

Governance and agile development

- Successful decision management projects adapt project methodology to be more iterative
 - Frequent requirement and model changes during early phases of project
 - Business owners provide early feedback on implementation
 - Respond to change instead of following a plan



Unit title

© Copyright IBM Corporation 2017

Figure 14-11. Governance and agile development

As you saw earlier in this course, the overall development of the decision follows an agile and iterative approach.

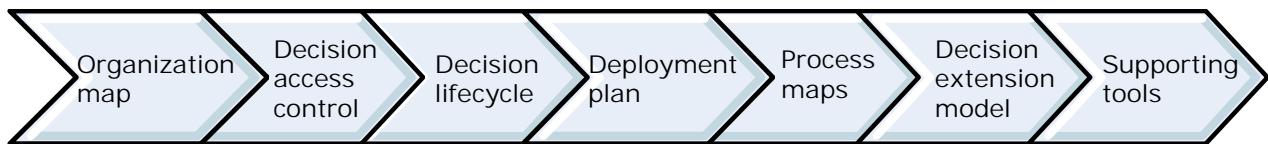
Recall that the Agile Business Rule Development (ABRD) process includes a sequence of phases. Each phase includes iterations on a set of activities. The goal is to deliver a workable set of rules: first on paper, then as a prototype in Designer, and finally published to Decision Center.

The following list describes the different phases and the tasks that are involved in each phase.

- *Rule discovery*: Harvest rules by using short workshop sessions.
 - *Rule analysis*: Understand and prepare rules for implementation.
 - *Rule design*: Define the business object model, the project structure, and the ruleset; prototype the rules.
 - *Rule authoring*: Develop the rules and test scenarios.
 - *Rule validation*: Perform functional tests that involve subject matter experts (SMEs) and their feedback.
 - *Rule deployment*: Use the Rule Execution Server staging platform.

Implementing governance

- The Agile Business Rule Development (ABRD) methodology defines a set of tasks to help implement governance
 - Develop the organization map
 - Assign responsibility and access control
 - Define the decision lifecycle
 - Define target deployment platforms, and who can deploy to which environment
 - Define each process with a Business Process Modeling Notation (BPMN) map
 - Implement a decision extension model
 - Design customizations to support the processes in Decision Center and Designer



Unit title

© Copyright IBM Corporation 2017

Figure 14-12. Implementing governance

The following set of tasks can help implement governance:

- Develop the organization map.

To develop the organization map, you first identify the project stakeholders, including internal and external groups, and try to understand their relationships along with how information flows between these groups.

An organization map defines roles, and can also involve setting up a team that is dedicated to decision management.

- Assign ruleset responsibility and access control.

Assigning an owner to a ruleset defines who is responsible for authoring and reviewing rules within that ruleset. The owner can create a table that outlines who has permission to *create*, *read*, *update*, or *delete* rules.

- Define the decision lifecycle.

Defining the lifecycle determines a status for each phase of development (such as “validated” or “deployed”), and defines who can promote a rule from one status to another.

The lifecycle forces the rule and event artifacts to go through a specific set of phases, which ensures that the artifacts pass through a testing phase. It also ensures that only certain roles have permission to do specific actions on an artifact at each phase of the lifecycle.

- Define target deployment platforms, and who can deploy to which environment.

Deployment platforms are determined according to testing requirements and application requirements. Planning the rule deployment controls how the rulesets are deployed to different server platforms: test, staging, and production.

- Define each process with a Business Process Modeling Notation (BPMN) map.

Processes include:

- Change management process
- Authoring process
- Testing process
- Deployment process
- Execution process
- Retirement process

- Implement a decision extension model.

- Design customizations to support the processes in Decision Center and Rule Designer.

Operational Decision Manager supports extending the rule model and customizing Decision Center to facilitate use of metadata and custom properties.

14.2. Operational Decision Manager support for governance

Operational Decision Manager support for governance

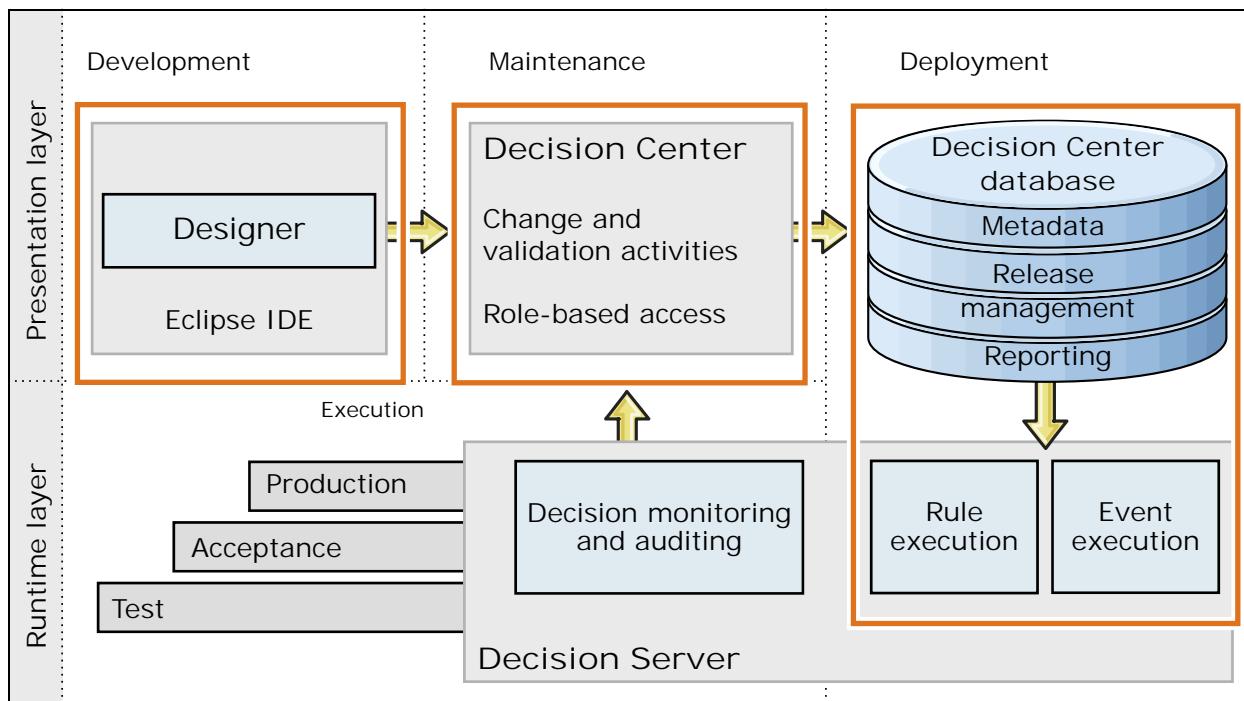
Unit title

© Copyright IBM Corporation 2017

Figure 14-13. Operational Decision Manager support for governance



Decision lifecycle in Operational Decision Manager (1 of 2)



Unit title

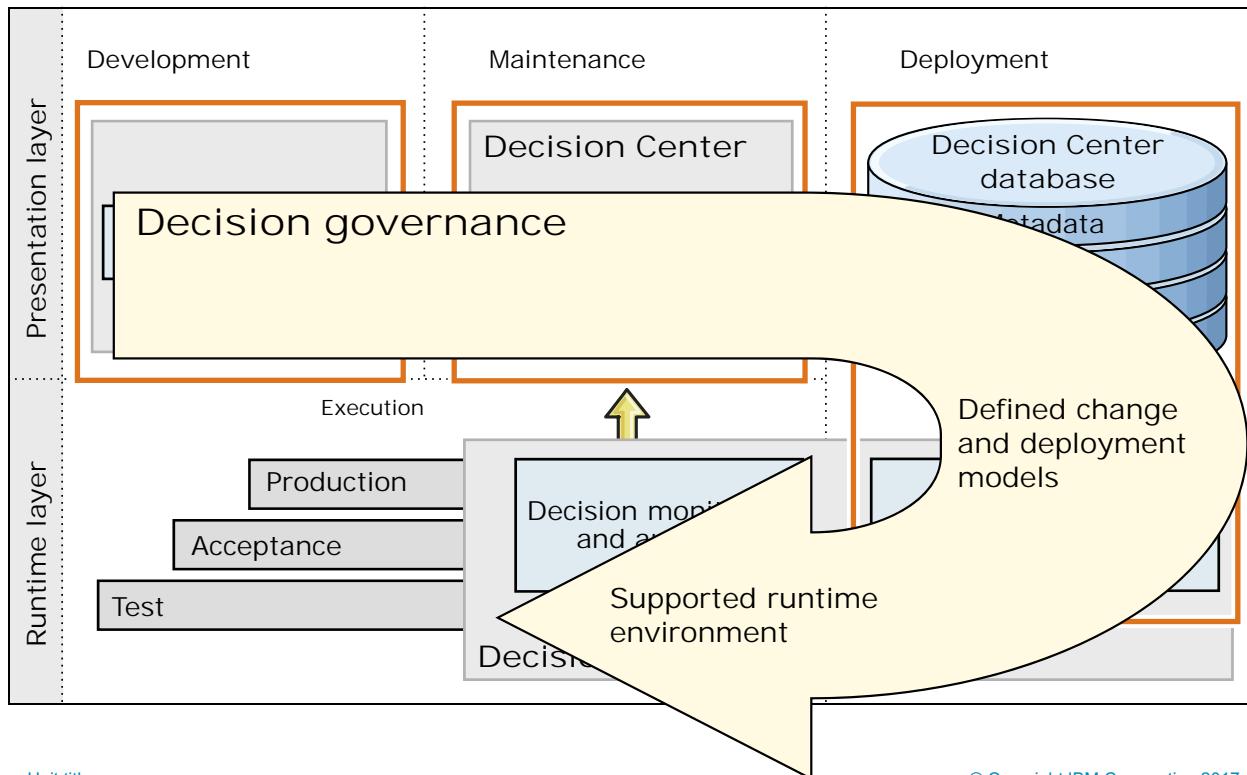
© Copyright IBM Corporation 2017

Figure 14-14. Decision lifecycle in Operational Decision Manager (1 of 2)

As seen throughout the course, ODM provides a set of mechanisms and tools to facilitate change while enforcing governance at the level of the individual decision artifacts and at the level of the overall decision.

- Developers can work in a single Eclipse-based environment.
- Decision Center provides a complete operational decision management environment for business users.
 - You can easily manage the level of access and control to enable various stakeholders to author, edit, transition, and deploy rules that are stored in the Decision Center database.
- The Decision Center database provides integrated storage for rule and event artifacts as a central “source of truth.”
 - A rich metadata layer is provided for decision logic that specifies all the custom properties that define how rules are used and how they interact with other rules in generating decisions.
- Decision Server provides a centralized execution environment to support streamlined change deployments.

Decision lifecycle in Operational Decision Manager (2 of 2)



Unit title

© Copyright IBM Corporation 2017

Figure 14-15. Decision lifecycle in Operational Decision Manager (2 of 2)

Coupled with the tools and deployment cycle that are depicted here, the suggested practices that are outlined in ABRD methodology ensure project success.

The lifecycle for rule and event artifacts is implemented through status management, access control, and permissions. Notice that this fine-grained level of governance must be defined carefully so that it does not become a hindrance to the change process.

Governance at the artifact level, while enforcing a division of responsibilities for the artifact authoring task, does not provide a vision of what happens at the decision level to manage change. Therefore, on top of artifact-level governance, a decision management solution must define the change lifecycle and governance at the level of the decision itself. The goal is to define who is responsible for which task, at what phase, and in which environment. The process defines change management of a business policy update, from its initial request by the policy manager to its deployment in the production environment, through to evaluation of the newly deployed business decision implementation. Decision changes follow a cycle of *define, deploy, measure, and update*. The key value is straightforward and predictable operational decision management.

Discussion

How can you apply governance?

Unit title

© Copyright IBM Corporation 2017

Figure 14-16. Discussion

For this discussion, follow the directions on the following slide.

If you are an instructor-led student, be prepared to discuss your ideas with the class.

If you are a self-paced student, think about the questions and write down your notes.

How can you apply governance?

1. Which tools and features did you work with during this course?
2. Considering the role or roles that you play in the decision management solution, which tools do you expect to work with?
 - Are you updating and authoring new business rules?
 - Are you involved in deploying to test or production servers, and monitoring execution?
3. How are you working with business rules?
 - How might you store and manage rule artifacts and assets?
 - Are you responsible for synchronizing artifacts across business and technical environments?
4. Based on what you learned during this course, how do you anticipate the application of governance principles and lifecycle management through the tools:
 - At the artifact level?
 - At the decision level?

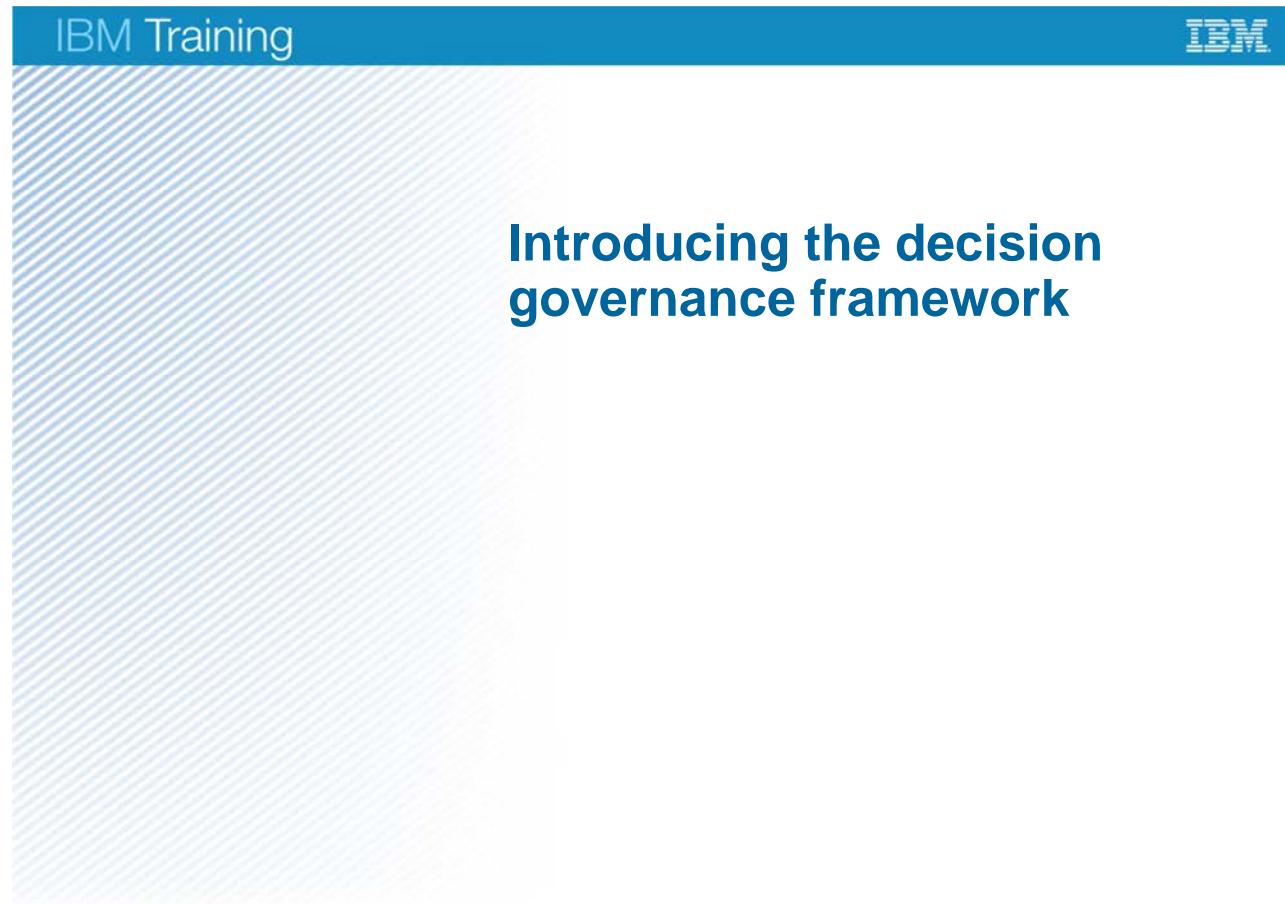
Unit title

© Copyright IBM Corporation 2017

Figure 14-17. How can you apply governance?

Before continuing, take a moment to consider what you learned in this unit by answering the questions displayed on the slide.

14.3. Introducing the decision governance framework



Unit title

© Copyright IBM Corporation 2017

Figure 14-18. Introducing the decision governance framework

Decision governance framework overview

- A well-defined decision change process that Decision Center tools support
 - User roles and permissions define who can do what
 - Releases capture and traces all changes that are related to a purpose and time period for the decision service
- Rule Designer
 - Publish decision services to Decision Center database
- Decision Center Business console
 - Work with decision service releases
 - Create, assign, and complete change activities
 - Create, assign, and complete validation activities, including testing and simulation
 - Approve change and validation activities
 - Approve releases for deployment
 - Deploy completed releases to production

Unit title

© Copyright IBM Corporation 2017

Figure 14-19. Decision governance framework overview

The decision governance framework uses tools in Rule Designer and Decision Center to provide a ready-to-use, prescriptive approach to change management and rule governance. It helps business users manage, report, and govern changes to rules and decisions.

The decision governance framework is based on decision service *releases*, which follow a well-defined change process that Decision Center tools support. Tasks that business users complete depend on user roles, such as rule author or tester.

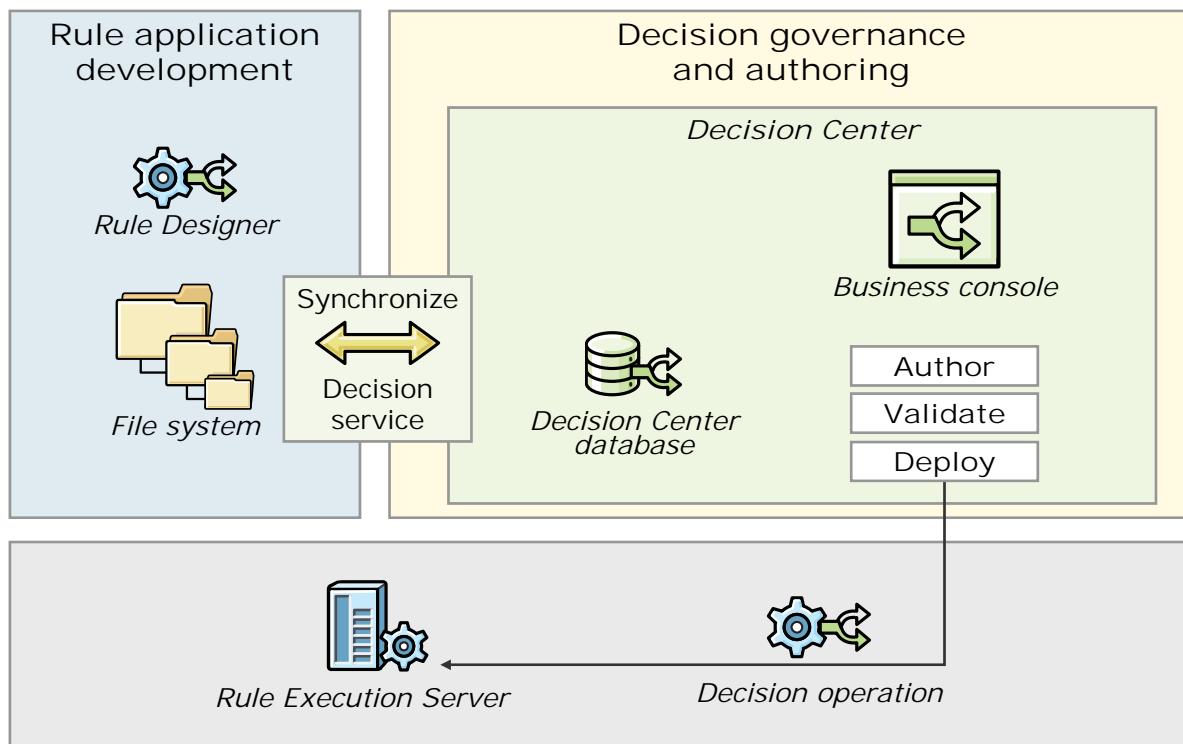
For effective workflow control, you set permissions for each phase to determine who can work with the rule during a particular phase, and who can promote an artifact from one phase to another.

Governance framework includes:

- Decision service releases.
- Change activities.
- Validation activities (testing and simulation).
- Release deployment.



Recall: Operational Decision Manager tools



Unit title

© Copyright IBM Corporation 2017

Figure 14-20. Recall: Operational Decision Manager tools

The decision governance framework in Operational Decision Manager encompasses the following tasks:

- **Synchronizing**

IT users use Rule Designer to publish a set of rule projects as a decision service that business users can access through Business console.

- **Authoring**

Business analysts and rule authors use Business console to create **change activities** and author rules within a release.

- **Validating**

Policy managers use Business console to create **validation activities** to track and manage test plans for a release. Assigned testers create and run test suites and simulations in Business console.

- **Deploying**

After all change and validation activities are complete and approved, policy managers approve the completion of the release. The release is ready for deployment. Release owners or

administrators can create deployment configurations and deploy decision services from Business console to Rule Execution Server.

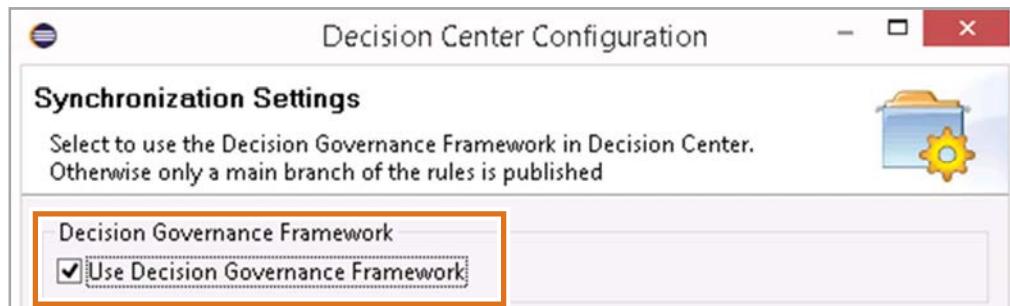
- **Administering**

Use Business console to manage user access and permissions. Additional administrative tools in Enterprise console are used to manage projects in the database. The Decision Center database provides snapshot and version features that support the auditing and rollback of business rules.



Using the decision governance framework

- Specify that a decision service is to be managed within the decision governance framework when you publish from Rule Designer to Decision Center
 - Connect to Decision Center from the top-level main project
 - Select **Use Decision Governance Framework** option



- Creates a main branch and an Initial Release branch

Unit title

© Copyright IBM Corporation 2017

Figure 14-21. Using the decision governance framework

To use the decision governance framework, you choose the decision governance framework option when you publish from Rule Designer to the Decision Center database.

You connect to Decision Center from the main project in the decision service and after the connection is established, you select **Use Decision Governance Framework**.

During the publish operation, you see the list of dependent rule projects in the decision service that are also published.

When the initial version of a decision service is published from Rule Designer, both a **main** branch and an **Initial Release** branch are created. The main branch is available if you want to work in a decision service but not use the governance framework. The Initial Release branch is published as the closed initial release of the decision service.

Governance in the Business console

Use the decision governance framework in a decision service release to manage these tasks:

- Create and manage releases
- Assign and complete change activities
 - Author and update rule artifacts
- Assign and complete validation activities
 - Run test suites and simulations
- Approve activities and releases
- Deploy decision services
 - Completed decision service releases (production or non-production)
 - From a change activity (non-production only)

Unit title

© Copyright IBM Corporation 2017

Figure 14-22. Governance in the Business console

The decision governance framework has built-in features to manage changes and support decision governance by using releases, change activities, and validation activities. You can also deploy a decision service. In the decision governance framework, you have two deployment options:

1. You can deploy completed decision service releases to either a production or non-production environment.
2. You can also deploy from a change activity, but only to a non-production environment.

States and user roles

- Governance in Decision Center is based on:
 - The **states** of decision service releases and activities
 - The **user roles** of participants who work on these releases and activities
- States
 - The state of a decision service release or activity determines what work can be done and by whom
 - The transition from one state to another can generate automatic snapshots or merges
- Roles
 - User roles have permissions for specific tasks that can be done within the context of a release

[Unit title](#)

© Copyright IBM Corporation 2017

Figure 14-23. States and user roles

The built-in governance feature in the Business console uses a workflow that is based on the status of a release or activity, and on the governance role of the participant.

The state of a release or activity can be one of the following states:

- **In Progress**
 - **Ready for Approval**
 - **Complete**
- **Canceled**
- **Rejected**

User roles include the following categories:

- All releases and activities have an **owner** and an **approver** role.
- Change activities also have an **author** role.
- Validation activities have a **tester** role.
- Users who have administrator privileges can perform the user operations of all roles.

Releases

- Captures and traces all changes to a decision service that are related to a purpose and period in time
 - A **purpose** is a set of business-driven goals
 - A **period in time** has a beginning date and an end date
- Releases are based on an existing, completed release
- Releases cannot be edited directly
 - Changes are managed through governance framework change activities
- Work on the release ends when it is completed, approved, and deployed

Unit title

© Copyright IBM Corporation 2017

Figure 14-24. Releases

A release is a branch of the rule projects that are contained in the decision service. However, a release has some characteristics that differentiate it from the ungoverned decision service branches that you worked with in [Exercise 16, "Working with management features in Decision Center"](#).

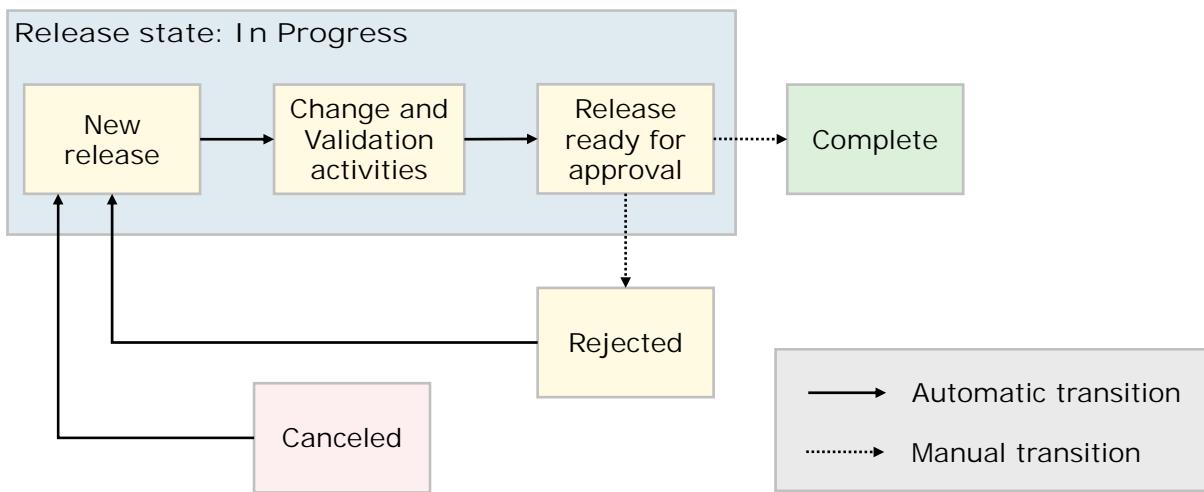
A release captures and traces all the changes that are related to a purpose and period in time. A release tracks and manages the changes that are made by its participant users and the validation of these changes.

All decision services contain a completed **Initial Release**, which you can use to create subsequent releases. A new release is based on an existing, completed release. Work occurs on the release while it is open, and it ends when you complete, approve, and deploy the release.

One important aspect of a release is that it cannot be edited directly. Instead, changes to the release occur through the completion of change activities.

Release governance

- Release governance involves management of the overall release lifecycle
 - A release is the container for rule content that is going to be deployed to production
 - Rules are maintained through change and validation activities within a release



Unit title

© Copyright IBM Corporation 2017

Figure 14-25. Release governance

The user who creates a release:

- Sets the owner and the goals of the release.
- Sets the date when the release must be completed.
- Assigns one or more participants as the approver of the release.

When a release is created, Decision Center automatically creates a snapshot.

- However, some release-related tasks are manual, such as some status changes.

When the release is in the **In Progress** state, the owner can:

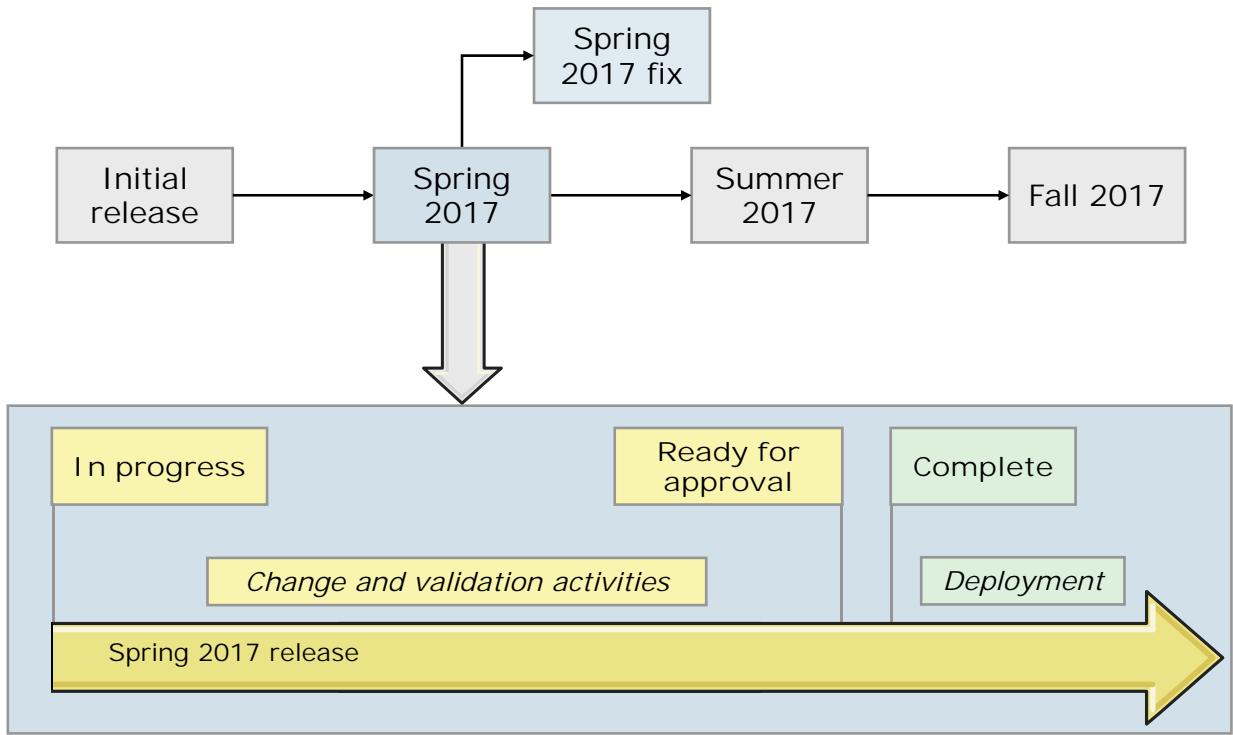
- Change the owner of the release.
- Change the goals of the release.
- Change the due date of the release.
- Create change and validation activities.

After all release activities are complete:

- The release owner changes the state of the release to **Ready for Approval**.
- Approvers can then approve or reject the changes to the release.



Release sequence



Unit title

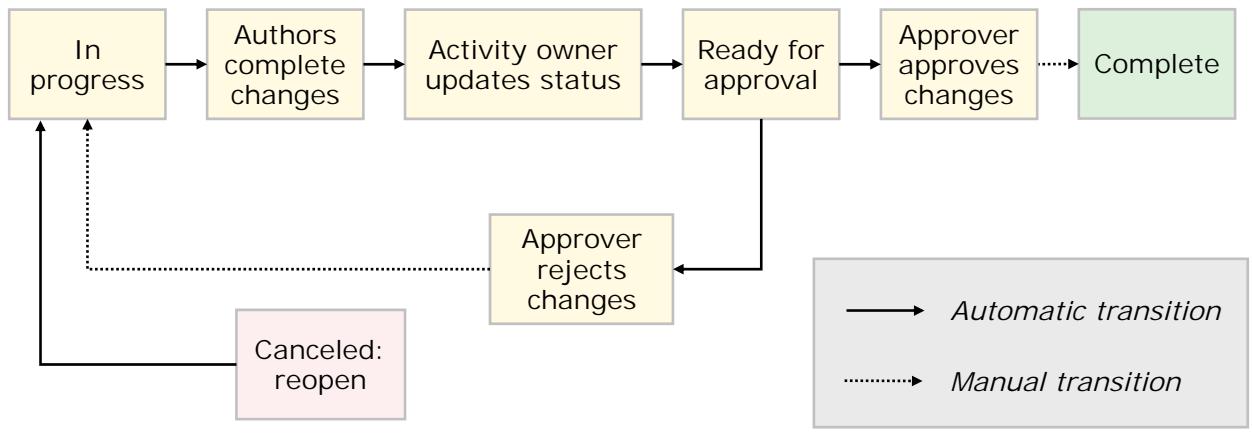
© Copyright IBM Corporation 2017

Figure 14-26. Release sequence

An initial release is automatically created when a decision service is published from Rule Designer. The content of the new release is based on the content of another complete release.

Change activity governance

- Changes to the release are initiated through the creation and completion of change activities
 - When a change activity is created, Decision Center takes a snapshot to record the starting state of the activity
 - Change activities also have states and transitions that are predefined



Unit title

© Copyright IBM Corporation 2017

Figure 14-27. Change activity governance

Rule authors are responsible for editing and updating the rules so that they align with the goals of the release. After rule authors finish their work, they can change the status of the change activity to **Finished**.

When all the authors finish their work, the owner of the change activity sets the state of the change activity to **Ready for Approval**.

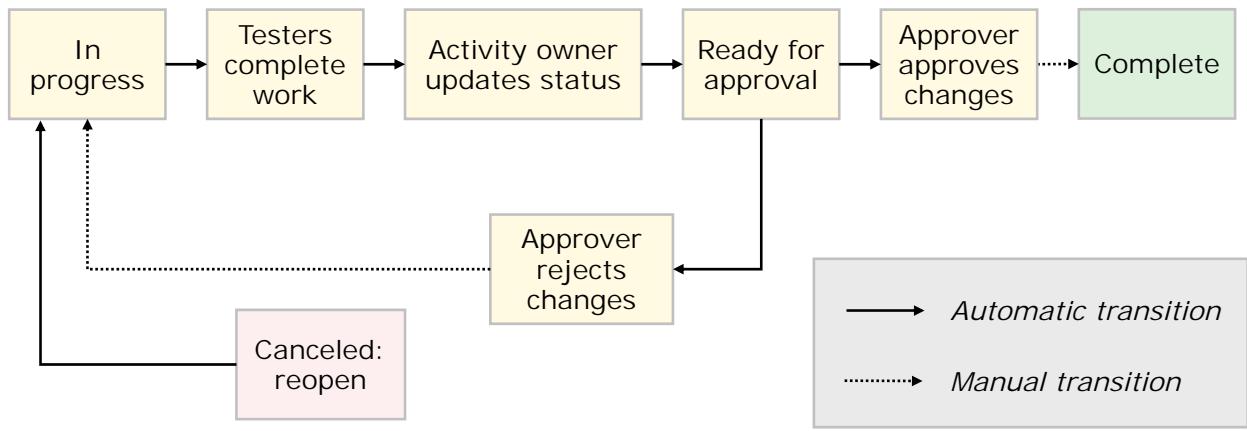
Then, the assigned approvers approve or reject the changes. If the change activity was not assigned approvers by the release owner, the change activity is automatically approved.

After all approvers approve the change activity, Decision Center takes another snapshot to record the state of the rules. Decision Center then merges the changes back into the release. The change activity is then marked **Complete**.

If a change activity is rejected, it returns to the state of **In Progress**.

Validation activity governance

- After change activities are completed, the release is validated through the creation and completion of validation activities
 - Policy managers create validation activities to track and manage test plans and test results for release to production
 - Validation tasks include creating test plans, running test suites, and running simulations
 - Tests and simulations can be run only in a validation activity



Unit title

© Copyright IBM Corporation 2017

Figure 14-28. Validation activity governance

Testers run different tests and simulations that are aimed at validating a release. After testers complete their work, and all the change activities of the release are **Complete**, they change their status to **Finished**.

The owner of the validation activity sets the validation activity state to **Ready for Approval**, at which point the approvers approve or reject the activity.

After all the approvers approve the validation activity, Decision Center sets the state of the validation activity to **Complete**.

Deploying releases

- After all activities in a release are complete, the release owner approves the release itself
- Approved releases can be deployed from Business console
- Only completed releases can be deployed to production
 - Change activities and branches can be deployed, but only to non-production environments

Unit title

© Copyright IBM Corporation 2017

Figure 14-29. Deploying releases

Unit summary

- Explain governance issues and good practices
- Identify Operational Decision Manager features that support decision governance
- Describe the decision governance framework

Unit title

© Copyright IBM Corporation 2017

Figure 14-30. Unit summary

Review questions (1 of 2)

1. True or False: Rules play a dual role as business assets that represent business logic and as part of the actual software that runs on enterprise data.

2. Governance encompasses which of the following tasks?
Select all that apply.
 - A. Defining expectations and assigning responsibilities
 - B. Establishing efficient collaboration between the business and IT teams
 - C. Selecting a group of productive developers to be in charge of business rule management and reporting their decisions to business stakeholders



Unit title

© Copyright IBM Corporation 2017

Figure 14-31. Review questions (1 of 2)

Write your answers here:

1.

2.

Review questions (2 of 2)

3. True or False: Implementing decision governance is outside the scope of the IT department.
4. True or False: Rules in a decision governance framework decision service release are maintained through change and validation activities.



Unit title

© Copyright IBM Corporation 2017

Figure 14-32. Review questions (2 of 2)

Write your answers here:

3.

4.

Review answers (1 of 2)

1. True or False: Rules play a dual role as business assets that represent business logic and as part of the actual software that runs on enterprise data.
The answer is True.

2. Governance encompasses which of the following tasks?
Select all that apply.
 - A. Defining expectations and assigning responsibilities
True.
 - B. Establishing efficient collaboration between the business and IT teams
True.
 - C. Selecting a group of productive developers to be in charge of business rule management and reporting their decisions to business stakeholders
False. Include business and developer stakeholders to ensure balanced representation for decision making.



Unit title

© Copyright IBM Corporation 2017

Figure 14-33. Review answers (1 of 2)

Review answers (2 of 2)

3. True or False: Implementing decision governance is outside the scope of the IT department.

The answer is False. Implementing rule governance must include collaboration from both business and IT stakeholders to provide an organizational framework that instills confidence in all stakeholders.



4. True or False: Rules in a decision governance framework decision service release are maintained through change and validation activities.

The answer is True.

Unit title

© Copyright IBM Corporation 2017

Figure 14-34. Review answers (2 of 2)

Exercise: Working with the decision governance framework

Unit title

© Copyright IBM Corporation 2017

Figure 14-35. Exercise: Working with the decision governance framework

Exercise objectives

- Work with releases
- Work with change and validation activities
- Deploy decision service releases



Unit title

© Copyright IBM Corporation 2017

Figure 14-36. Exercise objectives

Unit 15. Course summary

Estimated time

00:30

Overview

This unit summarizes the course and provides information for future study.

Unit objectives

- Explain how the course met its learning objectives
- Access the IBM Training website
- Identify other IBM Training courses that are related to this topic
- Locate appropriate resources for further study

Course summary

© Copyright IBM Corporation 2017

Figure 15-1. Unit objectives

Course objectives

- Describe the benefits of implementing an Operational Decision Manager solution
- Identify the main user roles and tasks that are involved in designing and developing an Operational Decision Manager solution
- Explain modeling concepts and the UML notation that is relevant to modeling for business rules
- Define and implement object models for business rules
- Set up the rule authoring environment in Designer by working with decision services and synchronizing across development and business environments
- Transform business policy into rule statements and make sure that they form a complete and coherent set of rules
- Use the Operational Decision Manager rule editors to author business rules and decision tables

Course summary

© Copyright IBM Corporation 2017

Figure 15-2. Course objectives

Course objectives

- Run tests and simulations in the Decision Center Business console to validate decision logic and rule changes
- Work with Decision Center decision service administration tools
- Manage user access and permissions in the Business console
- Use Operational Decision Manager tools to support decision governance

[Course summary](#)

© Copyright IBM Corporation 2017

Figure 15-3. Course objectives

Earn an IBM Badge

- After completing this course, you might be ready to take an IBM Badge test
- Use IBM Badges to share verified proof of your IBM credentials
- Find the your Badge test on this site:
 - <https://www.ibm.com/services/learning/ites.wss/zzen?pageType=badgesearch>
- The *IBM Operational Decision Manager Standard V8.9 Business Analyst* Badge test requires this course:
 - **WB401 / ZB401:** Managing Decisions in IBM Operational Decision Manager V8.9
- After completing this course, take the Badge test



Course summary

© Copyright IBM Corporation 2017

Figure 15-4. Earn an IBM Badge

IBM Training



To learn more on the subject

- IBM Training website:
www.ibm.com/training
- IBM Operational Decision Manager
www.ibm.com/software/products/en/odm
- Decision management:
www.ibm.com/software/decision-management
- Other IBM websites:
 - www.ibm.com/middleware
 - www.redbooks.ibm.com
 - www.ibm.com/developerworks

Course summary

© Copyright IBM Corporation 2017

Figure 15-5. To learn more on the subject

Enhance your learning with IBM resources

Keep your IBM Cloud skills up-to-date

- IBM offers resources for:
 - Product information
 - Training and certification
 - Documentation
 - Support
 - Technical information



- To learn more, see the IBM Cloud Education Resource Guide:
 - www.ibm.biz/CloudEduResources

Course summary

© Copyright IBM Corporation 2017

Figure 15-6. Enhance your learning with IBM resources

Unit summary

- Explain how the course met its learning objectives
- Access the IBM Training website
- Identify other IBM Training courses that are related to this topic
- Locate appropriate resources for further study

Course summary

© Copyright IBM Corporation 2017

Figure 15-7. Unit summary



Course completion

You have completed this course:

Managing Decisions in IBM Operational Decision Manager V8.9

Any questions?



Course summary

© Copyright IBM Corporation 2017

Figure 15-8. Course completion

Appendix A. List of abbreviations

ABRD	Agile Business Rule Development
API	application programming interface
ATM	automatic teller machine
B2X	BOM-to-XOM mapping
BA	business analyst
BAL	Business Action Language
BEP	business event processing
BOM	business object model
BPM	business process management
BPMN	Business Process Modeling Notation
BQL	Business Query Language
BRD	Business Rule Document
BRM	business relationship manager
BRMS	business rule management system
CPU	central processing unit
CSR	customer service representative
DHCP	Dynamic Host Configuration Protocol
DN	distinguished name
DVS	Decision Validation Services
EE	Enterprise Edition
ERC	edition revision code
GUI	graphical user interface
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
IBM	International Business Machines
IDE	integrated development environment
IP	Internet Protocol
IRL	ILOG Rule Language
ISSIS	IBM Systems Solution Implementation Standard

IT	information technology
Java EE	Java Platform, Enterprise Edition
Java SE	Java Platform, Standard Edition
KPI	key performance indicator
LAN	local area network
LDAP	Lightweight Directory Access Protocol
LOB	line of business
NSF	non-sufficient funds
ODM	Operational Decision Manager
OO	object-oriented
PDM	product data management
PMML	Predictive Model Markup Language
POS	point-of-sale
PVU	Processor Value Unit
QA	quality assurance
RES	Rule Execution Server
REST	Representational State Transfer
RFID	Radio Frequency Identification
RTS	Rule Team Server
SCC	source code control
SE	Standard Edition
SME	subject matter expert
SOA	service-oriented architecture
SPSS	Statistical Package for the Social Sciences
SQL	Structured Query Language
SSP	Scenario Service Provider
TCP/IP	Transmission Control Protocol/Internet Protocol
UC	use case
UML	Unified Modeling Language
URL	Uniform Resource Locator
VAT	value-added tax
VIN	vehicle identification number
XML	Extensible Markup Language
XOM	execution object model

XP	extreme programming
XSD	XML Schema Definition
XU	execution unit
z/OS	zSeries operating system

Appendix B. IBM ODM on Cloud

Estimated time

00:15

Overview

This appendix describes Operational Decision Manager on Cloud.

Introduction to IBM ODM on Cloud

- Enterprise-grade ODM cloud service for development, testing, and production
- Cloud-based, collaborative, and role-based environment
 - Capture, automate, and manage frequently occurring, repeatable rules-based business decisions
- Ready-to-use development, test, and production environments are available
- Monthly subscription plans
- Available exclusively on IBM Cloud infrastructure
- Managed by IBM
- Artifacts that are created with IBM ODM on Cloud are compatible with IBM ODM on-premises product

Unit title

© Copyright IBM Corporation 2017

Figure B-1. Introduction to IBM ODM on Cloud



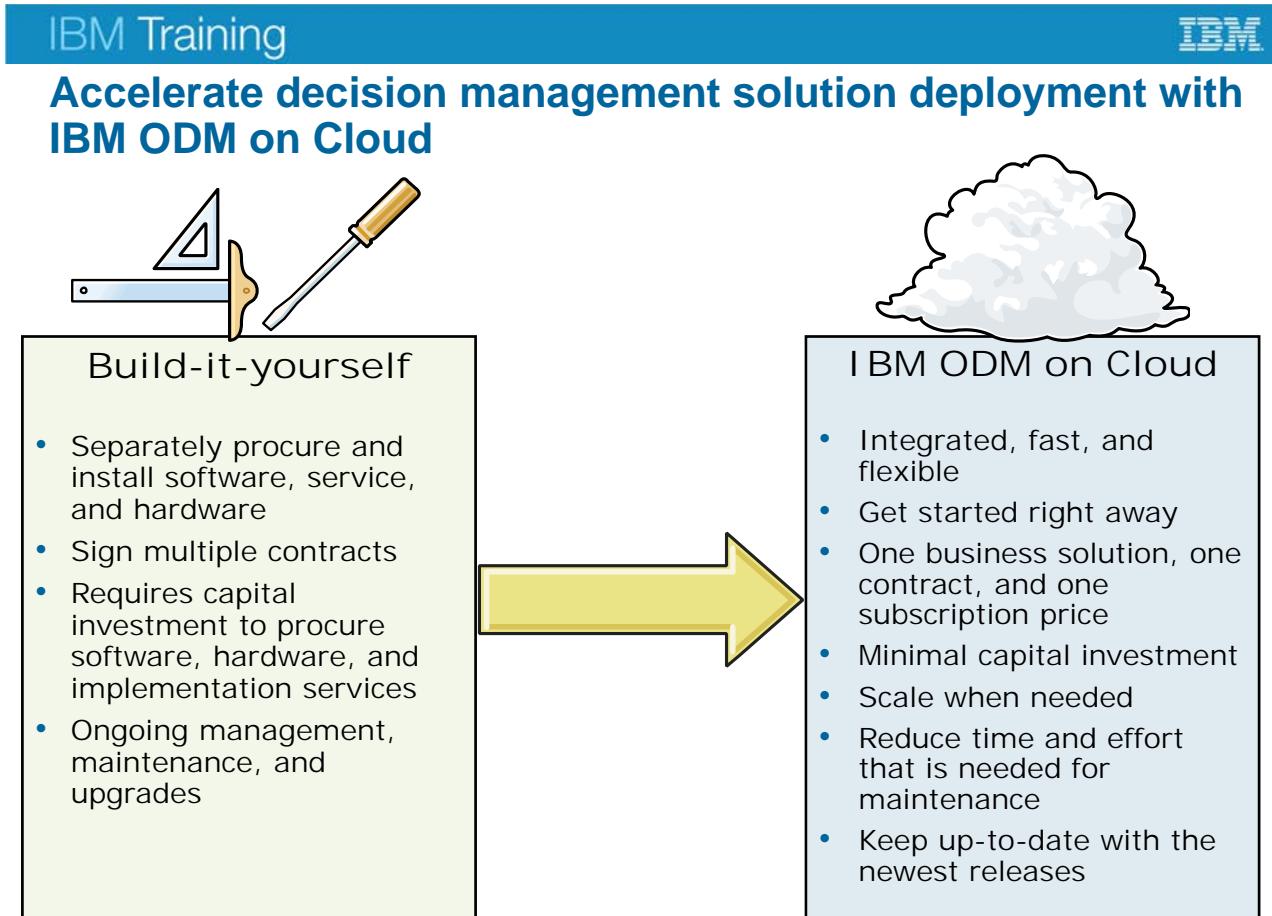
IBM ODM on Cloud

- Targets born-on-the-cloud projects and hybrid cloud scenarios
- Born-on-the-cloud project:
 - Development, testing, and production in the cloud
 - Prefer cloud solutions to on-premises solutions
 - Urgency for implementation
- Pilot project
 - Proving business value
 - Try a newer version of IBM Operational Decision Manager
- Development
 - Organization can manage the production solution on-premises, but needs to get started quickly to meet go-live dates

Unit title

© Copyright IBM Corporation 2017

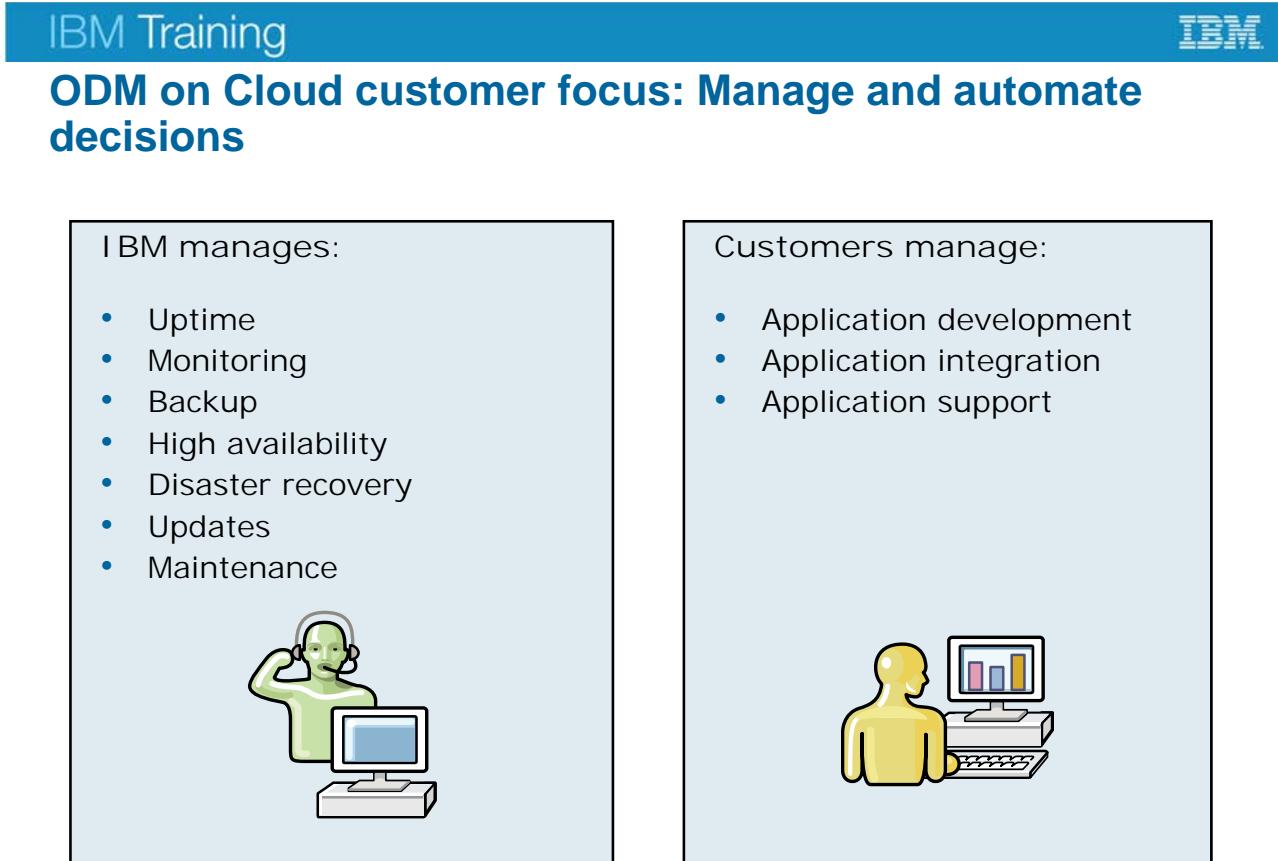
Figure B-2. IBM ODM on Cloud



Unit title

© Copyright IBM Corporation 2017

Figure B-3. Accelerate decision management solution deployment with IBM ODM on Cloud



Unit title

© Copyright IBM Corporation 2017

Figure B-4. ODM on Cloud customer focus: Manage and automate decisions

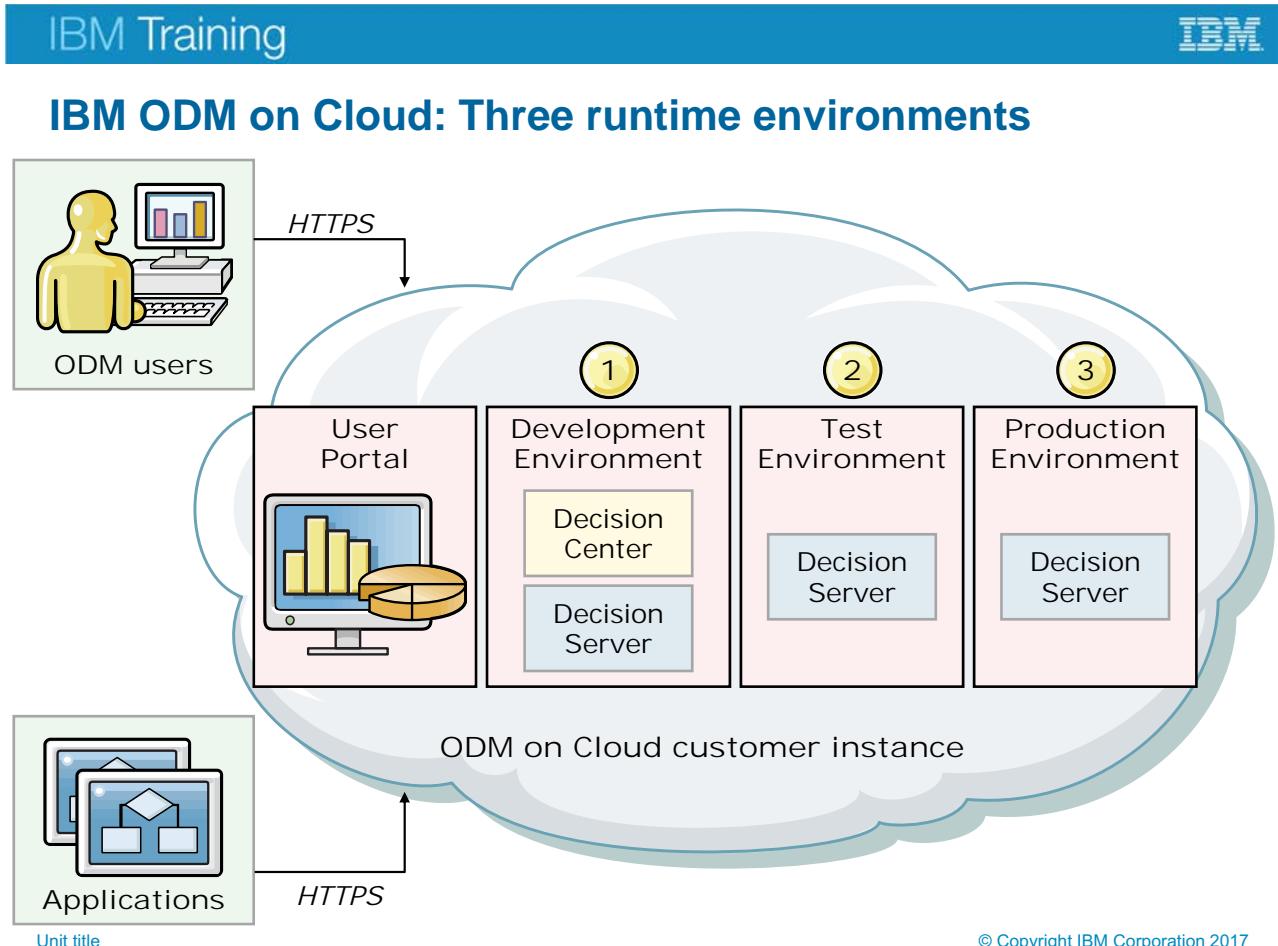


Figure B-5. IBM ODM on Cloud: Three runtime environments

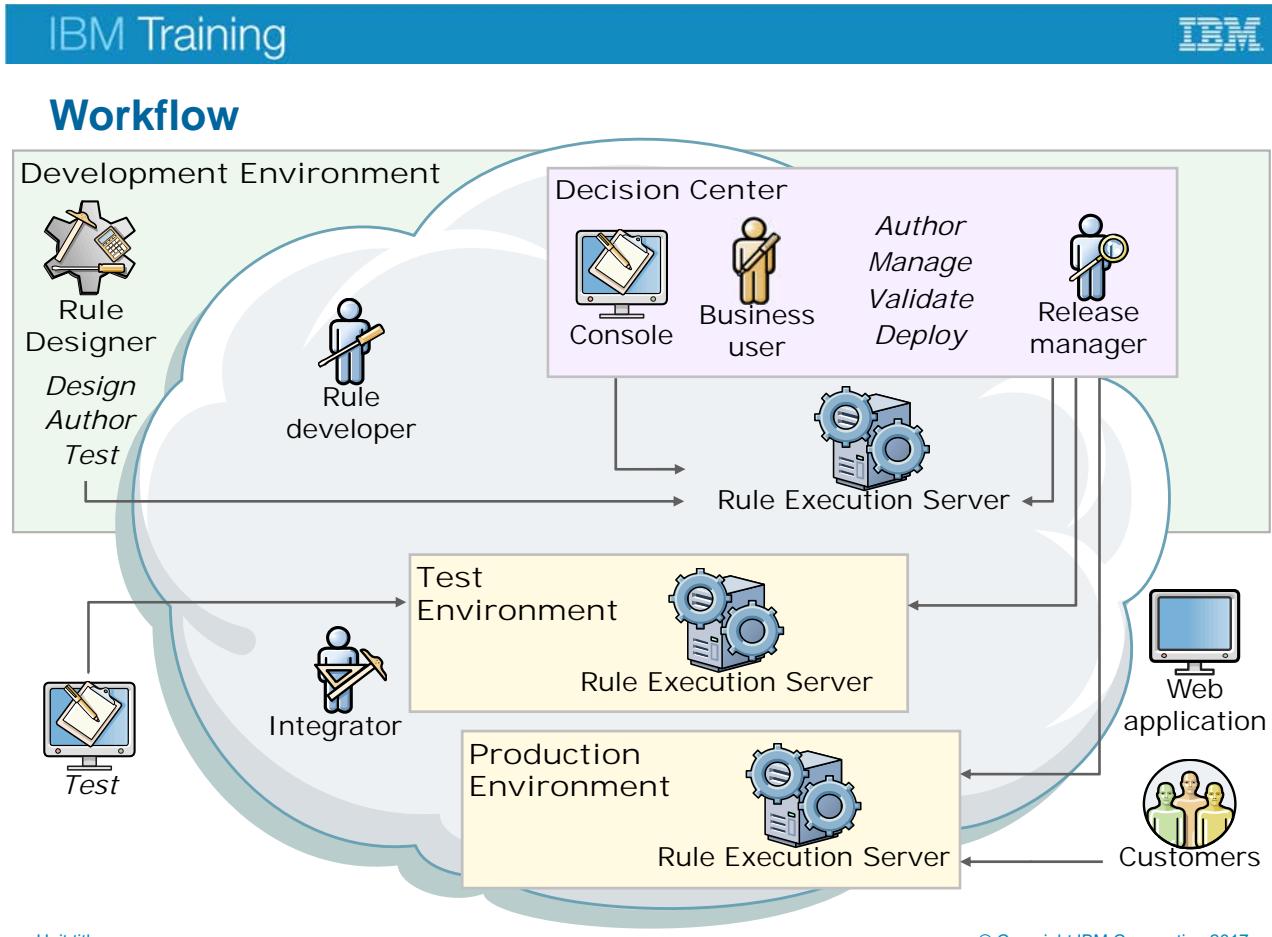
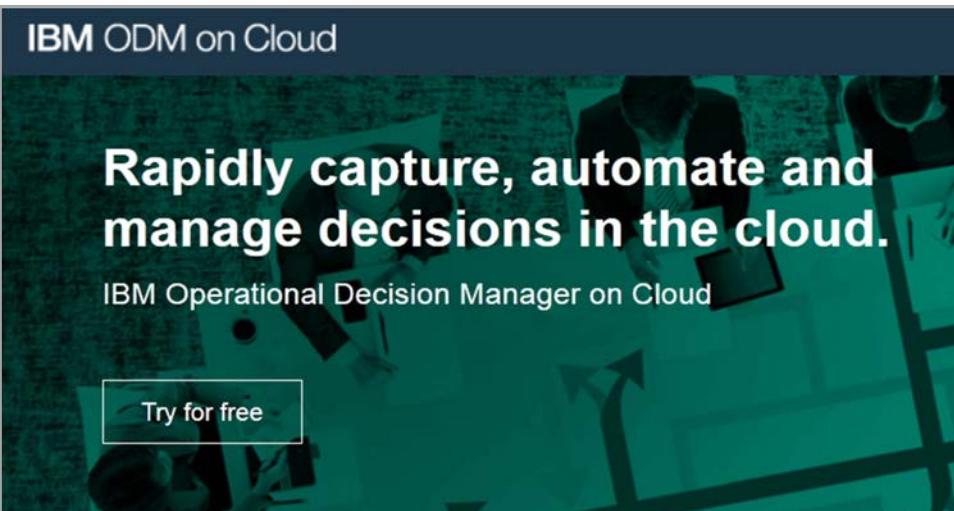


Figure B-6. Workflow

IBM Training 

IBM ODM on Cloud free trial

- Free trial for IBM ODM on Cloud is available
- Go to the following website and click **Try for free** to sign up:
www.bpm.ibmcloud.com/odm



Unit title

© Copyright IBM Corporation 2017

Figure B-7. IBM ODM on Cloud free trial



Activating access and logging in to IBM ODM on Cloud

- Welcome email includes the following information:
 - Link to activate ODM on Cloud access
 - Link to ODM on Cloud instance
- Activation link is tied to a specific email
- After activating access, you can log in to your ODM on Cloud instance



Unit title

© Copyright IBM Corporation 2017

Figure B-8. Activating access and logging in to IBM ODM on Cloud



IBM ODM on Cloud user portal

IBM ODM on Cloud APPLICATIONS

Development Environment

- Rule Designer** 
- Decision Center Business console** 
- Decision Center Enterprise console** 
- Rule Execution Server console** 

Build a decision service to author and run the business rules that implement your business decisions.

[!\[\]\(4527aaa5c419e74b7e7f051fe46a72a4_img.jpg\) Download](#) [!\[\]\(bed1d4e2b99850dc2912e78a14ec9960_img.jpg\) More info](#)

Author, test, govern and deploy decision services

[!\[\]\(a1595665b00b412b43c05ed3163f9e62_img.jpg\) Launch](#) [!\[\]\(6eccc16ad0d53b0d01a3d75001081a01_img.jpg\) More info](#)

Perform occasional administration or advanced management activities.

[!\[\]\(9af3fc2103b399dfbfabb112746ba60a_img.jpg\) Launch](#) [!\[\]\(298ca72f8a947c5b91b7eb8dc76a1876_img.jpg\) More info](#)

Manage and monitor decision services deployed from Rule Designer or Decision Center as part of your development activities.

[!\[\]\(e08f8661bb451eda5a110eb414cca568_img.jpg\) Launch](#) [!\[\]\(6a9e6e63818fac152947c445bf81d243_img.jpg\) More info](#)

Test Environment

- Rule Execution Server console** 

Monitor decision services deployed from Decision Center used for performance, system, and integration testing activities.

[!\[\]\(4c3fc2e2adeda3d1a974e4737e5941a9_img.jpg\) Launch](#) [!\[\]\(5562955b388b649240417d74df83a516_img.jpg\) More info](#)

Production Environment

- Rule Execution Server console** 

Monitor decision services deployed from Decision Center used in the context of a production application.

[!\[\]\(d06bd5599887589d599eacfefc29d9ee_img.jpg\) Launch](#) [!\[\]\(7862238a70f2be76073a71e5a1cf9856_img.jpg\) More info](#)

© Copyright IBM Corporation 2017

Figure B-9. IBM ODM on Cloud user portal

IBM Training

Using Rule Designer (1 of 3)

- Click **Download** from the user portal
- Two options for Rule Designer:
 - Stand-alone: Download a version of Rule Designer that is configured for use with IBM ODM on Cloud
 - Plug-ins: If you already use Eclipse, you can download Rule Designer plug-ins to use with your Eclipse installation
- Start Rule Designer by double-clicking `eclipse.exe`

The screenshot shows a section titled "Development Environment" with a sub-section titled "Rule Designer". It features a circular icon with a gear and a stylized letter 'A'. Below the icon is a descriptive text: "Build a decision service to author and run the business rules that implement your business decisions." At the bottom are two buttons: "Download" with a downward arrow icon and "More info" with a right-pointing arrow icon.

Unit title

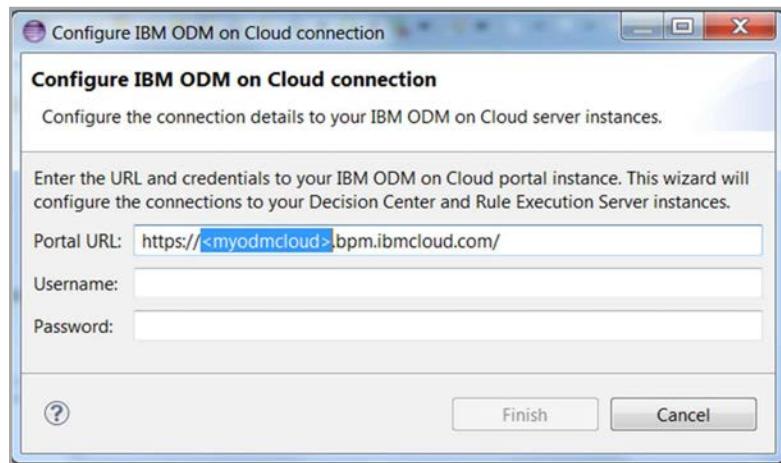
© Copyright IBM Corporation 2017

Figure B-10. Using Rule Designer (1 of 3)



Using Rule Designer (2 of 3)

- Configure Rule Designer to use the URL of your ODM on Cloud instance
- Rule Designer interface similar to on-premises version
 - However, some on-premises features (such as certain perspectives, like the Samples console) not available
- Switch to the Rule perspective to work on decision services
 - Create or import decision services
 - Publish decision services to Decision Center on the cloud



Unit title

© Copyright IBM Corporation 2017

Figure B-11. Using Rule Designer (2 of 3)

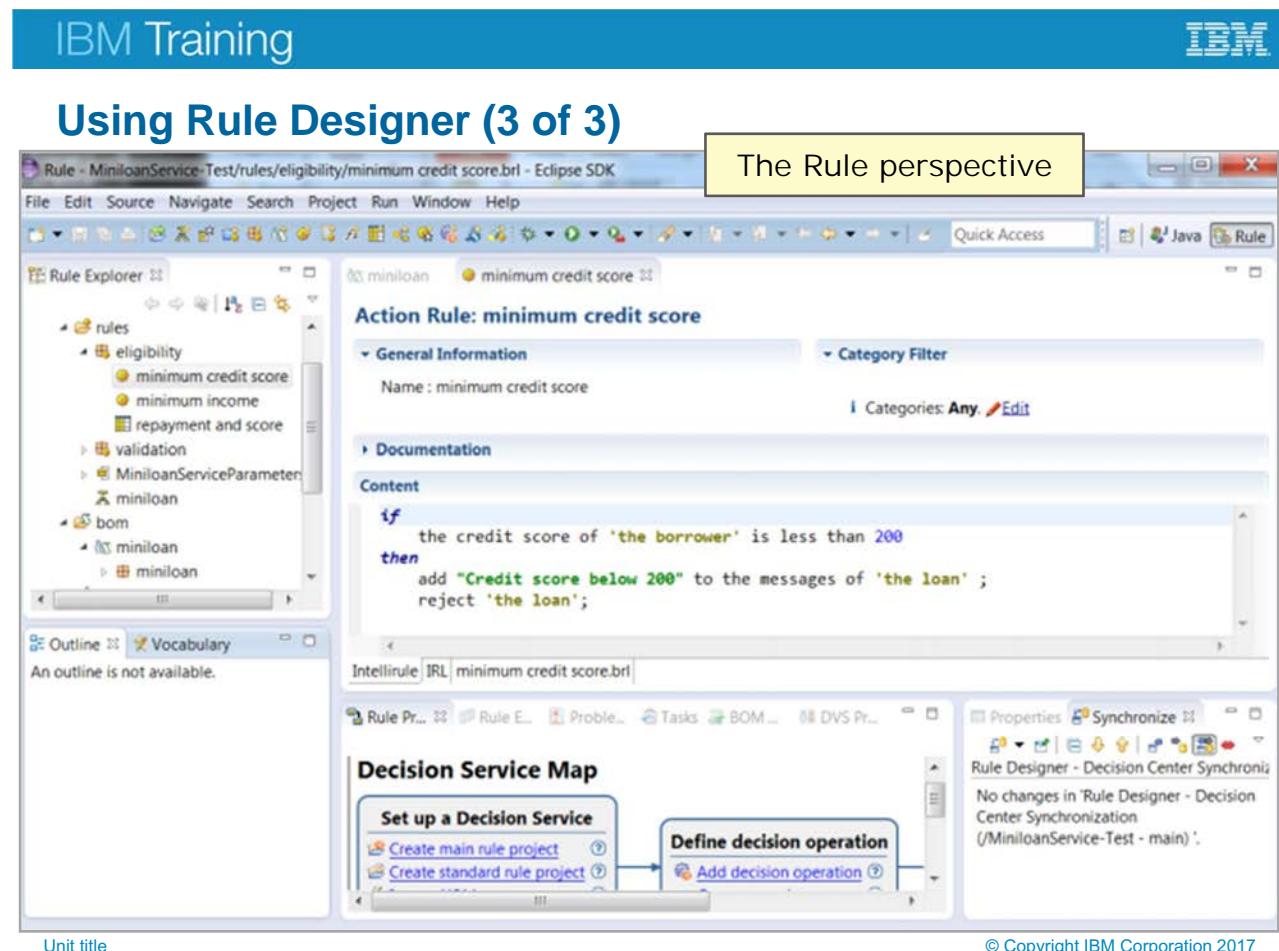
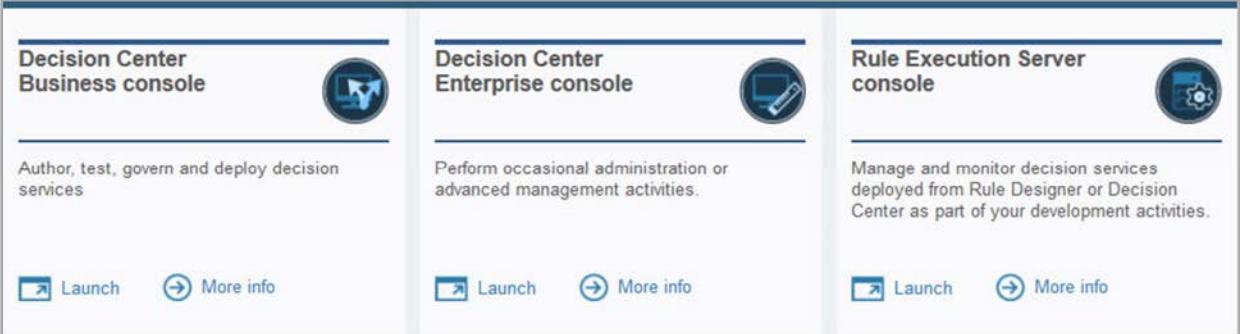


Figure B-12. Using Rule Designer (3 of 3)

IBM Training 

Using Decision Center and Rule Execution Server (1 of 2)

- In the user portal, find the environment that you want to use (Development, Test, or Production)
- Click **Launch** for the module that you want to start



Decision Center Business console  Author, test, govern and deploy decision services  Launch  More info	Decision Center Enterprise console  Perform occasional administration or advanced management activities.  Launch  More info	Rule Execution Server console  Manage and monitor decision services deployed from Rule Designer or Decision Center as part of your development activities.  Launch  More info
---	--	--

Unit title

© Copyright IBM Corporation 2017

Figure B-13. Using Decision Center and Rule Execution Server (1 of 2)

IBM Training

Using Decision Center and Rule Execution Server (2 of 2)

- Log in with your ODM on Cloud user name and password
- Module opens in web browser window
 - Interface is the same as on-premises version

Decision Center HOME LIBRARY WORK

What's New Stream

New Rules

- maximum amount
- repayment and score
- minimum income
- minimum credit score

New Updates on Followed Rules

There have been no new updates on followed rules.

New Comments in Activity Stream

There have been no new comments in the activity stream.

Followed Rules

- maximum amount (1) ★
- minimum credit score (1) ★
- minimum income (1) ★
- repayment and score (1) ★

Rules Recently Worked On

You have not worked on any rules yet.

Unit title

© Copyright IBM Corporation 2017

Figure B-14. Using Decision Center and Rule Execution Server (2 of 2)



Finding help for IBM ODM on Cloud

- IBM Knowledge Center for IBM ODM on Cloud

[http://www.ibm.com/support/knowledgecenter/SS7J8H/welcome/
kc_welcome_cloud.html](http://www.ibm.com/support/knowledgecenter/SS7J8H/welcome/kc_welcome_cloud.html)

- Complete product documentation for IBM ODM on Cloud, including a “Getting Started” tutorial
- IBM ODM on Cloud user portal also has direct links to the documentation

- IBM ODM Support Portal

[https://www.ibm.com/support/entry/portal/product/websphere/ibm
operational decision manager](https://www.ibm.com/support/entry/portal/product/websphere/ibm_operational_decision_manager)

- Support Portal provides tools and resources for help with IBM Operational Decision Manager
- Open service requests, view fix lists, access community resources, and more

Unit title

© Copyright IBM Corporation 2017

Figure B-15. Finding help for IBM ODM on Cloud

Appendix C. Business Rules Service on IBM Bluemix

Estimated time

00:15

Overview

This appendix describes the Business Rules Service on IBM Bluemix.

IBM Training

Introducing Business Rule Service on IBM Bluemix

- Provides a development environment to manage decision lifecycle from modeling to deployment
- Provides the execution environment to orchestrate and automate decision logic

IBM

IBM Cloud > Bluemix

Solve real problems

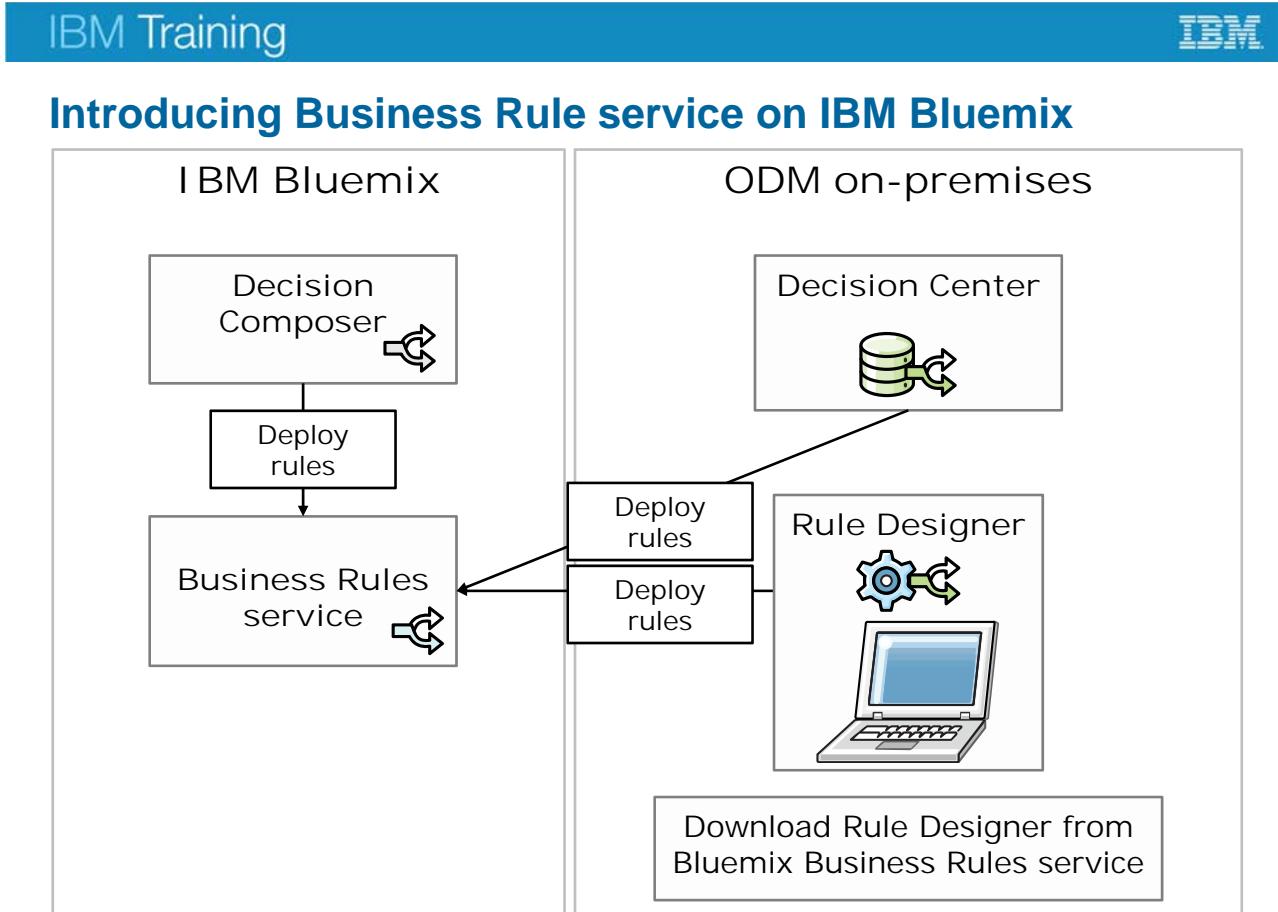
Build with infrastructure, Watson, software, and services on the Bluemix cloud platform

Get started free Learn how

Unit title

© Copyright IBM Corporation 2017

Figure C-1. Introducing Business Rule Service on IBM Bluemix



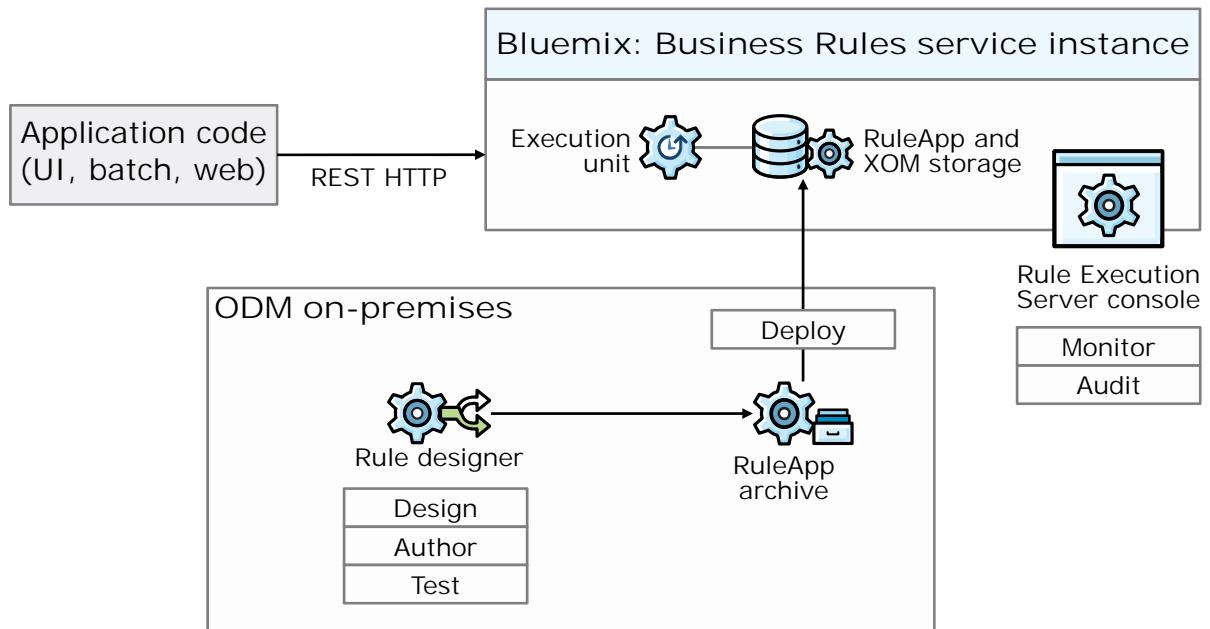
Unit title

© Copyright IBM Corporation 2017

Figure C-2. Introducing Business Rule service on IBM Bluemix



Business Rules service components



Unit title

© Copyright IBM Corporation 2017

Figure C-3. Business Rules service components

Advantages to using Business Rules service

- No installation of an application server, such as WebSphere Application Server
- No installation of Rule Execution Server
- No installation or configuration of the execution unit (XU) is required
- Rule data is persisted by the service, so you do not need a database to persist your rule data
- You can quickly and easily validate any ruleset in development, and test the client application, without needing to configure application servers for deployment and maintenance of the Decision Server
- You can free up precious CPU, memory, and database resources (used by the on-premises IBM ODM installation) to allocate to needier resource-intensive applications

Unit title

© Copyright IBM Corporation 2017

Figure C-4. Advantages to using Business Rules service

ODM tools that are not supported on Bluemix

- Decision Center
- Ruleset interceptors
- Ruleset execution traces
 - Ruleset monitoring properties (such as `rulesetSEQUENTIALTRACEENABLED`) are disabled in the Business Rules service
 - You can configure the payload to enable tracing

Unit title

© Copyright IBM Corporation 2017

Figure C-5. ODM tools that are not supported on Bluemix

Choosing ruleset candidates for Business Rules service

- Performance
 - Rulesets that are already hosted through REST transparent decision services or SOAP-hosted transparent decision services
 - Rulesets that can be started as a web service without negatively affecting the user or application in the transition to web services
- Customization
 - Rulesets that do not require configuration at the XU level (for example, XU connection pools or asynchronous ruleset parsing)
- Maintenance
 - Rulesets that are usually developed, deployed, and maintained by developers, and are not edited or deployed by business users

Unit title

© Copyright IBM Corporation 2017

Figure C-6. Choosing ruleset candidates for Business Rules service

More resources

- IBM Bluemix Catalog: Business Rules service
<https://console.ng.bluemix.net/catalog/services/business-rules>
 - Overview of Business Rules service
- IBM developerWorks
<https://www.ibm.com/developerworks/topics/businessrulesservice>
 - Articles and tutorials that are submitted by ODM experts

Unit title

© Copyright IBM Corporation 2017

Figure C-7. More resources



IBM Training



© Copyright International Business Machines Corporation 2017.