IBM® WebSphere®

Governing Operational Decisions in an Enterprise Scalable Way

IBM Operational Decision Manager support for release management

Decision governance overview and organization

Decision release management lifecycle

Pierre Berlandier
Eric Charpentier
Duncan Clark

Redbooks

ibm.com/redbooks

International Technical Support Organization

Governing Operational Decisions in an Enterprise Scalable Way

April 2013

SG24-8127-00

Note: Before using this information and the product it supports, read the information in “Notices” on page vii.

First Edition (April 2013)

This edition applies to Version 8, Release 0, Modification 1 of IBM Operational Decision Manager (product number 5725B69).

© Copyright International Business Machines Corporation 2013. All rights reserved.
Note to U.S. Government Users Restricted Rights -- Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

© Copyright IBM Corp. 2013. All rights reserved.

Notices

This information was developed for products and services offered in the U.S.A.

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, Armonk, NY 10504-1785 U.S.A.

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: 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 states 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 supply in any way it believes appropriate without incurring any obligation to you.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

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 the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. These and other IBM trademarked terms are marked on their first occurrence in this information with the appropriate symbol (® or ™), indicating US registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at http://www.ibm.com/legal/copytrade.shtml

The following terms are trademarks of the International Business Machines Corporation in the United States, other countries, or both:

IBM®	Redbooks®	WebSphere®
ILOG®	Redbooks (logo) ®	

The following terms are trademarks of other companies:

ITIL is a registered trademark, and a registered community trademark of The Minister for the Cabinet Office, and is registered in the U.S. Patent and Trademark Office.

Microsoft, and the Windows logo 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.

Other company, product, or service names may be trademarks or service marks of others.

Preface

This IBM® Redbooks® publication presents decision governance topics from a theoretical discussion perspective and then goes on to make links to the practical aspects of applying these concepts by using the IBM Operational Decision Manager platform.

This book explores enterprise governance context to clarify the bigger picture for how governance is carried out across the enterprise.

You will also find this book valuable if you are using or considering the usage of an operational decision management system (or business rules management system (BRMS)) in your organization. You might be following a standard such as the The Open Group Architecture Framework (TOGAF) Architecture Development Method (ADM) and decided to use a decision management system that lets the business people take control of the business decisions that are made by the technology systems in their organization.

This book also describes Control Objectives for Information and Related Technology (COBIT), which provides a comprehensive framework that assists enterprises in achieving their objectives for the governance and management of enterprise IT.

Another topic of great importance that this book covers is the relationship to ITIL, a public framework that describes best practices in IT Service Management. Of the five stages of the ITIL lifecycle, this book focuses on the objectives and processes of the Service Transition stage.

The team who wrote this book

This book was produced by a team of specialists from around the world working at the International Technical Support Organization, Raleigh Center.

Pierre Berlandier is a Senior Technical Staff Member with IBM Software Services for WebSphere® (ISSW). As part of the IBM ILOG® and IBM professional services for more than 17 years, Pierre has helped his clients succeed with the different incarnations of the rules programming paradigm. As part the Service Engineering group of ISSW, he leads the design of service delivery processes, mentors practitioners, and oversees the capture and organization of best practices for the IBM Operation Decision Management platform. Pierre received a Ph.D. in Computer Science from INRIA in France, where he started working on the synergy between the rules and the constraint programming paradigms.

Eric Charpentier is a Senior Consultant with IBM Software Services for WebSphere (ISSW) in Canada. Over the last 12 years, Eric has helped clients implement the Decision Management technology in projects of all sizes and in many different industries. As part of ISSW, he helps clients analyze their requirements, and designs solutions that use Decision Management technology. He also helps clients with the implementation of decisions through analysis, authoring, and testing. Eric has a B. Eng. in Computer Engineering from École Polytechnique de Montréal and an MBA from HEC Montréal.

Duncan Clark is a Product Manager for IBM Operational Decision Manager who is responsible for integrating Operational Decision Manager and other IBM products. He is based at IBM Hursley Labs, and over the last year has been responsible for product integrations and demonstrations with Business Process Management (BPM), IBM Case Manager, and Worklight products. He managed the integration of the ILOG embedded Rules into BPM V7.5 and the development of support pack LA71, which provides integration tooling and development patterns for using Operational Decision Manager with Business Process Manager. Before he performed this work, Duncan led the Governance and Policy architecture for WebSphere Service Registry and Repository.

Thanks to the following people for their contributions to this project:

Carla Sadtler
International Technical Support Organization, Raleigh Center

Antony Viaud, Product Manager, IBM Operational Decision Manager
IBM France

Pierre Chardin, Product Manager, IBM Operational Decision Manager
IBM France

Andy Ritchie, BPM and Operational Decision Manager Synergy lead
IBM UK

Alain Neyroud, PLM Lead, IBM Operational Decision Manager
IBM US

John Falkl, Distinguished Engineer
IBM US

Mike Rowling, Risk Analytics
IBM Canada

Now you can become a published author, too!

Here's an opportunity to spotlight your skills, grow your career, and become a published author—all at the same time! Join an ITSO residency project and help write a book in your area of expertise, while honing your experience using leading-edge technologies. Your efforts will help to increase product acceptance and customer satisfaction, as you expand your network of technical contacts and relationships. Residencies run from two to six weeks in length, and you can participate either in person or as a remote resident working from your home base.

Find out more about the residency program, browse the residency index, and apply online at:

ibm.com/redbooks/residencies.html

Comments welcome

Your comments are important to us!

We want our books to be as helpful as possible. Send us your comments about this book or other IBM Redbooks publications in one of the following ways:

- Use the online **Contact us** review Redbooks form found at:

  ibm.com/redbooks

- Send your comments in an email to:

  redbooks@us.ibm.com

- Mail your comments to:

  IBM Corporation, International Technical Support Organization
  Dept. HYTD Mail Station P099
  2455 South Road
  Poughkeepsie, NY 12601-5400

Stay connected to IBM Redbooks

- Find us on Facebook:

  http://www.facebook.com/IBMRedbooks

- Follow us on Twitter:

  http://twitter.com/ibmredbooks

- Look for us on LinkedIn:

  http://www.linkedin.com/groups?home=&gid=2130806

- Explore new Redbooks publications, residencies, and workshops with the IBM Redbooks weekly newsletter:

  https://www.redbooks.ibm.com/Redbooks.nsf/subscribe?OpenForm

- Stay current on recent Redbooks publications with RSS Feeds:

  http://www.redbooks.ibm.com/rss.html

1

Introduction

This chapter provides an introduction to enterprise governance concepts and practices and how they relate to other industry practices.

This chapter contains the following topics:

- Enterprise governance context
- Operational decision management
- Relationship to COBIT
- Relationship to ITIL
- References
- Decision management technology
- What is decision governance
- The evolution of concepts

© Copyright IBM Corp. 2013. All rights reserved.

1.1 Enterprise governance context

Before talking about governing decisions, the bigger picture for how governance is carried out across the enterprise must be clarified. In the context of overall enterprise decision making, several governance layers emerge across the organization:

- Corporate governance
- Service governance
- Operational governance

Corporate governance is the active process of creating strategy, making decisions, and applying policies by the board, executives, and other stakeholders. There is a clear relationship between the ability to have compliance with corporate governance and the specification of policies that drive the decision making in the enterprise. Corporate governance is concerned with the entire enterprise. Policy at this level tends to be broad and high level and must be interpreted and translated into business, architectural, and operational policy that provides specific standards and guidance that ensure compliance. Such policies should be traceable back to and be recognizable as an implementation of the corporate policy.

Service-oriented architecture (SOA) governance is a key ingredient in the bridging from corporate governance to operational governance. The planning process of SOA governance requires that the corporate policy be incorporated into the planning process for business, architectural, and operational policy, as shown in Figure 1-1.

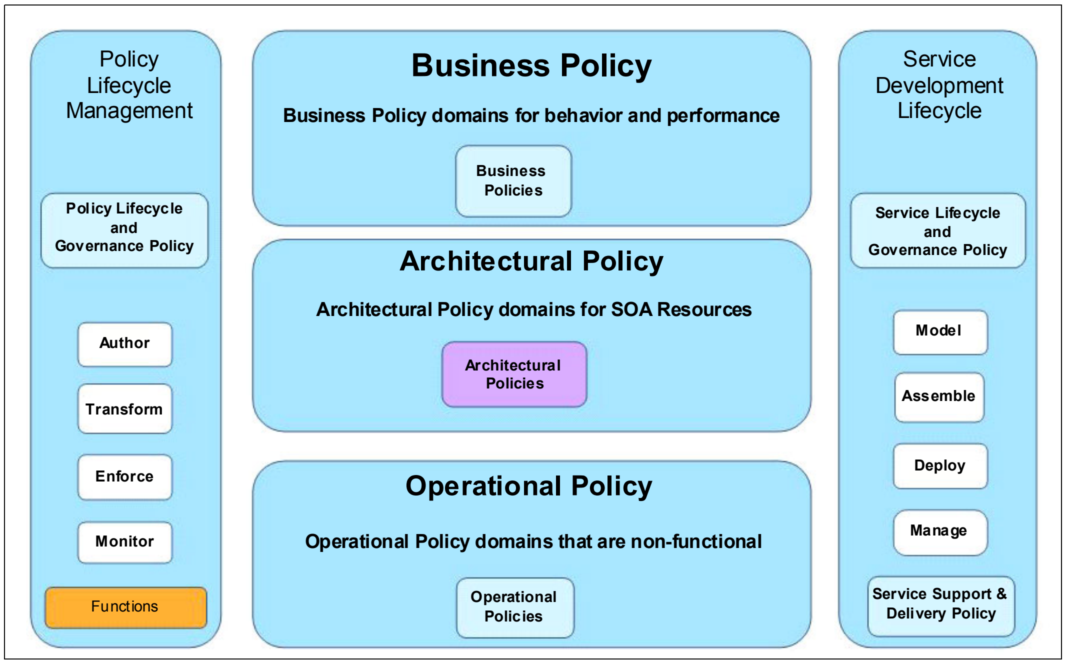

Policy Lifecycle Management
Policy Lifecycle and Governance Policy
Author
Transform
Enforce
Monitor
Functions
Business Policy
Business Policy domains for behavior and performance
Business Policies
Architectural Policy
Architectural Policy domains for SOA Resources
Architectural Policies
Operational Policy
Operational Policy domains that are non-functional
Operational Policies
Service Development Lifecycle
Service Lifecycle and Governance Policy
Model
Assemble
Deploy
Manage
Service Support & Delivery Policy

Figure 1-1 SOA governance planning process

Business policy reflects both the real-time decision management rules that any compliant organization must have in place as well as providing the standards and guidelines that architectural and operational policy must follow. It is these real-time decision management policies that this book focuses on. Changes to these policies go through a policy lifecycle, as shown in the left part of Figure 1-1. This book aims to describe how the decision management governance realizes this business decision change lifecycle and delivers agile control of the business according to the corporate governance policies.

Architectural policy identifies standards for building and deploying shared services in a compliant fashion that maximizes quality of service (QoS). In this context, the architectural policy that we focus on is that business decisions should be realized by using an operational decision management solution, such as IBM Operational Decision Manager. This situation recognizes that the decisions are exposed as services and integrated into solutions according to the organizations service development lifecycle shown in the right part of Figure 1-1 on page 2.

This *service development lifecycle* is governed according to the IT governance standards, such as ITIL and COBIT, so all development and deployment of these decision services should adhere to these standards. Participation in a software development lifecycle implies that some policy changes cannot be realized simply through changes to policy or rules but require changes to the deployed software. This is considered a technical decision change lifecycle in this book.

Operational policy identifies the runtime policy for operational excellence and the monitoring that is required to validate that corporate compliance is indeed taking place. This book does not consider the operational policies in detail, but the assurance of delivering the appropriate QoS for the decision service in terms of responsiveness and quality of decision must be confirmed through testing, validation, and monitoring activities in the policy lifecycle.

The *policy management lifecycle* provides organizations with an opportunity and an obligation to perfect their decision making capabilities through the refinement of policy and optimization of operations. Enterprises are also implementing more forward-looking views through predictive models that must be in synchronization with corporate, service, and operational decision management solutions. Governing remediation and change management against policies, regulations, and other factors is an ongoing challenge to keeping corporate governance in synchronization with service definition and operational implementations. Furthermore, incident management, audit, and other processes are also necessary when you govern operational decision management. It is this aspect of governing operational decisions in an enterprise scalable way that is the focus of this book.

1.2 Operational decision management

This book assumes one main thing from the reader's perspective: You decided to use or are considering the usage of an operational decision management system (or business rules management system (BRMS)) in your organization. Because this architecture change is important to you, you gather as much information as possible on the topic and want to know how the decisions are governed in your organization.

You might be following a standard such as the The Open Group Architecture Framework (TOGAF) Architecture Development Method (ADM) and decide to use a decision management system that lets the business people take control of the business decisions that are made by the technology systems in their organization. Decision management is a business discipline that has the following characteristics:

- Identifies decisions that have value for the organization.
- Creates a common vocabulary and categorization to facilitate communication about decisions.
- Provides a systematic approach for identifying, documenting, and analyzing existing decisions as well as new decisions or updated decisions.

- Transforms the documented decisions into executable decisions that have the following characteristics:
  - Are readable by business users.
  - Can be run by IT systems within the organization.

This last bullet is achieved by using a specific technology that is called a *decision management system*, which provides transparency to what was previously implemented in an obscure fashion in some application code by technical people. For more information, see 1.5, “Decision management technology” on page 6.

Decision management systems are meant to provide the following benefits to your organizations:

- Visibility of the decisions
- Agility in making changes to decisions
- Empowering business people to make the changes
- Governance over the changes

Many of the details about decision management can be related to knowledge management. Both COBIT and ITIL refer to knowledge management in their processes. You are managing the decisions that are an important part of the knowledge that is critical to an organization. To paraphrase the COBIT definition of the process:

“Management of decisions: Maintain the availability of relevant, current, validated, and reliable knowledge to support decision. Plan for the identification, gathering, organizing, maintaining, use, and retirement of decisions.”[1]

The goal of decision management is to use this process and information to achieve agility with the decisions of the organization.

One of the challenges that all organizations face is to support the changes that are required for decisions. If you allow people to change existing decisions, how do you make sure that the changes made are always valuable to the business?

IT might be able to manage decisions at a project level, but to achieve some level of governance at a program or enterprise-wide level, you must look more deeply at the roles, responsibilities, and the processes that are used to support what you do when you perform decision management.

This book tries to provide the information that you need to implement this governance in your organization. The guidance in this book is not prescriptive. You should expect to adapt the information within to fit your organization rather than to try to apply it to the letter. To make an analogy: *It is a list of ingredients, but not the recipe*.

This book is about providing you with the information about operational decision management that you need in the hope that you can use it to relate to other standards that might be used in your organization or at least to give you some guidance to have good practices around the management of your operational decisions.

[1] Source: *COBIT 5: Enabling Processes*, 2012, © ISACA©, All rights reserved. Reprinted by permission.

1.3 Relationship to COBIT

Control Objectives for Information and Related Technology (COBIT) provides a comprehensive framework that assists enterprises in achieving their objectives for the governance and management of enterprise IT. COBIT presents a set of governance and management processes, some of which are closely related to the governance and management processes that we present in this book. In regard to the COBIT framework, most of the information that is found in this book relates to the following COBIT processes:

- “BAI06 - Manage Changes: Manage all changes in a controlled manner, including standard changes and emergency maintenance relating to business processes, applications, and infrastructure. This includes change standards and procedures, impact assessment, prioritization and authorization, emergency changes, tracking, reporting, closure, and documentation.”[2]
- “BAI07- Manage Change Acceptance and Transitioning: Formally accept and make operational new solutions, including implementation planning, system and data conversion, acceptance testing, communication, release preparation, promotion to production of new or changed business processes and IT services, early production support, and a post-implementation review.”[3]

For both of these processes, this book focuses on operational decision changes.

1.4 Relationship to ITIL

ITIL is a public framework that describes best practices in IT Service Management. It presents the following five stages of the service lifecycle:

- Service Strategy
- Service Design
- Service Transition
- Service Operation
- Continual Service Improvement

More information about these service lifecycle stages can be found in *The Official Introduction to the ITIL Service Lifecycle* by the Office of Government Commerce (OGC).

This book focuses primarily on the operational decision management aspects of the Service Transition stage of the ITIL lifecycle where new and updated services are moved from development and test environments to supported environments.

[2] Source: *COBIT 5: Enabling Processes*, 2012, © ISACA©, All rights reserved. Reprinted by permission.
[3] Source: *COBIT 5: Enabling Processes*, 2012, © ISACA©, All rights reserved. Reprinted by permission.

1.5 Decision management technology

There are two forms of decision management technology: operational decision management, using business rules and event, and analytical decision management, using predictive analytics and optimization techniques. Their applicability overlaps and for some classes of problems is complementary, as shown in Figure 1-2.

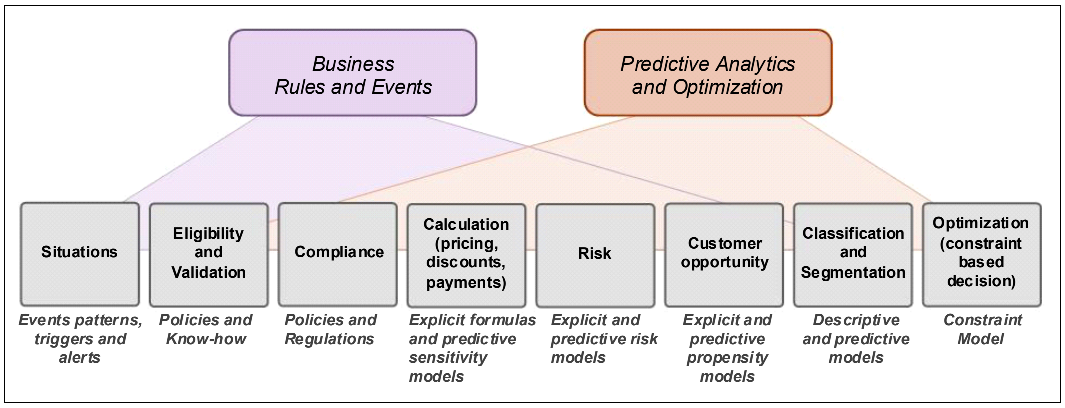

Business Rules and Events
Predictive Analytics and Optimization
Situations
Eligibility and Validation
Compliance
Calculation (pricing, discounts, payments)
Risk
Customer opportunity
Classification and Segmentation
Optimization (constraint based decision)
Events patterns, triggers and alerts
Policies and Know-how
Policies and Regulations
Explicit formulas and predictive sensitivity models
Explicit and predictive risk models
Explicit and predictive propensity models
Descriptive and predictive models
Constraint Model

Figure 1-2 Decision management technology

1.5.1 What is decision management

In both cases, the business processes and applications invoke the Decision Services that encapsulate the behavior of the decision that is being automated.

In analytical decision management, the history of decisions is analyzed to build a model that can be used to predict the best decision response for the future.

In operational decision management, policy, best practices, and business experience is used to write rules that describe how to make those decisions or identify situations that must be reacted to. In many cases, the models that result from analytical decision management can also be used in the operational decision management automation.

There are situations where all the decision automation that is needed for a solution can initially be provided by one of these forms of decision management. However, as decision making becomes more complex, the need for synergistic usage of both forms of technology becomes greater. When you start from analytical decision management, the models that define best practices or probability can be encapsulated and applied in a broader systems context through operational decision management. When you start from operational decision management, the decision logic that defines best practices can benefit from the data mining, segmentation, and insight that are provided by analytical decision management.

In both cases, there must be some measure of how good the decisions are. These key performance indicators (KPIs) relate to the overall goals of the business and, when used with Scenario Analysis and Simulation, allow for a real-world method of assessing how decision changes affect the behavior of business systems.

1.5.2 IBM Operational Decision Manager

This book focuses on the operational decision management aspect and provides examples by using IBM Operational Decision Manager and its capabilities.

Figure 1-3 gives an overview of the Operational Decision Manager.

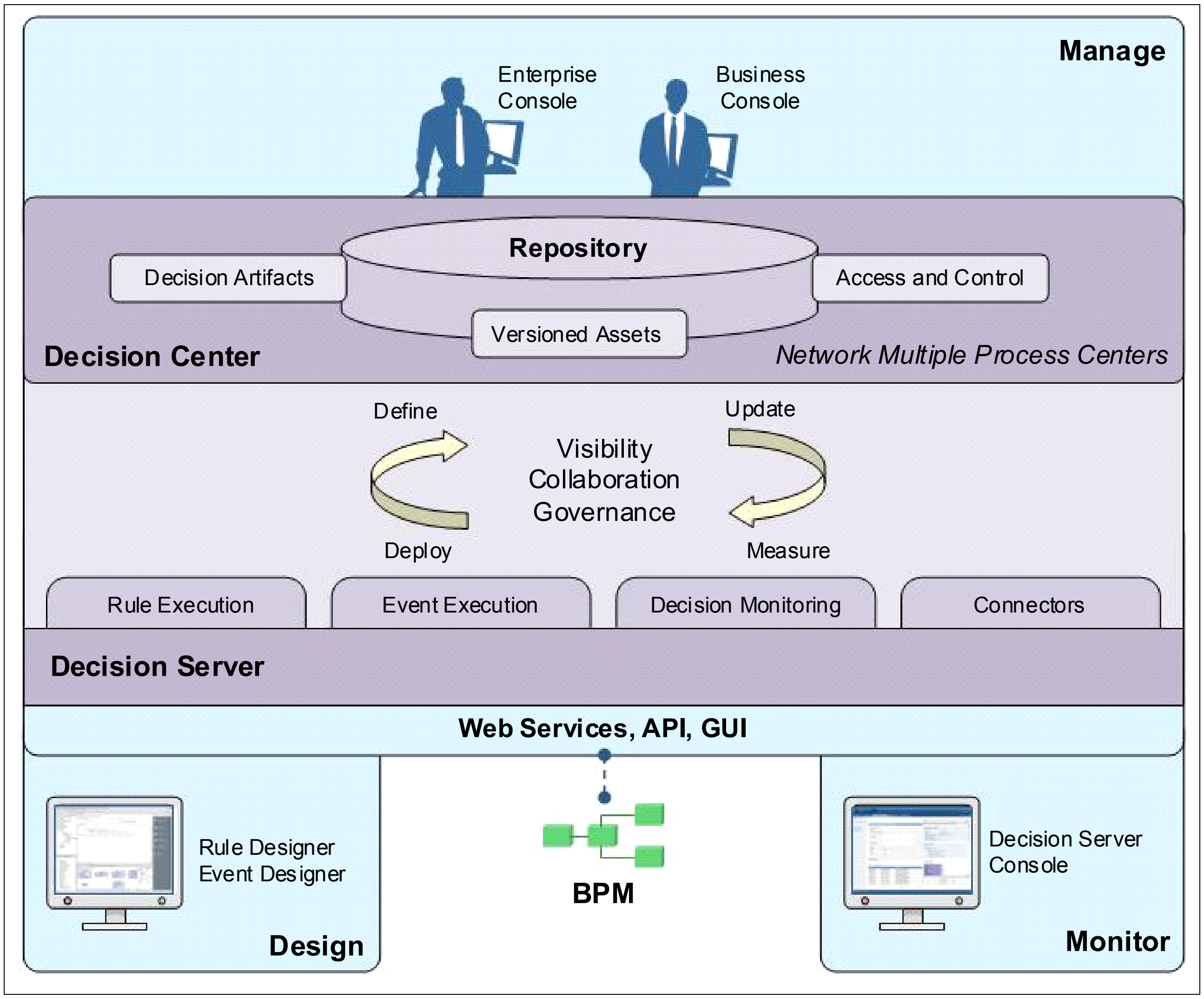


Figure 1-3 IBM Operational Decision Manager components

Here are descriptions of the components:

- *Decision Center Business console* allows business teams to view changes as they occur, communicate about changes, and share information and author rules by using a web-based collaboration environment. The console provides a highly interactive and social experience for non-technical subject matter experts (SMEs) to understand what is being worked on and to control the maintenance of rules as they evolve to meet business policy requirements.
- *Decision Center Enterprise console* is a web environment that assists business analysts in managing changes through the rule lifecycle by providing a set of functional capabilities, including testing, simulation, analysis, and reporting of decision logic. Administrative capabilities in this console control specify repository access and change permissions for individual users in both console interfaces.

- The *Decision Center repository* is the foundation for change governance. The repository provides a single source for automated decisions that are used by applications, transactions, and processes. Users of the repository can make a change in one place that can be deployed over many systems. With role-based access controls, each LOB participant has specific access and management capabilities along with multi-level version control for individual definitions, sets of definitions, and entire business rule and business event projects.
- *Decision Server* comes with specific execution run times for business rules and business events. Decision management applications can be designed and deployed to handle specific types of automation:
  - *Business rule execution* runs data against sets of rules to determine a decision response that is requested from a process or application. The rule execution run time can handle many rule conditions for any request. It can run a designated order of rules (a rule flow) or use an inference-based execution in which the rule engine determines which rules are required based on the context of the request.
  - *Business event execution* can process data from many different sources, detecting and responding to event patterns among related events, missing events, and aggregated events. The Event Execution Runtime includes a number of integration connectors to maintain a persistent state with different systems, which makes it possible for the Event Execution Runtime to track data patterns over long periods and correlate events from multiple sources. The result of an event pattern correlation can be an alert or an automated action. In many cases, the action starts business rule execution to determine the appropriate decision response, although it can also trigger actions in processes and applications.

  Decision Server also includes the Eclipse-based development tooling that you can use to build decision management applications. The *Rule Designer* and *Event Designer* are separate perspectives that can be run in a single Eclipse instance, providing developers a “one-stop” development environment in which virtually all the artifacts and operations that are needed to create and maintain rule and event-based applications are included.
- *IBM Decision Server console* is a web environment that is designed for administrators for monitoring execution and deployed artifacts.

You can find more information about IBM Operational Decision Manager at the following websites:

- IBM Operational Decision Manager home page:

  http://www.ibm.com/software/decision-management/operational-decision-management/websphere-operational-decision-management/
- IBM Operational Decision Manager, Solutions Brief, WSS14123-USEN-01:

  ftp://public.dhe.ibm.com/common/ssi/ecm/en/wss14123usen/WSS14123USEN.PDF

1.6 What is decision governance

This section describes the concepts of decision governance.

1.6.1 Governance and management

In COBIT 5, the fifth principle highlights the difference between governance and management. It is an important set of definitions for you to consider when we describe decision governance.

Here is how COBIT 5 defines governance and management:

"Governance:

- Ensures that stakeholder needs, conditions, and options are evaluated.
- Determines balanced, agreed-on enterprise objectives to be achieved.
- Setting direction through prioritization and decision making.
- Monitoring performance and compliance against agreed-on direction and objectives.

In most enterprises, overall *governance* is the responsibility of the board of directors under the leadership of the chairperson. Specific governance responsibilities may be delegated to special organizational structures at an appropriate level, particularly in larger, complex enterprises.

Management:

- Plans, builds, runs, and monitors activities.
- Alignment with the direction set by the governance body to achieve the enterprise objectives."[4]

In most enterprises, *management* is the responsibility of the executive management under the leadership of the chief executive officer (CEO).

There is one thing that is missing from this context, and that is the team that runs the management activities. After decision services are deployed, this group, usually referred to as Operations, performs the day to day activities to support the decision management.

1.6.2 Decision governance

Decision governance, like other governance subjects, is the responsibility of management. It is not an isolated discipline or activity, but is integral to IT governance and enterprise governance.

Here are the primary goals of decision governance:

- Provide leadership to support decisions management.
- Make sure that decisions changes are aligned strategically and generate business value.
- Mitigate the risks that are associated with decision changes by performing the following tasks:
  - Implement an organizational structure with defined roles and responsibilities.
  - Provide a set of processes to manage the changes and mitigate the risks.
- Providing applications and infrastructure to support the execution of the processes.
- Measure the performance of the processes and implement improvements.

[4] Source: *COBIT 5: A Business Framework for the Governance and Management of Enterprise IT*, 2012, © ISACA©, All rights reserved. Reprinted by permission.

In traditional organizations, the different levels of governance can be represented by the left part of the relationship that is shown in Figure 1-4.

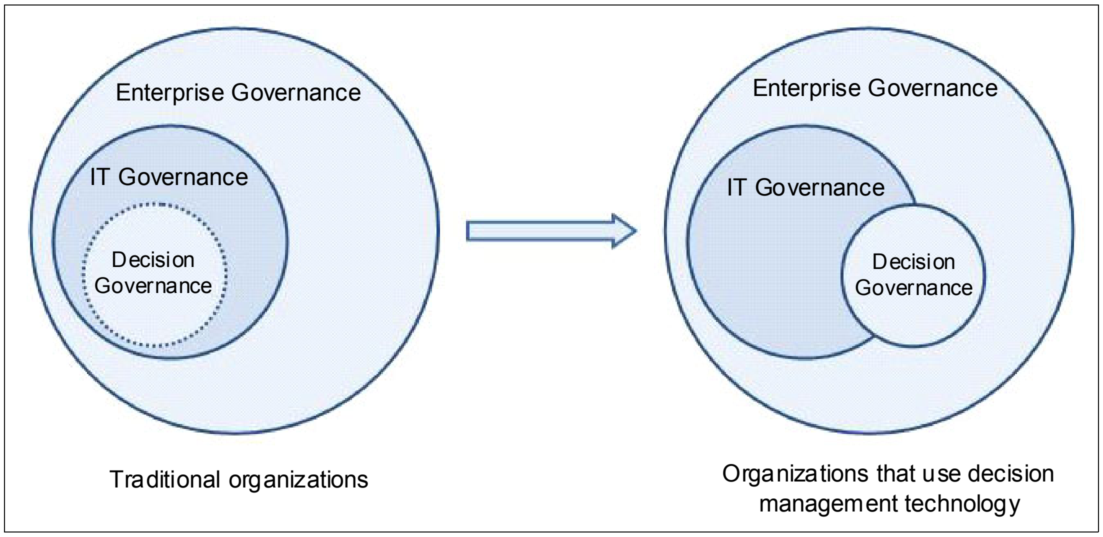


Figure 1-4 Decision governance

In traditional organizations:

- Enterprise governance is the overarching governance.
- IT governance is part of the enterprise governance.
- Decision governance is non-existent as a separate governance (hence the dashed line). It is implicit to IT governance and the business has no visibility into it.

The traditional organization does not usually provide much visibility into the implementation of the decisions in the existing systems. Business must feed IT with specifications about a specific request for change, which means that implementation depends on IT resources being available to process the changes. It takes time for business and IT to come to agreements in some cases, which makes the implementation costly because changes took weeks or months (or more) before they could be completed, so the organization might suffer from lower sales or profits.

In short, the development cycle to update a decision can sometimes be long and not responsive enough for a business.

The right part of Figure 1-4 shows the evolution as the organization starts using decision management technologies and starts discussing decision governance as a separate entity with its own specific roles and responsibilities. In that model, the business is empowered to make some of the changes themselves under the supervision of IT. This improved collaboration reduces the time to market and costs.

In organizations that use decision management technology:

- Decision governance becomes more defined and formalized (shown by the line that becomes solid).
- Decision governance becomes visible to business actors (shift to the right, overlapping enterprise governance and IT governance).
- Decision governance requires the collaboration of both technical actors and business actors.

The roles and responsibilities for decision governance are described in Chapter 3, “Roles and responsibilities in governing decisions” on page 25.

Governance enablers

The COBIT 5 framework describes seven categories of governance enablers:

- “Principles, policies and frameworks
- Processes
- Organizational structures
- Culture, ethics, and behavior
- Information
- Services, infrastructure, and applications
- People, skills, and competencies”[5]

Using these enablers as a base, the guidance in this book focuses primarily on the processes that you can use to achieve your objectives and goals around decision governance as well as the people and skills that you need to complete all the activities successfully.

This book provides some of those enablers that you can use to build governance in your organization.

Some of the principles for governing decisions are described in Chapter 3, “Roles and responsibilities in governing decisions” on page 25. A set of processes to use are described in Chapter 8, “Processes” on page 95.

1.7 The evolution of concepts

This section describes how the concepts described in this chapter evolved.

1.7.1 From rules to decisions

Over the last few years, there has been a natural evolution from the management of business rules to the management of decisions. Business rules are still there, but most business decisions are composed of multiple business rules, and for the purposes of discussions and management, the level of granularity of business rules was too fine for most practical purposes. Hence, a business decision that can be composed of multiple rules but that makes more sense from a business perspective is a more accurate term than rule.

As such, technology products that support this evolution are also evolving, and focusing on decision management as opposed to business rules management. As a result, the topics that are described in this book present an updated view about the governance of decisions in the enterprise.

1.7.2 Decision change management

After you have decisions that are implemented in a decision management system, there are times where a change must be made to those decisions and you want to follow a process that has a set of governed activities that results in the specific requirements for that change to be distributed to the new deployed version or release of that decision.

[5] Source: *COBIT 5: Enabling Processes*, 2012, © ISACA©, All rights reserved. Reprinted by permission.

Decision changes occur for different reasons:

- Proactive changes:
  - Policy improvement (to reduce risk, improve revenues, and so on)
  - Compliance (imposed by legislation outside of the organization's control)
- Reactive changes:
  - Test fixes to resolve production issues

Tip: The fact that a decision management system such as IBM Operational Decision Manager empowers business users to make changes does not mean that these changes should not be managed and governed.

Whatever the reason, the changes to decisions must be managed so that you avoid potential issues that could occur:

- Conflicts with other changes to decisions.
- Incomplete analysis and testing that results in an incorrect decision being deployed.

To reduce the possibility of having issues when you deal with decision changes, implement decision change management processes that do the following actions:

- Allow decisions changes to be implemented in a timely manner.
- Ensure that decision changes are:
  - Identified, analyzed, and recorded
  - Evaluated, authorized, and prioritized
  - Planned, implemented, and tested
  - Deployed

Identified, analyzed, and recorded

Here are factors that affect change management:

- Changes are identified by business people.
- Changes are recorded in a change management system.
- Changes are analyzed so that enough detailed information is documented for further action.

Evaluated, authorized, and prioritized

Assume that there is a steering committee or a Change Control Board (CCB) or some other authority that can evaluate all the change requests, prioritizes them by importance and groups them into a release, and then authorizes the next steps that are required to implement these changes.

Planned, implemented, and tested

These stages are affected by the following items:

- The people that participate in a release start planning the work, separating the pieces, and assigning them to the appropriate participants to work on the implementation.
- After the implementation is complete, the changes can be reviewed and then tested to make sure that there are no issues with the implementation, such as:
  - Issues with the implementation itself. It should behave as expected and provide the correct results.
  - Issues that affect other parts of the system in an unforeseen way (often referred to as *regression* tests).

Deployed

After the team is satisfied with the results of the tests, the changes can be released for deployment. The deployment can be to User Acceptance Testing (UAT) environments as well as production environments.

Change activities, validation activities, and releases

To support these activities, businesses use a group of people to perform these tasks for development activities. Chapter 3, "Roles and responsibilities in governing decisions" on page 25 highlights the roles that are the most important to identify to support these activities.

To support this approach, three important concepts must be introduced:

- Releases
- Change activities
- Validation activities

Releases

The release is a container for change activities and validation activities. In essence, the CCB approves a list of change activities to be included in a release.

Associated with this release, and based on the change activities, is a list of validation activities that, after they are completed and approved, should allow the release to be closed and eventually deployed.

Figure 1-5 shows a conceptual diagram that represents these concepts.

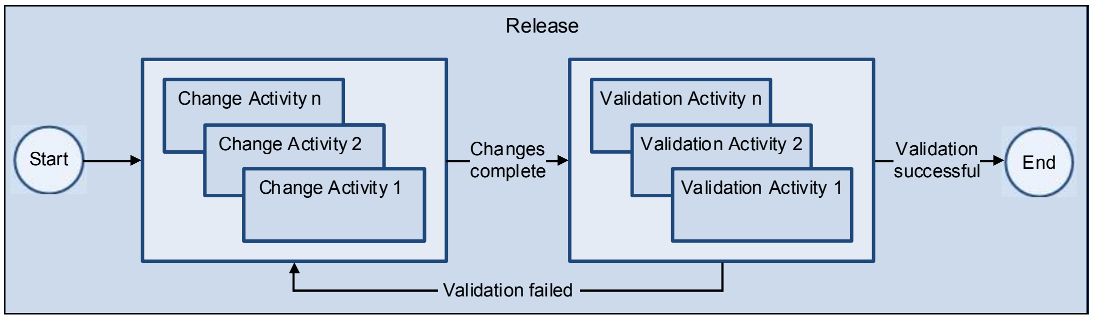


Figure 1-5 Release change validation

Change activities

A change activity is the smallest unit of management of changes to decisions. It is typically initiated by one of the following situations:

- Business policy change (for internal purposes or compliance).
- Technical change (new data, changes to system interfaces, changes to web services, and so on).
- Test fixes of existing decisions that are in production.

These situations initiate a change activity that is justified, prioritized, and tracked to provide visibility and traceability of the change.

Each change activity might materialize as a series of changes to individual business rules, technical components, and so on. The change activity has a status that provides a view of where this change activity is in its lifecycle.

Because the status is on a change activity, there is no need to track the status at an individual rule or component level, which addresses one of the pain points you see when you deal with rule statuses in real-life projects. The granularity of rule statuses is often too fine to be manageable and useful. By changing the level of granularity to the change activity, the governance is simplified.

Change activities are addressed in more detail in 6.2, “Change activities” on page 75.

Validation activities

Validation activities are the activities that are required by governance to make sure that the goals of the change activities are met, and that they are correct and complete. The following validation activities exist:

- Peer review of the changes.
- Decision Validation Service (DVS) tests of the decision changes.
- Simulations (and the use of key performance indicators (KPIs)) of the decision changes
- Integration testing with the systems outside of Operational Decision Manager

When you run the validation activities and the validation fails, you might have to go back to change activities to implement a fix to the change.

Validation activities are addressed in more detail in 6.3, “Validation activities” on page 79.

The end of rule statuses

This book promotes an updated way to manage rules as part of your governance process. The branching and merging functionality that is now available in Operational Decision Manager provides a way to simplify the management of rules by using individual rule statuses. This situation is a departure from what was previously promoted to help with the governance and management of rule changes.

The use of branches allows users to manage the changes to multiple rules that are part of a change activity for a decision update. This situation removes the sometimes painful necessity of micro-managing individual rule statuses that might have been updated for a specific change request.

Does this mean that rule statuses have no use anymore? Not necessarily, but the usage of the branching capability of the product can simplify the decision governance a great deal by allowing users to change multiple rules and test all of those changes as part of a branch. After the changes are validated, they can be merged back into the original branch they came from as a whole.

If you are using rule statuses to manage changes to rules in your organization, start looking at a transition to using branches and updating your processes accordingly. If you are not using decision management technology yet, consider using branches and not individual rule statuses.

This concept is reiterated throughout the book when it is appropriate. You can also find a more detailed introduction to branching and merging in 5.5, “Branching and merging to manage releases” on page 55.

1.7.3 About previous versions of IBM Operational Decision Manager

Some readers might be using an older version of the IBM Operational Decision Manager product, and although the latest version of the product provides more facility at implementing and using the concepts that are presented in this book, it should be possible, with a bit more work, to apply the same concepts with the previous versions of the product.

Although the concepts of releases, change activities, and validation activities might not be explicit in the version that you use, it is possible to apply the ideas behind the concepts in an adapted way through the usage of the branches and merging in the repository.

1.8 References

For more information, see the following books:

- *CMMI for Development, Version 1.3 (CMU/SEI-2010-TR-033)*, found at:

  `http://www.sei.cmu.edu/library/abstracts/reports/10tr033.cfm`

- *Business Decision Maturity Model (BDMM)*, found at:

  `http://www.kpiusa.com/index.php/The-Decision-Model/business-decision-maturity-model-bdmm.html`

- von Halle, et al, *The Decision Model - A Business Logic Framework Linking Business and Technology*, Taylor & Francis, 2009
- *Making Better Decisions Using IBM WebSphere Operational Decision Management*, REDP-4836, found at:

  `http://www.redbooks.ibm.com/abstracts/redp4836.html`

- *COBIT 5: A Business Framework for the Governance and Management of Enterprise IT*, found at:

  `http://www.isaca.org/COBIT/Pages/default.aspx`

- *COBIT 5: Enabling Processes*, found at:

  `http://www.isaca.org/COBIT/Pages/COBIT-5-Enabling-Processes-product-page.aspx`

- Office of Government Commerce (OGC), *The Official Introduction to the ITIL Service Lifecycle*, TSO, 2007, ISBN 0113310617
- Office of Government Commerce (OGC), *ITIL Service Transition*, TSO, 2011, ISBN 9780113313068
- *The Open Group Architecture Framework (TOGAF)*, found at:

  `http://www.opengroup.org/togaf/`

- *SOA Policy*, found at:

  `http://www.ibm.com/developerworks/webservices/library/ws-soa-policy/index.html/`

2

Sample scenario

This book uses a fictional auto insurance organization to illustrate how IBM Operational Decision Manager can be used to govern decision making in an enterprise scalable way.

The goal is to show how the pricing for auto insurance quotes can be:

- Represented as a pricing decision service.
- Broken down into rule projects to support appropriate access by underwriters, marketing, and regional business roles, as well as the IT roles.
- Changed in an agile way by the appropriate roles with appropriate skill levels by using the appropriate capabilities, ensuring traceability of changes to the business goals.
- Tested and validated to ensure that the changes meet the business requirements.
- Deployed to the operational systems in a manner that conforms with an existing IT governance process.

This chapter provides a high-level overview of the insurance organization solution and the change management scenarios that are illustrated.

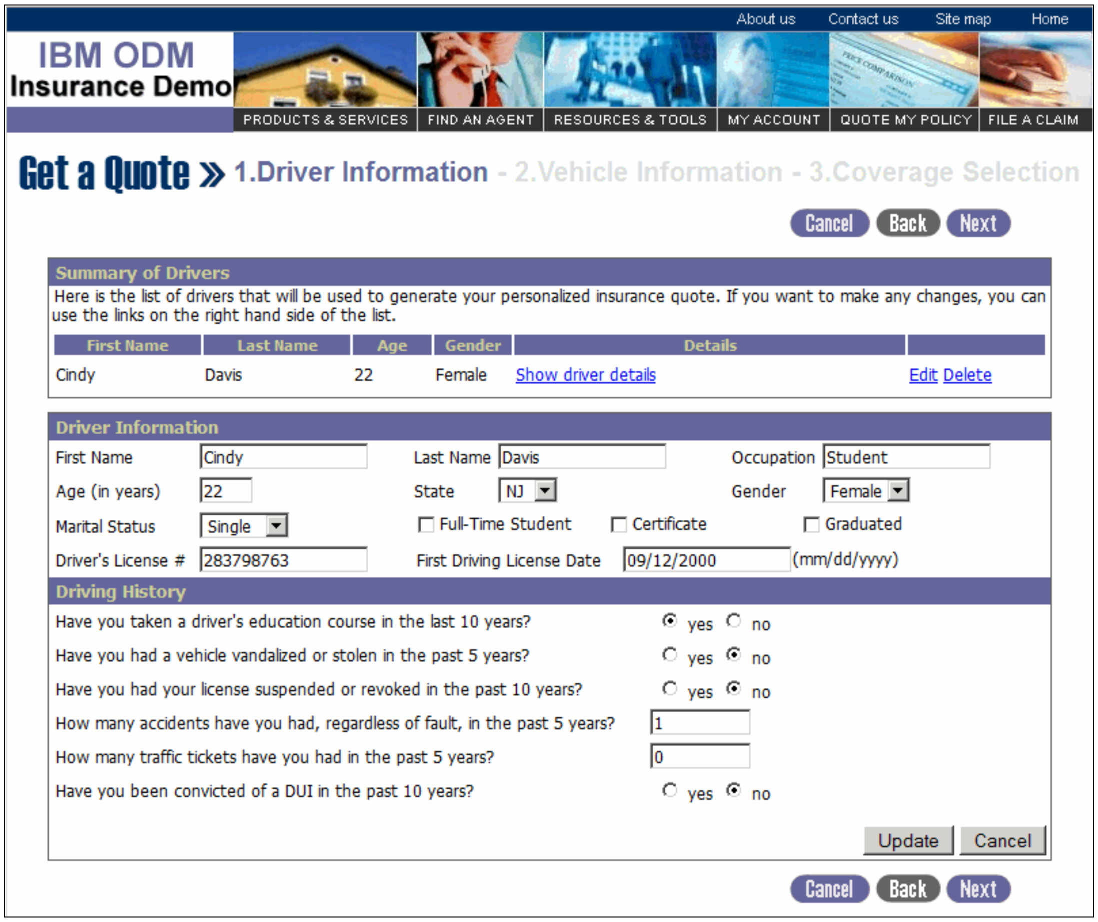
This chapter contains the following topics:

- Insurance organization background
- The insurance solution
- Change management scenarios

© Copyright IBM Corp. 2013. All rights reserved.

2.1 Insurance organization background

The auto insurance organization has two channels to market: a self-service website and a call center. Using either channel, customers can apply for insurance quotes by providing the driver characteristics, vehicle characteristics, and type of coverage that is required, as shown in Figure 2-1.



Summary of Drivers

Here is the list of drivers that will be used to generate your personalized insurance quote. If you want to make any changes, you can use the links on the right hand side of the list.

First Name	Last Name	Age	Gender	Details	
Cindy	Davis	22	Female	Show driver details	Edit Delete

Driver Information

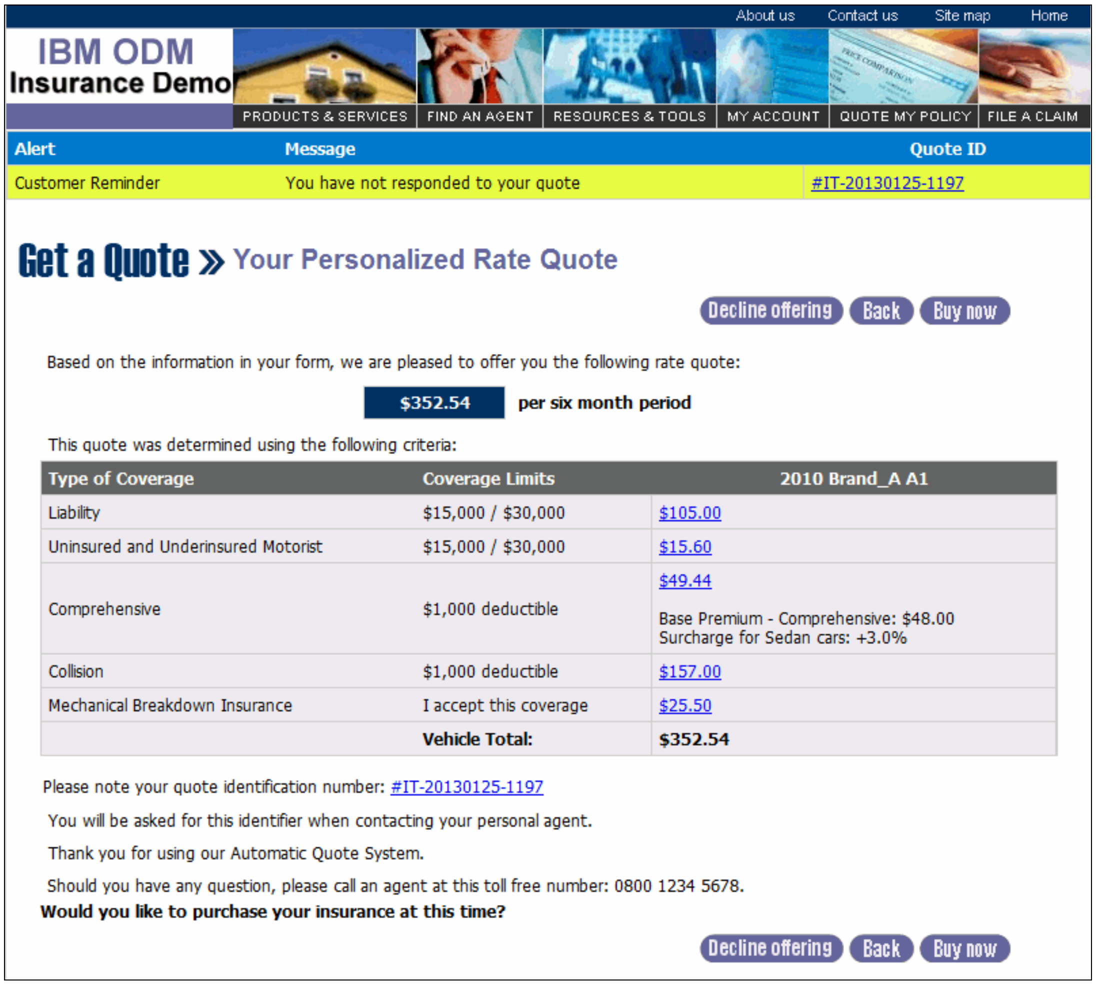
First Name	Cindy	Last Name	Davis	Occupation	Student
Age (in years)	22	State	NJ	Gender	Female
Marital Status	Single	☐ Full-Time Student	☐ Certificate	☐ Graduated	
Driver's License #	283798763	First Driving License Date	09/12/2000	(mm/dd/yyyy)	

Driving History

Have you taken a driver's education course in the last 10 years?	[✓ yes] no
Have you had a vehicle vandalized or stolen in the past 5 years?	yes [✓ no]
Have you had your license suspended or revoked in the past 10 years?	yes [✓ no]
How many accidents have you had, regardless of fault, in the past 5 years?	1
How many traffic tickets have you had in the past 5 years?	0
Have you been convicted of a DUI in the past 10 years?	yes [✓ no]

Figure 2-1 Quote request website

Based on these characteristics, underwriting decisions are made about the risk of providing insurance and, if approved, a price is calculated and a quote offer is made to the customer, as shown in Figure 2-2.



Alert	Message	Quote ID
Customer Reminder	You have not responded to your quote	#IT-20130125-1197

Get a Quote » Your Personalized Rate Quote

Based on the information in your form, we are pleased to offer you the following rate quote:

$352.54 **per six month period**

This quote was determined using the following criteria:

Type of Coverage	Coverage Limits	2010 Brand_A A1
Liability	$15,000 / $30,000	$105.00
Uninsured and Underinsured Motorist	$15,000 / $30,000	$15.60
Comprehensive	$1,000 deductible	$49.44 Base Premium - Comprehensive: $48.00 Surcharge for Sedan cars: +3.0%
Collision	$1,000 deductible	$157.00
Mechanical Breakdown Insurance	I accept this coverage	$25.50
	Vehicle Total:	**$352.54**

Please note your quote identification number: #IT-20130125-1197

You will be asked for this identifier when contacting your personal agent.

Thank you for using our Automatic Quote System.

Should you have any question, please call an agent at this toll free number: 0800 1234 5678.

Would you like to purchase your insurance at this time?

Figure 2-2 Quote offer

The insurance organization uses IBM Operational Decision Manager to automate the underwriting decisions and pricing rules to reduce risk, improve profitability, and ensure that they can react quickly to marketplace trends. They are also concerned about the amount of market share they have and must attract a higher proportion of profitable customers. They intend to accomplish this task by identifying situations where there is an opportunity to retain a desirable customer by offering personalized promotions.

In the scenario, this solution is available and working and you consider how governed changes can be made to this solution in an enterprise scalable way.

2.2 The insurance solution

The insurance organization's solution is shown in Figure 2-3 and consists of four components:

- Website (self-service interface)
- Call Center
- Decision execution (Decision Server)
- Decision management (Decision Center)

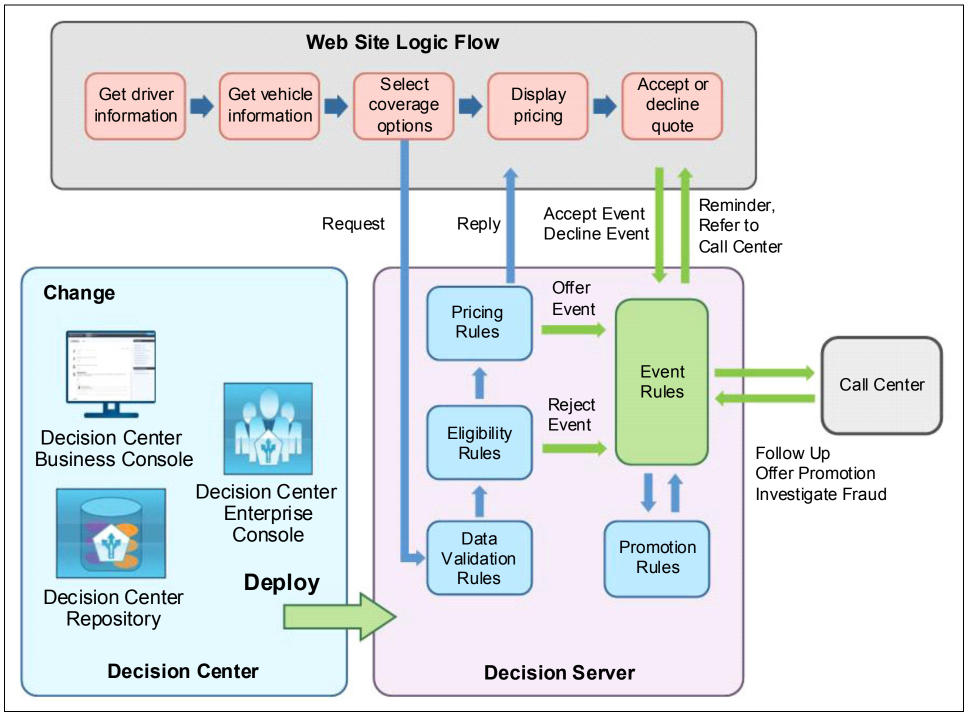


Figure 2-3 Insurance solution

2.2.1 Website (self-service interface)

The website provides the main means for a customer to obtain quick quotations. On arriving at the website, the customer is asked for details about the driver, the vehicle, and the type of insurance coverage that is required.

The website application first validates (Data Validation decision) the information that is provided and then invokes Decision Server to check the underwriting rules and assess if this driver is eligible for insurance.

The Eligibility decision assesses the risk and identifies the quote as eligible, ineligible, or requiring manual approval. If the decision is ineligible or manual, the quote is rejected and a reason for rejection is passed to the customer.

If the quote is eligible, the Pricing decision is made and a quote is offered to the customer. When a quote is offered to the customer, they then have the option of Accepting the quote or Declining the quote. As with any web application, they can also close the browser and leave the website.

The website application also integrates with the Decision Server Event run time. Events are generated whenever a quote is Offered or Rejected by Decision Server and also when a customer Accepts or Declines a quote.

These events are used in the Customer Acquisition Situations project to determine customer behavior patterns and decide what actions to take to close a sale.

Two actions are possible to influence the customer though the website:

- Reminder messages can be sent to the customer about quote offers, encouraging them to accept the quote.
- The customer can also be referred to the Call Center, where they can obtain quotes that cannot be provided through the website.

If a promotional situation is identified, the Customer Acquisition Promotions decision is made to determine the most effective promotion. This decision uses business rules in association with events in a Detect - Decide - Respond pattern.

2.2.2 Call Center

The Call Center provides a similar interface to the website, but it is used by a call center operator who enters the customer details that are based on information that is received on the phone. After the details are provided, the Eligibility decision is made, and if the customer is eligible, the Pricing decision is made and a quote is obtained. Then, the Call Center operator can see the result of any promotion decisions and can then accept the quote on behalf of the customer or reject the quote.

This situation illustrates the consistent reuse of the Web Site Eligibility and Pricing decisions.

In addition, the Call Center maintains a list of referrals from the website with guidance about how to respond to those particular customers:

- Following up a customer that was unable to obtain a quote through the website.
- Providing a promotion to a desirable customer to close a sale.
- Alerting to potentially fraudulent quote applications where, for example, several quotes were requested for the same vehicle by different drivers in different states.

2.2.3 Decision execution (Decision Server)

Decision Server provides the main run times for the rules-based and event-based decisions. The design and integration of the decisions with the solution are carried out using Eclipse tools that are called Rule Designer and Event Designer. These tools provide the mechanisms to define the vocabulary that is used in the rules that make the decisions and integrate the decisions, events, and actions into the solutions by using web services, EJBs, Java, or other event connectors.

In this book, you see how Rule Designer can be used to make engineering changes to the pricing decision service, which affect how it integrates with the solution and how those changes can be synchronized with Decision Center as part of the governed release.

Further changes and refactoring to the rules are then undertaken by the business in Decision Center and are then deployed to the Decision Server run times. These are the Rule Execution Server for business rules and the Event Runtime for event processing or Situation Rules.

2.2.4 Decision management (Decision Center)

Decision Center provides business users the means to manage and govern decisions. Facilities exist for authoring, simulating, testing, and deploying the rule and event projects that realize the decisions. Changes to the rules, decision tables, and other artifacts that are used in the decisions are undertaken as part of a release, and the governance processes described in this book then ensure that the changed decisions meet the overall goals when that decision is deployed.

The unit of deployment for the business rules is a RuleApp. In this case, the InsdemoRuleApp contains the Data Validation, Eligibility, and Pricing Decisions. This RuleApp also contains another Decision that is called Customer Acquisition Promotions, which provides rules to determine the promotional discounts that are invoked directly by the event processing. For events, a single event project (Customer Acquisition Situations) is deployed to the event run time to define the situations of interest to the business (patterns of events) and deciding what action to take when these situations arise.

2.3 Change management scenarios

This section introduces some of the details about the scenarios that are covered in this book. It starts by introducing the people that are involved in the scenarios and their roles and then presents the two main change management scenarios that are used for discussion purposes.

2.3.1 Personas and roles

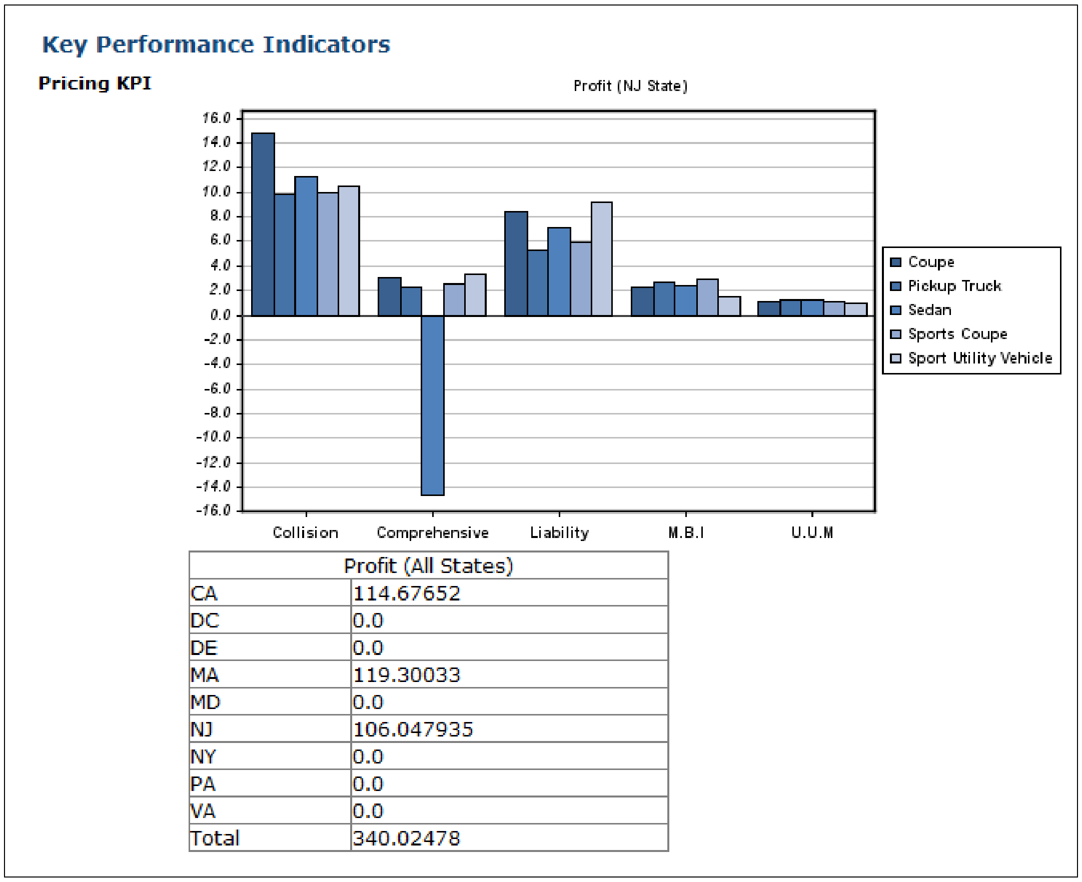
These scenarios use the following personas:

- Barbara: The pricing analyst with overall responsibility for business performance, setting the business goals, and reacting to market changes. Barbara adopts the role of the release sponsor, setting the objectives for any changes and approving the deployment of any release into production.
- Paul: The insurance pricing manager is responsible for managing the Pricing policies team and coordinating the various activities that are needed in the release. He adopts the role of Release Manager.
- Rachel: A regional manager with responsibility for managing the pricing policies for New Jersey. Rachel is a release contributor in these scenarios, and she applies her experience to authoring rules in specific areas of the pricing policies.
- Adam: The IBM Operational Decision Management Administrator with responsibility for testing and deploying decisions. In these scenarios, Adam is a Release contributor, but also undertakes a role liaising with the Configuration Management solutions to ensure that all deployments to production systems are compliant with the insurance organization's operations management policies (for example, policies that are laid down in ITIL and COBIT).
- Ivan: The IT Architect has responsibility for a team of developers that developed the Web Quoting application and its integration with IBM Operational Decision Manager. Ivan has responsibility for any Technical change and validation activities. Ivan represents the IT Architect, Rule Developer, and Integration Developer roles.
- Terry: The QA Engineer has responsibility for System Integration Testing (SIT) and User Acceptance Testing (UAT) environments and ensuring that the quality objectives for the solution are maintained.

2.3.2 Business change management scenario

The decisions that are automated in Decision Server can be managed and governed by the line of business in Decision Center. In this scenario, you see how the Line of Business (LOB) users can collaborate and change the Pricing rules to meet the business need. You learn how to ensure that those changes are undertaken securely and consistently by using the governance features of IBM Operational Decision Manager.

Barbara was performing her monthly analysis of insurance profitability and notices that the organization is suffering losses in New Jersey, especially for comprehensive coverage for sedan cars. This information comes from her business dashboards and is illustrated in a customized Operational Decision Manager simulation KPI chart that is shown in Figure 2-4.



Profit (All States)	
CA	114.67652
DC	0.0
DE	0.0
MA	119.30033
MD	0.0
NJ	106.047935
NY	0.0
PA	0.0
VA	0.0
Total	340.02478

Figure 2-4 Pricing Profitability KPIs

Pricing Profitability KPIs

Based on this information, Barbara asks Paul to include changes to resolve this matter in the next planned monthly release of the pricing policies.

In this monthly release, there are also change activities that take new promotional pricing policies that will be released at the same time into account.

IBM Operational Decision Manager supports the concepts of branches, which can be used to scope the changes that are being made to a release or within an individual change activity.

In the scenario, you learn how governance can be applied to these release and change activity branches to show the following activities:

- Rule authoring that is appropriate to LOB users
- Performing simulations to understand how the changes improved profitability and understand how you might identify other changes that are needed for the future
- Testing the Pricing project release after the changes to ensure that it still works
- Verifying that the changes operate as required by running them in a User Acceptance Test environment
- Approving the release for deployment to production
- Deploying the changed rules to production Decision Servers.

2.3.3 Technical change management scenario

The business change management scenario provides the means to quickly provide changes to pricing policies according to a planned cycle that can be managed by the LOB with reduced support from IT. However, when new information is required to make a decision or the way the information is provided from a solution changes, IT must be involved and changes must be coordinated so that the solution and pricing decision service continue to work correctly.

In this scenario, Barbara requested that Paul introduce a new roadside assistance policy that extends the Motor breakdown insurance, which allows different pricing according to whether coverage is required from a home state, nation wide, or overseas.

In this scenario, you learn how governance can be applied to technical release change and validation activities, such as:

- Changing the information model (XOM) used within the rules
- Changing the interface between the solution and the pricing decision service
- Changing the rule flows and variables to reflect the new information
- Testing the revised integration in System integration environments

3

Roles and responsibilities in governing decisions

As mentioned in the introduction of the book, one of the main goals of decision governance is to provide the support for the organizational changes that are required to run the governance processes. One of the first steps towards this goal is to identify the roles and responsibilities that are required to support and run the processes. This chapter defines governance roles and responsibilities and provides guidance for organizational changes.

This chapter contains the following topics:

- Governance roles
- References
- Organizational change

© Copyright IBM Corp. 2013. All rights reserved.

3.1 Governance roles

Table 3-1 shows a list of roles that are part of an implementation of decision governance in an enterprise. These roles are examples only and are not meant to be an exhaustive list that might be required for your organization. This list is a starting point and organizations should adjust and augment this list to fit their specific situation. The list of applicable roles evolves with time as the processes themselves evolve.

A role is not equal to a position or a job title. A specific person, holding a specific position in an organization with a job title might play multiple roles. Similarly, a single role might end up being played by multiple people or a team.

A governance role has the following characteristics:

- It is a set of defined and related responsibilities.
- It is performed by people.
- It is required for operational decision governance and management to be successful in an organization.
- It should not be confused with an organizational position that is the job title and function of a specific individual in the organization.

The relationship between the responsibilities defines the role. Table 3-1 shows these relationships. The Also Known As column provides some additional names that are sometimes used.

Table 3-1 Governance roles and responsibilities

Governance role	Responsibilities	Also known as
Business Owner	► Requests business decision changes. ► Provides business approval for a release.	► Policy Owner ► Change Initiator (ITIL)
Change Control Board (CCB)	► Triages and selects change requests. ► Selects change requests to include in a specific release. ► Delegates the delivery of a release to a release manager.	► Steering Committee ► Change Advisory Board (ITIL)
Release Manager	► Coordinates the contributors work toward a release so that it gets released in a timely manner. ► Gathers the appropriate approvals for a release. ► "Releases" the release to the Configuration Management Engineer.	► Release Owner ► Process Owner (ITIL) ► Process Manager (ITIL)

Governance role	Responsibilities	Also known as
Business Analyst	► Provides business domain knowledge. ► Acts as a liaison with Subject Matter Experts (SMEs). ► Oversees change and validation activities. ► Resolves business issues that relate to decisions. ► Creates test cases to test the decisions. ► Is accountable for the quality of the decisions. ► Approves releases by performing User Acceptance Testing.	

Governance role	Responsibilities	Also known as
Business Policy Analyst	► Determines and manages the implementation of a business policy. ► Assists business in identifying existing decisions in the business process and gathering the required supporting documentation around them. ► Formalizes the business terminology (vocabulary) used in Operational Decision Manager. ► Creates and manages the domain model that represents the business entities and attributes and their relationships. ► Designs the structure of the implementation to support the design requirements and evolution of the decisions (shared with Rule Developer). ► Specifies templates for rule developers to create. ► Analyzes decisions for completeness, correctness, optimization (from a logical, not performance, perspective) conflicts, and redundancies. ► Ensures consistency of rules across functions, geographies, and systems. ► Conducts impact analysis and simulations of updates to rule sets. ► Makes recommendations for rule changes that are based on business knowledge. ► Acts as a liaison between the business and IT at the time of the design and implementation of the decisions.	► Business Policy Architect ► Rule Analyst ► Rule Architect
Rule Author	► Authors rules to support decisions. ► Manages and runs test cases to test the decisions. ► Runs simulations.	► Rule Writer

Governance role	Responsibilities	Also known as
Rule Developer	► Designs the structure of the implementation to support the design requirements and evolution of the decisions (shared with Business Policy Analyst). ► Creates and update rule flows. ► Creates templates that are specified by the Business Policy Analyst. ► Authors technical rules to support decisions. ► Develops technical functionality to support authoring (XOM and utility functions). ► Performs impact analysis for potential changes to the rules from a technical perspective. ► Debugs decision logic. ► Challenges authoring for ambiguity, inconsistency, and conflict from a technical perspective. ► Creates and manages technical components to support test cases to test the decisions.	
Operational Decision Manager DC Administrator	► Creates a.nd configures projects ► Manages permissions and access rights on projects and artifacts.	
Configuration Management Engineer	► Performs configuration management tasks that are related to the release. ► Acts as a liaison between business and IT at the time of the deployment.	

Governance role	Responsibilities	Also known as
IT Architect	► Designs the solution to ensure the performance of the rule execution and usability of the BRMS platform. ► Architects and designs the infrastructure and deployment patterns of Operational Decision Manager in the enterprise environment to support the requirements and evolution of the decisions. ► Supervises the development efforts. ► Requests technical changes to support business decision changes (new data sources and XOM changes). ► Provides technical approval for a release.	► Software Architect ► Application Architect
IT Integration Developer	► Develops components around integration to other systems, web services, and data services.	
IT Quality Assurance (QA)	► Performs integration testing.	Tester

Experience scale: Many of the roles that are described in Table 3-1 have different levels that are based on the maturity and experience of the person that plays that role. It is easy to make the parallel to functional positions that are advertised as junior, intermediate, or senior positions. Similarly, you can fully expect this type of experience scale to be applied to some of the governance roles that are listed above, although we chose not to detail them here because each organization probably wants to use their own experience scale

Here are some specific comments about the roles:

- Subject matter expert versus business analyst: In the context of decision management, a business analyst often provides the domain knowledge that is required for the project and in some cases acts as a liaison to subject matter experts. That is not to say that on a project you might not have a *real* subject matter expert that is available, but from experience, we often see SME availability as a challenge to projects.
- Business analyst versus business policy analyst: Business analysts, as they become more familiar with decision management technology, the products, the methodology, and so on, can sometimes evolve into business policy analysts.
- Business policy analyst versus business policy architect: The difference between the first name and the second name is mostly around the experience scale. Some of the responsibilities of the business policy analyst are initially better handled by someone with more experience.

- Business policy analyst versus rule analyst: Both names are analysts and have the same responsibilities. The usage of the word rule can be considered limiting in its meaning, so it is therefore more appropriate to use business policy.
- Business policy analyst versus rule architect: The distinction and comments of the two items above apply here.
- Rule Author versus Rule Developer: The first name focuses on authoring rules through the Business Console or the Enterprise Console of the Decisions Center. The second name is more technical, works mainly in the Rule Designer component of the platform, and also develops technical components outside of rule projects.
- IT Architect, Software Architect, Application Architect: These three names are equivalent and the name might vary in each organization so the responsibilities are really what we are concerned with for this role.
- Operational Decision Manager RES Administration: The administration of the Rule Execution Server (RES) is an operational task that happens after a release has made it to production. We chose to omit it from the governance roles, although it is expected that someone has those responsibilities.

In addition to the roles described in Table 3-1 on page 26, Table 3-2 describes some additional roles that are required roles for the touch points between the decision governance and IT governance. These roles are sometimes collectively referred to as *IT Infrastructure Administration* in the rest of the book.

Table 3-2 More governance roles

Role	Touchpoint	Responsibility
LDAP administrator	► User creation ► Group creation	► Creates and maintains users and groups to support the security requirements.
Database administrator	► Installation of Operational Decision Manager ► Updates to Operational Decision Manager Databases	► Creates and maintains databases as required by IBM Operational Decision Manager. ► Operational Decision Manager Database installation and configuration.
Application server administrator	► Installation of Operational Decision Manager ► User creation ► Group creation	► Deployment of the Operational Decision Manager EAR files. ► Creates and maintains users and groups to support the security requirements.
Networking administrator	► Installation of Operational Decision Manager	► Configuration of DNS. ► Configuration of firewall security.

3.2 References

For more information about governance, see the following resources:

- Boyer, et al, *Agile Business Rules Development*, Springer, 2011, ISBN 3642190405
- *Business Rules Governance and Management*, found at:

  `http://www.primatek.ca/blog/?dlname=Primatek-BusinessRulesGovernanceManagement¬`
  `WhitePaper.pdf`

- ABRD website guidance on governance:

  `http://epf.eclipse.org/bp/abrd_1.5.1_20100928/process.abrd.base/deliveryprocess`
  `es/abrd_governance_BF9D60E1.html?nodeId=464d376e`

3.3 Organizational change

This section describes how changes within an organization impacts governance and how it can best be handled.

3.3.1 Planning for organizational change

To use the information that is shown in Table 3-1 on page 26 and Table 3-2 on page 31, you must do some initial work to help you create the vision for the organizational change that is needed in your organization.

Here are some steps to help you get started:

- Start by identifying the specific governance roles that your organization needs, including the responsibilities that these roles have. This is your governance vision.
- Map the governance roles that you identified to functional roles within your organization. This action might possibly involve identifying specific people to fill a governance role, although from a functional role perspective that person is not in the correct place. This is your governance reality.
- Identify the gaps in people or skills as well as the changes that are required to move your governance reality towards your governance vision. Here are some possible changes:
  - Temporarily using external consultants while internal teams acquire the wanted skills.
  - Training people (formal or on the job) to give them the skills they need.
  - Hiring new people to complement your existing team.

Tip: Identify the gaps in people or skills as well as the changes that are required to move your governance reality towards your governance vision.

Before you attempt any organizational changes, it is important to consider the emotional response of the people who might be affected by such a change. The organizational change that is required for supporting operational decision governance and management may not be extensive, especially compared to other types of organizational changes, but it does have an impact. With this situation in mind, you should these focus on these two fundamental points when you consider this specific type of organizational change:

- Develop your workforce.
- Communication is key.

Workforce

As the organization works towards supporting operational decision management, the future needs of the organization are more clearly defined and the following questions should be answered:

- What are the new roles and their responsibilities?
- How many people are required to fill each role?
- What are the skills that are required for each role?

To answer these questions, start with a small subset of key stakeholders that can provide their input and participate in constructive discussions around the organizational changes, what the changes mean to the other stakeholders, and to make sure that most potential roadblocks are identified and mitigated early on.

The answers to the questions above then lets you identify in more detail how the gaps to achieve the goals of the organizational change can be filled. For example:

- Identify specific employees who play the roles that are identified.
- Identify skills or knowledge gaps and create development plans for each employee.
- Identify training and education that might be required to help employees achieve the goals.

The key stakeholders can help provide details about all this information and prepare the material that is required to present to all the stakeholders in follow-up meetings. These meetings are important in supporting your communication goals for the organizational change.

Communication

Because you might encounter some resistance from some employees, it is important for you to address this resistance quickly and help them align with the vision and direction the organization is taking. To achieve success, it is critical that communication with stakeholders take place early so that they are made aware of the change and provided with information about the change. With time, they start understanding and accept the change before they take ownership of the change and committing to it.

Here are some key points to remember:

- Communicate the vision of the change early by providing details about the benefits and impact:
  - Reasons for the change
  - Benefits for them and the organization
  - Details of the change (when, where, and who is involved)
- Communicate often and repeat the message through multiple channels.
- Involve the stakeholders by having an open dialog with them.

The changes might be as simple as identifying the roles and responsibilities and assigning them to specific people in your organization. In that case, the communication may be achieved by calling a meeting with all the stakeholders that are involved and discussing with all of them the roles and responsibilities that are required, reviewing them, and assigning them to the people in the room. Then, the details can be discussed and clarified. Should you take this approach, be ready to discuss the topics around the workforce development as detailed above.

Further references on the organizational change

For more information, see Chapter 5, “Managing People through Service Transitions”, in *ITIL Service Transition* by the Office of Government Commerce (OGC). That chapter can be used as a starting point for managing your own organizational change.

3.3.2 The initial role of the Center of Excellence

It is important to highlight the potential role a Center of Excellence (CoE) has with helping an organization through a change and with helping the people build the skills and knowledge they need to fill their new roles.

Tip: The CoE does not have to be an actual organizational entity of its own accord, or at least not at the beginning.

The CoE can take a simplified form at the beginning where it simply consists of a community of practitioners within the organizations where the participants share their experiences, practices, and guidance with others.

This community can play an important role in the success of decision management and governance in your organization. At first, the participants are the people that run the first project in your organization, but as the usage of decision management continues to expand, these people possibly become some key participants in the community.

The CoE community should help with the tasks:

- Acquiring and keeping required skills so that the skills are available for other projects internally. This task allows for sharing of resources among multiple projects.
- Developing a broader set of people with the required skill base by supporting new people in developing their skills and knowledge.
- Providing coaching and mentoring for new people by providing them with a way to communicate with more experienced people.
- Documenting good practices in your specific organization.
- Providing a first point of contact for people to discuss issues or challenges they are facing.

It is important to make sure that there is some enticement that is tied to the participation in the CoE and that the time for participants to contribute to the community might have to be absorbed by each project, but this action might be a critical success factor for using this technology in your organization.

The building of a CoE might require external expertise to start the process and to get the CoE established. As it is built, the internal staff that is involved in the CoE should acquire the knowledge and skills that are required to keep the CoE running after the external resources leave. The CoE should also capitalize on the experience and practices of the first few projects as a starting point for good practices (and practices to avoid in that specific organization).

Tip: It might be a good idea to temporarily bring in external help to start the CoE in your organization.

The evolution of the Center of Excellence

With time, this initial form of the CoE can evolve into something more:

- Project people become CoE champions to support the next project.
- The participation in the CoE community is essential for gathering common knowledge.
- Initially, the *funding* of the time that is required by participants to participate in the CoE comes out of the project budgets.
- Eventually, there might be a need to fund a specific role to lead the CoE.
- Champions can possibly and eventually evolve into the next level as *internal consultants* as part of the CoE.
- The CoE must be adapted so that it fits with your processes and evolves over time.

Much more could be said about the topic of the Center of Excellence, but it falls out of the scope of this book.

3.3.3 Relationship to Business Process Management

Some organizations might already have started an organizational transformation to support Business Process Management (BPM) and the required roles to support it. BPM and Decision Management have much in common, and the organizational changes that are required to add Decision Management may be simplified by using the organizational changes that are required for BPM.

One important point is that the roles that were listed in 3.1, “Governance roles” on page 26 are still required and should be identified specifically for decision governance. Again, a single person in the organization might play multiple roles for decision governance, which applies for BPM roles as well.

For example, the person that plays the role of process analyst might also end up playing the role of business policy analyst. Another example is the Operational Decision Manager Administrator role, which could be played by the same person who plays the role of BPM Administrator.

Tip: A governance role does not necessarily map to a single individual in an organization.

Here are some factors that might affect the mapping of governance roles to individuals:

- The size of the organization for the availability of resources.
- The number and complexity of the projects, which might require more resources.
- The specific skill set of the individual players, which might lead to training (or to the identification of a training plan).

Decision governance most probably requires that existing management processes for BPM be adapted to include decision management processes or to identify how they relate and interact.

Tip: BPM processes must be adapted or adjusted to consider the Decision Management processes.

4

Organizing decision management

This chapter describes the goal of operational decision management to provide the means to represent and manage high-value business decisions, as well as make them executable in the context of an IT-based business solution. This decision making capability can be considered a service in the context of a service-oriented architecture (SOA) so that you can apply the change management practices that are expressed in ITIL and COBIT.

The IBM Operational Decision Manager platform provides many artifacts to support the definition and the structuring of decision services, both in the context of the rule authoring phase (by using rule projects and rule packages artifacts) and the rules deployment phase (by using artifacts such as rule sets and RuleApps). This chapter describes how to use these constructs to structure a decision service. After this chapter establishes the function of the different structuring artifacts, it then examines how to partition the management of these artifacts along the different project roles through the concepts of security and permissions that are available in the IBM Operational Decision Manager platform. The resulting permission scheme is essential to implementing the separation of responsibilities that are prescribed by the rule governance processes.

This chapter contains the following topics:

- Decision service structure
- Roles and permission management

© Copyright IBM Corp. 2013. All rights reserved.

4.1 Decision service structure

The main idea behind the business rules management paradigm is that part of the decision making capability of an enterprise can be automated by analyzing its business policies, and decomposing and organizing them into manageable and executable business rules artifacts, such as action rules, decision tables, and decision trees.

The foundation for a decision service is a collection of rule projects that contain these rule artifacts. Within each rule project are rule packages that group the artifacts in a functional hierarchy. Rule packages, from the different rule projects, are then strung together into rule flows that determine the sequence and sets of rule artifacts that are applied in response to a particular decision request.

IBM Operational Decision Manager groups these rule artifacts, flows, and parameters in an executable unit that is called a *rule set*. This rule set has a defined interface in terms of the inputs and outputs that are processed, and often forms the basis for validation activities, such as simulation and testing. In SOA terms, you can consider this rule set as a decision service operation. Building on rule sets, the unit of deployment is the *RuleApp* or rule application. Each RuleApp represents a separate collection of rule sets that are deployed to a decision server with bindings to indicate how they may be invoked by client applications.

The decision service is formed by the combination of all the artifacts that we mentioned above. In the rest of this chapter, we study in more detail each artifact and how they come together to define a full-fledged decision service.

4.1.1 Role of rule projects

Rule projects are a top-level entity in the Operational Decision Manager platform. They can reference (or be referenced by) other rule projects, but they cannot encapsulate other rule projects. In the context of a simple application, a single rule project might be enough to define all the necessary artifacts (business object model, rule set parameters, rules, rule flow, and so on) and produce the wanted executable rule set.

However, when you address the construction of enterprise decision services, which are composed of multiple decision operations that are large, complex, and usually share some of their logic, the rule projects must be organized in a more modular fashion, each with a specific function. The following paragraphs provide a recommended function decomposition for rule projects, and show how to assemble the different functions to achieve the wanted flexibility in the design of a decision service.

4.1.2 Business Object Model project

The Business Object Model (BOM) is an entity that is reusable across the different operations of a service, and sometimes part of it reusable across the overall organization. It is also an entity that evolves less frequently than the rules during the decision lifecycle. Thus, it makes sense to define a common BOM project for the decision service. As a convention, no other artifact than the BOM is defined by this project. Any decision service involves at least one such BOM project, but there can be more than one depending on the complexity of the BOM and the scope of the decisions. These BOM projects are referenced (directly or transitively) as needed by all other rule projects to support the definition of the rule artifacts.

4.1.3 Context variables project

The role of the context variables rule project is to provide a definition of all the variables that are needed to write the rules in the different functional rule projects. These rule set variables eventually are connected to the input and output rule set parameters that are defined in the signature projects.

The definition of these variables in a separate project facilitates the building of the signature of the rule sets independently from the rule projects where the rules are defined, and thus provide the flexibility to define multiple service operations that might have rules in common but differ in the context of the information they are using.

For example, the first version of a Pricing decision for an auto insurance policy quotation can rely only on the characteristics of the insured car itself and some core information about the drivers. Then, as the system matures, a more sophisticated Pricing decision can be designed that uses external information, such as a DMV report on the drivers accidents and traffic violations. The DMV report variable can be introduced in the context variables project, and new pricing rules can be written that use this DMV report information, without impacting the existing Pricing decision. Later, a new signature project can be created that establishes the flow for pricing with the DMV report information, as well as introduce the DMV report object as part of the new pricing operation signature. Again, this situation is achieved without interfering with the existing service operations.

One inconvenience of this practice must be noted: Rule set variables, unlike rule set parameters, are not associated with a concept of direction. So you cannot specify that a variable has an IN, OUT, or IN_OUT direction. Therefore, the rule authoring components may present the rule authors with operations (in particular, the rule actions) that should not be available. This drawback is a relatively small one compared to the flexibility brought by the approach.

4.1.4 Functional rule projects

The role of functional rule projects is to define the business rules that are derived from the different business policies, and that are run by the different operations of the decision service.

A functional rule project is usually written to support one main decision service operation or a class or functionally related operations. To continue with the example of Pricing from 4.1.2, "Business Object Model project" on page 38, the Pricing functional rule project can encapsulate the rules for both the standard Pricing decision, and the advanced one, which uses the DMV report. Two separate rule flows can then be created in the project, one for the standard and one for the advanced Pricing decision. This model is illustrated in Figure 4-1 on page 41.

Another possible organization can rely on two separate projects: one to hold the standard pricing rules and the related flow, and the other to hold the advanced rules and the related rule flow. The latter project then has a reference to the former one. This model is illustrated in Figure 4-2 on page 42.

Deciding how to divide the rules into different functional projects and which rules are hosted by which functional rule project depends on the complexity of the decisions and the number of business dimensions (for example, geography or products). You also must take into account whether different groups of users must have different access rights to different projects. Within the same pricing function, some core risk assessment rules might be sensitive material and have their access restricted to a limited group of users and thus placed in a separate project, while others, such as adjustments and fees, might be accessible by a larger group of users and placed in another project. For more information, see 4.2, “Roles and permission management” on page 44.

4.1.5 Signature project

The role of a signature rule project is to orchestrate and use the rules as a decision service operation. A signature rule project is the lead project from which a rule set is generated. As such, it contains the elements that are necessary to define a standard rule set: a main rule flow that is used to define the entry point and guide the execution, and the signature of the operation through the project's rule set parameters.

The first task that is performed by the flow of a signature project is to connect the project's rule set parameters to the rule set variables from the context variables rule project. Then, in its simplest form, the flow then simply references a subflow from one of the functional rule projects, through a flow task. When you combine rule projects, you must make sure that rules have different names across the different rule projects to avoid any collision during the rule set archive generation process.

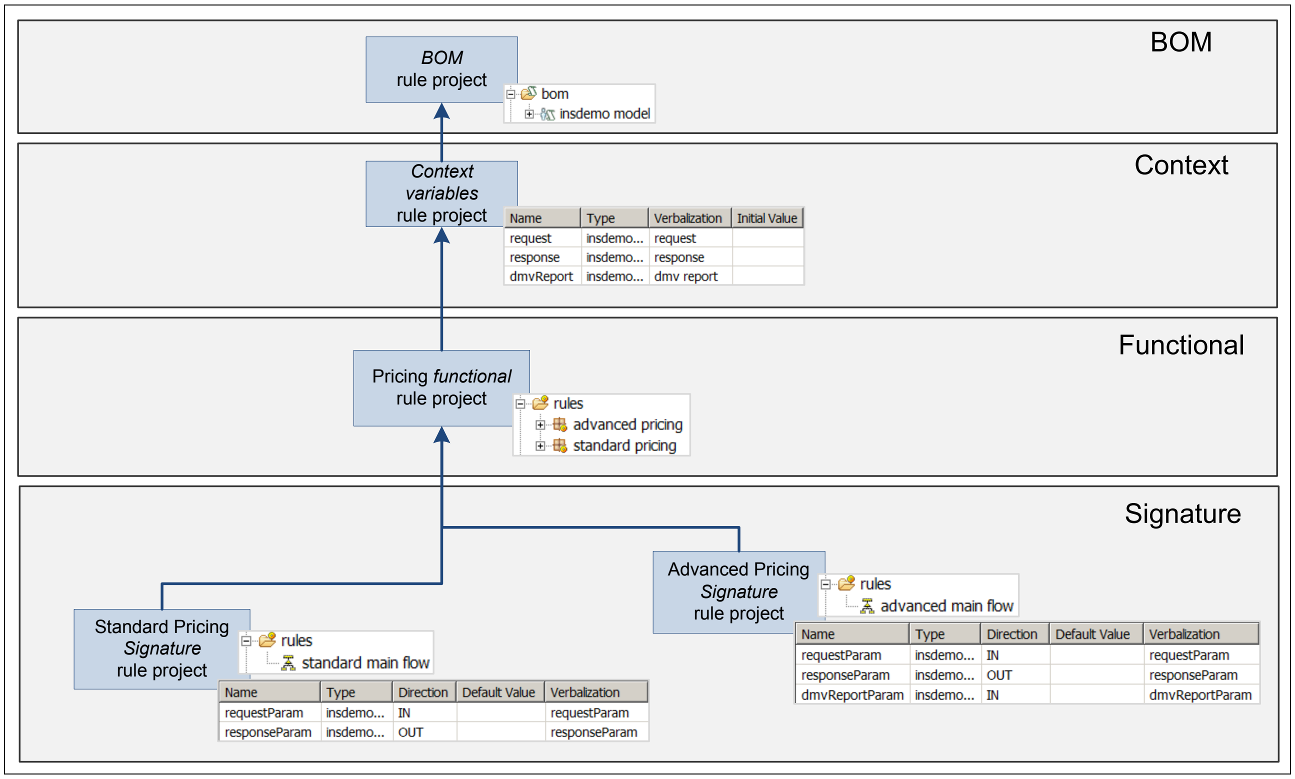
One of the main benefit of decoupling signature projects from the functional rule projects is that you are able to create decision operations without impacting the definition of the existing ones, and thus not requiring any change by its clients. If a new operation needs more context information than the existing one, you do not have to change the object model to integrate the new information. Instead, you create a signature rule project and adjust the signature to include the information through more parameters.

Another benefit of using signature projects is related to functional testing and simulation with the Decision Validation Services (DVS) component. Suppose that our insurance decision service has three main operations: Validation, Eligibility, and Pricing. Using DVS, these operations can be tested independently by the business users. However, these business users might also want to validate the behavior of the combined execution of the three operations. This validation can be achieved by creating a signature rule project that is based on the combined set of parameters that are used by Validation, Eligibility, and Pricing, and which rule flow is the sequence of these three operations. The rule set that is derived from this signature project is not exposed to the production environment, but provides a business-friendly way to validate the combination of operations. The opposite situation is also found where a complex decision operation is composed of multiple high-level tasks, and the business wants to be able to test or apply simulations to rule sets that would be built from each individual high-level task.

4.1.6 Project organization examples

This section contains two examples of organizing the different types of projects.

Figure 4-1 shows how different signature projects can use the same rule project to derive different decision operations. In this case, a standard pricing operation is defined initially by using only a Request and Response parameter. An advanced pricing policy is then introduced by using more information from a DMV report. The report is added to the list of variables of the context variables projects so that the advanced pricing rules can be authored in the pricing functional project. Finally, the advanced pricing signature project is created independently from the standard pricing signature project, which provides a new pricing operation.



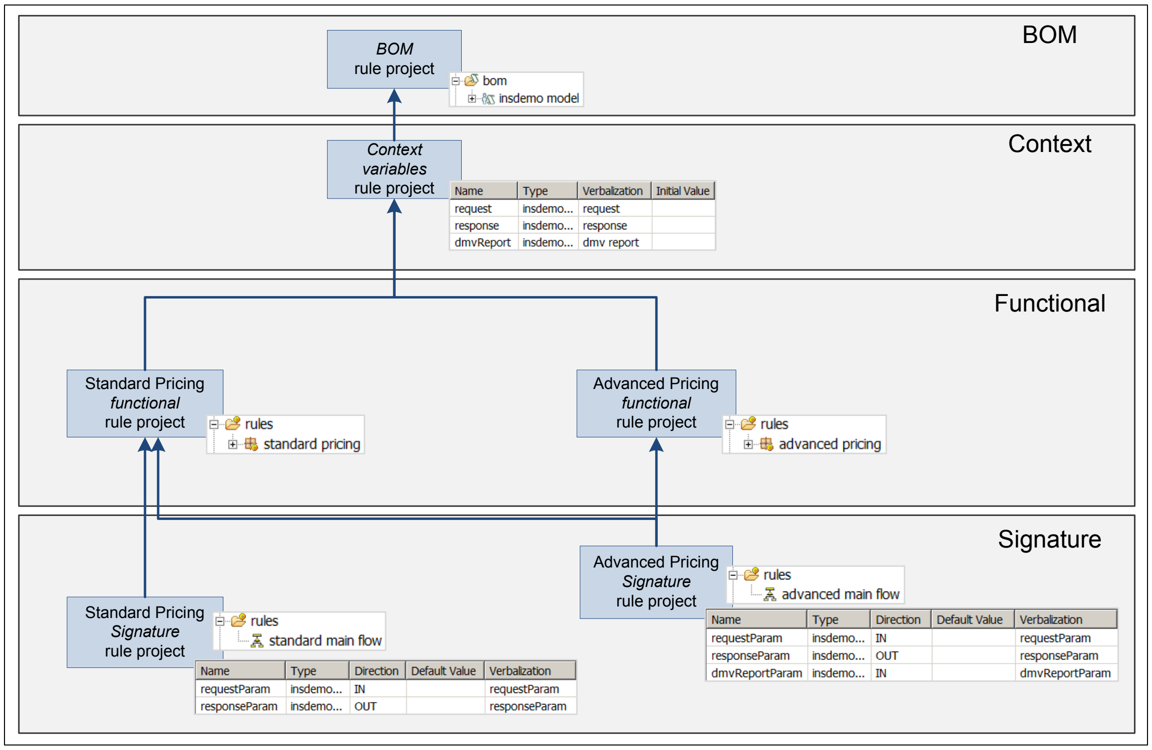
Name	Type	Verbalization	Initial Value
request	insdemo...	request	
response	insdemo...	response	
dmvReport	insdemo...	dmv report	

Name	Type	Direction	Default Value	Verbalization
requestParam	insdemo...	IN		requestParam
responseParam	insdemo...	OUT		responseParam

Name	Type	Direction	Default Value	Verbalization
requestParam	insdemo...	IN		requestParam
responseParam	insdemo...	OUT		responseParam
dmvReportParam	insdemo...	IN		dmvReportParam

Figure 4-1 Rule projects organization - example 1

In another project organization, which is shown in Figure 4-2, the rules for advanced pricing are in a separate functional project. This configuration allows you to easily define different access permissions for the two functional projects.


BOM
BOM rule project
bom
insdemo model
Context
Context variables rule project
Functional
Standard Pricing functional rule project
rules
standard pricing
Advanced Pricing functional rule project
rules
advanced pricing
Signature
Advanced Pricing Signature rule project
rules
advanced main flow
Standard Pricing Signature rule project
rules
standard main flow

Name	Type	Verbalization	Initial Value
request	insdemo...	request	
response	insdemo...	response	
dmvReport	insdemo...	dmv report	

Name	Type	Direction	Default Value	Verbalization
requestParam	insdemo...	IN		requestParam
responseParam	insdemo...	OUT		responseParam

Name	Type	Direction	Default Value	Verbalization
requestParam	insdemo...	IN		requestParam
responseParam	insdemo...	OUT		responseParam
dmvReportParam	insdemo...	IN		dmvReportParam

Figure 4-2 Rule projects organization - example 2

4.1.7 Role of rule packages and folders

Inside a project, rules are organized in a hierarchy of rule packages. The packages hierarchy allows you to group rules that are logically related by a certain business dimension. The top level of the hierarchy usually follows the division along the successive steps of the decision process and these package end up being associated with rule tasks in the rule flow. Subpackages are then divided along other business dimensions, such as a specific product characteristic or the geography in which the rules apply.

This static organization makes it easy for the rule author to quickly and intuitively navigate to an existing rule or to a location where the author can create one. It also facilitates the design of the rule orchestration. From a rule governance point of view, packages can be used to control the user access to the rules. Permissions can be established on a package so that its access is authorized only to the users that are part of the group that is designated by the package. For more information, see 4.2, “Roles and permission management” on page 44.

Sometimes, the single hierarchical organization that is provided by packages is not sufficient to correctly model and organize the different business dimensions of a rule project. With a package, you are provided with an entity that contains multiple rules. You might need to represent a dual state where a rule is associated to one or more groups. For example, you might want to represent a situation where a fee rule is applicable for two different products or maybe only in four different regions or states. This situation can be achieved by associating some metadata (or property) with the rules. The fee rules can have, for example, an applicableProducts and an applicableStates property that record when and where the fee is applicable.

Property values can be used within the rule conditions themselves at run time. However, they are more often used during the rule set extraction process through the usage of extractors, which filter the rules according to their property values.

Properties are also frequently used as the basis for defining Smart Folders in Decision Center. A Smart Folder is a dynamic folder, the content of which is driven by a repository query. The query can be defined against the values of the rule properties. The Smart Folder then shows the rules that satisfy the query expression. For example, you can write a query for a Smart Folder that gathers all the rules for an applicableStates property that contains “NY” as follows:

Find all business rules such that the applicable states of each business rule contain "NY".

This query provides you with the hierarchy of rules in the project, which applies to the state of New York. As for any other rule repository elements, specific permissions can be applied to Smart Folders so that their elements are accessible only to the group of users that are intended by the rule governance.

4.1.8 Rule sets as exposed decision operations

A rule set is the executable entity that is used by the Operational Decision Manager rule engine. Rule sets can be invoked individually, designated by a rule set path that might include the mention of a version, and providing values for the rule set parameters. Rule sets represent the implementation of decision operations and are exposed as such by the Rule Execution Server.

A rule set is defined by a source rule project and, optionally, an extractor. Using the recommended rule projects decomposition that we described previously, rule sets are associated with a signature project, which defines the signature of the decision operation and the main rule flow. An extractor allows you to create a view of the rule project so that only rules that satisfy the extractor's condition are included in the rule set. This can be used to support different goals:

- Create multiple decision operations that are based on a single signature rule project. For example, you can have an extractor that uses the applicableStates property to create rule sets that are dedicated to handling the pricing operation in certain states.
- Filter out rules that are not applicable in the context of the rule set. One common practice is to exclude the rules that have passed their expiration date. One side effect of using extractors in this way is that you cannot assume that a rule set is extracted from the same rule projects baseline, but at different times are dentical.

4.1.9 RuleApps as units of deployment

Rule applications (RuleApps), are the last component of the decision service structure. RuleApps are the unit of deployment for the Rule Execution Server. They encapsulate a set of related rule sets, and are associated with a version number.

Having reviewed the different elements that participate in the structure of a decision service, Figure 4-3 now provides the recommended organization for these elements to get the maximum flexibility and agility to handle change.

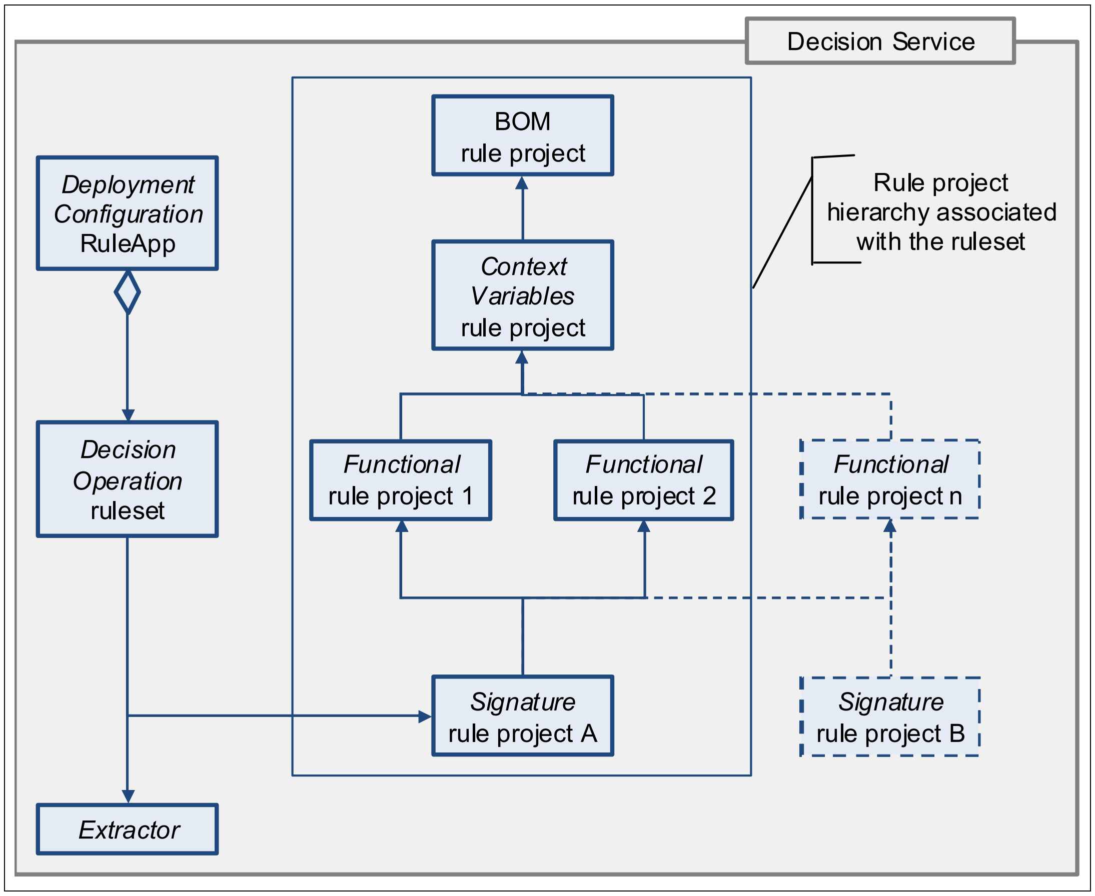

Decision Service
BOM rule project
Deployment Configuration RuleApp
Rule project hierarchy associated with the ruleset
Context Variables rule project
Decision Operation ruleset
Functional rule project 1
Functional rule project 2
Functional rule project n
Signature rule project A
Signature rule project B
Extractor

Figure 4-3 Decision Service structure diagram

4.2 Roles and permission management

To correctly govern the execution of change activities on the rule artifacts, you must define who can perform which operation on what type of rule artifacts. This means you first must identify the relevant groups of users, the possible operations, and the different criteria for grouping the rule artifacts. You can then establish the wanted permission relationships between the user groups, the operations, and the rule artifact groups.

The possible list of operations on the rules artifacts is the easy part; it is the usual create, read, update, and delete operations. The remaining task is to identify the different groups of users and the partition of the different sets of rule artifacts. These are concomitant activities, which are driven mainly by the business dimensions that are involved in the decision.

The most common business dimension is the set of functional groups that participate in the creation, the change, and the validation activities of the rule artifacts. For example, the rules to price the comprehensive, the collision, and the liability part of an insurance contract might be managed by different actuarial groups. You might want the actuaries that deal with the collision pricing to be able to view the comprehensive rules, but not modify them.

This functional dimension introduces both a partition of users into functional groups of users (comprehensive, collision, and liability actuaries) and a partition of the rule artifacts into a functional group of rule artifacts (comprehensive, collision, and liability rules). Additional dimensions, such as geography-specific or broker-specific rules, might establish further partitions of the users and of the rule artifacts.

The following sections introduce the concept of user groups and show how permissions can be established between the user groups, the create, retrieve, update, and delete operations, and groups of rule artifacts. They also show how rule elements can be partitioned and organized by using projects and possibly packages and smart folders containers to support permissions across multiple dimensions.

4.2.1 User groups

The IBM Operational Decision Manager platform provides the concept of security regarding project access and the concept of permission regarding elements of a project. Both are based on the groups that can be assumed by the user when he logs in to the Decision Center.

Each user of the Decision Center can be associated with a number of groups. The groups are divided in to two categories:

- *Predefined groups*: These groups, such as rtsUser, rtsConfigManager, or rtsAdministrator, are used to control which tabs can be accessed and which feature can be used by the user in the Decision Center consoles. For example, users of the rtsAdministrator group have an additional Configure tab available when you sign in, more buttons than the other groups in some of the toolbars, and more features in the Analyze and Project tabs. All users must belong to one of these predefined groups. Most users usually fall in to the rtsUser group.
- *Custom groups*: These groups are used to establish the functional role of the users in relation to a specific decision service or group of decision services, in a LOB for example. These groups are created on the application server that hosts the Decision Center application or the LDAP server and imported in the Decision Center database.

The custom groups usually reflect the roles that were determined as being active stakeholders in the management of the decision service artifacts. Examples of groups are it_architect or rule_author. Such high-level groups are usually enough to characterize IT functions (it_architect), which apply horizontally to all rule projects.

For Business functions, generic groups such as rule_author are reserved to a limited number of users with privileges over all rule projects. Large and complex decision services need more fine-grained groups. As mentioned before, you can have a first level of user grouping by business function. For example, our rule projects might need to differentiate eligibility_rule_author and pricing_rule_author. Then, within a business function, more decomposition can be applied along other dimensions such as geography, yielding, for example, specific groups such as pricing_rule_author_nj for the authors of the specific pricing rules in the New Jersey division.

The groups in IBM Operational Decision Manager are defined at the level of the Decision Center instance, and are not specific to a certain set of rule projects. This means that the permission properties that are associated with a group are applicable to all rule projects that are hosted by the Decision Center instance that have the group in their access control list. This means that a user who is part of a certain group has the permissions that are associated with this group for all rule projects that have this group in their access control list.

Therefore, it is important to establish a cross-line of business strategy for the creation of groups that takes into account the type, the structure, and the complexity of the rule projects that are planned to be developed and that will coexist on the Decision Center instance. This strategy allows you to balance the number of groups with the need for fine-grained permissions.

4.2.2 Project security

After groups are defined, access control can be established for the rule projects. By default, the access to a rule project in Decision Center is not protected, so any user who can log in to a Decision Center instance can select the project to access and modify its content. To limit access to a project, or a specific branch within the project, you must enable security on the project or branch.

This operation is performed in Decision Center by using the Edit branch security feature of the Configure tab, where you can select the list of groups that can access a certain project and branch combination. Because this is a configuration step for Decision Center, it is the responsibility of the Operational Decision Manager DC Administrator. This step can be achieved either manually or through a script by using the Decision Center API, which is usually applied after the creation of the rule project.

Project security can be changed during the project lifecycle, and different groups can be authorized to access different branches of the project. However, security is usually set up once on project creation and on the project's main branch. Branches' security usually use the option to inherit security configuration from their parent branch.

An example of user groups and their access to projects is presented in Figure 4-4.

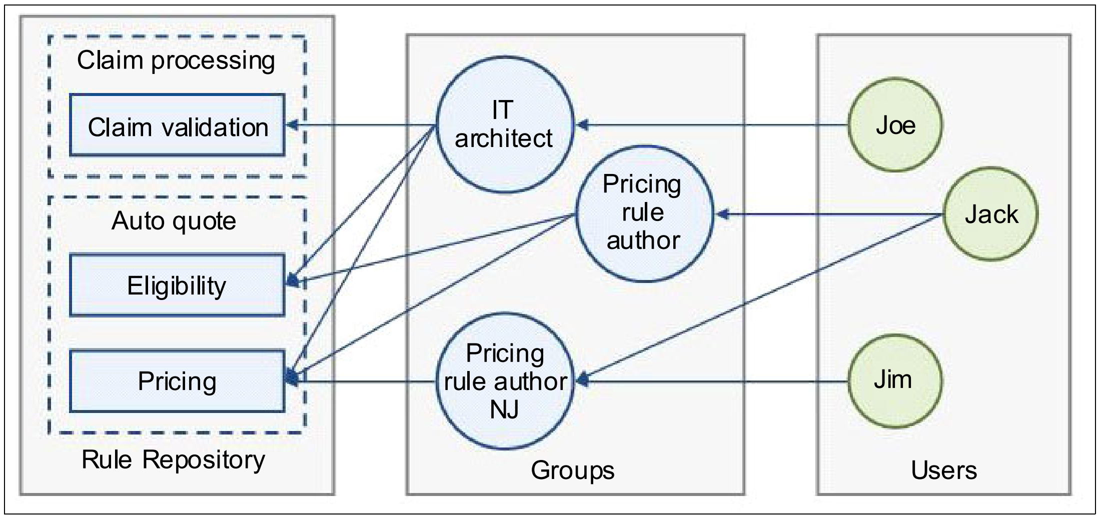


Figure 4-4 User groups security

Figure 4-4 shows the access for three groups:

- The it_architect group represents a cross-projects and cross-LOB function. It is added to the access control list of all projects. Thus, Joe, a member of the architect group, has access to all the projects in the repository. The exact nature of the access is determined by the permissions on the group.

- The pricing_rule_author group represents a function that is specific to the auto quote decision service. It is added to the access control list of all the auto quote projects. Members of this group probably do not have permission to update the rules in the Eligibility project, but they should be able to view the Eligibility rules. Again, the specific permissions for the group are defined separately. Jack is a member of the pricing_rule_author group and therefore has access to both the Eligibility and the Pricing projects. He is also a member of the pricing_rule_author_nj group. This membership does not give him access to more projects, but might affect the permissions that he has within the projects.
- The pricing_rule_author_nj group is dedicated to local managers who can access only the pricing project and not access any other auto quote projects or projects from other LOBs.

After the access control model is established, fine-grained permissions can be defined for the different groups.

4.2.3 Permissions

Permissions in Decision Center are defined for a user group and not for a specific project. Thus, if you allow the pricing_rule_author group to create and delete action rules (for example), a user who belongs to this group can create or delete action rules in any rule project that he has access to. The scope of the permissions for a group is defined by whether a project can be accessed by that group.

Permissions are established for a specific type of operation and a specific type of target artifact. By default, a group cannot perform any operation on any artifact. Operations must be explicitly granted through permission definitions:

- The possible operations are create, view, update, and delete.
- The possible target artifacts are the ones that are managed in the Decision Center repository, for example, action rules, decision tables, and test suites. The scope of the permission can be further refined to the level of a property for a certain type of artifact.

The permissions for a group regarding an action on a type of artifact can be absolute, for example, defined as yes or no, and thus applicable to all instances of the type of artifact. The permissions can also be relative to the value of the Group property of the artifact. When the permission is set to group, the value of the group property of the artifact is matched against the set of groups that the user belongs to. If the user belongs to the group defined by the artifact, then he is granted permission to perform the action.

This permission setting is especially useful when it is applied to folders for organizing fine-grained access to parts of a rule project. For example, if you want to limit the permission of the pricing_rule_author_nj group to create or modify rules to dedicated New Jersey folders, you can do so by completing the following tasks:

- Assigning the pricing_rule_author_nj value to the “Group” property of the New Jersey specific folders.
- Setting up a permission of type group for the pricing_rule_author_nj group for the folder artifacts.

The same type of approach can be used with Smart Folders. However, managing group permissions at the level of folders to partition the access within a project can be a difficult exercise. It is easier and more manageable to rely only on project-level permissions, and multiply the number of projects, if needed. In this example, instead of using folders to manage rules and permissions that are specific to the New Jersey pricing, a separate project could be used to hold these rules. The state-specific rules projects are then part of the set of functional rule projects that are referenced by the signature rule project, as described in 4.1, “Decision service structure” on page 38.

Tip: Managing group permissions at the level of folders to partition the access within a project can be a difficult exercise. It is easier and more manageable to rely only on project-level permissions and multiply the number of projects, if needed.

5

Release management

Chapter 4, “Organizing decision management” on page 37 described the idea of organizing important business decisions as services. It described how ITIL provides a systematic way to govern and manage services. The decision services that are described in this chapter are no exception.

This chapter contains the following topics:

- Overview
- Managing decision change through releases
- Technical Release Change Patterns
- Branching and merging to manage releases
- Release identification and versioning
- Decision release lifecycle
- Definition: Planning activities

© Copyright IBM Corp. 2013. All rights reserved.

5.1 Overview

The decision service is an IT resource that performs the following functions:

- Encapsulates a decision making capability that is important to the business. It must be driven by business policies
- Automates frequently occurring business decisions in the business solutions that it supports by using a well-bounded interface.
- Provides a vocabulary and structure that allows the line of business (LOB) to express the rules and policies that define that decision making behavior.

Changes to this decision service require the same level of release management as any other IT service that is described in this chapter.

5.1.1 Decision services

The goal of decision services is to enable the business to respond quickly to changing business requirements. This situation is achieved by minimizing the need for skilled IT resources and empowering LOB users to express the changes that they require to these decisions. By expressing the wanted business policies as rules that are interpreted in the decision service, you can introduce a much more agile business release lifecycle while you consider the essential principles and best practices features that are provided by ITIL.

The mechanism for introducing changes to services is through releases, with each release following a well-planned traceable lifecycle that does the following tasks:

- Specifies the policy changes that are required to be in the service release.
- Changes the rules that are used to determine the decision service behavior.
- Tests and validates the decision service to ensure that it meets the business goals.
- Distributes and deploys the validated decision service release into the live environment.

ITIL principles should also be applied to this decision service release lifecycle to ensure the following items:

- Releases are planned with agreement from both IT and business sponsors.
- IT and business resources are used effectively by using the appropriate level of skill and experience for the job.
- The interactions between resources and activities are coordinated and managed to ensure the integrity of each decision service release.
- It is possible to apply emergency fixes or reinstate earlier releases according to agreed procedures.
- The effectiveness of each release should be measurable by allowing the planned releases to deliver continual decision improvement.

Although the goal of decision services is to enable agile business empowered releases, there are situations where IT must be involved because of changing interface requirements or changes in the information that is needed to make a decision. In these cases, a different type of technical release is required that must go through a similar lifecycle but involves more resources and activities with a larger dependency on IT.

This chapter focuses on these two main types of release cycles:

- *Technical change management*, which results in a major release where new functionality or interfaces mean that the release is not compatible with solutions that used earlier major releases.
- *Business change management*, which should allow compatibility with earlier version and where the business changes only the way that the decision is made and thus does not require major IT rework. Within this business change management cycle, how quick fixes can be applied is also considered.

5.1.2 Change management for releases

Changes to a decision service can be managed by using a release. The changes that are made are defined through some formal change request mechanism that can be traced back to the business goals, policies, or optimizations that are required. It is important to recognize that some changes are more complex than others, so the release lifecycle must be varied to suit the type of change request that is supported.

Three different approaches are commonly encountered for changes:

- *Standard changes* are pre-authorized changes that are low risk, relatively common, or planned and follow a repeatable procedure for implementation. These changes are part of a decision optimization strategy that plans to adjust the way automated decisions are made to use the insight that is obtained from analytics and business monitoring. These standard release cycles are part of the normal business operations and must be approved and documented in the organizations quality system.
- *Emergency changes* respond to situations where a change must be implemented as quickly as possible, for example, to resolve a major incident or policy breach. These emergency release cycles must be documented in the organization's quality system as the approach to dealing with incidents and risk management.
- *Normal changes* cover other situations where the release must go through a full planning and acceptance cycle with the necessary checkpoints and approvals to ensure successful delivery. This type of change is used to cover any technical changes that require IT involvement and normally requires a project setup to define and deliver the release.

Changes are often also categorized as major, significant, or minor, depending on their level of cost and the risk that is involved, their scope, and their relationship to other changes. This categorization may be used to identify an appropriate change authority and thus the level of approval that is required for each release.

Planning releases in this way allows the scope of the change to be controlled, the risk managed, and thus allows the authority to be delegated appropriately in the approvals for that type of change.

The remainder of this section describes the following items in more detail:

- Decision service release types that support the types of changes that are described above.
- Decision service release lifecycle patterns that identify the key activities and approval points that allow the types of release to be planned and governed.
- Decision service release identification, deployment, and usage patterns that allow the configuration items that are produced from the release to be used reliably and effectively by the business solutions.

5.2 Managing decision change through releases

Decision management offers businesses the ability to quickly refine the way that their decisions are made by releasing new versions of the decision services. This section describes some common decision change patterns that occur within organizations, together with the business reasons for making those changes. The pattern is characterized by the activities that are needed to complete the release. As a result, a plan can be put forward for the frequency and timescale of each release that follow a certain pattern.

Any release goes through many phases, as shown in Figure 5-1.

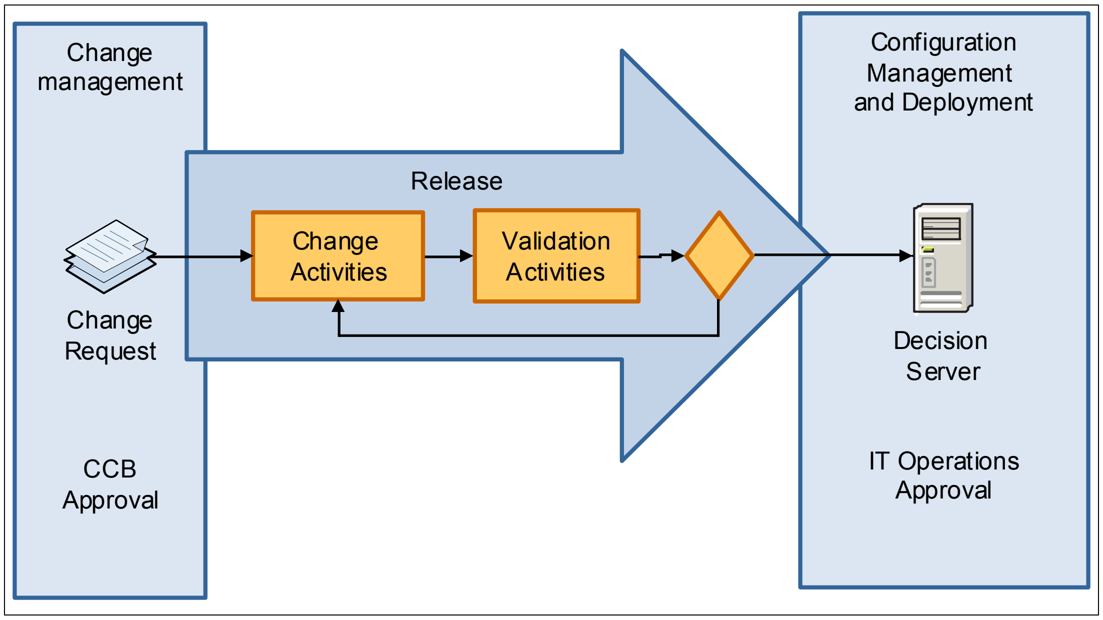


Figure 5-1 Release overview

The release starts with an approval from the Change Control Board (CCB) or other authority for a set of changes or change requests to the decision service.

The release lifecycle supports the governance of the following activities:

- Realizing those changes though Change Activities
- Validating that the requirements are met through Validation Activities.

On approval of the release, a deployable configuration item (the RuleApp archive in the context of the Operational Decision Manager platform) can be produced, which may then be deployed to the production Decision Servers according to the necessary ITIL procedures.

The rest of this chapter identifies some of the common reasons for changing a decision service and thus initiating a new release.

5.3 Business release change patterns

The need for change is often driven by various business pressures. These pressures might be imposed by external regulations or driven by the need for the business to adapt and optimize to local market conditions.

New releases might need to be produced for many reasons, each of which might have an impact on the governance activities that must be performed during the realization of the changes. Some common patterns of change are shown in the following sections.

5.3.1 Compliance change

In many financial and government applications, there is a requirement that business applications must be compliant with external regulations (for example, Sarbanes-Oxley) or internal corporate policies. This compliance can be provided by using rules within an application. When the regulations change, it becomes necessary to update the rules to ensure that the updated policies are being followed. In a compliance change, it is important to be able to trace how the modified rules support the new regulations and, when applied, to be able to prove that the correct version of the regulations was being followed. Although these changes do not occur that frequently, the accuracy with which the rules reflect the updated policies is key to ensure that compliance is achieved.

5.3.2 Decision improvement

Policy driven decision automation is often the main reason for investment in a decision management solution. In these cases, the business knows that they must adapt the way that they make these decisions to be competitive in the marketplace. Fraud detection and targeted promotions are examples where the policies must evolve over time to improve performance in the different market conditions.

Verification of changes is often difficult in these cases and must be undertaken by comparing actual behavior against expected behavior for historic data or manufactured test scenarios.

5.3.3 Decision improvement that is based on key performance indicators

In some cases, it is possible to define key performance indicators (KPIs) either for the decision itself, or for the effect of the decision on the business. These KPIs must be calculated and displayed graphically, allowing a comparison of the new rules against existing rules. This champion / challenger approach can be used in simulated scenarios or in production systems to identify which of two releases provides the better performance.

5.3.4 Decision evolution

Automated decisions are often enriched incrementally over time to cover more facets of the decision and increase the rate of straight-though processing across the different dimensions of the business. For example, a decision can initially be defined to handle the processing of a request in the most significant geographical business regions, and then augmented to support less significant regions in later releases. In the same way, the initial version of the decision can be geared toward the handling of the most popular product option. Less common options can then be introduced through successive decision changes.

These business changes might prompt a major technical change (such as a signature), as described in 5.4, “Technical Release Change Patterns” on page 54.

5.3.5 Decision policy test fixes

In any governance or change management system, there are always situations where problems are found and must be resolved quickly. Although the release governance process might require many activities to ensure that the release goals are met safely, there are times where this process is too burdensome and must be bypassed. The implication is that the release governance process for a particular decision service might have to support many different routes from requirement specification to release deployment.

5.4 Technical Release Change Patterns

Decision automation engines always require integration with the solutions that they support. In addition to managing changes because of changing business requirements, release management also must able to handle changes in the IT integration interfaces. As these changes require software development, a much longer lifecycle is expected and there is more sophisticated testing that is required before an engineering release can be deployed into a production solution. Two different forms of technical change are commonly encountered.

5.4.1 Technical changes in the decision object model

Information that is processed by the decision is passed from the solution through parameters that are based on an eXecution Object Model (XOM). This XOM is used as a basis for a Business Object Model (BOM) and defines the vocabulary on which the rules are based. Changes that are required in the XOM to accommodate changes to solution information models, or more information that is needed in the decision vocabulary implies that the interface between the solution and decision service is modified and must be validated, which might require changes to the solution and refactoring to existing release rules.

5.4.2 Technical changes in the decision operation signature

It is also important to consider how changes to the information that is available to a decision can be handled. These changes are usually based on the XOM, but even if the XOM does not change, requirements for more or changed information in these parameters can lead to incompatibility between client solutions and the decision signature. In general, changes can be considered compatible with earlier versions (only new optional parameters added) or not compatible with earlier versions (mandatory parameters are added or deleted or types of parameters are modified). When you consider technical changes, consider the level of compatibility with earlier versions that is required and how the solutions are migrated to use the new interface. In many cases, the approach might be to leave the existing interface available for the existing solution and add an operation to support the new interface. New solutions can immediately use the new interface and the existing solutions continue to use the original interface until they can be migrated.

5.5 Branching and merging to manage releases

This section introduces concepts that might be familiar to technical users but not to business users. Branching and merging are concepts that are borrowed from source code control in software development.

Leaving the main branch behind: The concept of releases that was introduced in 5.2, “Managing decision change through releases” on page 52 requires us to question the relevance of the main branch for decision management. Releases represent a unit that can be deployed as a decision service. It is possible to have multiple concurrent releases that focus on some specific decision changes that must be kept together.

Start considering each release as being its own “main branch”, which we will call the release branch. With this approach, each release is independent from other releases and multiple releases can be managed in parallel. Changes from one release can also be merged into another release if required.

If the Release branch becomes the main branch, change activities then must be managed as part of subbranches that are created off of the release branch.

Should your organization have a single release at one time, using the main branch might still make sense.

5.5.1 Branching and merging

The underlying principle for branching and merging is that you must have traceability of what specific changes were performed as part of a release or as part of a specific change activity. Which rules were changed? Which decision table values were changed? What were the changes? Who performed them?

In traditional software development, the answers to these questions are found in the source code control system (SCCS) used by the development team. Decisions are no different in that you must be able to have the answers to all those questions. In IBM Operational Decision Manager, that functionality is provided by the repository.

Knowing that one specific rule has changed is important, but you must turn away from managing individual rules and focus on change activities and releases instead. Because change activities are expected to include changes to multiple rules, and releases are expected to include multiple change activities, you need a mechanism that you can use to group these changes together so that you can look at them as a whole. This situation is where branching comes in.

Branching

To understand branching, you must understand the concept in more depth.

First, you must consider the structure of the packages and realize that the rules that are part of decision service are similar to some folders and files on a computer. The packages (folders) provide a structure for the rules (files). Creating a branch is essentially like taking a copy of the packages and rules and putting that copy in a different location so that it does not interfere with the original copy. In this case, the original copy is called the main branch or trunk.

If you were to do this action with directories and files on your computer, you could then make changes to any file within the copy without interfering with the original copy of the files.

As you make these changes to the *copy* (the branch), you start enhancing some of the original documents and improving them. These changes are not part of the main branch, but it makes sense to have the main branch benefit from these enhancements. This is where merging comes in.

Merging

The goal of merging is to have the changes from a branch merged into the main branch so that the main branch can benefit from the enhancements that were performed. Merging changes is a tricky task:

- Rules that were not changed in the branch do not need to be merged.
- Rules that did not exist in the main branch must be created.
- Rules that are deleted in the branch might need to be deleted in the main branch (you must be careful about this task).
- Rules that were changed in both the main branch (trunk) and the branch can be in conflict, which must be resolved.

Although it might be a tricky task to perform, it is important that a merge is performed appropriately so that the rules remain consistent and behave according to expectations. After a merge is complete, there is no need for the branch anymore because the change history is shown in the history of the rules of the main branch.

5.5.2 Good practices

Here are some good practices to consider:

- Create a separate branch for each release and one subbranch per change activity.
- Develop a merging strategy:
  - Regularly merge changes from the release branch into each change activity subbranch so that each change activity subbranch considers other completed change activities for conflict resolution and testing.
  - Only when everything works in the change activity subbranch should you consider merging back to the release branch.

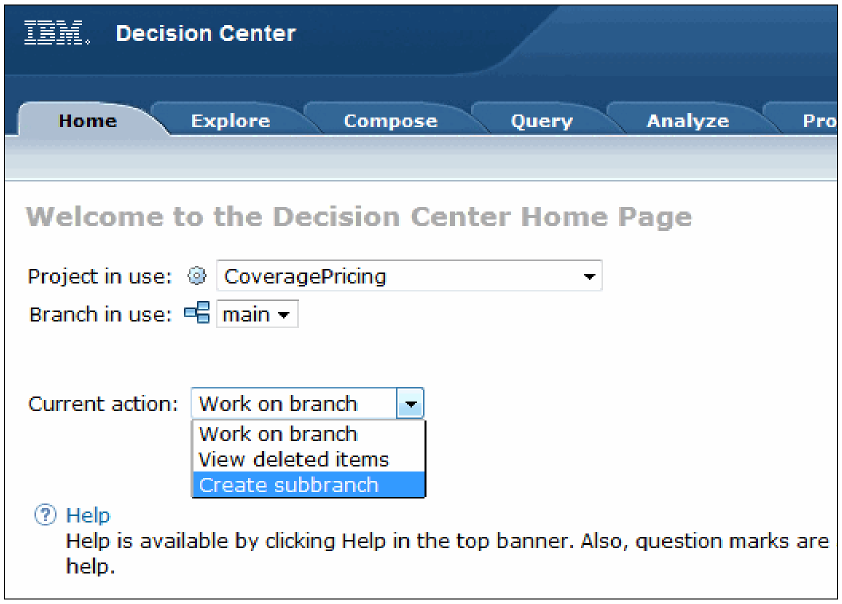
  - Only after everything is complete and working in the release branch should you merge it back in to the original branch it came from, if required. This action might not be taken if the release branch is to remain separate.
- Make sure that someone is playing the role of release manager, that is, a person that manages the releases and the merging into the main branch.

In the end, make sure that everyone understands the procedures and that there is a way to enforce them.

5.5.3 Example of branching and merging in IBM Operational Decision Manager

This section shows some simple examples of what branching and merging might look like in IBM Operational Decision Manager. All of the operations are shown in the Decision Center component of IBM Operational Decision Manager. The example also assumes that the user that is logged in to the application has the appropriate rights to perform the operations that are indicated. Complete the following steps:

1. On the home window, create a subbranch for a specific project, as shown in Figure 5-2.



Welcome to the Decision Center Home Page

Project in use: CoveragePricing

Branch in use: main

Current action: Work on branch

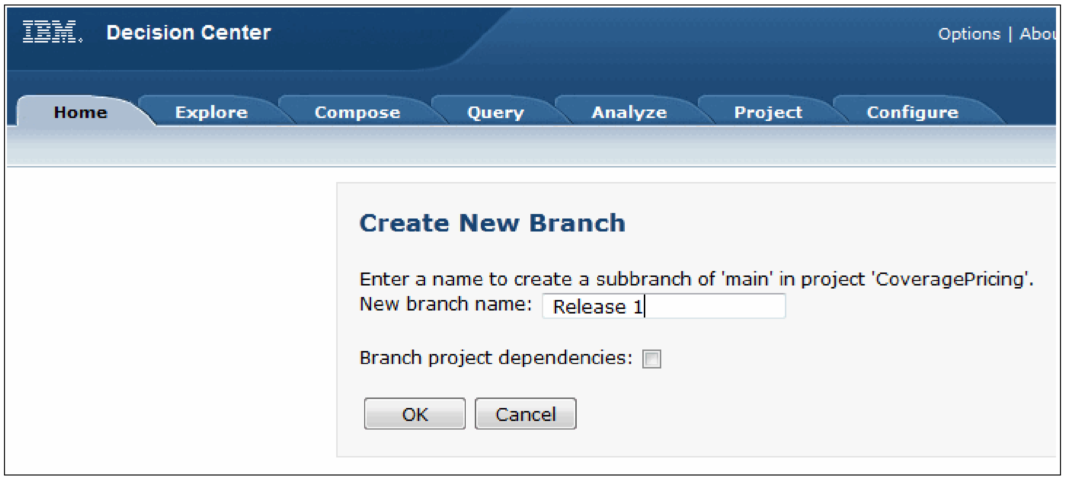
Work on branch
View deleted items
Create subbranch

Help

Help is available by clicking Help in the top banner. Also, question marks are help.

Figure 5-2 Create a subbranch

2. Specify a name for the subbranch, as shown in Figure 5-3. Click **OK**.



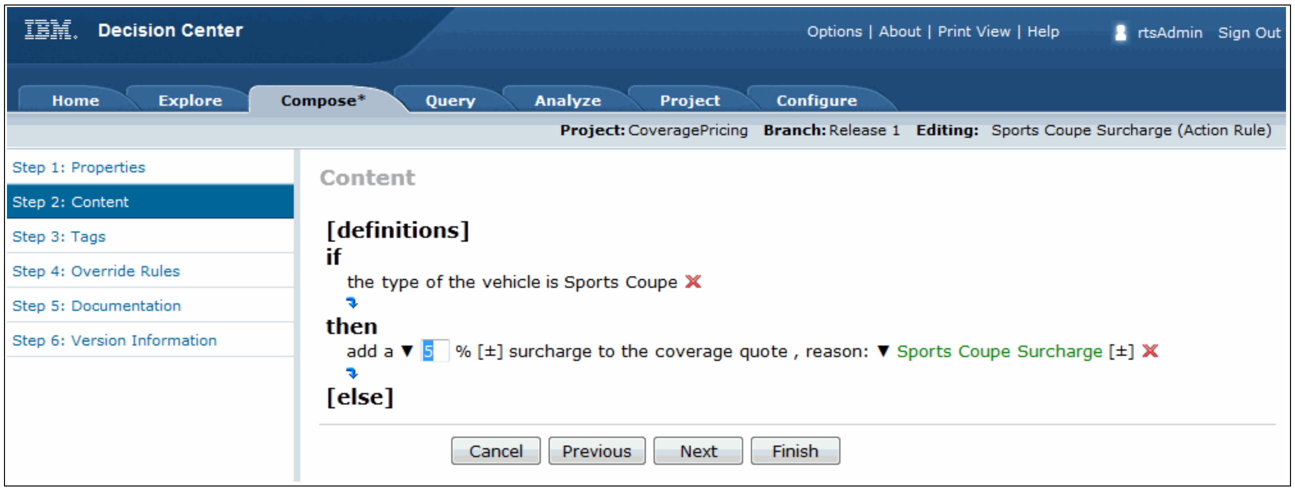
Create New Branch

Enter a name to create a subbranch of 'main' in project 'CoveragePricing'.
New branch name: Release 1

Branch project dependencies: ☐

Figure 5-3 Name a subbranch

3. Make any changes to the rules, as required. Figure 5-4 shows an example.



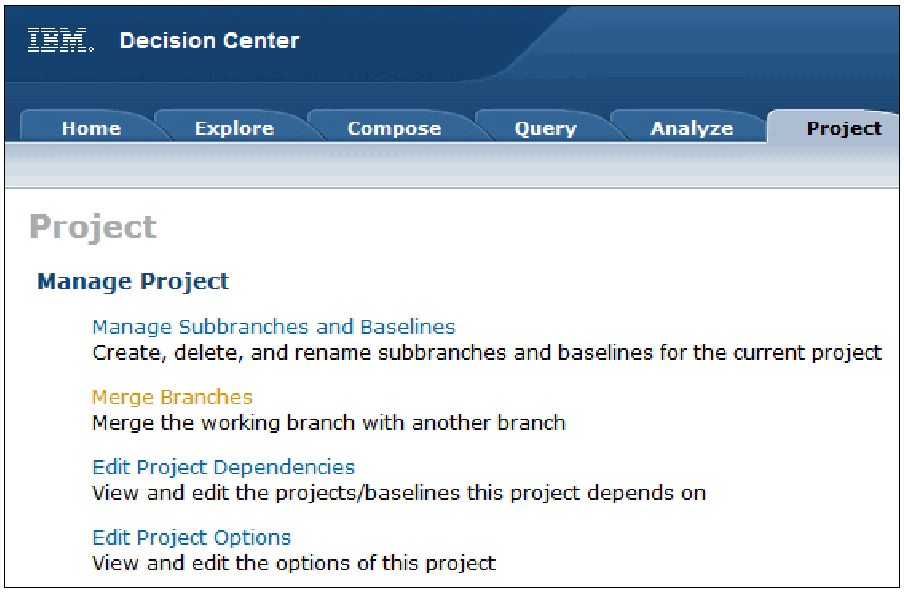
Project: CoveragePricing **Branch:** Release 1 **Editing:** Sports Coupe Surcharge (Action Rule)

Content

```
[definitions]
if
  the type of the vehicle is Sports Coupe
then
  add a ▼ 5 % [±] surcharge to the coverage quote , reason: ▼ Sports Coupe Surcharge [±]
[else]
```

Figure 5-4 Change rule definition

4. After all the changes are complete, navigate to the **Project** tab and select the **Merge Branches** menu, as shown in Figure 5-5.



Project

Manage Project

Manage Subbranches and Baselines
Create, delete, and rename subbranches and baselines for the current project

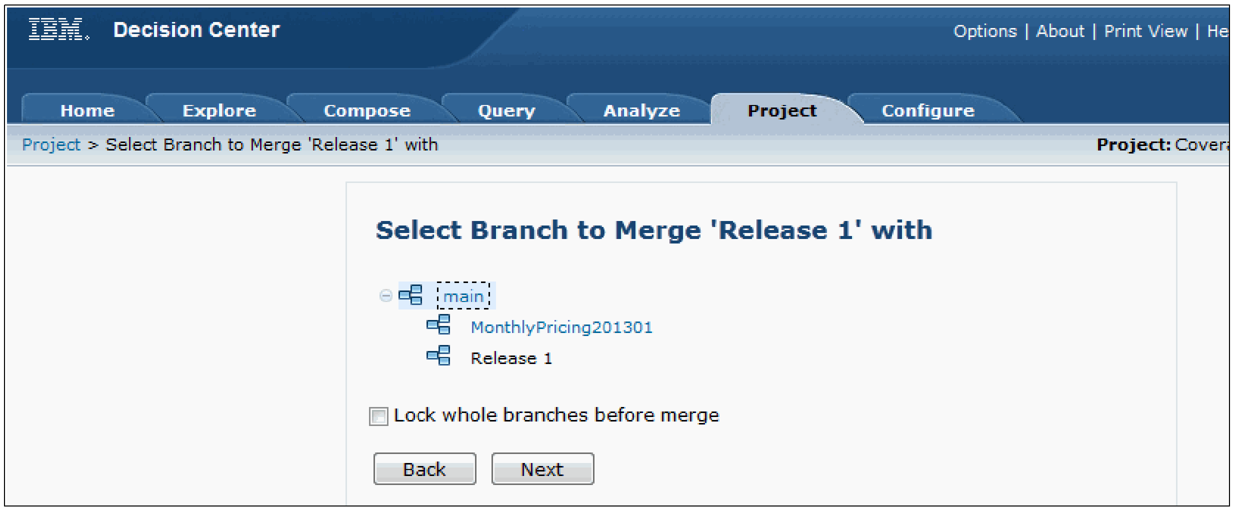
Merge Branches
Merge the working branch with another branch

Edit Project Dependencies
View and edit the projects/baselines this project depends on

Edit Project Options
View and edit the options of this project

Figure 5-5 Merge branches

5. Select which branch this subbranch should be merged with. Figure 5-6 shows an example.



Project > Select Branch to Merge 'Release 1' with

Project: Cover

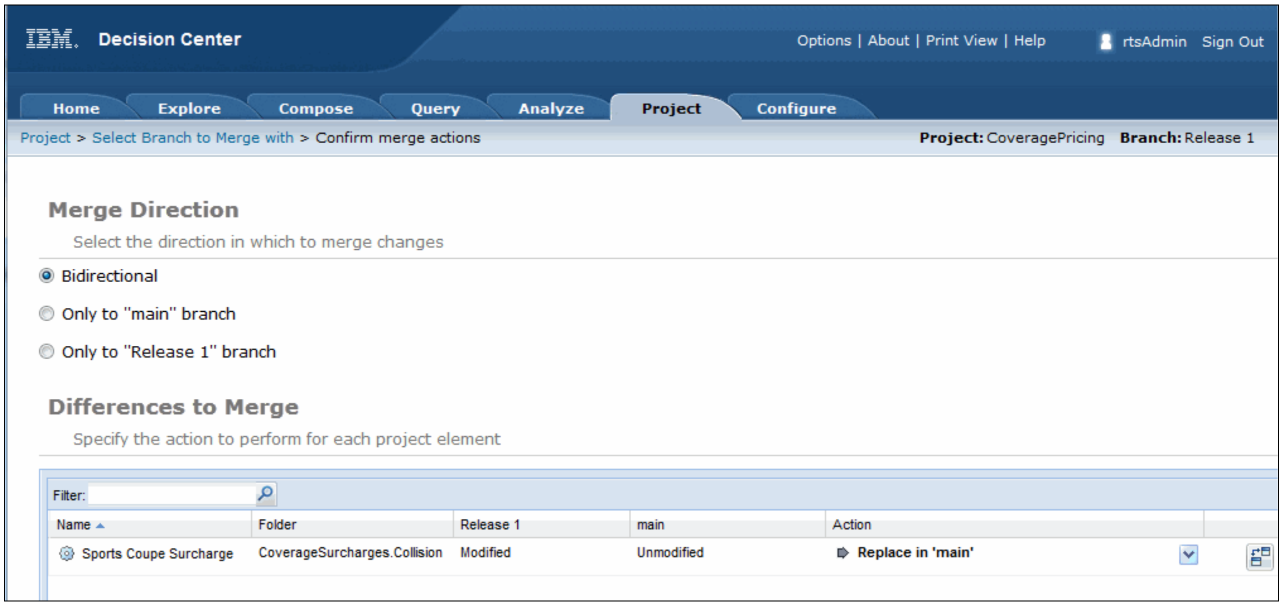
Select Branch to Merge 'Release 1' with

main
MonthlyPricing201301
Release 1

☐ Lock whole branches before merge

Figure 5-6 Select a branch to merge with

6. Depending on the nature of the changes, the next step can show you some options to simply merge. For example, Figure 5-7 shows no conflicts.


IBM. Decision Center
Options | About | Print View | Help
rtsAdmin Sign Out
Home Explore Compose Query Analyze Project Configure

Project > Select Branch to Merge with > Confirm merge actions **Project:** CoveragePricing **Branch:** Release 1

Merge Direction

Select the direction in which to merge changes

(●) Bidirectional

( ) Only to "main" branch

( ) Only to "Release 1" branch

Differences to Merge

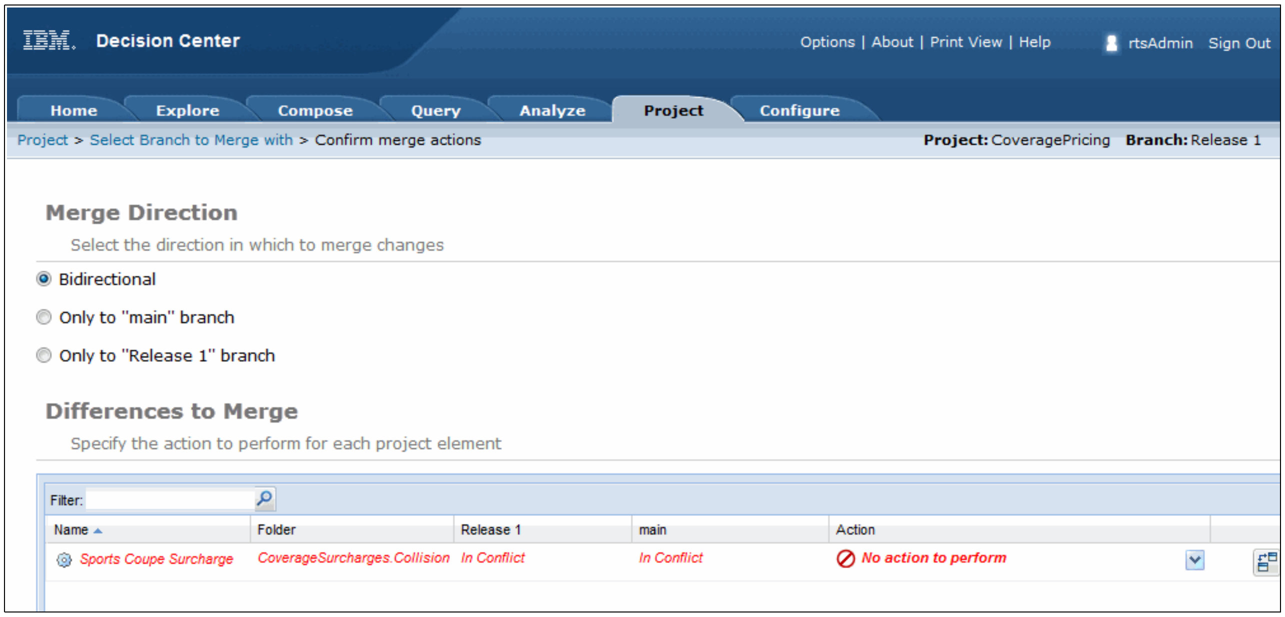
Specify the action to perform for each project element

Filter:

Name	Folder	Release 1	main	Action
Sports Coupe Surcharge	CoverageSurcharges.Collision	Modified	Unmodified	Replace in 'main'

Figure 5-7 Merge with no conflicts

7. Some changes might be in conflict with changes that were performed on the main branch and you must resolve those conflicts by selecting the specific action that you want to perform. Figure 5-8 shows an example.


IBM. Decision Center
Options | About | Print View | Help
rtsAdmin Sign Out
Home Explore Compose Query Analyze Project Configure

Project > Select Branch to Merge with > Confirm merge actions **Project:** CoveragePricing **Branch:** Release 1

Merge Direction

Select the direction in which to merge changes

(●) Bidirectional

( ) Only to "main" branch

( ) Only to "Release 1" branch

Differences to Merge

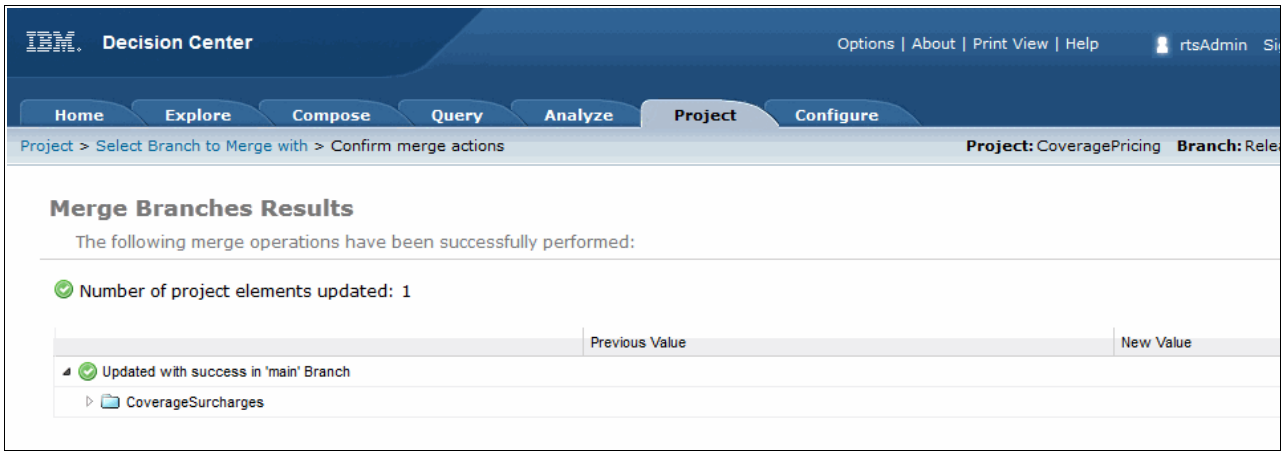
Specify the action to perform for each project element

Filter:

Name	Folder	Release 1	main	Action
Sports Coupe Surcharge	*CoverageSurcharges.Collision*	*In Conflict*	*In Conflict*	***No action to perform***

Figure 5-8 Merge with conflicts

8. After the merge operation is complete, you see a status window that shows the results of the merge. Figure 5-9 shows an example.


IBM. Decision Center
Options | About | Print View | Help
rtsAdmin
Home
Explore
Compose
Query
Analyze
Project
Configure
Project > Select Branch to Merge with > Confirm merge actions
Project: CoveragePricing Branch: Rele

Merge Branches Results

The following merge operations have been successfully performed:

Number of project elements updated: 1

	Previous Value	New Value
Updated with success in 'main' Branch		
CoverageSurcharges		

Figure 5-9 Merge results

More information about this topic can be found in thE Information Center for IBM Operational Decision Manager at `http://pic.dhe.ibm.com/infocenter/dmanager/v8r0m1/index.jsp`.

5.6 Release identification and versioning

This section describes standard, emergency, and normal release identification and versioning and release runtime versioning strategy.

5.6.1 Release branching strategy

The decision service release is a way to take account of agreed change requirements and produce a configuration artifact that can be deployed into production systems.

Section 5.1.2, “Change management for releases” on page 51 determined that releases must be planned as part of governance and identified three different types of release:

- Standard releases: Pre-planned regular releases that are performed as part of normal business operations.
- Emergency fixes: A release where an existing release must be corrected.
- Normal releases: A release that involves a project to establish the new behavior of the decision service.

Each release specifies a set of changes regarding an existing release. When you create these releases, two patterns can be used:

- *Sequential releases* represent successive changes to a decision service, where it is expected that there is only one release open at any time. This pattern is used for changes where it is expected that each successive release replaces the existing one. This situation is especially the case with standard releases or emergency fixes.
- *Concurrent releases* represent branches in the decision service behavior where after the branch occurs, the two releases can be developed in parallel until they are merged. This situation is the case with normal releases.

This section describes how these releases affect the behavior of the production decision service.

5.6.2 Standard releases

Standard releases are planned regular updates that allow an organization to plan, for example, daily releases to adapt decisions to meet a market state that is defined by analytics, and monthly releases that take into account sales results or profits. Each standard release is deployed to the production systems and replaces the current release.

Figure 5-10 shows the 1.0 release that corresponds to the Monthly Pricing Release that is described in the sample scenario.

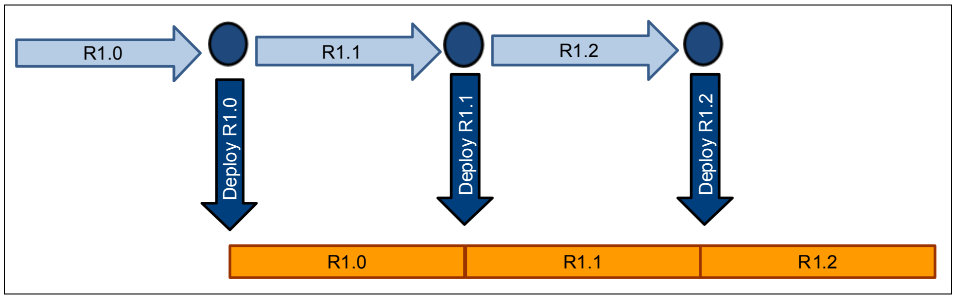


Figure 5-10 Standard release

5.6.3 Test fix or emergency releases

A test fix release quickly amends a deployed release, as shown in Figure 5-11. In this case, in addition to fixing the deployed RuleApp, the changes should be merged into the next planned release.

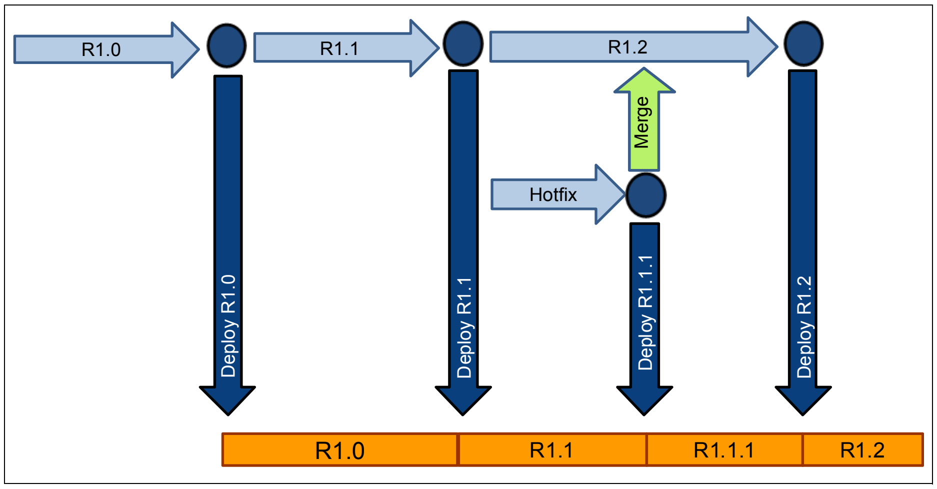


Figure 5-11 Test fix release

5.6.4 Normal releases

Normal releases might be reviewed on an annual basis and projects set up to deliver a release that takes into account new solutions services and information models.

This type of release is typical of the Roadside Assistance release that is described in the sample scenario. A normal release might introduce changes that are not compatible with the applications and solutions that use the decision service, so the release must be deployed concurrently with existing major releases. This situation means that you must be careful when you identify the releases and versioning strategy that is described in 5.6.5, "Decision release runtime versioning strategy" on page 63. Figure 5-12 shows an example.

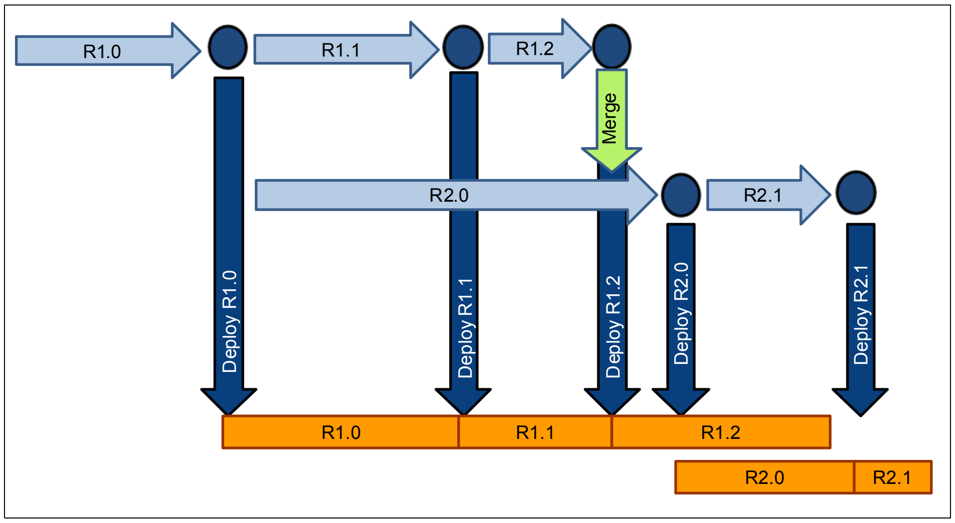


Figure 5-12 Normal release

5.6.5 Decision release runtime versioning strategy

Releases provide the means to manage changes to and thus version the decision service. Some releases may introduce technical changes to the XOM or signature that means that those releases are not compatible with an earlier version of client applications. To allow client applications to distinguish between releases at run time and ensure that they use a compatible version, a convention must be adopted for the versions.

For any specific release, a number of rule sets might be deployed. The unit of deployment is called a RuleApp, which identifies the rule sets that are created from a specific branch or release.

Each rule set that is deployed in this manner can be individually identified by client applications by using a rule set path that is broken down in the following manner:

```
RuleApp/RuleApp_major.RuleApp_minor/Ruleset/Ruleset_major.Ruleset_minor
```

Where:

- `RuleApp` identifies the particular deployment configuration or RuleApp name.
- `RuleApp_major` identifies non-compatible releases as different major numbers. The lack of compatibility might be because of BOM or XOM changes in the decision service.
- `RuleApp_minor` identifies different releases of the same deployment configuration. Although technical changes might have been made, they do not cause problems with interoperability between a client application and a release that is using an earlier minor version.
- `Ruleset` identifies the specific decision service operation or rule set that is being used.

- `Ruleset_major` identifies a major version of the rule set where incompatible rule sets might be identified as different major numbers. Signature changes are a typical reason for introducing a new rule set major version.
- `Ruleset_minor` identifies compatibility changes to a decision operation or rule set to work with an earlier version, such as changes to the extractor or rule flows.

In the sample scenario, the two major releases are identified as follows:

- Monthly Pricing - Standard Release: `Monthly_Pricing_RuleApp/1.*/Pricing/1.*`
- Roadside Assistance - Normal release: `Monthly_Pricing_RuleApp/2.*/Pricing/2.*`

When a RuleApp is deployed to a decision server, a versioning scheme is adopted that identifies whether the existing behavior is replaced or supplemented with the new behavior. Applications are designed so that they work with specific major versions but can accept the latest minor versions, which means that in general Standard and Emergency releases using minor RuleApp version increases always are picked up applications.

When a decision signature changes, a new decision operation or rule set can be developed with an incremented major version number. When a new release is deployed, this new version is available at the same time as the earlier version, allowing both decision operations to be used by their respective applications with the same release behavior.

Adopting this approach allows releases to be managed by using the described lifecycle while still maintaining interoperability with the client applications.

5.7 Decision release lifecycle

This section describes the objectives and roles of a release lifecycle.

5.7.1 Release goals

Any changes to a decision service happen in the context of a release, as described in 5.4, “Technical Release Change Patterns” on page 54.

Any change requires investment in time and resources, so most decision service release cycles are initiated from some form of change request. This request specifies the goals of the release and might be broken down into many smaller change requests that are realized through the same release.

Although these changes requests define the intention of the release, the decision release lifecycle describes the activities that are undertaken by various roles to realize those goals.

5.7.2 Activity representation

Section 5.1.2, “Change management for releases” on page 51 provided an overview of how releases could be used to govern the changes for a decision service over time. Within a release, you must decompose the change requirements into activities that allow each activity to be undertaken by resources of the appropriate skill level. The branching capabilities also can be used to segment the individual change activities, as shown in Figure 5-13.

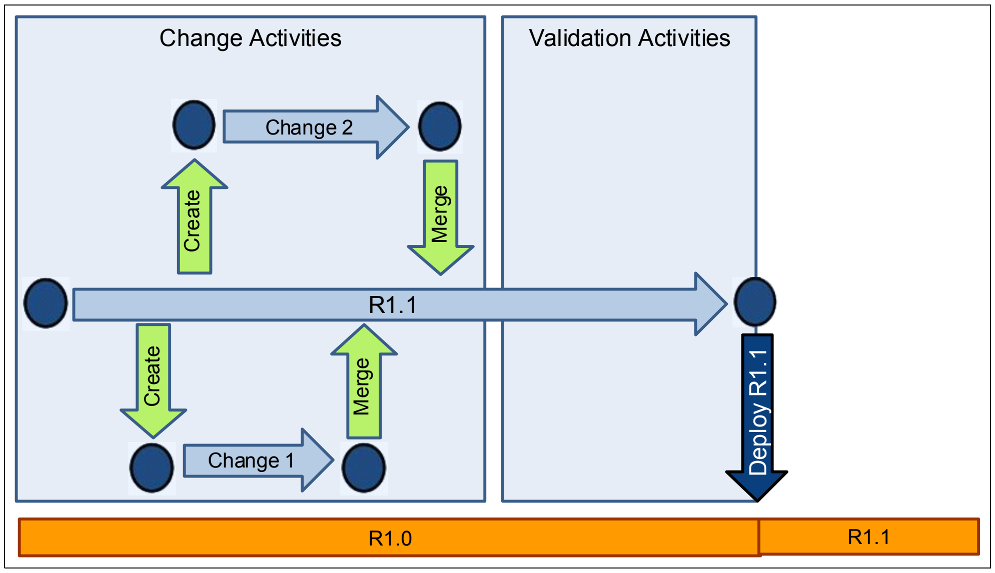


Figure 5-13 Change activities branching

To separate individual change activities and allow them to proceed in isolation, a subbranch of the release must be created. This subbranch then allows the users to undertake their changes and change specific validation activities in the context of their own branch. After the change is complete and verified, the change branch must be merged back into the release. Then, any conflicts that occur because the same rules are modified in different change activities must be resolved. When all change activities are merged into the release branch, the validation activities can be undertaken by using baselined release content.

5.7.3 Release roles

The activities that are involved in developing and deploying a decision service release are varied and require many distinct skill sets. As described in Chapter 3, “Roles and responsibilities in governing decisions” on page 25, organizations use many different roles with different perspectives for managing decisions.

From the release perspective, the roles can be grouped into the following roles:

- Release sponsor: Identifies decision changes, and prioritizes and groups them in to a release. Delegates the delivery to the release manager.
- Release manager: Responsible for delivering the release.
- Release contributor: Contributes to the delivery of the release based on their knowledge and specialty.

Table 5-1 summarizes this grouping.

Table 5-1 Release roles

Release role	**Governance role**
Release Sponsor	► Business Owner
Release Manager	► Change Control Board (CCB) ► Release Manager
Release Contributor	► Subject Matter Expert ► Business Policy Analyst ► Rule Author ► Rule Developer ► Operational Decision Manager DC Administrator ► Configuration Management Engineer ► IT Architect ► IT Integration Developer ► Operational Decision Manager RES Administrator ► IT Quality Assurance (QA)

5.7.4 Release lifecycle

Each release has different goals and involves different activities, but to apply a consistent approach, each release goes through many distinct activities and choice points for each of the release roles that were identified earlier, as shown in Figure 5-14. This release cycle is described from the perspective of a business release change pattern.

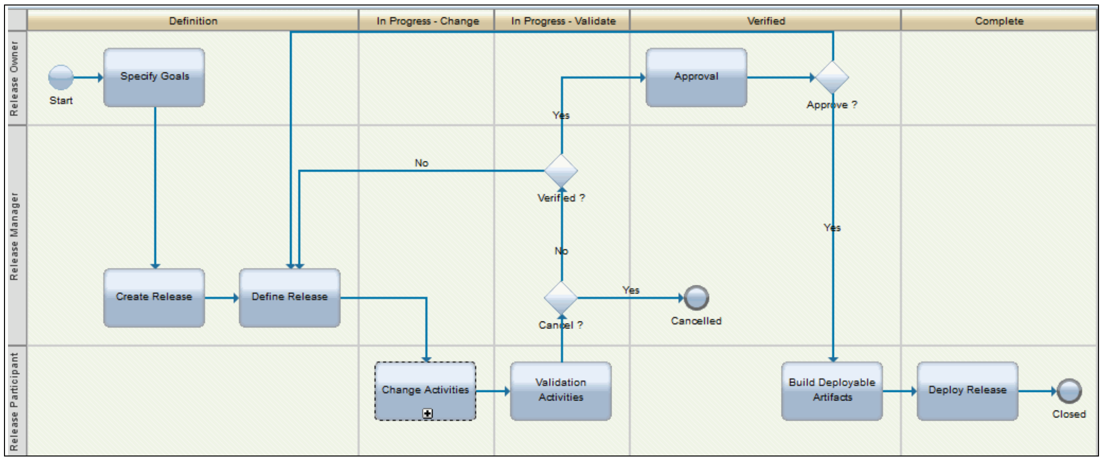


Figure 5-14 Decision release lifecycle

Definition: Planning activities

The release sponsor defines the requirements and goals for the release and the release manager creates a release and defines the various activities that are needed to realize those goals. The release manager then decomposes the release goals into subgoals and defines the activities and appropriate resources to carry out those activities. This situation allows for the effective usage of the appropriate resources within the release. The planning activities depend on the type of release:

- Standard: As the release is pre-planned, all activities and decision points should be predefined for this type of release. You can consider this release a release template, and examples of this template are provided in the sample realization. The authority to approve the release is delegated and each iteration of a standard release can be consistently managed. As this is a planned release, the emphasis becomes more on managing the activities to ensure that the planned release timelines are met, which involves the release manager allocating due dates for the activities so that he can effectively manage the release through to a timely deployment.
- Emergency: In this case, the goal is defined by the incident to be resolved: a minimal release is provided with one change activity and sufficient validation to prove that the fix worked. Each emergency fix is different and requires specific personnel to be assigned to the change and validation activities to ensure a fast and effective response is provided. Approval must reside with the identifier of the incident that needed the change.
- Normal: In this case, the definition phase involves decomposing the release requirements into the necessary activities and allocating resources with the appropriate skill sets. The order of the activities becomes important, especially when IT changes occur that imply changes to the language that is used in the rules. This situation means that the IT changes must be completed before the LOB users start their policy authoring activities.

In-Progress: Change activities

The release participants modify the rules and policies according to the change activities that are defined. Change activities are described in more detail in 6.2, “Change activities” on page 75. When all change activities are complete, the release moves on to the validate phase and no further changes are permitted to the rules and policies.

Change activities represent authoring or editing tasks that meet a specific goal or requirement of the release. The idea is to provide the means to change and approve a group of artifacts (rules or decision tables) rather than having to micro-manage each individual artifact by using its rule status. Each change activity should be performed in isolation from other changes and, when complete and approved, the artifacts are merged back into the release.

- Standard: The goal with a standard release is to empower the LOB users to make the changes that directly influence the part of the business that they are involved with. For the monthly pricing releases, this task involves letting the regional managers modify the rules that influence their regions profitability. To minimize risk, these activities must constrain what can be changed and also support users that might not be experts in the underlying rules technology.
- Emergency: For emergency fixes, the changes are identified by technical experts and are specific. These changes must be undertaken as quickly as possible by users with the appropriate (often administrative) permissions.
- Normal: In a normal release, the definition of the change activities provides the decomposition and traceability to the requirements to be included in this release. Change activities are varied and undertaken by users with a wide range of skills.

In Progress: Validation activities

The release participants undertake a set of validation activities that are appropriate to the changes that are undertaken. These validation activities act on the release as a whole and cover a range of review, simulation, and testing activities. When all validation activities are complete, the release manager must decide whether the release conforms with the validation that he set up. In effect, he determines whether he believes the goals of the release are met and documented in the various validation activity reports to allow the release sponsor to approve the release. If changes are still required, the release is replanned and another cycle of change and validation occurs. At any point in the release, requirements might change and therefore the release might be canceled.

Validation activities represent testing, simulating, or reviewing tasks that are applied to the release content as a whole. These activities normally are undertaken on a snapshot of the release. Validation activities can also be used to represent deployment and testing on System Integration Testing (SIT) and User Acceptance Testing (UAT) environments where the release is deployed to a copy of the solution and can be tested in a realistic environment.

- Standard: For a standard release, the goal is to minimize the effort that is required to validate the changes that are made. By constraining what might be changed and how those changes are undertaken, the validation activities can be focused on achieving the business goals with standard regression tests that are used to ensure that nothing else is broken. This situation provides the agility that is needed for these business-driven releases.
- Emergency: Validation of emergency fixes focuses on the validation of the key feature that is being fixed. By constraining the changes to a specific area of the rules, validation can again be minimized although the usage of standard regression testing, which should be applied to minimize the chance of new problems being introduced by the fix.
- Normal: For a normal release, the validation activities must be designed to ensure that they adequately verify the new behavior and interfaces that are being modified. This action normally requires a full set of validation activities at both the business and the technical level. After these validation activities are established and run, they can form the basis for regression testing for standard releases that are based on this release.

Verified: Approval activities

In the Verified state, the release is considered complete, together with all the documentation that shows what was changed and how it corresponds to the release goals. The release sponsor must decide whether to approve or reject the release that is based on that documentation or reject it by stating reasons. If the release is not approved, the release manager must replan the release to respond to the rejection reasons and the cycle begins again. If the release sponsor approves the release, the release is closed and deployment configuration items are produced ready for deployment into production.

- Standard: Approval is often based on the release sponsor ensuring that the process that is identified was followed, that the validation activities show that the changes are implemented correctly, and that the release can be deployed into production. In some cases, the approvals might be delegated or automatic if the validation activities were followed correctly.
- Emergency: The authorization / approval of the emergency fix is often at the deployment level rather than within the release lifecycle.
- Normal: Approval of a new normal release requires approvals from many different roles across both IT and the business. Identifying those approvers is a key part of the governance cycle.

Complete: Deployment activities

At this point in the release deployment, configuration items can be managed and deployed to the production systems according to existing ITIL practices.

Deployments may occur to non-production environments as part of the validation activities. However, the main goal of a release is to complete and validate the changes to the point where the release can be deployed to the production systems. In many organizations, this release deployment and configuration management to the production environments are governed under IT policies such as those expressed in ITIL and COBIT. Chapter 7, “Deployment” on page 89 describes these release deployment processes.

- Standard: Standard deployments that occur regularly may be undertaken by using scripts that build and deploy the RuleApps to the production systems. This action depends greatly on the IT configuration management processes that are set up in the organization.
- Emergency: The fix / emergency release often is managed through the IT configuration management system with the final stage being the deployment of the fix to the production systems according to the organization’s practices.
- Normal: In a normal release, the deployment often must be synchronized with deployment of other parts of the solution. Thus, the release forms a configuration item that gets deployed as part of a larger deployment.

6

Release activities

Decision Management systems provide the tools to author, manage, and test decision releases. These decision releases are built from more fine-grained rule artifacts (individual rules, decision tables, and so on).

Historically, this situation means that decision management was often undertaken at an individual artifact level by using a status to maintain the point in the lifecycle for each individual artifact. This approach was a good step in the right direction, but when attempting to scale this approach to organizations that had several thousand rules in multiple projects, the approach was unmanageable.

This chapter contains the following topics:

- Overview
- Change activities
- Validation activities

© Copyright IBM Corp. 2013. All rights reserved.

6.1 Overview

Managing rule artifact statuses individually is like a mason that tries to manage the status of each brick individually. It is near impossible to manage that level of detail. Instead, a mason instead manages a section of a wall as a whole and this is the approach that you must take for managing decisions.

Consider the following items:

- Stop managing individual rule artifact statuses and start managing change activities.

  The concept of activities allows a high-level and task-oriented approach to be adopted, where each activity defines changes to a related set of artifacts to achieve a particular release subgoal of requirement.

- A change activity is a task-oriented group of rule artifact changes that are aimed at achieving a specific requirement goal.

  A validation activity is an activity that is used to validate that all the release changes completed correctly, and that the release behaves as expected by giving the correct results.

- A validation activity is a task-oriented validation of a particular aspect of the release as a whole that may be used to validate a specific change in the context of the complete release.

  The combined change activities and validation activities are the building blocks of a release.

- A release is a set of change requests that are tracked as change activities that are validated by validation activities, which results in a decision service configuration item that can be deployed into the production systems.

  This chapter starts detailing some of the information that you need for using change activities and validation activities.

Figure 6-1 on page 73 provides a good over view of the release activities. It shows the following items:

- A request for change is created and submitted for review to the change control board (CCB).
- The CCB either approves the request and assigns it to a specific release or does not (it may be postponed to a future release, for example).
- The change requests are assigned specific change activities, some of which can be grouped into a single change activity.
- After all the change activities for a release are completed, the required validation activities for that release are performed.

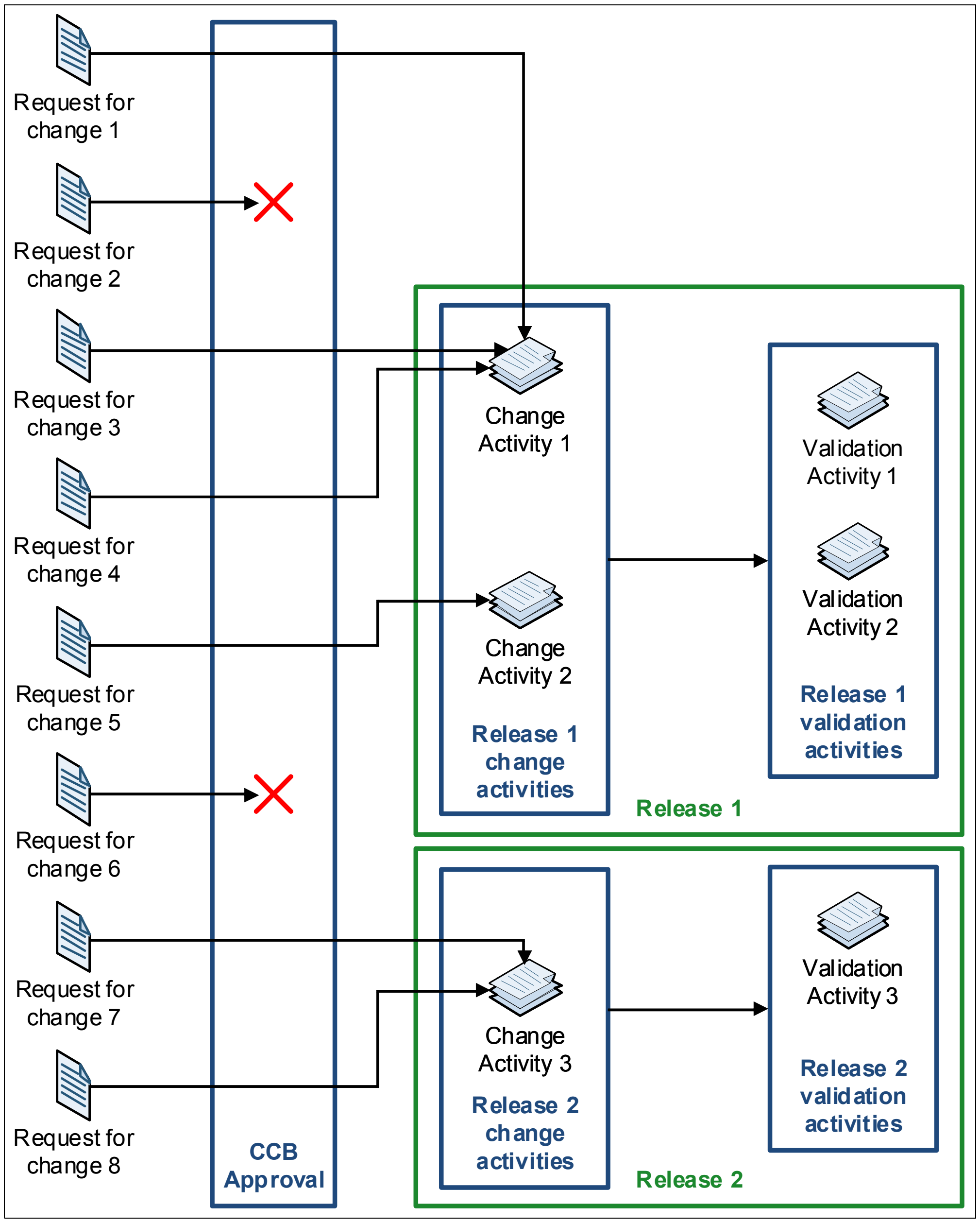


Figure 6-1 Release activities

6.1.1 Activity classification

The first step that is required to understand activities (both change and validation) is that there are different types of activities that you must consider. To help put some structure around the activities, we provide some examples of the activities and classify them.

Here are the classification of the activities:

- Release Class: Defines whether this activity usually implies that the associated release is considered a business release or a technical release (requiring IT involvement). A technical release usually refers to a release that requires changes that can affect the structure of the decisions or the interfaces that are used with the external systems. A business release does not usually touch on those structural parts of the decisions (it does not mean that IT is not required to support some more advanced functionality to support the business release).
- Activity Class: This item defines a name for the particular type of activity.
- Description: Provides an overview of the goal of the activity class and thus the types of tasks that are undertaken. In addition to listing the different activities and description, you can also show the activities and the expected business maturity of the people that perform the activity.
- Business Maturity: Provides an indicator of the level of business expertise that is required to undertake this activity class. See Table 6-1.

Table 6-1 Business maturity

Business maturity	Description
High	Implies that the activity requires contributions from users with an excellent understanding of the business and able to understand the implications of changing a rule on the overall business solution.
Medium	Implies that the activity requires an understanding of the implications of a rule on the decision that is being made. This understanding allows the user to safely modify rules that are tied to a specific business context.
Low	Implies that although a basic understanding of the business is required, tasks can be performed without that much risk on the business solution.

- Platform Maturity: Provides an indicator of the level of technical expertise in rules and the decision manager platform that is required to undertake the activity. See Table 6-2.

Table 6-2 Platform maturity

Platform maturity	Description
High	The contributor has a detailed understanding of all aspects of rule development and management and is able to use Rule Designer and Decision Center to perform sophisticated tasks such as integration, vocabulary design, and rule flow design.
Medium	The contributor has a good understanding of the rule language and is able to create and edit rules by using Decision Center.
Low	The contributor has a basic understanding of rules, but normally prefers to work with templates and decision tables rather than write by using rule statements.

6.1.2 Activity realization

Just as the decision service release can be realized by a branch, the same can be true of a change activity where each change activity is a subbranch of the release. This situation allows each change activity to proceed independently and, when complete, changes are merged back into the release.

6.2 Change activities

Change activities are part of a release. Each change activity is focused around meeting a particular goal and can occur in isolation within the release it is part of.

You should classify the different change activities that are possible so that you can better understand the impact this change activity might have and the risk that is introduced into the release by carrying it out. The next section shows how the validation activities can be selected to mitigate the risks that are introduced by the change activities.

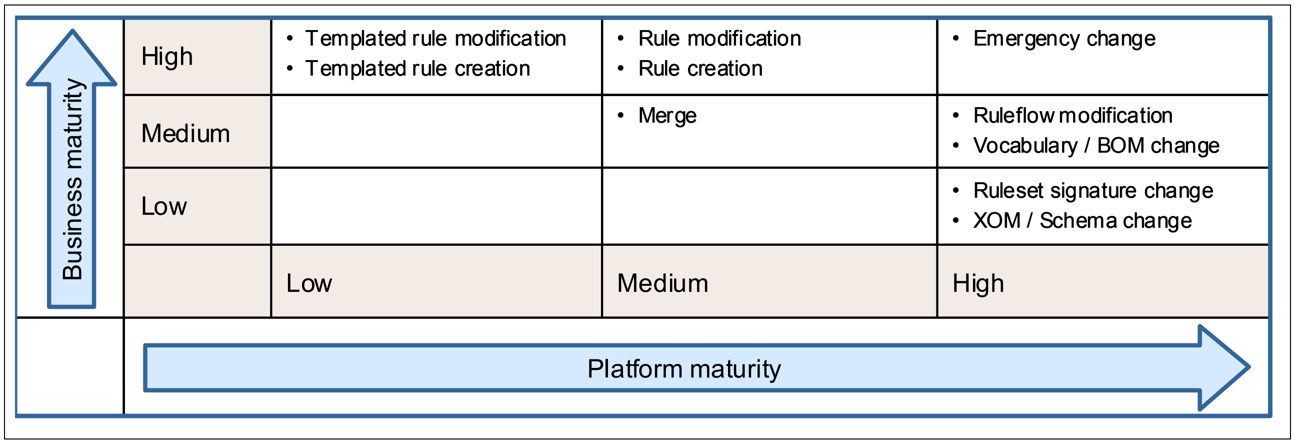
6.2.1 Change activity classification

The list of change activities that are described in Table 6-3 shows the most common types of change activities that can be required for filling the change requests.

Table 6-3 Change activity classifications

Release type	Change class	Description
Business	Templated Rule Modification	Changes involve only replacing values in rules. Values are constrained through the usage of templates.
Business	Templated Rule Creation	New rules are created from templates only. Rows to decision tables can be added. There is no changing of decision table columns or free format rule writing.
Business	Rule Modification	There is modification of rules and decision tables by using the full capability of the vocabulary. The decision table structure is changed by adding columns.
Business	Rule Creation	New rules and decision tables are created from scratch and inserted into the rule flow packages.
Business	Rule flow Modification	The rule flow and package structure are modified. New packages are created.
Business	Merge Activity	Rules are merged in from another completed release. A good understanding about both the business goals of the merged and target rules and a good understanding of rule technology is needed to resolve any conflicts.
Business	Emergency	A specific change is applied to fix a problem. This activity relies on high release participant skill levels.
Technical	Vocabulary / BOM changes	Verbalization changes require rule refactoring. There are changes to rule variables.
Technical	Rule set Signature Change	Input or output information is changed. Signatures might not be compatible with an earlier version.
Technical	XOM / Schema Change	Underlying information model is changed. Requires the refactoring of vocabulary and BOM. Might imply rule set signatures change types and are not compatible with an earlier version.

The change activities that are shown in Figure 6-2 shows the required level of business knowledge maturity and platform knowledge maturity that is required to support these change activities.



Business maturity				
	High	• Templated rule modification • Templated rule creation	• Rule modification • Rule creation	• Emergency change
	Medium		• Merge	• Ruleflow modification • Vocabulary / BOM change
	Low			• Ruleset signature change • XOM / Schema change
		Low	Medium	High
		Platform maturity		

Figure 6-2 Business change activities

In Figure 6-2, the specific distribution of the change activities can be changed to meet your own specific needs. What is important is that you must ensure that the people that are performing the changes have the appropriate level of business and platform maturity to perform the change activities in your organization.

6.2.2 Change Activity Lifecycle Overview

Chapter 5, “Release management ” on page 49 described how the change requirements for a release were decomposed into individual activities that could be undertaken independently. In order for this task to work effectively, the change activity must itself have some structure or lifecycle to allow it to be managed.

The proposed lifecycle for the change activity is shown in Figure 6-3.

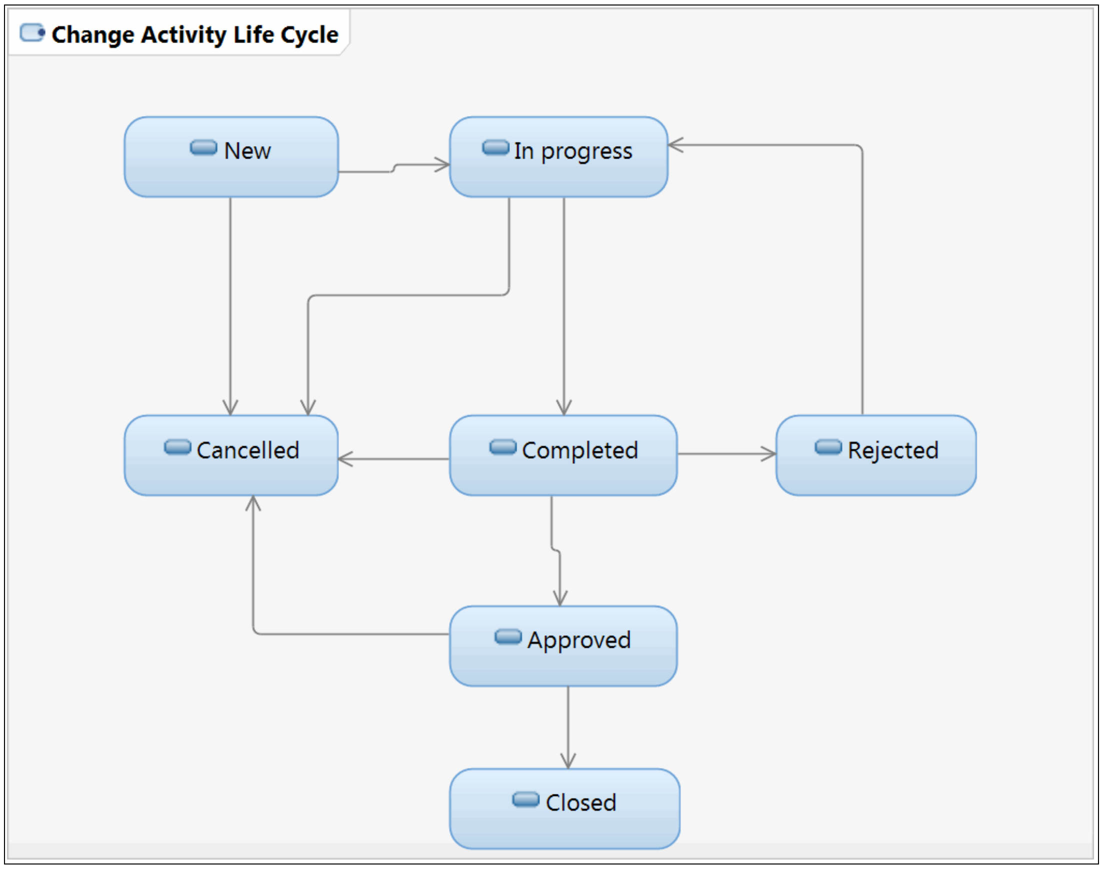


Figure 6-3 Change activity lifecycle

Every change activity starts in a status of *New*. From this status, after someone starts working at implementing the required changes within the rules, the change activity should go to the *In progress* status. After the work on a change activity is finished, the status should be changed to *Completed*. You may undertake some validation activities as part of a change activity, especially simulation and test suites. In this case, the validation applies to the single change itself. Based on this validation, the change activity might be *Approved* or *Rejected*. The final status for the change activity is when everything is complete. In most cases, if there are no conflicts, the change may be merged back into the release branch and it can now be considered *Closed*. If, however, there are conflicts, it might be necessary to bring the release differences back into the change branch and resolve the conflicts before merging back into the release. At any time during the lifecycle, a change activity can be *Cancelled*.

6.2.3 Granularity of a change activity

Chapter 1, “Introduction” on page 1 described the fact that the *rule lifecycle* was too fine-grained a level to be of use because a change probably involves multiple rule changes and that tracking each rule status individually is counter productive.

The same principle applies to the level of granularity of a change activity. If a change activity makes updates to a single rule, then it is possible that using the change activity might be overkill. It might make more sense to group a few simple change requests into a single change activity so that the impact of a change activity does not become problematic.

Figure 6-1 on page 73 represents this situation by showing multiple requests for changes that are combined into a single change activity.

6.2.4 Change activities in IBM Operational Decision Manager

Now that you know what a change activity consists of, you can consider how IBM Operational Decision Manager supports the concept of a change activity. In short, and similar to releases, a change activity can be handled through the branching mechanism that is offered by IBM Operational Decision Manager. See Figure 6-4.

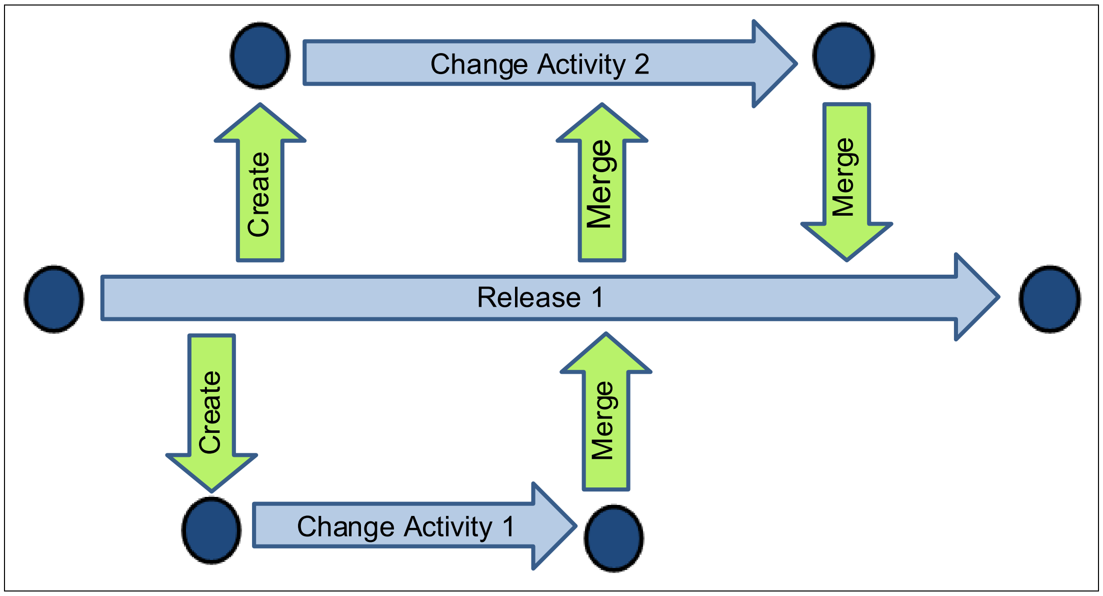


Figure 6-4 Branches for change activities

As Figure 6-4 shows, after the Release 1 branch is created, you create subbranches to keep the changes associated with a change activity together. If a specific release is composed of multiple change activities, these activities each have their own subbranch. After the work on a specific change activity is completed, it can be merged into the release branch in preparation for the validation activities.

Figure 6-4 shows that change activity 1 is merged back into the release branch and also shows that the change activity 2 subbranch must merge the resulting changes back in to its subbranch to be able to handle conflicts as early as possible within the subbranch. After the work on the change activity 2 is complete, the subbranch can then itself be merged back into the release 1 branch.

Important: You should identify the strategy to use in your organization should this situation occur so that you are prepared for it when the time comes.

6.3 Validation activities

Validation activities are the activities that are required for quality assurance of a release. The change activities introduce new rules or modify the behavior of the decision, and validation activities provide the confidence that these changes are implemented correctly to meet the new requirements for the decision service. Validation activities are undertaken against the release when all the change activities are complete for that release.

The underlying idea is that catching an issue with a decision costs less to fix before it goes to production than after it goes to production.

The changes that are performed as part of the change activities of that release drive the selection of the validation activities. For example, if a release includes only simple changes of a value within rules or decision tables, you do not require the same amount and type of testing as a release that includes a new set of rules that require new data elements to implement, thereby changing the interfaces to external systems.

Note: The type of changes that were performed as part of the change activities of that release drive the selection of the validation activities.

The list of validation activities in Table 6-4 provide an example of validation activities that could be used as part of a release.

Table 6-4 Validation classes

Release type	Validation class	Description
Business	Peer Review	Confidence in the changes is obtained by having another person review the rules and policies that were changed.
Business	User Acceptance Test	The release is tested in an end to end environment that exercises the release against the business goals. The result from this testing is confirmation of the business behavior from the business sponsor.
Business	Smart Query Based Analysis	Rule metadata is used to query the rules that are used within the decision service against specific criteria. This situation can be status-related or specific to organization requirements, for example, identifying all the changed rules that implement Sarbanes Oxley. This activity might also include actions that change rule metadata, but it should not change rule content.
Business	Simulation	Rule sets are simulated against a predefined set of scenarios to ensure that business goals are met. Scenarios are usually measured with KPIs, which allows a quantitative assessment of the decision and thus allows decision improvement to take place. The number of scenarios are usually greater than in testing.
Business	Historic Simulation	Rule sets are run against historic data from a decision warehouse. This situation provides the means to test the new release against real historic data and to compare the results after the fact.

Release type	Validation class	Description
Business	Champion / Challenger in production	Champion / Challenger provides the means to perform a direct comparison between the release under development (Challenger) and a release already in production. It may sometimes be considered by deploying both releases to production and directing a certain percentage of requests to one option or the other from within the rule flows.
Business	Audit	Before closing the release, reports might need to be produced that ensure that all changes are clearly documented for the release and traced back to requirements.
Business and Technical	Configuration Management	The commit to production often requires the creation of a configuration item (CI) in the Change Management system. This activity ensures that the relationship between the release and the change management system is maintained.
Technical	Unit Test	Rules are tested by using development tools in Rule Designer.
Technical	Test Suite	Rule sets are tested against predefined data sets with prescribed outcomes. There is coverage of the rules, regression, and performance.
Technical	System Integration Test	The release is tested in an end to end environment to ensure that the rule set signatures and XOM that are used work with the client solutions for which the release is intended. This activity also includes ensuring that the release operates correctly over the bounds of operation of the solution.
Technical	Rule Analysis	Rule analysis provides the means to check the decision service rules for completeness and consistency. This activity highlights overlap / under-lap and problems within the rules.

Similar to the change activities, validation activities can be shown in a spectrum that shows the required business maturity and platform maturity. See Figure 6-5.

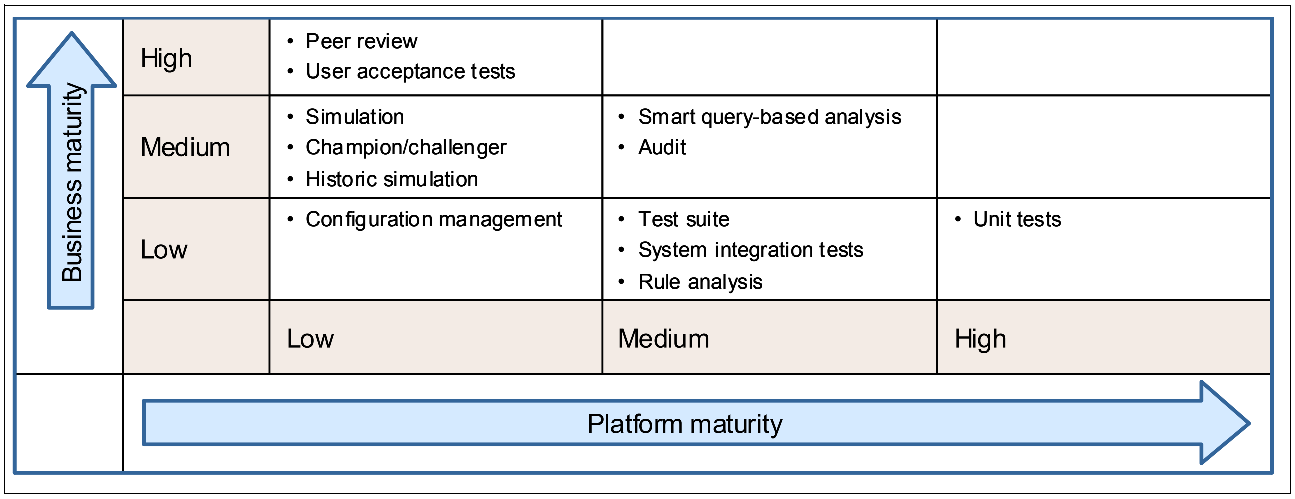


Figure 6-5 Validation activities

Table 6-5 shows a potential match between the change activities in a release and the validation activities that might be required for that release. Selecting the appropriate validation activities is based on the amount of risk that is associated with the change activities in a release. The riskier the changes that are included, the more the validation activities require time and resources.

Table 6-5 Change activities and validation activities

Change activity	Validation activity
Templated Rule Modification	► Peer review
Templated Rule Creation	► Peer review ► User acceptance tests
Rule Modification	► Peer review ► User acceptance tests
Rule Creation	► Peer review ► Rule analysis ► User acceptance test
Merge Activity	► Peer review ► Smart query-based analysis ► Rule analysis ► Audit ► User acceptance tests
Rule flow Modification	► Peer review ► Rule analysis ► Test suite ► Simulations ► User acceptance tests
Emergency	► Peer review ► Rule analysis ► Test suite ► Audit ► User acceptance tests
Vocabulary / BOM changes	► Peer review ► Rule analysis ► Test suite ► Audit ► User acceptance tests
Rule set signature change	► Peer review ► Rule analysis ► Unit Tests ► Test suite ► Audit ► System integration tests ► User acceptance tests
XOM / Schema Change	► Peer review ► Rule analysis ► Unit Tests ► Test suite ► Audit ► System integration tests ► User acceptance tests

Table 6-5 on page 81 does not list some of the validation activities:

- Champion-Challenger in production
- Historic simulation

These approaches to validation are more involved and time-consuming and usually happen when you are making substantial changes to your approach in the rules and must perform some impact analysis before you let the release go to production.

6.3.1 Peer review

Peer review is one of the easiest validation techniques to implement. A peer review consists of having a peer of the rule author or rule developer perform a manual review and inspection of the changed rules or new rules that are implemented as part of the change activity. Typically, this person is a person of similar or more skill and experience so that the reviewer has enough knowledge and understanding of the business decisions and the platform to be able to perform a useful review of the changes.

The peer reviewer should be looking for, for example, the following things:

- Following authoring standards for naming rules, variables, and so on.
- Reviewing the rules to make sure that they are correct, complete, and do not create conflicts.
- That the documentation and other associated properties are completed appropriately.
- That the changes line up with the change request.

In IBM Operational Decision Manager, some of the tools that available to users can help support this activity. For example, in the repository, a history of the changes can let a reviewer look at the evolution of the changes of a rule artifact. Queries can also be used to identify some of the associated artifacts that might be of interest.

6.3.2 User Acceptance Testing

User Acceptance Testing (UAT) are tests that are performed by business users that are not involved in the actual implementation of the decisions. It is performed in an *end to end* testing environment where the testers are using their usual working tools to submit requests through the decision management system.

The testers still must create a list of formal tests that test different scenarios that confirm that the behavior of the decisions is performing as expected. The tests must emulate real world scenarios and usage and can also include some limit cases as well. Because users are expected to perform the tests (or possibly be supported by some automated tools), the overall number of tests for the UAT is usually smaller than in some of the simulation approaches described above.

UAT is usually one of the final stages of approval and verification before the deployment of a change into the production environment. After the software works as expected with no issues, it is reasonable to think that the decision changes can be deployed to production.

6.3.3 Smart query-based analysis

Smart query-based analysis is an extension of the manual analysis of the changes that use queries of the rule content and metadata to identify where information is used in the decision making process. The queries can be used to perform a semantic analysis as a validation activity for checking for completeness across the rules.

Semantic queries are a different way of looking across the whole rule project to understand where information may be used when you make a decision. It requires a higher level of skill and maturity in both business and technical dimensions than a simple peer review.

In IBM Operational Decision Manager, queries can be saved and presented to users in the form of smart folders that let them quickly rerun specific queries of interest.

6.3.4 Concept: Champion-Challenger

Almost everyone has seen this concept in a sports-related situation. The reigning champion (individual or team) is opposed to a challenger to decide who really is the best in this specific sport. The one with the best results (or the win) at the end of the match is then declared champion.

In decision management, champion and challenger have specific meanings:

- Champion: The champion corresponds to the most effective decisions, usually the existing production decisions.
- Challenger: The challenger is compared against the champion. The decisions that generate the best results then become the new champion.

In IBM Operational Decision Manager, there are many ways to approach the concept of champion-challenger:

- Simulations and historic simulations
- Champion-Challenger in production

6.3.5 Simulation and historic simulation

Simulations are a means of performing a champion-challenger comparison of the decisions without deploying both sets of rules to production. There are two important prerequisites that enable simulations:

- Access to a stable source of data. To perform the tests, you need a set of stable and predefined test data that you can run repeatedly while the simulations are taking place. The number of scenarios that are required depend on the specific situation, but it is usually large.
- A list of key performance indicators (KPIs). The KPIs must be measurable metrics that have business value and that can be used to judge the effectiveness of the decisions that are being simulated. For example, for a financial institution that is working on decisions around mortgages, the KPI might be the mortgage application rejection rate (percentage of mortgage applications that are declined). Multiple KPIs might be necessary to obtain enough information to make an informed decision about the quality of the decisions that is being evaluated.

After these two prerequisites are present, it is possible to run the data set through the existing set of decisions and to obtain the benchmark value for the KPIs. Changes to decisions can then be implemented, and you can then run the same set of data through the new set of decisions to obtain the associated KPIs. You are then in a position to compare the results of the existing decisions and the new decisions and to make an informed decision about the next steps to take.

In IBM Operational Decision Manager, simulations require the help of developers to implement the KPI calculations that must take place when you run a simulation.

Historic simulations are a variation of the simulation described above. In this case, the source of data is historic data that is available from a data warehouse. In this specific situation, the data is still relatively stable, but you might have the ability to tweak the queries that identify the subset of data to be used in the simulation. For example, the set of dates to use would be the same for a specific set of simulations, but might be slightly different when the next set of simulations takes place two months later, for example. In any case, it is assumed that the data that is being used is not being modified between simulations.

Tip: Simulations and historic simulations are really a *non-production* champion-challenger scenario.

A drawback of the implementation of the champion-challenger concept through simulations is that the KPIs must be identified before the running of the simulation, which is implemented by a developer, and then ran. This process might be sufficient in some cases, but if the KPIs were not properly identified before the execution, the benefits might be limited because a new or updated KPI requires more development and a rerun of the simulation.

One main advantage of this approach is that because the simulated rules are not in production, the production data is not affected by these simulations, and the challenger rules can be adjusted and modified until confidence in those rules is sufficient.

6.3.6 Champion-Challenger in production

As described above, simulations are really a champion-challenger concept that is applied to a subset of data elements and where you measure KPIs from the champion and KPIs from the challenger.

The simulation KPIs can provide some indication of the results you might get, but to take this one step further, you might decide to compare the two sets of decisions in a production environment. There are requirements for this situation to be possible:

- A means to identify which subset of rules was run. A new piece of information must be returned with the results so that you can identify which set of rules was run, that is, the champion rules or the challenger rules.
- A means to perform post-execution analysis of the results. Because this process is not a simulation anymore but works with real production data, you must have a means to perform post-execution analysis to compare the results after the execution completes. This process may be performed by a Business Intelligence (BI) tool.
- Introduction of a controller. Modifications to existing decisions (rule flows) to allow a percentage of the requests to be directed to the challenger decisions through a controller. In IBM Operational Decision Manager, this controller can take many forms, from a rule set interceptor to a simple rule flow decision. In either case, there is usually a percentage of randomly selected cases to be processed through the challenger decisions.

- Confidence that the challenger decisions are good enough for production. Because you are dealing with production data, there might be implications to consider before you implement the challenger decisions in production.

Confidence in the challenger decisions: Before you deploy the challenger decisions to production, you might want to consider the legal or financial implications of incorrect or bad decisions. Having the confidence that the challenger decisions are good enough for production before you deploy them is one of the biggest challenges of this approach and is also why simulations exist.

At the beginning, the percentage of requests that is routed to the challenger decisions might be small; over time, this number can be increased.

This approach might require more work to set up and support because the number of decisions to maintain is, at least temporarily, doubled to support the implementation of the challenger decisions. Although the costs might be higher, this approach might be valuable in some situations.

In IBM Operational Decision Manager, this approach could be implemented through a change in a rule flow that diverts a portion of the requests through the challenger rule flow or through a rule set interceptor.

The rule flow implementation provides an easy way for business users to update the selection criteria over time. For example, after two weeks of 5% randomly selected cases to run through the challenger decisions and after some analysis, this percentage could be updated to 10% or more.

In short, and as you might have realized by now, each approach has its advantages and you must consider your options and the implementation so that your specific requirements are met.

6.3.7 Audit

Audits and audit reports might need to be provided as official documents before you close a release. The content and structure of the reports and how they must be built or generated vary by organization. After you identify your specific requirements around audits and audit reports, you can use much f the functionality that is offered by IBM Operational Decision Manager to support these requirements.

IBM Operational Decision Manager offers the following abilities:

- Ability to build custom queries to look for specific information
- Ability to build custom reports
- History of changes in the repository
- Ability to generate the documentation from the existing rules
- Ability to export rules to Microsoft Office documents

6.3.8 Configuration management

This validation activity is listed here mostly so that business people understand that changes to business decisions are not necessarily autonomous and to make sure that the IT change management system is made aware of business decision change, which is important. You should assume that the IT change management system is used to some degree by the business people to have traceability of the request for changes in the decisions, a history, and a link to the actual configuration items that are associated with the deployment of the changes into production.

6.3.9 Unit test

The unit tests validation activity is one of the more technical options for testing some rules. A rule developer can create a specific test case by using some of the technical tools that are available to him and run the rules, adding breakpoints and tracing the execution step by step, and so on. Typically, this testing is used when a defect is discovered and it is impossible to identify the source of the issue based on business testing. This testing possibly identifies some bug in the underlying code or something else that was unexpected and that must be handled differently.

In IBM Operational Decision Manager, this testing is performed by using the Rule Designer developer tool.

6.3.10 Test suite

Test suites are a set of predefined test cases, including input data and expected outcomes that allow rules to be tested. This testing can be used to perform the following testing:

- Coverage testing: This testing uses multiple scenarios to ensure that most (if not all) of the possibilities are tested, although their occurrence might not be as common as others.
- Regression testing: The existing test scenarios can be rerun to make sure that the latest update to the rules has not broken something else in the system without anyone noticing.
- Performance testing: The test suites can help you make sure that the execution does not encounter a performance bottleneck or that if a bottleneck is identified, it can be investigated further to try and resolve any issue that might exist.

In IBM Operational Decision Manager, test suites are implemented by using the Decision Validation Service (DVS). Technical users are usually required for creating the DVS templates that are used during testing, but business users can easily be shown how to complete these templates so that they can also create the type and the number of test cases that they are expecting for the decision service.

6.3.11 System integration tests

System Integration Testing (SIT) does not concern itself with the results of the execution of the decisions. SIT is more concerned with the ability of the decision services to *work* with the external connection point it has to the outside world. Here are some examples:

- Making sure that the client applications can call the decision service by using the agreed upon interface and get an expected response.
- Making sure that other systems that are involved (for example, databases) are working appropriately to provide the information that is expected by the decision service.

SIT rarely involves business people because it is not concerned with the actual decisions and rules.

6.3.12 Rule analysis

Rule analysis is a manual analysis of the rules by using some of the tools that are available to support it. The rule analysis that is performed as a validation activity is similar to rule analysis that should be performed at the discovery and analysis time (before implementation) but is focused on the rules and decisions that are actually implemented in the system.

For example, here are some items you should look for:

- Rules that are atomic. A rule cannot be decomposed further without losing meaning.
- Removing redundant rules or rules that overlap with other rules.
- Ensuring consistency across the rules.

IBM Operational Decision Manager provides the following tools to help you with this analysis:

- Rule analysis is a feature for checking that your rule project contains consistent rules with no conflict or redundancy. Using the rule analysis feature, you can analyze your rule project to check that it contains no ambiguities or conflicts, and that no rules are missing. If your project was modified, you can use rule analysis to make sure that no inconsistencies or gaps were introduced.
- Consistency checking is a mechanism for checking whether rules do not contain semantically conflicting elements. Consistency checks belong to one of the following categories:
  - Checks that analyze an individual rule. These checks are activated when you build the rule and when you run the consistency checking analysis:
    - Rules that are never selected
    - Rules that never apply
    - Rules with range violation
  - Checks that analyze rules in relation to other rules. These checks are activated only when you run the Consistency checking analysis.
    - Rules with equivalent conditions
    - Equivalent rules
    - Redundant rules
    - Conflicting and self-conflicting rules
- Completeness analysis is a mechanism for checking whether there are rules that are missing from a set of rules. A rule set is considered complete if, for any possible case, at least one rule applies. You run completeness analysis for the following purposes:
  - Check whether a rule set is complete.
  - Obtain a report of the missing rules.
  - Add the missing rules to the rule set.

6.3.13 Validation process

Here is a sample validation and testing process:

- Plan all testing (functional, integration, regression, and so on)
- Prepare the testing environments and execute the tests according to the plans.
- Document testing issues, reporting procedures, and problem resolution processes.
- Finalize the execution of the testing plan.

Similar to IT validation testing, decision validation and testing must be planned and organized so that the risk of deploying a release in production is minimized.

Do not underestimate the amount of time and work required around the following tasks:

- Setting up testing environments
- Obtaining representative test data
- Creating test scenarios
- Running and debugging test scenarios

6.3.14 References

- von Halle, *Business Rules Applied: Building Better Systems Using the Business Rules Approach*, John Wiley & Sons, 2002, ISBN 9780471412939
- IBM WebSphere Operational Decision Management Version 8.0 Information Center, found at:

  http://pic.dhe.ibm.com/infocenter/dmanager/v8r0m1/index.jsp
- Office of Government Commerce (OGC), *ITIL Service Transition*, TSO, 2011, ISBN 9780113313068

7

Deployment

Deployment is the activity of creating a RuleApp and making it available for use by a Rule Execution Server (RES) instance. Planning the deployment process involves multiple choices and design decisions. This chapter explores some of these decisions, the possible alternatives, and the decision criteria to be considered. In particular, this chapter reviews decisions about the following topics:

- The source of truth for the business rules, that is, what repository is used for the rules.
- The deployment of the Decision Center component of Operational Decision Manager.

This chapter contains the following topics:

- Choosing the source of truth for the rules
- Deploying the Decision Center component
- Deploying and promoting RuleApps

7.1 Choosing the source of truth for the rules

The IBM Operation Decision Manager platform allows the management of rule projects both from an IT perspective, through the usage of the Rule Designer Eclipse plug-in, and from a business perspective, through the usage of the Decision Center business and enterprise consoles.

The prescribed lifecycle for Operational Decision Manager rule projects is that they initially are created and structured by using the Rule Designer component and then are published to the Decision Center, where most of the day-to-day rule management operations (rule authoring, testing, deployment, and so on) take place. In this concept of operations, after the projects are created, the Rule Designer environment is used only when some technical rule project updates, such as a change to the BOM or a rule flow modification, are needed. All other operations are performed from one of the Decision Center consoles or through scripts that use the Decision Center API. In particular, the RuleApp elements are managed in the Decision Center to perform deployments to the RES. This model is illustrated in the Figure 7-1 on page 91 on the left side.

In this model, the source of truth for the business rules is the Decision Center rule repository, and the rule project and their artifacts exist outside of the rule repository only for a short period, when they are modified and eventually synchronized back to the repository. Although this model is the preferred one, it is also possible to approach the rule management from a perspective that matches more closely the usual application source code (for example, Java) management model.

In this case, the source of truth for the rule projects is the project Source Code Control System (SCCS), which likely also manages the source code for the applications that use the decision services. By contrast with the previous model, it is now the rule projects in the Decision Center rule repository that are transient. They are published to allow the business users to author and modify the rules, and are then synchronized back to the SCCS through the Rule Designer and the underlying rule projects workspace. The RuleApp elements are managed through RuleApp projects in Rule Designer, and the RuleApp deployments are likely to be managed by a build server. This approach is illustrated in Figure 7-1 on page 91, on the right side.

This rule management and deployment model might be perceived as easier to integrate in an IT-centric organization, which uses preexisting and well-defined governance processes for managing and deploying application source code. However, it must be noted that the management of branches in rules projects on the Rule Designer side can quickly become complex, tedious, and error prone. One main source of complication is the merging operation, as the rule project artifacts comparison is based on the content of their persisted file version, which makes it difficult to isolate the changes between two versions. This situation is in contrast with the rule artifacts comparison available in Decision Center, which provides a clear and intuitive outlining of differences, as shown in Figure 7-1 on page 91.

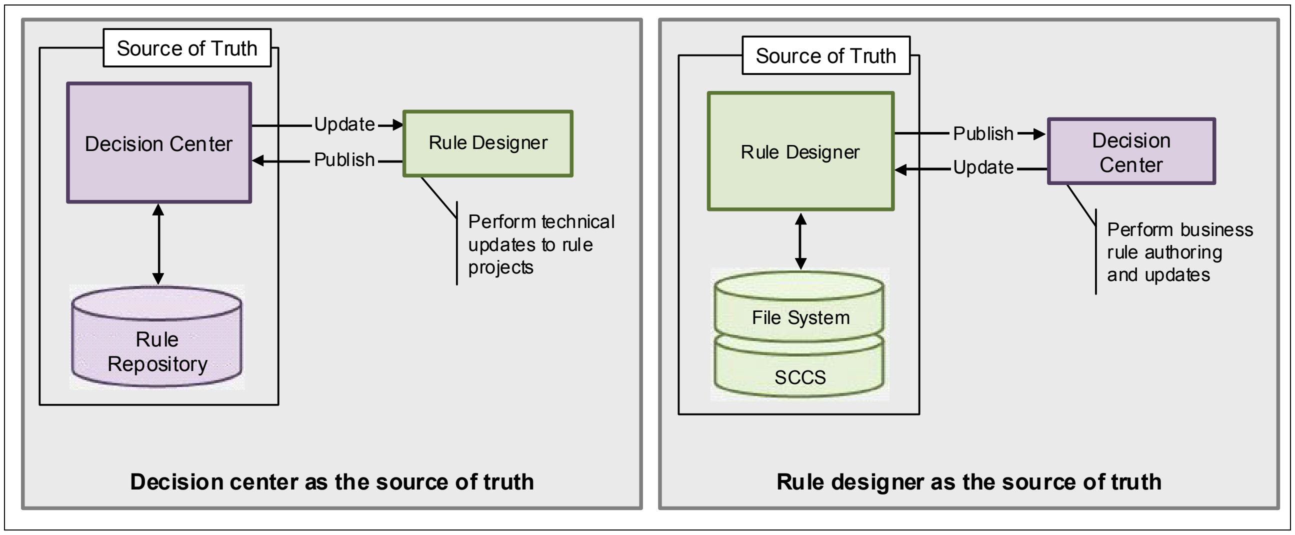

Source of Truth
Decision Center
Update
Publish
Rule Designer
Perform technical updates to rule projects
Rule Repository
Decision center as the source of truth
Source of Truth
Rule Designer
Publish
Update
Decision Center
Perform business rule authoring and updates
File System
SCCS
Rule designer as the source of truth

Figure 7-1 Rule management and deployment model

7.2 Deploying the Decision Center component

After an enterprise commits to the adoption of the Business Rules paradigm, and then to the usage of the IBM Operation Decision Manager platform, the business rules become a critical asset for the enterprise, comparable to the production data or the IT applications source code.

When the Decision Center rule repository is selected as the source of truth for the business rules, the component should be subject to an appropriate Service Level Agreement (SLA). In particular, consider the following items:

- The Decision Center database content (the rule repository) should be managed and backed up in the same way as it would be for production data.
- The availability of the Decision Center application should be compatible with the expected rate of changes in the business policies, and the need to explore the rules that are established with the business users. Ideally, it should have at least the same availability as the SCS server that supports the application code that is using the decision services.

Besides the service level agreement aspect, the Decision Center should also support the authoring and the governance of the rules among different roles, and potentially the deployment and promotion of the executable components (rule sets and RuleApps) though the different rule execution environments.

One common question then becomes, *where should you deploy the Decision Center component in your target component architecture?* The correct answer always depends on the particular situation. However, consider the following points.

There are some potential issues with deploying DC in the production environment:

- The internal users (rule authors and IT developers) might not have access to servers that are running in a production environment (they often do not in order to keep customer information confidential). In particular, DC must have an HTTP connection that is available to the development environment so that it can synchronize with Rule Designer.
- The production environment might be more exposed to hacking by external users. As confidential and business-critical enterprise assets, the Business Rules should be protected as much as possible from potential hacking.

There might also be issues with deploying DC in the development or System Integration Testing (SIT) environment:

- These environments might not have the SLA that is required to correctly support the rule authoring and management operations (the database backup is lacking, the hardware configuration is not sufficient, or high availability is not ensured).
- If RuleApps are deployed directly from DC to the RES in the different environments, DC might not have access to the Production RES.

It might be worth considering deploying DC in a separate environment, with an SLA that is comparable to what is provided to the other existing asset management components, such as the SCCS, but with enough flexibility so that various user profiles, in particular business users, can connect to it.

7.3 Deploying and promoting RuleApps

The RuleApp archive is the main artifact that is deployed to the RES. The are different options that are offered by the IBM Operational Decision Manager platform to create the archive from the Decision Center repository and to deploy it to the wanted RES.

The Decision Center center enterprise console provides the capability to generate a RuleApp archive, and also to directly deploy the archive to a certain RES instance. This capability is often used to deploy the archive to the development instance of the RES. However, this solution is usually not a viable one to deploy the archive to more controlled environments, such as test, User Acceptance Testing (UAT), and production.

To integrate the RuleApp deployment process with a Release Management (RM) system, the RuleApp generation and deployment are usually separated:

- The RuleApp generation is performed through the Decision Center deployment facility API. The RuleApp generation code is harnessed by a script that can be started by the RM system and output the archive in a well-defined location in scope of the RM system. It is then up to the release management team to decide how to manage the archive so that it is available for deployment to the different environments.
- The RuleApp deployment can use the deployment predefined res-deploy Ant task. The role of the deployment script is simply to pick up the archive that is provided by the RM system in a well-defined location and deploy it to the RES instance in the wanted environment.

Figure 7-2 shows both the interactive deployment from the Decision Center console and the deployment and promotion that is orchestrated by a release management system.

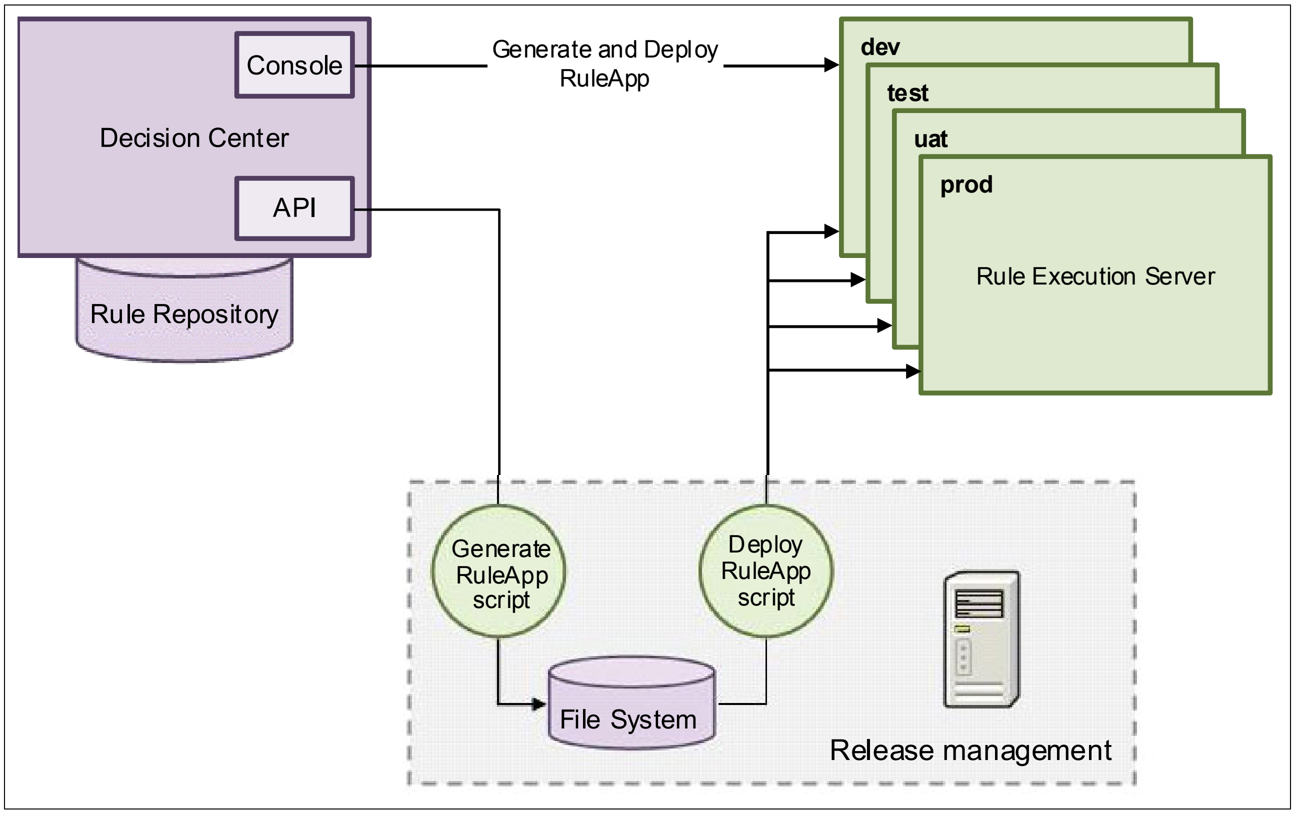


Figure 7-2 Interactive deployment

8

Processes

Governance processes are part of the implementation of governance. The documentation of the processes helps formalize the communication and responsibilities between the different actors that are involved in the processes. The documentation can also be used as a communication and training tool for new team members. It also reduces the knowledge that is lost when other team members leave. The documented processes also provide a baseline that can be used for process improvement for your organization.

Although there are many processes to consider in the governance of decisions, this chapter aims to provide a few sample processes specifically for change management that can be used as a starting point by different organizations in building their own internal processes.

The processes can be affected by many factors:

- Your specific list of roles and responsibilities that affect the actors in the processes.
- The different level of changes you want to recognize and differentiate with a different process.
- The types of validation that might be required for the different types of change.
- Service Level Agreements (SLA) for each type of change.
- The specific branching and merging strategy that you choose to take, especially with regards to test fixes and scheduled releases.

The processes that are shown in this chapter can be used as inspiration for what change processes that your organization can put in place. The branching and merging tasks are omitted from the processes because specific strategies for branching and merging make it difficult to create a generic diagram. When you work on your own diagram, you must consider this component and include the specific tasks that are required for branching and merging so that it is clear who has what responsibility.

This chapter contains the following topics:

- Business test fix process
- Pure Business change process
- Business with IT change process
- Technical change process
- Supporting validation processes

© Copyright IBM Corp. 2013. All rights reserved.

8.1 Business test fix process

This process aims to show the bare minimum set of activities that are required to implement simple business rule changes. Figure 8-1 shows the process to follow in the case of a business test fix.

For example, this process is used if a production issue is found and must be fixed quickly by making a simple change to a rule or a decisions table.

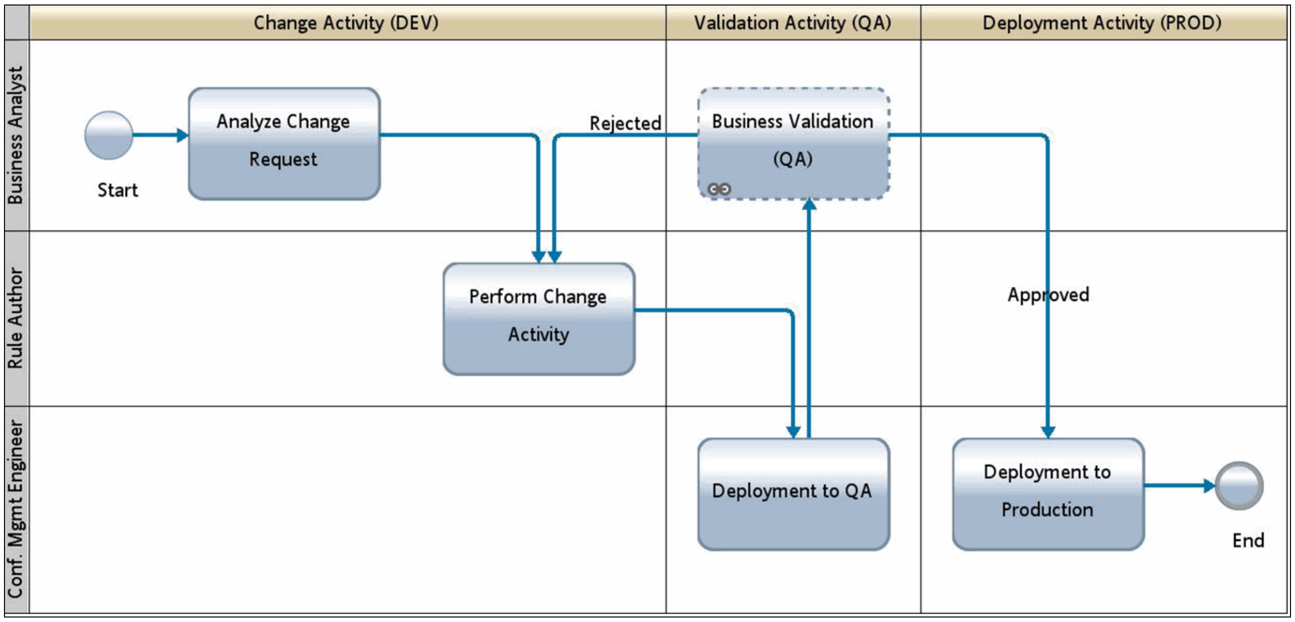


Figure 8-1 Business test fix process

Here are some notes about the process:

- The change activity is assumed to include the rule author's own validation that the change works in the development environment.
- The deployment of the test fix occurs immediately after the validation is approved.
- The change is considered simple enough to not require a second validation step before deployment to production.

For the test fix process, the usage of branches (and merging) can be optional should you choose to not create separate branches for creating test fixes, but choose to work on the existing and current production branch.

8.2 Pure Business change process

This process describes typical activities to be undertaken if the release requires changes only to business rules and no technical support from IT is required.

When compared to the business test fix process, the Pure Business change process includes a second validation step that is performed after the branch, with the changes merged into the release branch. See Figure 8-2.

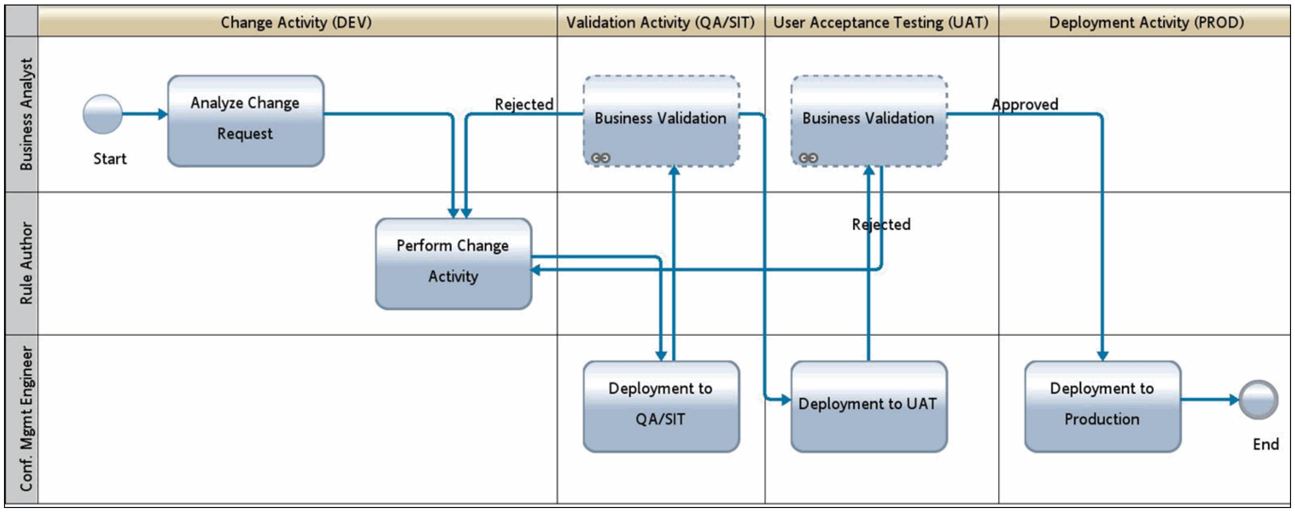


Figure 8-2 Pure Business change process

8.3 Business with IT change process

This process describes the activities that are likely to be undertaken where technical support is required from IT but there are not any integration changes. See Figure 8-3.

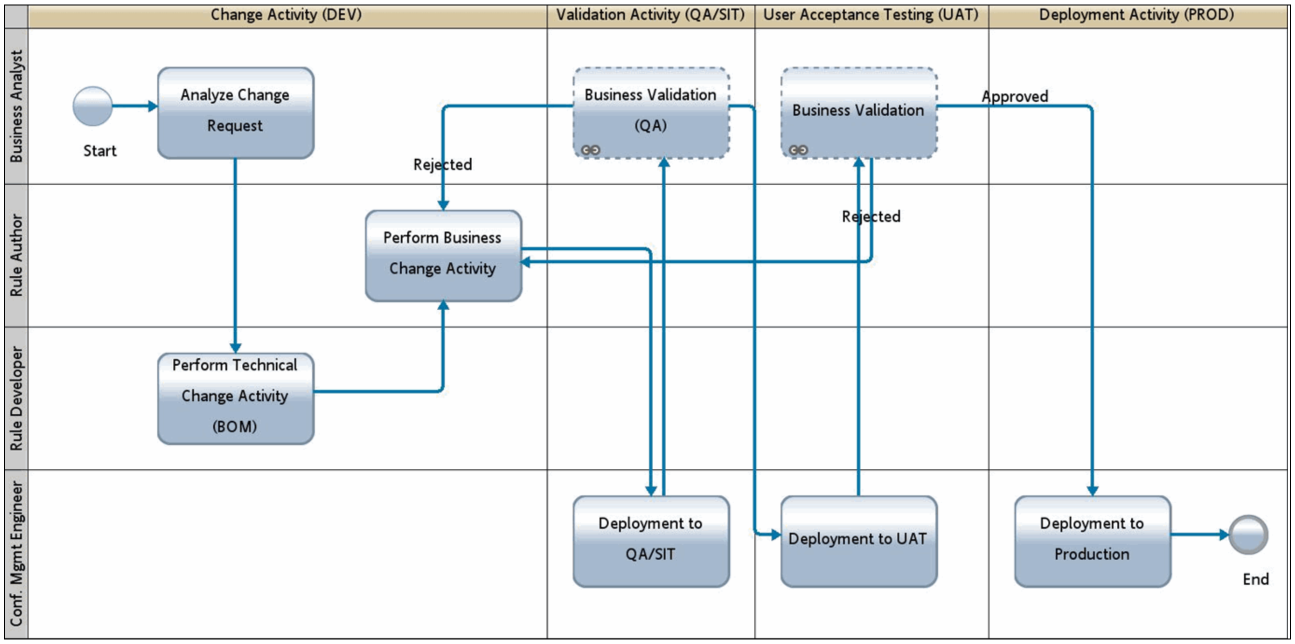


Figure 8-3 Business with IT change process

Note the following items:

- In this case, assume that the change that requires IT support does not change the interfaces to other systems, but rather provides additional functionality in the BOM to support new requirements in the rules, perhaps a new verbalization or a new utility function that can be implemented in the BOM
- Not shown in Figure 8-3 on page 97 is the situation where the Technical Change Activity does not work as expected and the Business Change Activity cannot be completed until the technical change activity is fixed. The technical and business change activities are expected to possibly be iterative in nature until the technical change works as expected for the business.

8.4 Technical change process

This process shows the activities that are required if the release changes imply that significant technical integration work is required from IT. See Figure 8-4.

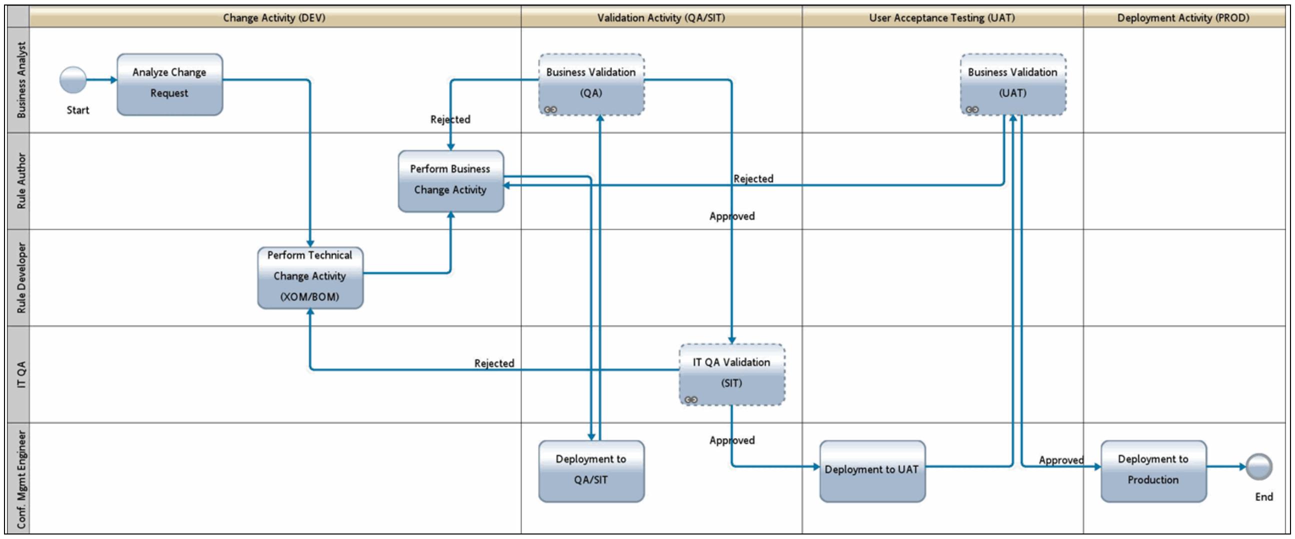


Figure 8-4 Technical change process

8.5 Supporting validation processes

The following processes are used in multiple places in other processes and hence are pulled out to simplify the other process diagrams. They essentially represent the same thing except that the activity is performed by a different person.

The business validation is performed by a business person, in this example, a business analyst. The technical validation is concerned with the integration testing and is performed by IT QA.

Figure 8-5 shows the business validation process.

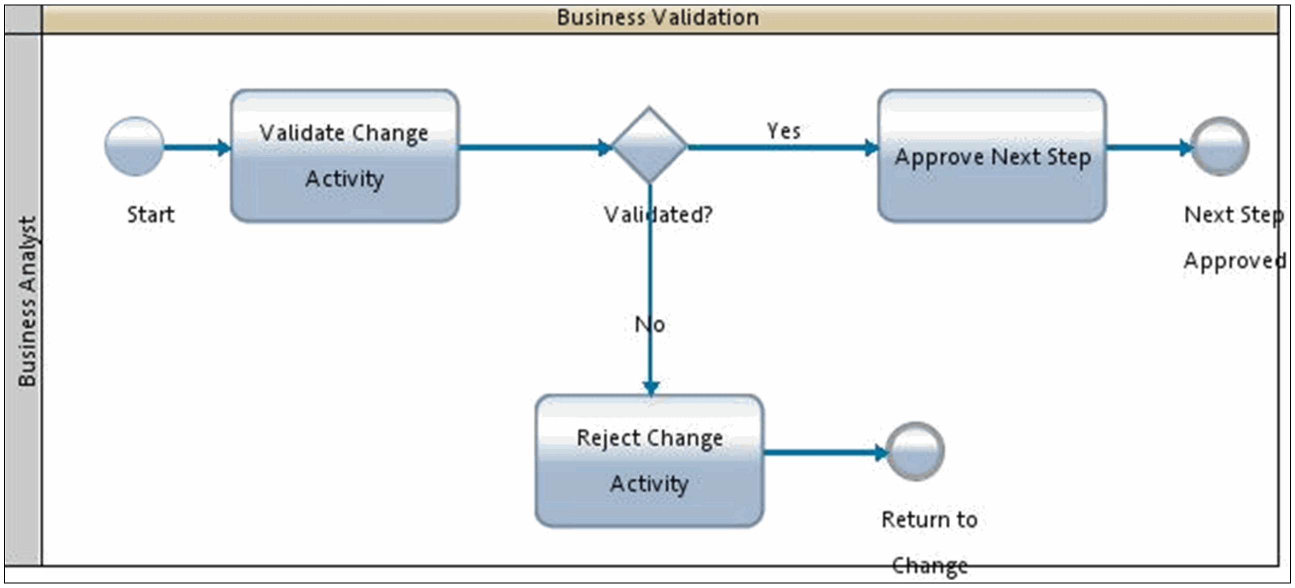


Figure 8-5 Business validation process

Figure 8-6 shows the IT QA validation process.

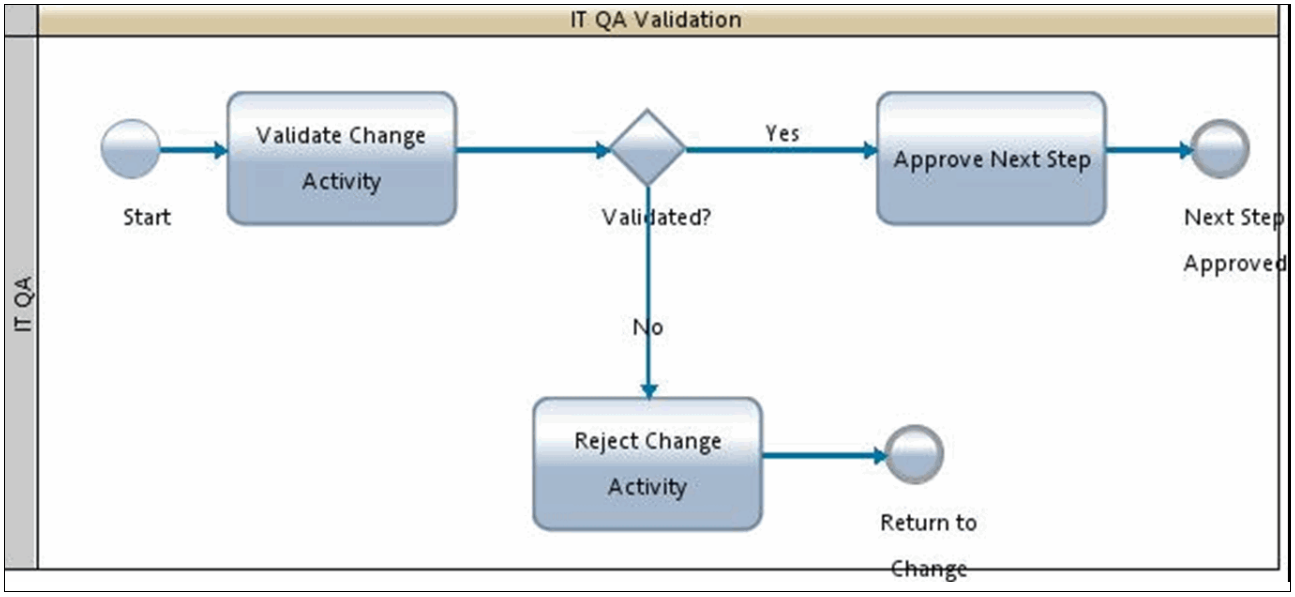


Figure 8-6 IT QA validation process

9

Realization of the insurance scenario

This chapter provides a practical example of applying the governance techniques that are described in the previous chapters to the auto insurance sample scenario.

This chapter contains the following topics:

- ► Overview
- ► Decision service project organization
- ► Defining roles responsibilities and permissions
- ► Pure Business Change process example
- ► Technical Change Process example

© Copyright IBM Corp. 2013. All rights reserved.

9.1 Overview

Figure 9-1 shows the insurance company solution from the sample scenario, highlighting the Pricing rules that we concentrate on in this realization.

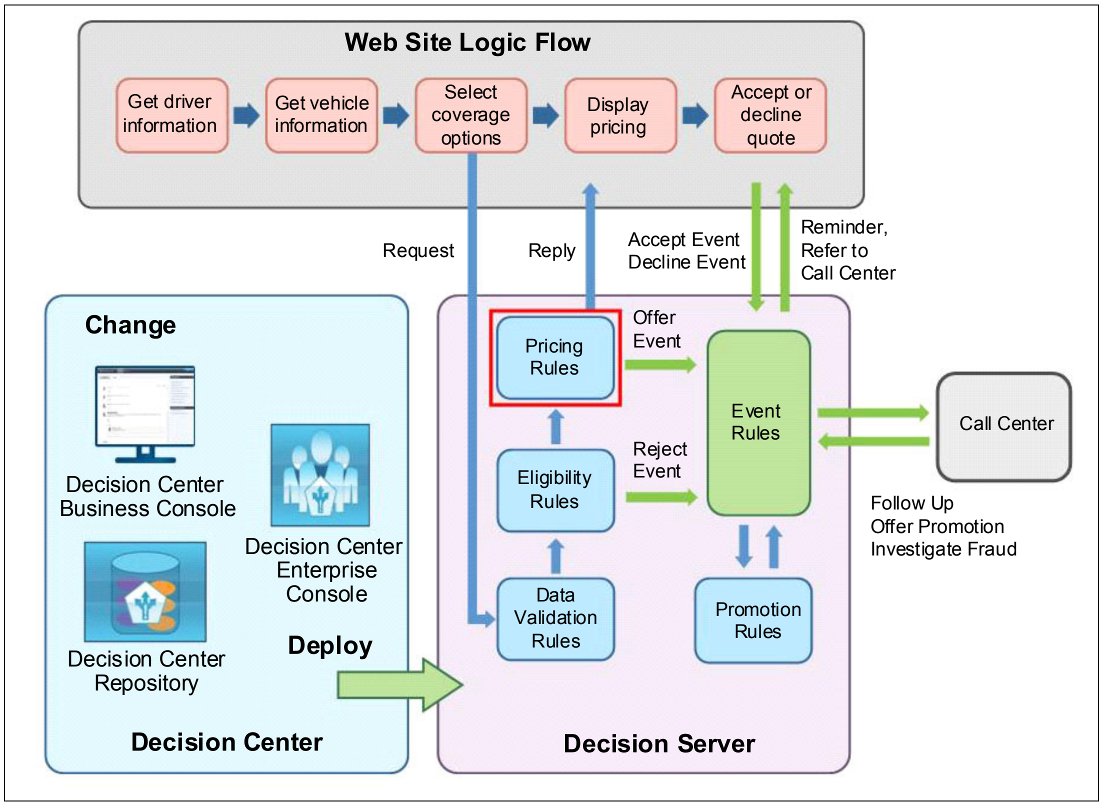


Figure 9-1 Solution realization context

To keep the descriptions simple, the other decisions and rules within this solution are not considered and we concentrate on how changes to the pricing rules can be governed using IBM Operational Decision Manager.

The section first describes how the pricing rules are organized into rule projects to realize the Pricing decision service. The Pricing decision service projects are organized by taking into account the split in responsibilities between regional underwriting pricing policies and global marketing-based reductions and surcharges. The concepts behind this structure are described in Chapter 4, “Organizing decision management” on page 37.

The next section describes how the various roles and responsibilities are mapped onto the permissions that allow the correct individuals to perform the correct tasks. In this case, Barbara has the role of the business sponsor, defining the key requirements for change to the pricing decision service and approving new releases that realize those changes. Paul is the Release Owner and has the responsibility of ensuring that all changes are undertaken in a timely and safe manner. Rachel is a regional manager, undertaking the policy changes to improve New Jersey profitability. Ivan is an IT specialist with responsibility for integrating the pricing decision service into the solutions and realizing any technical changes that are illustrated in the technical release lifecycle. Adam is the administrator of the system that is responsible for deployment to the operational systems.

The following change processes are also described:

- A Pure Business change process example illustrates the means to bring day to day changes that arise from the business into a regular planned release. This lifecycle is typical of a *standard* release and demonstrates how line of business (LOB) users can safely manage the changes to business policies. In this case, we show how to define a Monthly_Pricing release to manage the regular change cycle and a Regional_Profitability change activity to show how the LOB users can modify the rules as part of that release process. In this example, Barbara (Business Analyst) identifies the goal for the January release to improve the New Jersey profitability. Rachel (New Jersey Regional Manager) then modifies the rules within the change activity to realize those goals. The process then illustrates some typical validation activities, such as simulation and testing, that can be used to confirm that the goals are met before Adam (IT Administrator) deploys the new release into production.
- A technical change process example illustrates the means to plan a new major release that requires changes in the insurance solution itself as well as the decision service. This lifecycle is typical of a *normal* release and demonstrates the key interactions with the software development lifecycle when changes occur in both the solution and the decision service release. In this case, we show how to define a *Roadside_Assistance* release whose goal is to extend the pricing decision service to support a new initiative that requires new information to be considered. We show how to create the Roadside_Assistance release and how that can be synchronized into the IT tools so that Ivan (IT Architect representing IT development and integration roles) can change the interface between the solution and the pricing decision service. We then show how Ivan can change the information model that is used by the pricing decision service to accommodate the new RoadsideAssistanceType and make this type available in the rules language so that rule authors can use it easily in their rules. We then show how Ivan changes the signature that is used between the new Insurance Solution and the new decision service to minimize the impact on the deployed solution with its existing decision service interface. This allows later planned monthly releases of the pricing decision service to support both the new and existing versions of the solution until the existing solution can be retired.

9.2 Decision service project organization

The insurance organization's pricing decision service must be structured to support the different decision making capabilities that are required and allow the different roles to have access to the rules and project content for which they are responsible.

Figure 9-2 shows the project structure.

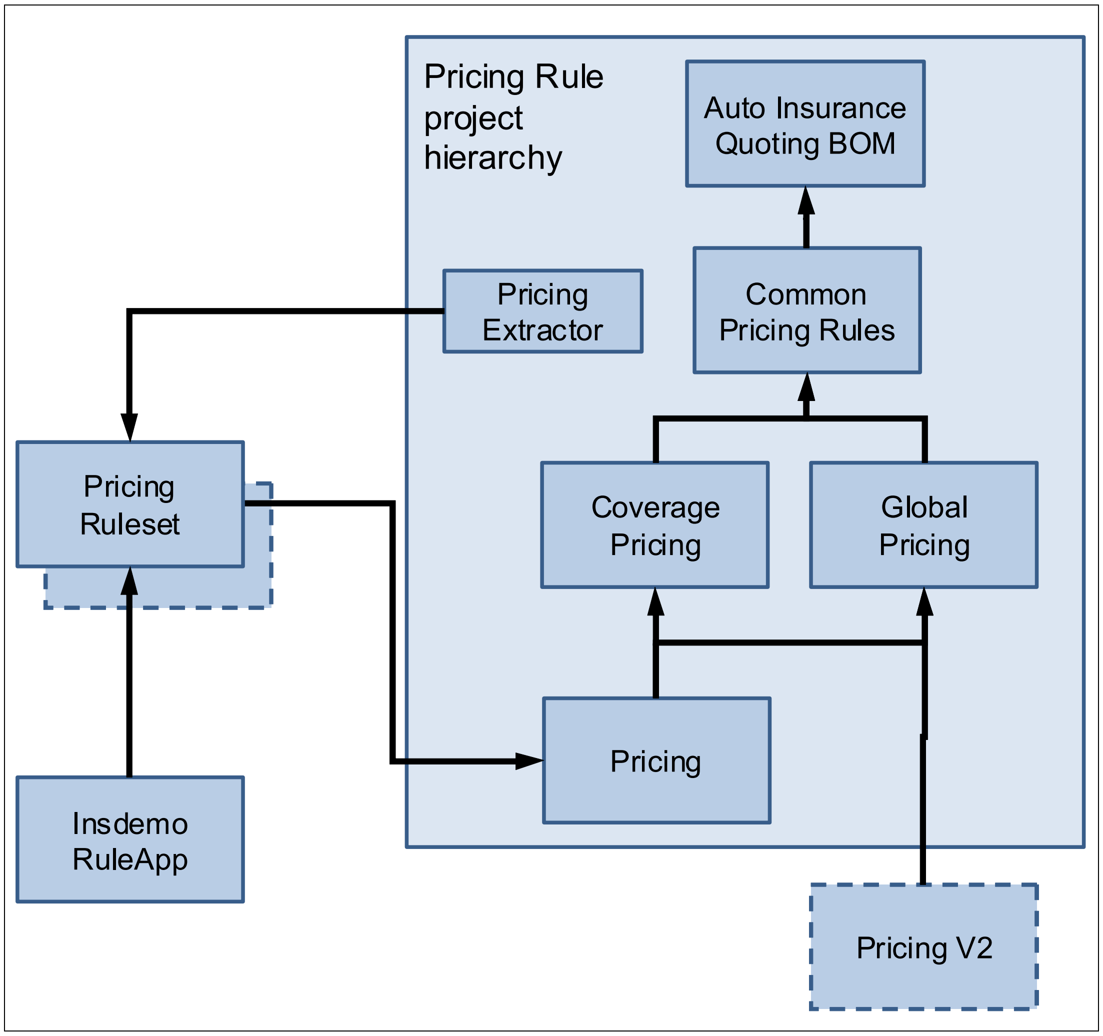


Figure 9-2 Pricing decision service structure

Each project can be configured to allow the different groups to access it, but can also define dependencies on other projects, allowing vocabularies and variables (information about which the decisions are made) to be shared between these projects.

The role of each project is described in Table 9-1.

Table 9-1 Project roles

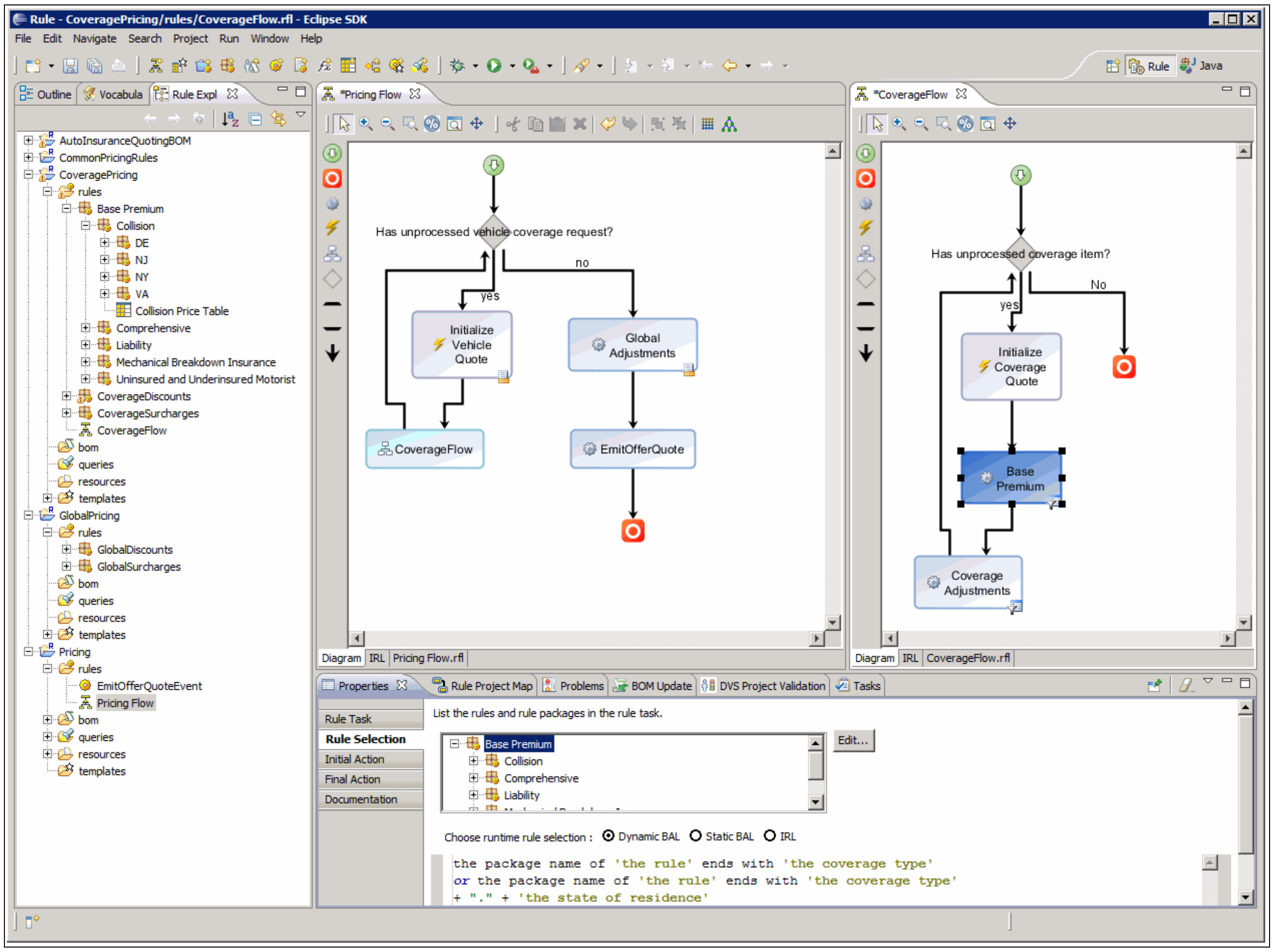
Project	Type	Description
Pricing	Rule set Definition	The Pricing project defines the parameters that are exposed by the service and the rule flows that are followed whenever the service is started. This is configured by IT so that the rules in the LOB projects are applied in the correct sequence. This project also defines the following items: ► Extractors that are used to build the deployable rule sets. ► Simulations and test suites that are used to validate the overall decision service.

Project	Type	Description
Coverage Pricing	Pricing rules	The Coverage Pricing project holds the main rules for calculating the price of an insurance quote. This is used by the LOB to manage the behavior of the decision. Within the project, group access control can be used to restrict visibility to certain regions (for example, New Jersey).
Global Pricing	Marketing pricing rules	The Global Pricing project's policies are applied after the coverage pricing. This represents discounts or surcharges that may be used for marketing promotions or to balance risk for particular cases. The regional managers would not have access to this project.
Common Pricing Rules	Shared Rules and variables	The Common Pricing project's rules provide a location where shared rules and variables are provided. In this example, the project contains the variables that are populated by the rule set parameters when the decision service is started. This means that both the Coverage Pricing and Global Pricing rules have access to this information.
Auto Insurance Quoting BOM	BOM & Vocabulary	This project contains the shared vocabulary and Business Object Model (BOM) that is used by all the projects in the decision service. This would be defined by IT and should not need to be visible to LOB users.

9.2.1 Definition of decision service structure in Rule Designer

The projects that are used in the decision service are created in Rule Designer together with any technical artifacts, such as Java projects or schemas that form the basis of the XOM. These projects are defined by IT, and the key activities that are undertaken when you define or change this structure are described in the technical release lifecycle section.

An example of the projects and key folder structure is shown in Figure 9-3.



Choose runtime rule selection : (• Dynamic BAL) ○ Static BAL ○ IRL

```
the package name of 'the rule' ends with 'the coverage type'
or the package name of 'the rule' ends with 'the coverage type'
+ "." + 'the state of residence'
```

Figure 9-3 Rule Designer Project structure

In the Rule Designer workspace, the projects are created together with the folder structure that is used to organize the rules.

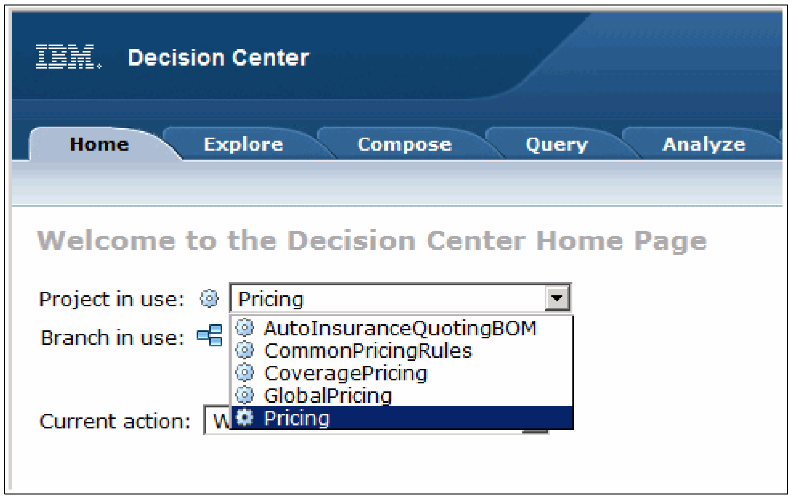
Figure 9-3 shows how the Pricing project defines the Pricing Flow rule flow, which first iterates through each of the individual coverage pricing rules (CoverageFlow) before applying the Global Adjustments from within the Global Pricing project.

The Coverage pricing project is structured into Base Premium, CoverageDiscounts, and CoverageSurcharge for each type of cover. The CoverageFlow rule selection shows how the rules from each cover type (Collision and Comprehensive) are selected according to the coverage type that is being considered. The Rule Selection tab also shows how regional rule variations can be taken into account by including rules in the regional folder appropriate to the quote request.

9.2.2 Navigating the decision service structure in Decision Center

After the structure is defined in Rule Designer together with the artifacts that are defined by IT, the projects can be synchronized with Decision Center, allowing the rules to be navigated and edited within the releases.

In Decision Center Enterprise console, you can select the project that you want to examine, as shown in Figure 9-4.



Welcome to the Decision Center Home Page

Project in use: Pricing

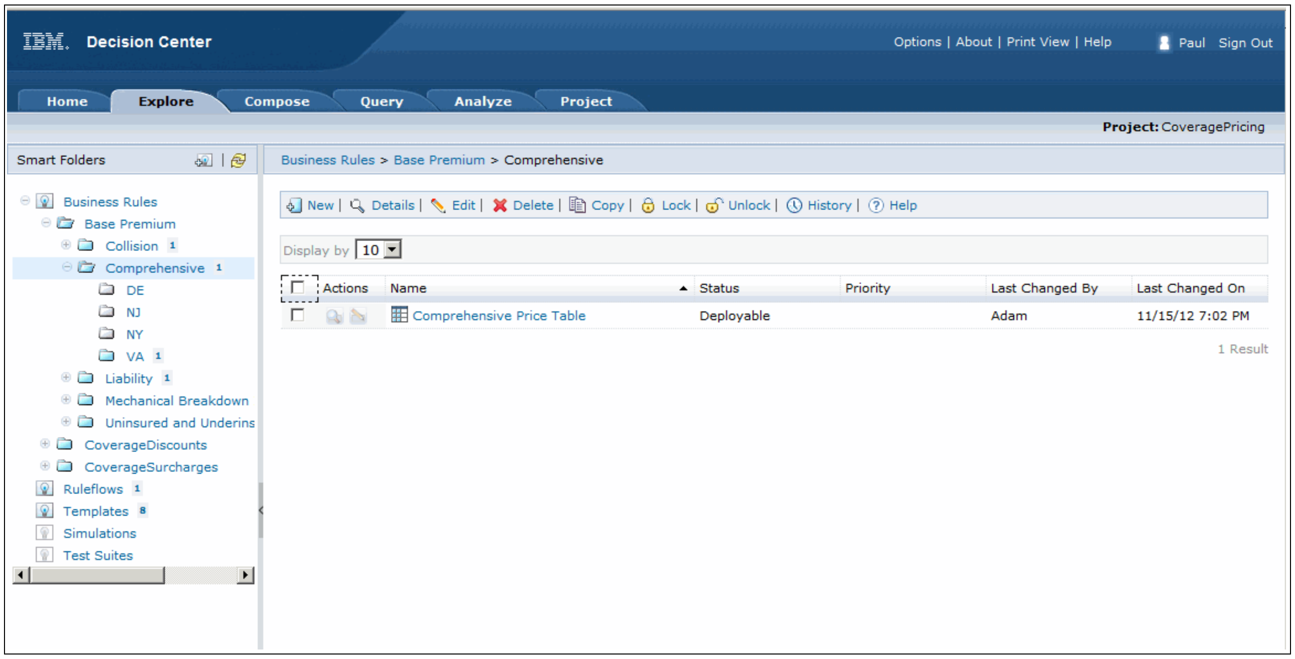
Branch in use:

Current action: W

AutoInsuranceQuotingBOM
CommonPricingRules
CoveragePricing
GlobalPricing
Pricing

Figure 9-4 Decision Center Enterprise console navigation

After you select the Release or Change activity branch that you require and select the **Explore** tab, the folder structure of the project can then be seen, as shown in Figure 9-5.


IBM. Decision Center
Options | About | Print View | Help
Paul Sign Out
Home
Explore
Compose
Query
Analyze
Project
Smart Folders
New | Details | Edit | Delete | Copy | Lock | Unlock | History | Help
Display by 10

Project: CoveragePricing

Business Rules > Base Premium > Comprehensive

- Business Rules
  - Base Premium
    - Collision 1
    - Comprehensive 1
      - DE
      - NJ
      - NY
      - VA 1
    - Liability 1
    - Mechanical Breakdown
    - Uninsured and Underins
  - CoverageDiscounts
  - CoverageSurcharges
- Ruleflows 1
- Templates 8
- Simulations
- Test Suites

Actions	Name	Status	Priority	Last Changed By	Last Changed On
	Comprehensive Price Table	Deployable		Adam	11/15/12 7:02 PM

1 Result

Figure 9-5 Decision Center Enterprise console folder navigation

In the business console, projects are selected from the **Library** tab, as shown in Figure 9-6.

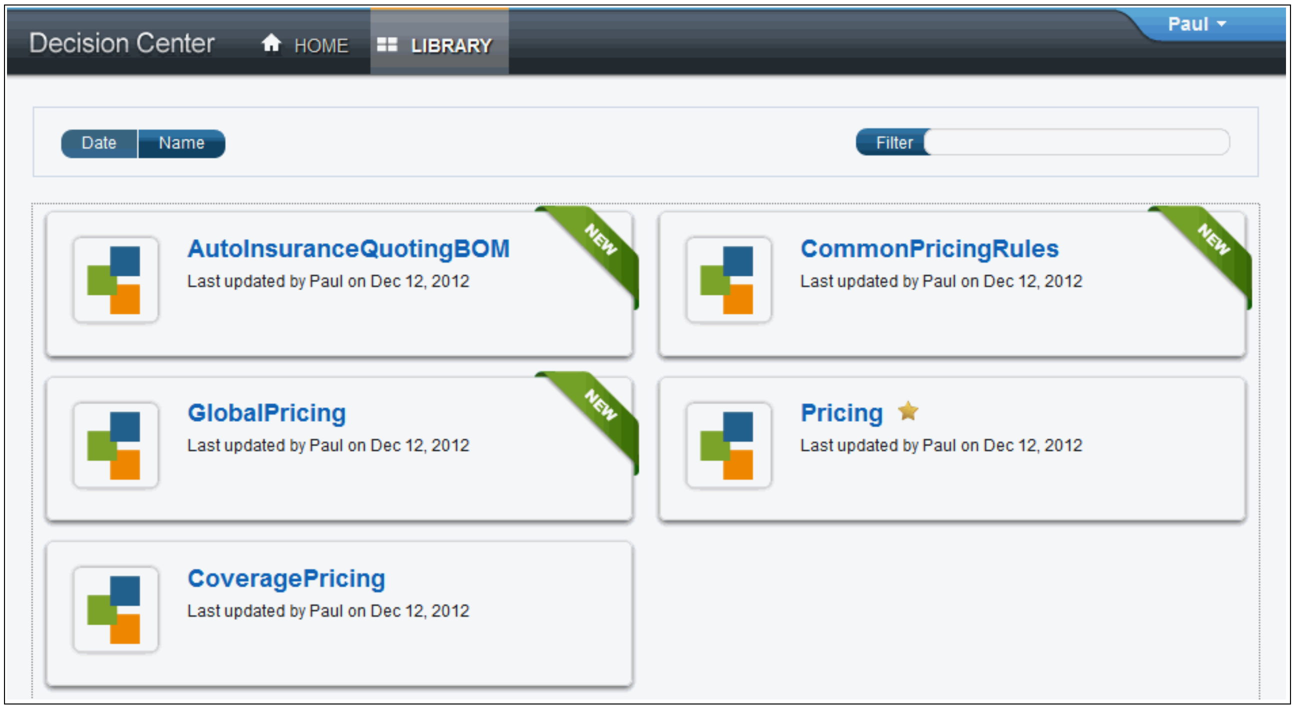

Decision Center
HOME
LIBRARY
Paul
Date
Name
Filter
AutoInsuranceQuotingBOM
Last updated by Paul on Dec 12, 2012
NEW
CommonPricingRules
Last updated by Paul on Dec 12, 2012
NEW
GlobalPricing
Last updated by Paul on Dec 12, 2012
NEW
Pricing
Last updated by Paul on Dec 12, 2012
CoveragePricing
Last updated by Paul on Dec 12, 2012

Figure 9-6   Decision Center Business console project navigation

When you select the Coverage Pricing Project and the appropriate branch (release or change activity), the rules can be navigated alphabetically or by using the folder structure of the project. See Figure 9-7.

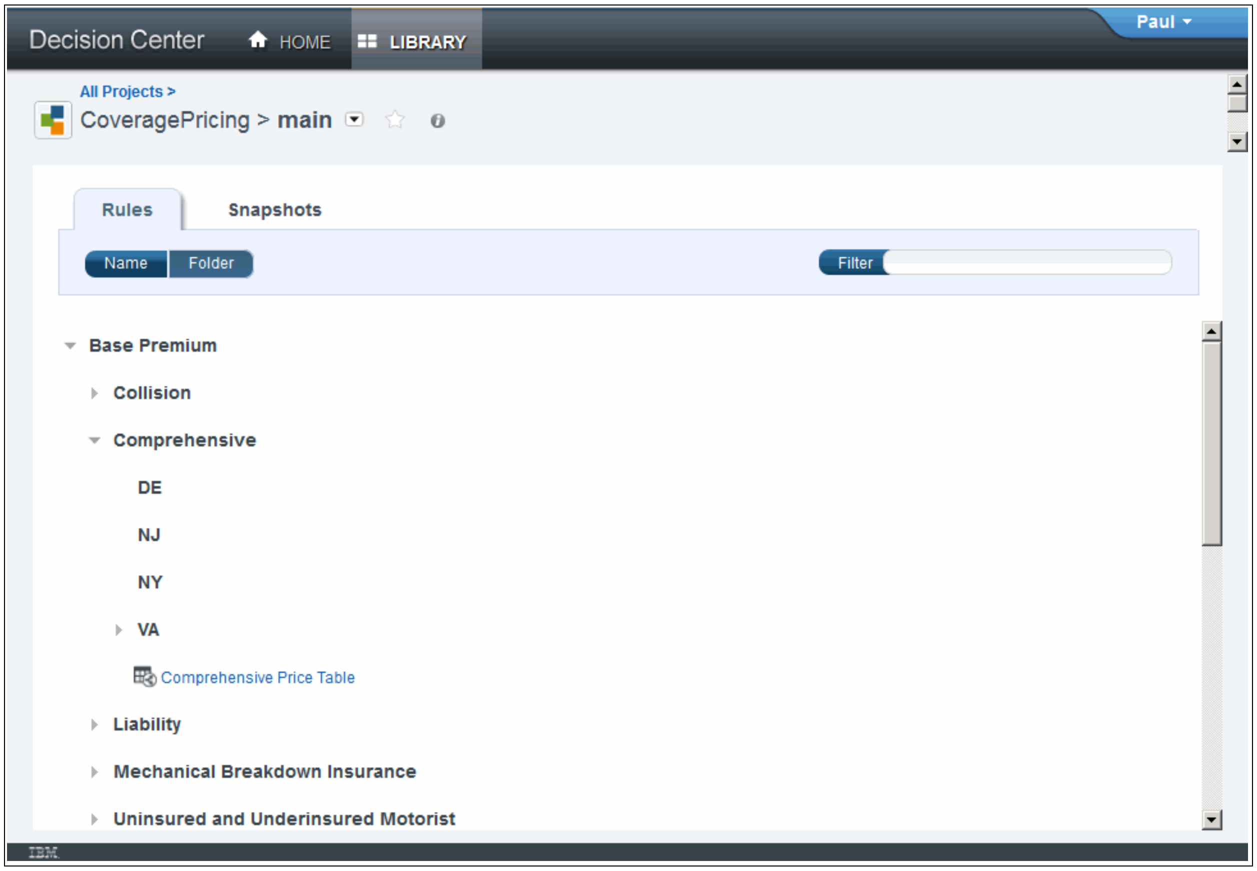


Figure 9-7 Decision Center Business console folder navigation

These capabilities allow the structure of a decision service to be established in Rule Designer to support the way that the decision is made. This structure is then exposed in Decision Center, where it can be used to control the roles and rights of the users that modify the rules that influence the decision service behavior.

9.3 Defining roles responsibilities and permissions

IBM Operational Decision Manager uses permissions to define role responsibilities and access control and thus ensure that the authorized users make the changes to a release according to the governance policies established.

The key users and their roles in the sample scenario were described earlier and are shown in Table 9-2.

Table 9-2 Auto insurance scenario user roles and responsibilities

User	Role	Description
Barbara	Analyst	Barbara has the role of the business sponsor, defining the key requirements for change to the pricing decision service and approving new releases that realize those changes.

User	Role	Description
Paul	Policy Manager	Paul is the Release Owner and has the responsibility of ensuring all changes are undertaken in a timely and safe manner.
Rachel	Regional Manager	Rachel is a regional manager, undertaking the policy changes to improve New Jersey profitability.
Ivan	Architect	Ivan is an IT specialist with responsibility for integrating the pricing decision service in to the solutions and realizing any technical changes that are illustrated in the technical release lifecycle. He represents the IT Architect, Rule Developer, and Integration developer roles.
Adam	Administrator	Adam is the administrator of the system, and is responsible for deployment of the operational systems.

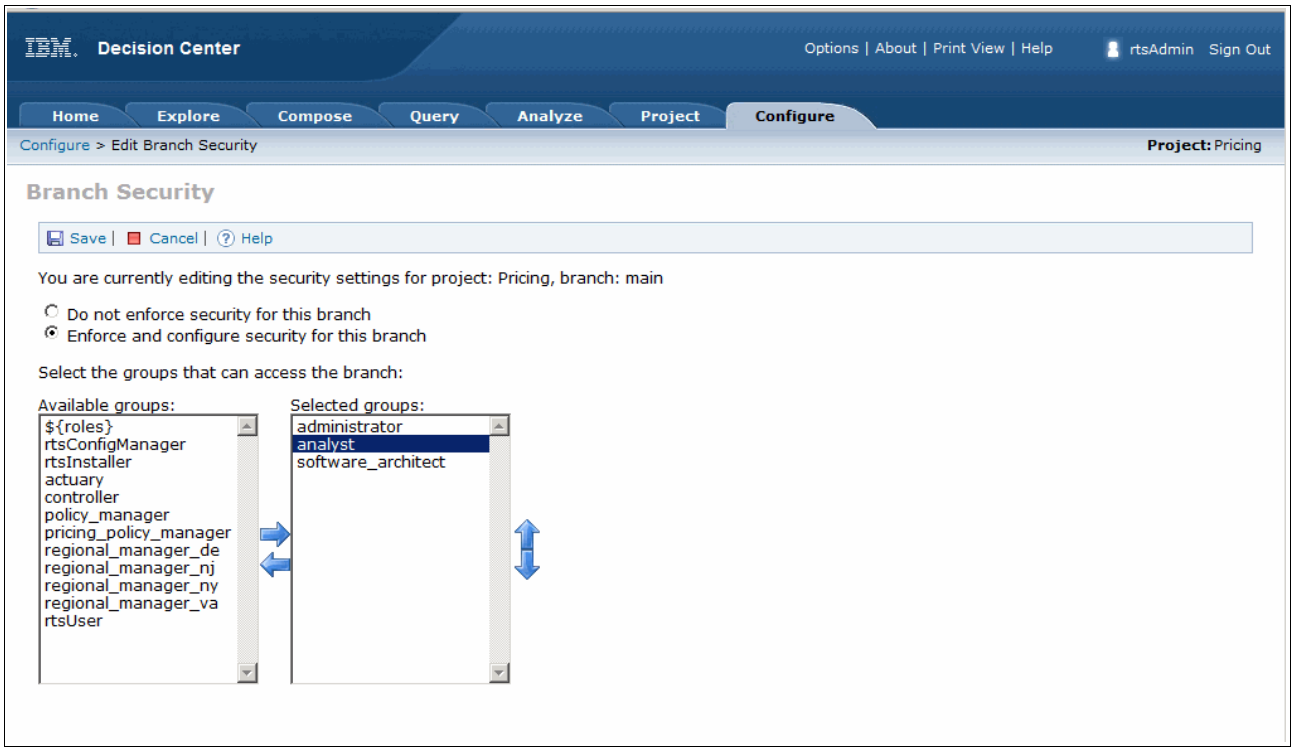
9.3.1 Project permissions

Each project within the decision service may be configured to define the roles that have access to it. In the roles that are shown in Table 9-3, the Architect is acting on behalf of development changes that are introduced by IT development.

Table 9-3 Project roles

Project	Type	Roles	Role usage description
Pricing	Rule set Definition	► Architect ► Administrator ► Analyst	► The architect has access to the Pricing project to define the interfaces and test cases that are exposed to the business solutions. ► The administrator must be able to make emergency changes to any of the artifacts that are defined in this project. ► The analyst must be able to run the simulation and test suites to validate the end to end business impact of the pricing decision.
Coverage Pricing	Pricing rules	► Policy Manager ► Regional Manager ► Analyst	The Coverage Pricing holds the main rules for calculating the price of an insurance quote. This is used by the LOB to manage the behavior of the decision. Within the project, group access control can be used to restrict visibility to certain regions (for example, New Jersey).
Global Pricing	Marketing pricing rules	► Policy Manager ► Analyst	The Global pricing policies are applied after the coverage pricing. This represents discounts or surcharges that may be used for marketing promotions or to balance risk for particular cases. The regional managers would not have access to this project.
Common Pricing Rules	Shared Rules and variables	Administrator	The common Pricing rules provide a location where shared rules and variables are provided. In this example, the project contains the variables that are populated by the rule set parameters when the decision service is started.
Auto Insurance Quoting BOM	BOM & Vocabulary	► Architect ► Analyst	This project contains the shared vocabulary and Business Object Model that is used by all the projects in the decision service.

These roles can be configured by using the Decision Center Enterprise console for each project. See Figure 9-8.



Branch Security

You are currently editing the security settings for project: Pricing, branch: main

( ) Do not enforce security for this branch
(●) Enforce and configure security for this branch

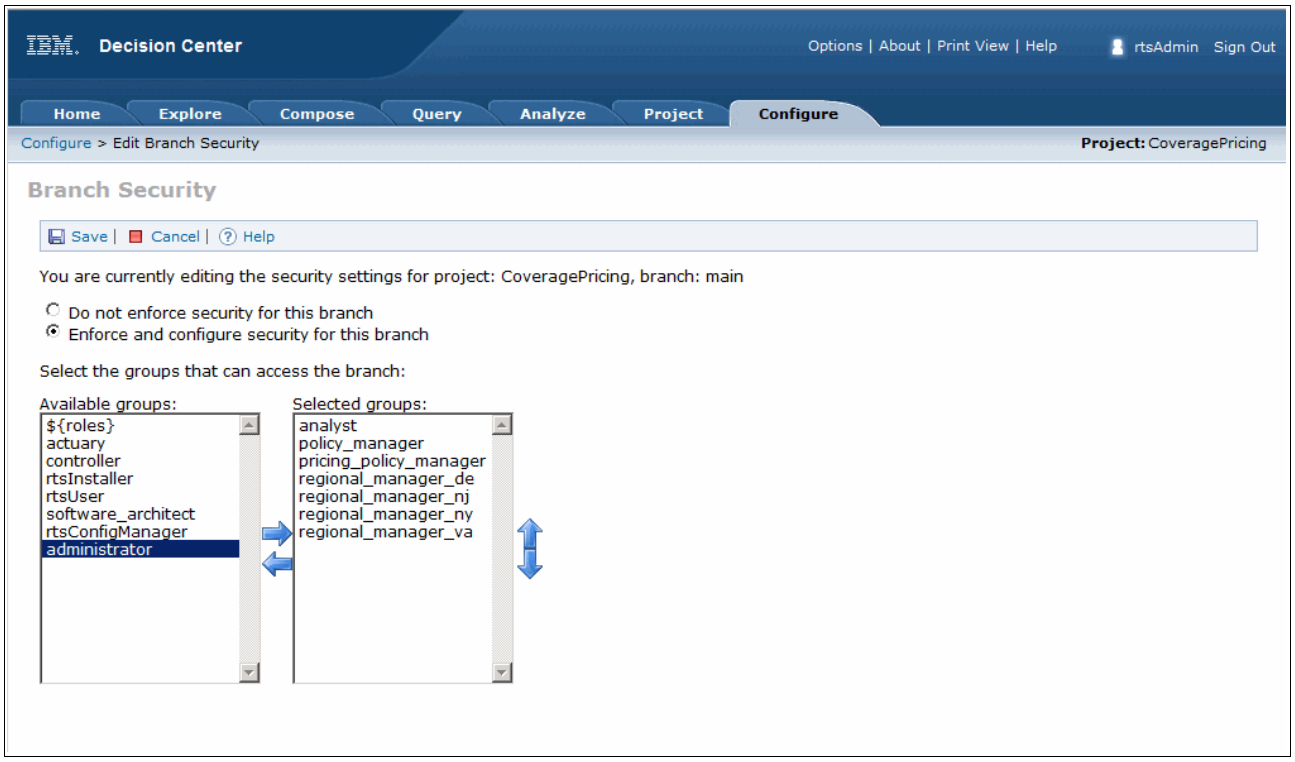
Select the groups that can access the branch:

Available groups:
${roles}
rtsConfigManager
rtsInstaller
actuary
controller
policy_manager
pricing_policy_manager
regional_manager_de
regional_manager_nj
regional_manager_ny
regional_manager_va
rtsUser

Selected groups:
administrator
analyst
software_architect

Figure 9-8 Pricing Project Access Control

In contrast, a functional project such as Coverage Pricing provides access control to the LOB users that need to manage the policies within the rules. See Figure 9-9.



You are currently editing the security settings for project: CoveragePricing, branch: main

○ Do not enforce security for this branch
● Enforce and configure security for this branch

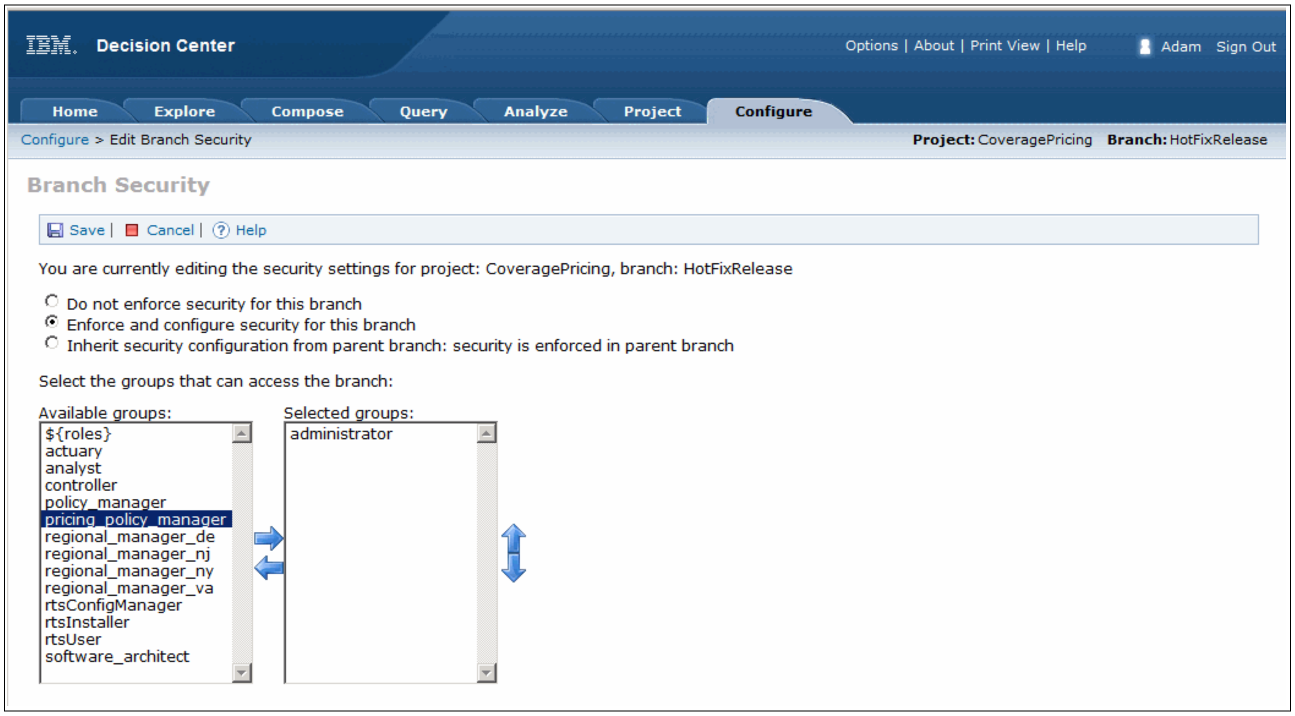
Select the groups that can access the branch:

Available groups:
${roles}
actuary
controller
rtsInstaller
rtsUser
software_architect
rtsConfigManager
administrator

Selected groups:
analyst
policy_manager
pricing_policy_manager
regional_manager_de
regional_manager_nj
regional_manager_ny
regional_manager_va

Figure 9-9 Coverage Pricing Project Access Control

It is also possible to modify the access control at the release level for any project, as shown below. Figure 9-10 shows that only administrators have access to the coverage pricing rules for a test fix release. The default behavior is to inherit the permissions from the parent branch. This means that normally the permissions that are established for the decision service projects are applied automatically to any new releases.



Project: CoveragePricing Branch: HotFixRelease

Branch Security

You are currently editing the security settings for project: CoveragePricing, branch: HotFixRelease

- ○ Do not enforce security for this branch
- ● Enforce and configure security for this branch
- ○ Inherit security configuration from parent branch: security is enforced in parent branch

Select the groups that can access the branch:

Available groups:
${roles}
actuary
analyst
controller
policy_manager
pricing_policy_manager
regional_manager_de
regional_manager_nj
regional_manager_ny
regional_manager_va
rtsConfigManager
rtsInstaller
rtsUser
software_architect

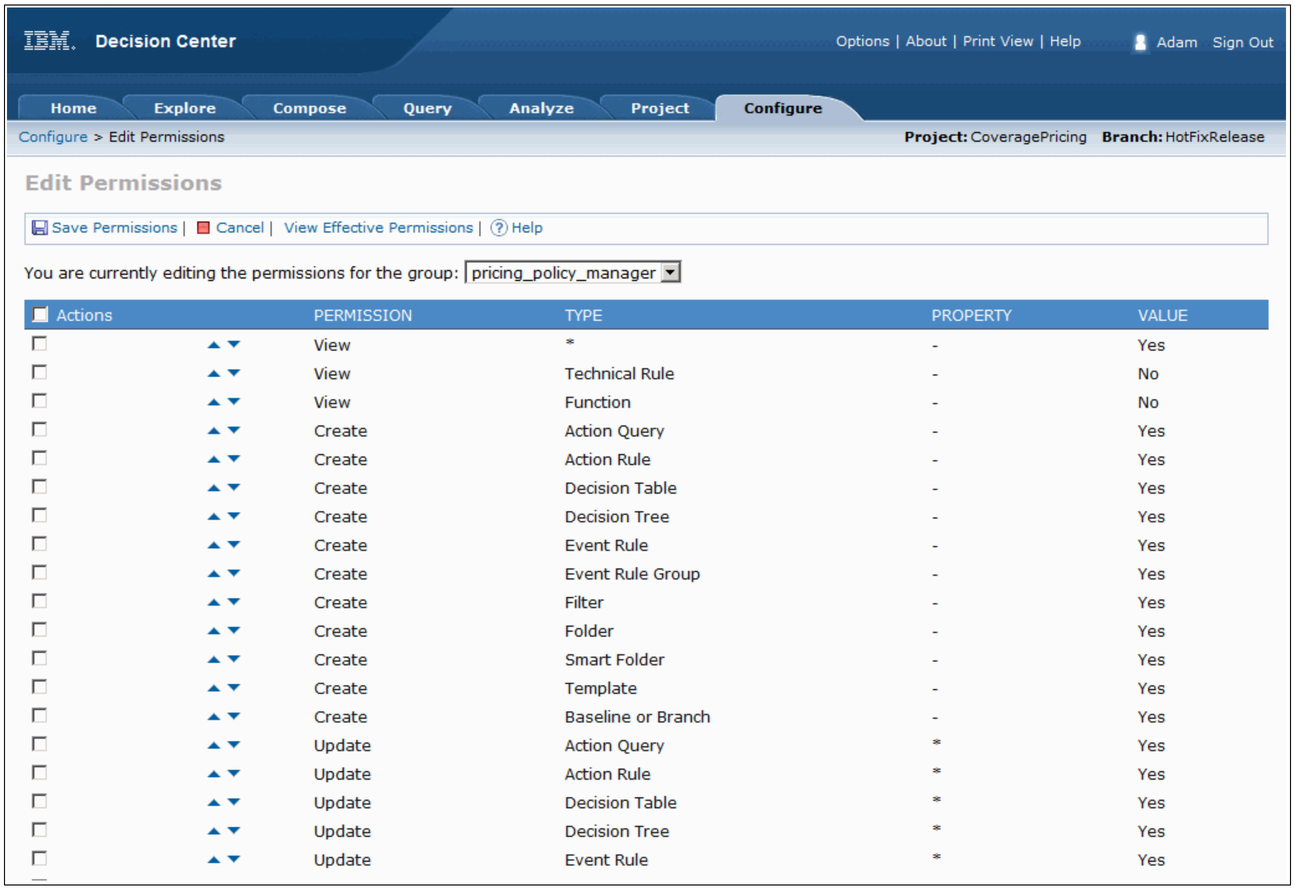
Selected groups:
administrator

Figure 9-10 Branch level access control

9.3.2 Defining role-based action permissions

In addition to defining the project and branch level permissions that identify who can access which branches and releases, a finer-grained access control can be applied to indicate what roles have what type of access to the different artifacts. This allows the different roles and their skill levels to be mapped to the actions that they can perform.

Figure 9-11 shows the permissions for the pricing policy manager role (Release Manager).


IBM. Decision Center
Options | About | Print View | Help
Adam Sign Out
Home
Explore
Compose
Query
Analyze
Project
Configure
Configure > Edit Permissions
Save Permissions | Cancel | View Effective Permissions | Help

Project: CoveragePricing **Branch:** HotFixRelease

Edit Permissions

You are currently editing the permissions for the group: pricing_policy_manager

Actions	PERMISSION	TYPE	PROPERTY	VALUE
	View	*	-	Yes
	View	Technical Rule	-	No
	View	Function	-	No
	Create	Action Query	-	Yes
	Create	Action Rule	-	Yes
	Create	Decision Table	-	Yes
	Create	Decision Tree	-	Yes
	Create	Event Rule	-	Yes
	Create	Event Rule Group	-	Yes
	Create	Filter	-	Yes
	Create	Folder	-	Yes
	Create	Smart Folder	-	Yes
	Create	Template	-	Yes
	Create	Baseline or Branch	-	Yes
	Update	Action Query	*	Yes
	Update	Action Rule	*	Yes
	Update	Decision Table	*	Yes
	Update	Decision Tree	*	Yes
	Update	Event Rule	*	Yes

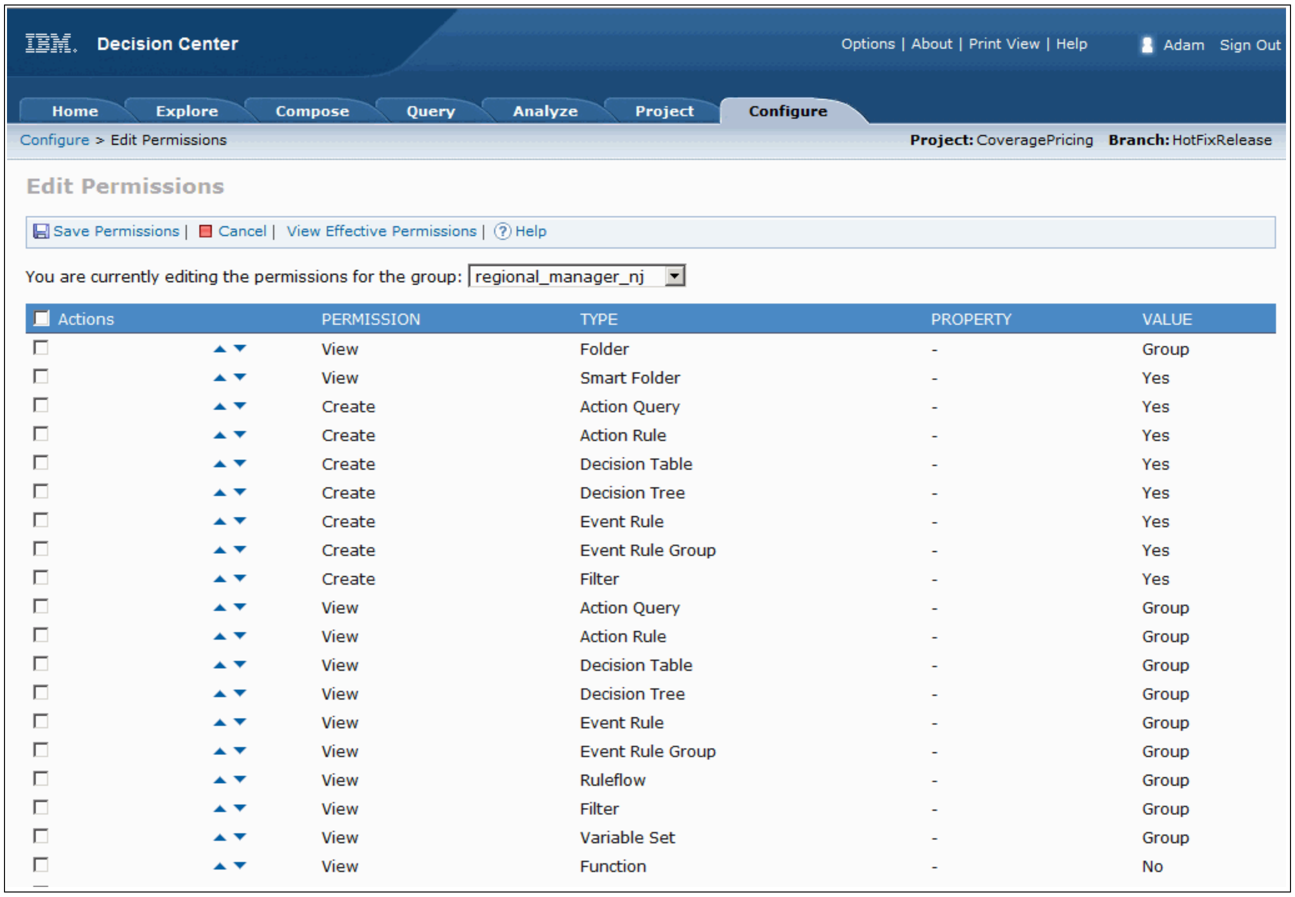
Figure 9-11 Role-based action permissions

In this case, the Release Manager can see all types of artifact except the detailed technical artifacts. They can also create and update most types of artifact. However, they do not have any delete permissions, so deletes must be performed by administrators in this configuration.

9.3.3 Group level action permissions

A finer level of control can also be applied based on the group property that is attached to rules and folders.

This approach allows the regional managers to have actions that are constrained to those rules and folders that apply to their region or group. Figure 9-12 on page 115 shows how some of the permissions are set up for the New Jersey regional manager.


IBM. Decision Center
Options | About | Print View | Help
Adam Sign Out
Home Explore Compose Query Analyze Project Configure
Configure > Edit Permissions
Save Permissions | Cancel | View Effective Permissions | Help

Project: CoveragePricing **Branch:** HotFixRelease

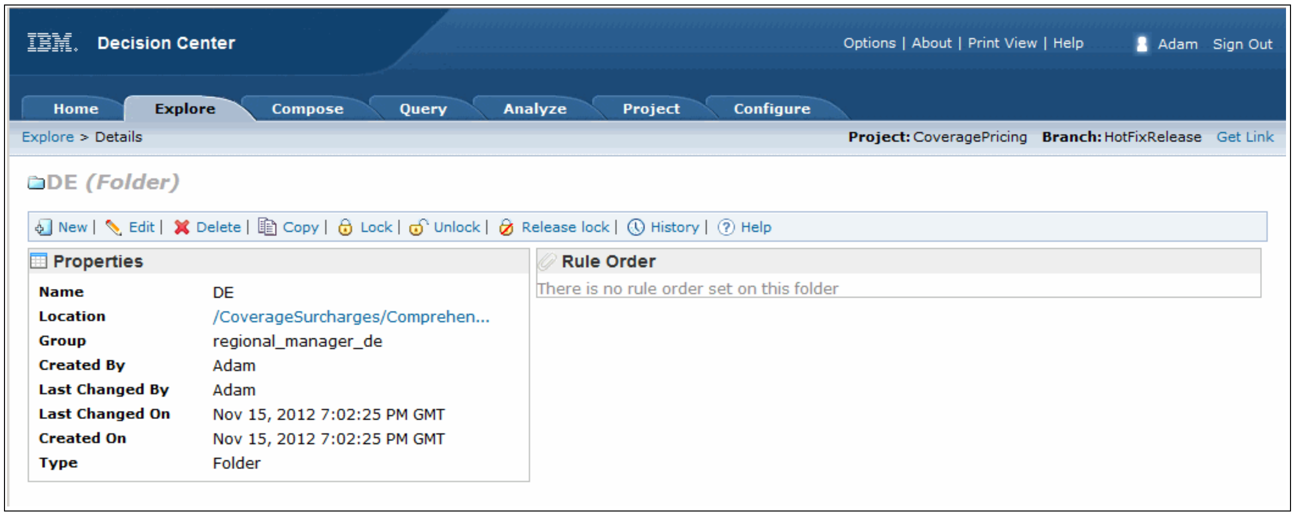
Edit Permissions

You are currently editing the permissions for the group: regional_manager_nj

Actions	PERMISSION	TYPE	PROPERTY	VALUE
☐	View	Folder	-	Group
☐	View	Smart Folder	-	Yes
☐	Create	Action Query	-	Yes
☐	Create	Action Rule	-	Yes
☐	Create	Decision Table	-	Yes
☐	Create	Decision Tree	-	Yes
☐	Create	Event Rule	-	Yes
☐	Create	Event Rule Group	-	Yes
☐	Create	Filter	-	Yes
☐	View	Action Query	-	Group
☐	View	Action Rule	-	Group
☐	View	Decision Table	-	Group
☐	View	Decision Tree	-	Group
☐	View	Event Rule	-	Group
☐	View	Event Rule Group	-	Group
☐	View	Ruleflow	-	Group
☐	View	Filter	-	Group
☐	View	Variable Set	-	Group
☐	View	Function	-	No

Figure 9-12 Group based permissions

These settings constrain the operations that a regional manager can perform to those artifacts that are marked with their group. This means that the regional folders and rules within them that are marked with a group access control are visible only to users that are members of that group. This can be applied to any of the View, Update, or Delete permissions. Consider the regional folders that are shown in Figure 9-13, each of which is marked with their own appropriate group.


IBM. Decision Center
Options | About | Print View | Help
Adam Sign Out
Home Explore Compose Query Analyze Project Configure
Explore > Details
Get Link
New | Edit | Delete | Copy | Lock | Unlock | Release lock | History | Help
Properties
Rule Order

Project: CoveragePricing **Branch:** HotFixRelease

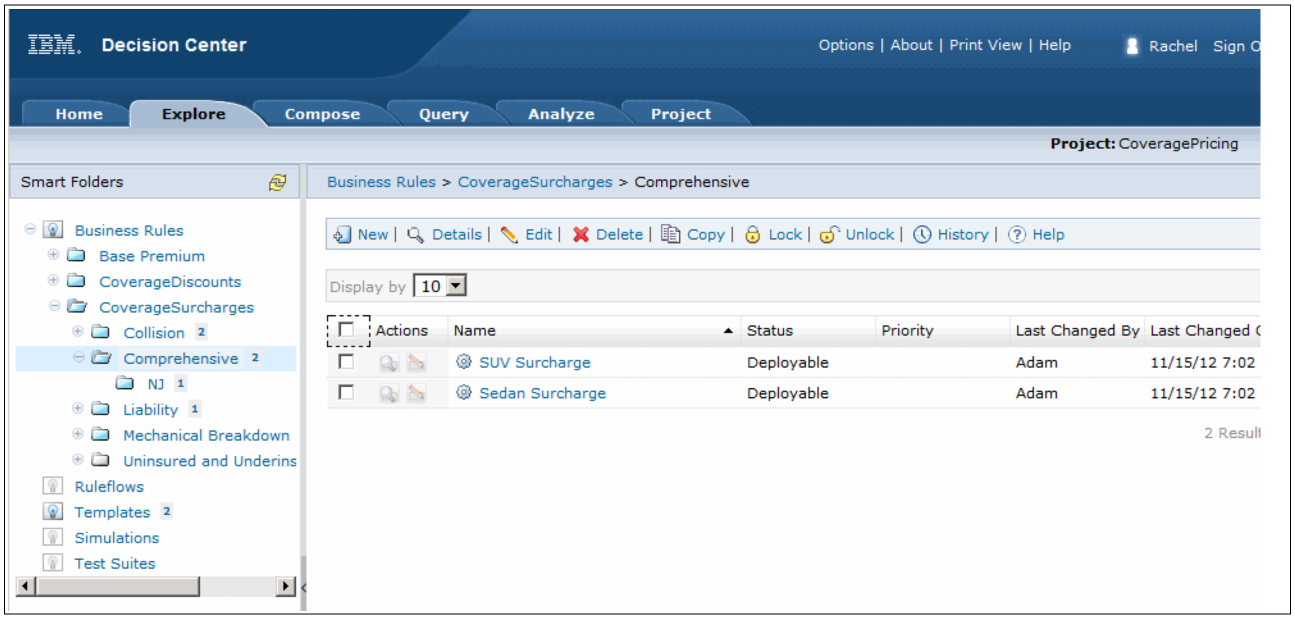
DE (Folder)

There is no rule order set on this folder

Name	DE
Location	/CoverageSurcharges/Comprehen...
Group	regional_manager_de
Created By	Adam
Last Changed By	Adam
Last Changed On	Nov 15, 2012 7:02:25 PM GMT
Created On	Nov 15, 2012 7:02:25 PM GMT
Type	Folder

Figure 9-13 Folder group properties

In this case, it shows that the DE folder is allocated to the regional_manager_de group. If Rachel, who belongs to the regional_manager_nj group, looks at this folder hierarchy, she cannot see the DE folders; she can see only the NJ folders for which she has access control. See Figure 9-14.


IBM. Decision Center
Options | About | Print View | Help
Rachel Sign O
Home
Explore
Compose
Query
Analyze
Project
Smart Folders
New | Details | Edit | Delete | Copy | Lock | Unlock | History | Help
Display by 10

Project: CoveragePricing

Business Rules > CoverageSurcharges > Comprehensive

- Business Rules
  - Base Premium
  - CoverageDiscounts
  - CoverageSurcharges
    - Collision 2
    - Comprehensive 2
      - NJ 1
    - Liability 1
    - Mechanical Breakdown
    - Uninsured and Underins
- Ruleflows
- Templates 2
- Simulations
- Test Suites

	Actions	Name	Status	Priority	Last Changed By	Last Changed (
☐		SUV Surcharge	Deployable		Adam	11/15/12 7:02
☐		Sedan Surcharge	Deployable		Adam	11/15/12 7:02

2 Result

Figure 9-14 Group based folder visibility

In this case, as Rachel is only a member of the regional_manager_nj group; she cannot see any of the other regional folders or rules. Shared folders (such as CoverageSurcharges/Comprehensive) are granted readonly access by using the rtsuser group, allowing Rachel to see but not modify these folders or the rules in them.

9.4 Pure Business Change process example

This section looks at how a planned release is set up and how you can define the goals that can be taken into account in each release iteration.

This scenario shows how the LOB users can collaborate and change the Pricing rules to meet the business needs by following the Pure Business Change Process outlined in Chapter 8, “Processes” on page 95.

Barbara is performing her monthly analysis of insurance profitability and notices that the organization is suffering losses in New Jersey, especially for comprehensive coverage for sedan cars. Based on this information, Barbara asks Paul to include changes to resolve this situation in the next planned monthly release of the pricing policies.

This example concentrates on how the NJ Comprehensive Pricing can be improved as part of a planned monthly release process. It walks through a number of typical release activities that are relevant to a planned release where the changes are made by Rachel, a regional manager that does not understand the technicalities of authoring business rules. It also looks at typical validation activities that are used to ascertain that the changes were made correctly and that the business goals are met.

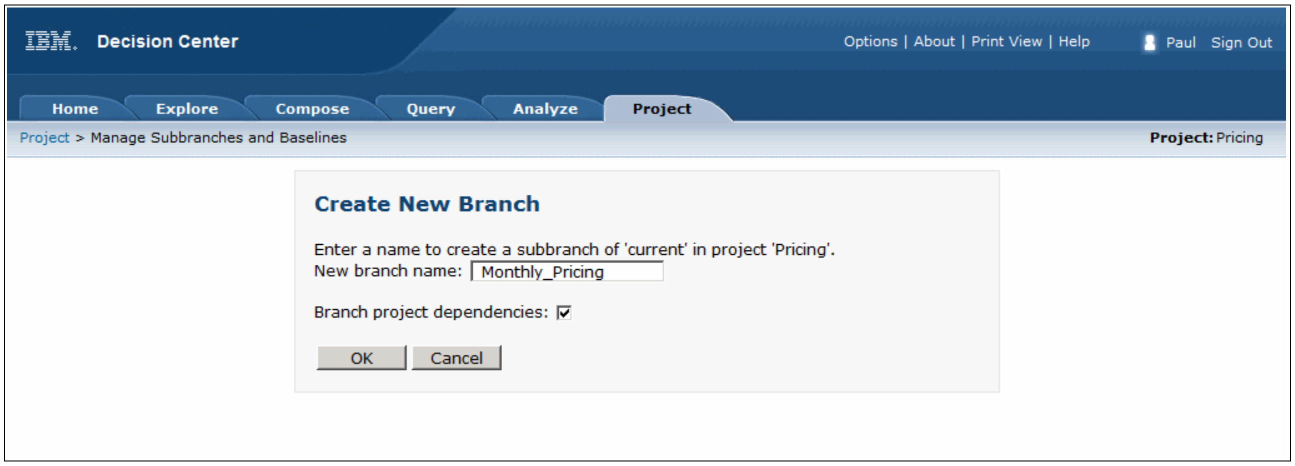
In this case, it shows how Barbara can run simulations against the new release to ensure that the NJ profitability is improved. It also looks at more IT-centric validation activities where testing for performance and interface correctness is undertaken by Ivan. Finally, with the release changes complete and validated, we show how Adam can deploy the release into the production systems according to his IT operating procedures.

9.4.1 Release definition

This section describes the steps to create and define a monthly planned release that takes the insight that is obtained from day to day operations into account. As this is a planned release, the release itself and activities within it are reused with each release cycle. This section first defines setting up the branch structure for the release, and then shows how the requirements for a particular release can be configured so that the team knows what work they must perform.

Release creation

Creation of the Monthly_Pricing release is undertaken by Paul the release manager. In this case, he creates the release as a branch based off the main Pricing rule project hierarchy. See Figure 9-15.

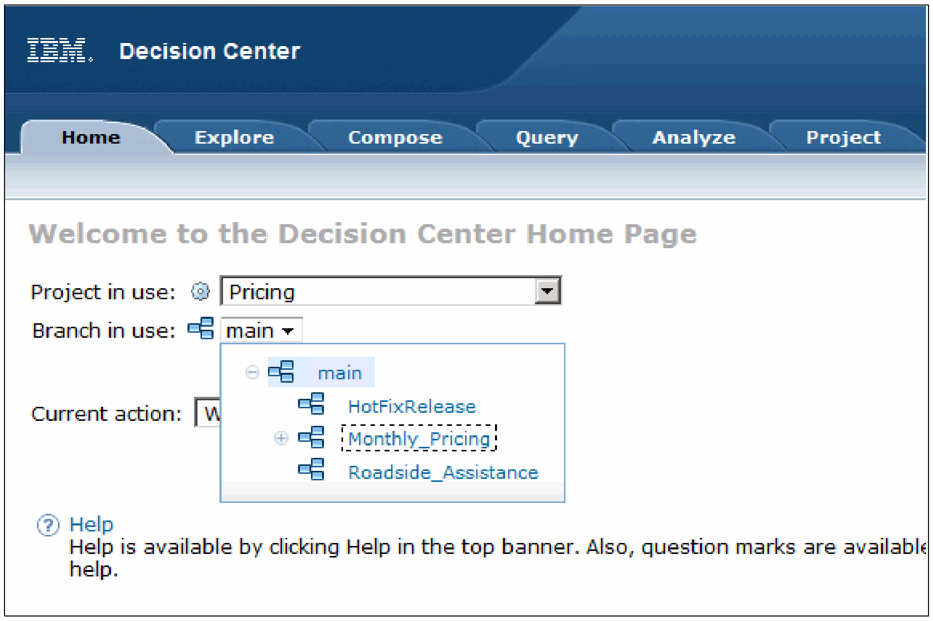


Enter a name to create a subbranch of 'current' in project 'Pricing'.
New branch name: Monthly_Pricing

Branch project dependencies: ✓

Figure 9-15 Creating a branch for the Monthly_Pricing Release

This action produces a branch in which Paul and his team can work. By branching the project dependencies, each project in the hierarchy also has a branch with the same name. Now that this release is created, any users that have access to projects in the branch can now navigate to that branch to work on the release content. See Figure 9-16.



Welcome to the Decision Center Home Page

Project in use: Pricing

Branch in use: main

Help
Help is available by clicking Help in the top banner. Also, question marks are availabl
help.

Figure 9-16 Pricing Release selection in the Enterprise Console

A similar selection is possible in the Business Console, which allows any user to scope their work to the particular release. See Figure 9-17.

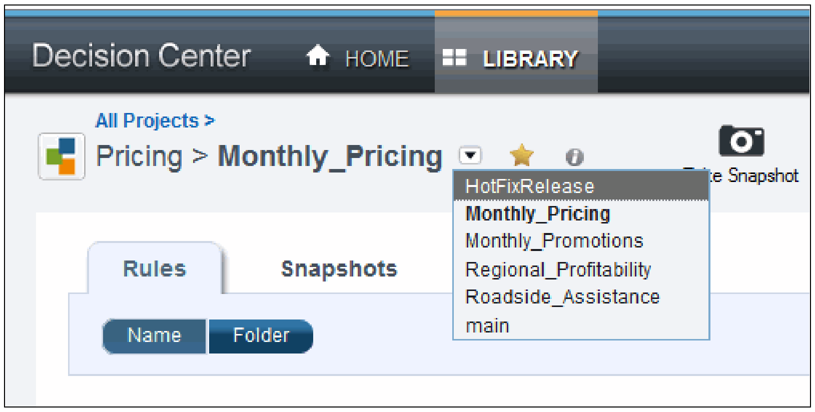
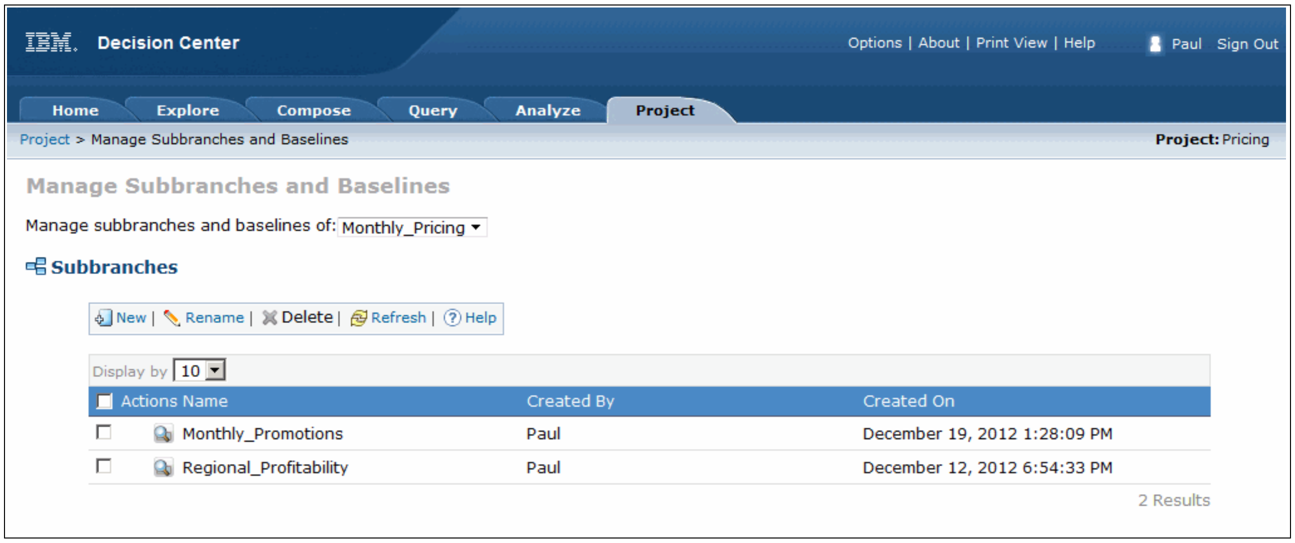


Figure 9-17 Pricing Release selection in the Business Console

Change activity definition

Having defined the release, Paul can now define the Change activities. In this case, he defines a Change activity for the New Jersey Comprehensive Sedan Profitability changes called Regional_Profitability and another change activity that is called Monthly_Promotions to allow the Global Pricing adjustments to be modified. This is undertaken in the Enterprise console by subbranches of the Monthly_Pricing release, as shown in Figure 9-18.


IBM. Decision Center
Options | About | Print View | Help
Paul Sign Out
Home
Explore
Compose
Query
Analyze
Project
Project > Manage Subbranches and Baselines
Project: Pricing
Manage Subbranches and Baselines
Subbranches
New | Rename | Delete | Refresh | Help
Display by 10

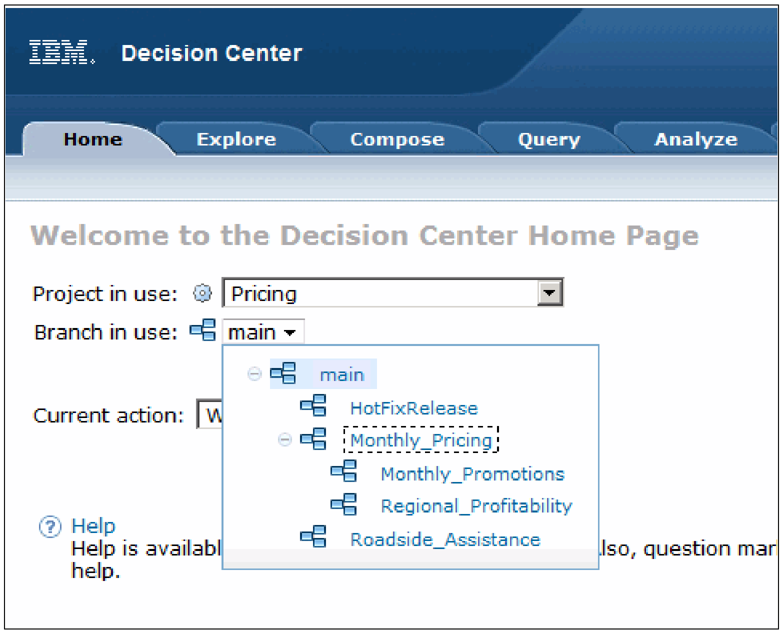
Manage subbranches and baselines of: Monthly_Pricing

Actions Name	Created By	Created On
Monthly_Promotions	Paul	December 19, 2012 1:28:09 PM
Regional_Profitability	Paul	December 12, 2012 6:54:33 PM

2 Results

Figure 9-18 Defining change activities as subbranches of releases

When you select projects in the Enterprise Console, this branching hierarchy is now available to select the activity to work in, as shown in Figure 9-19.



Project in use: Pricing
Branch in use: main
Current action: W

Help is availabl ... lso, question mar help.

Figure 9-19 Selecting change activities in the Enterprise Console

Selection of the change activity (and the releases) can also be made in the business console for any of the projects that are used in the decision service, as shown in Figure 9-20.

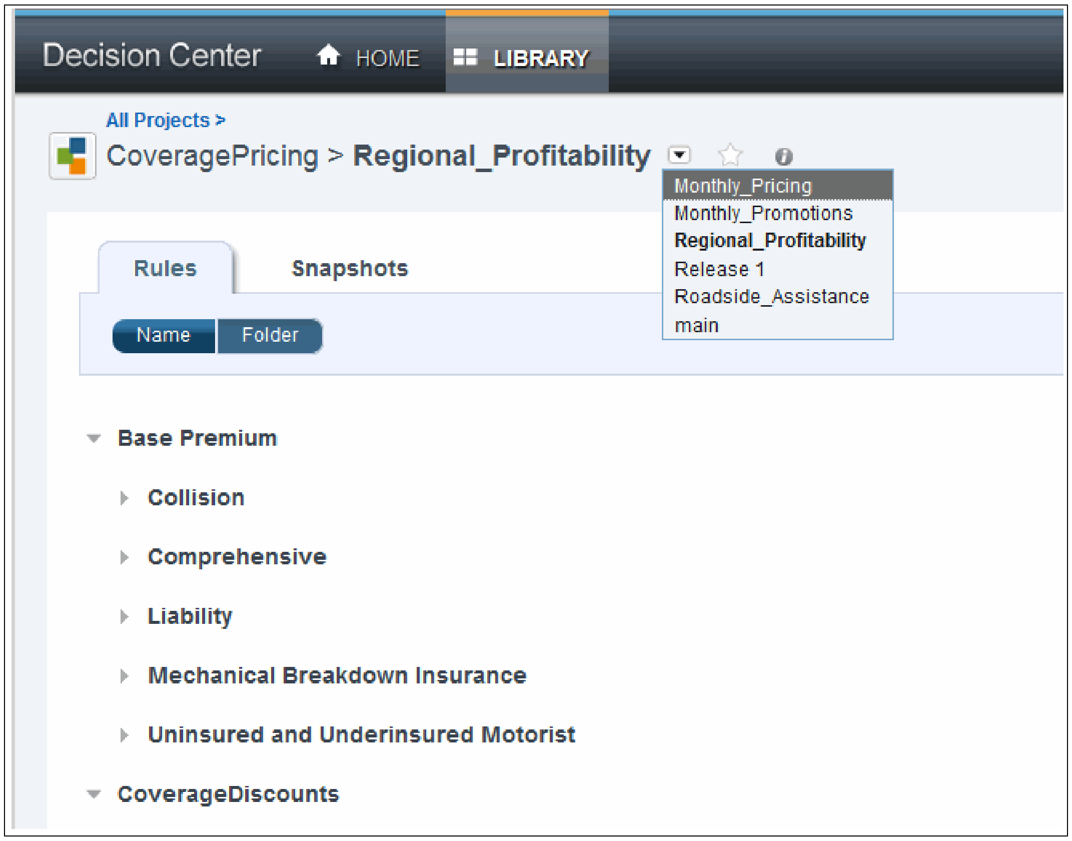


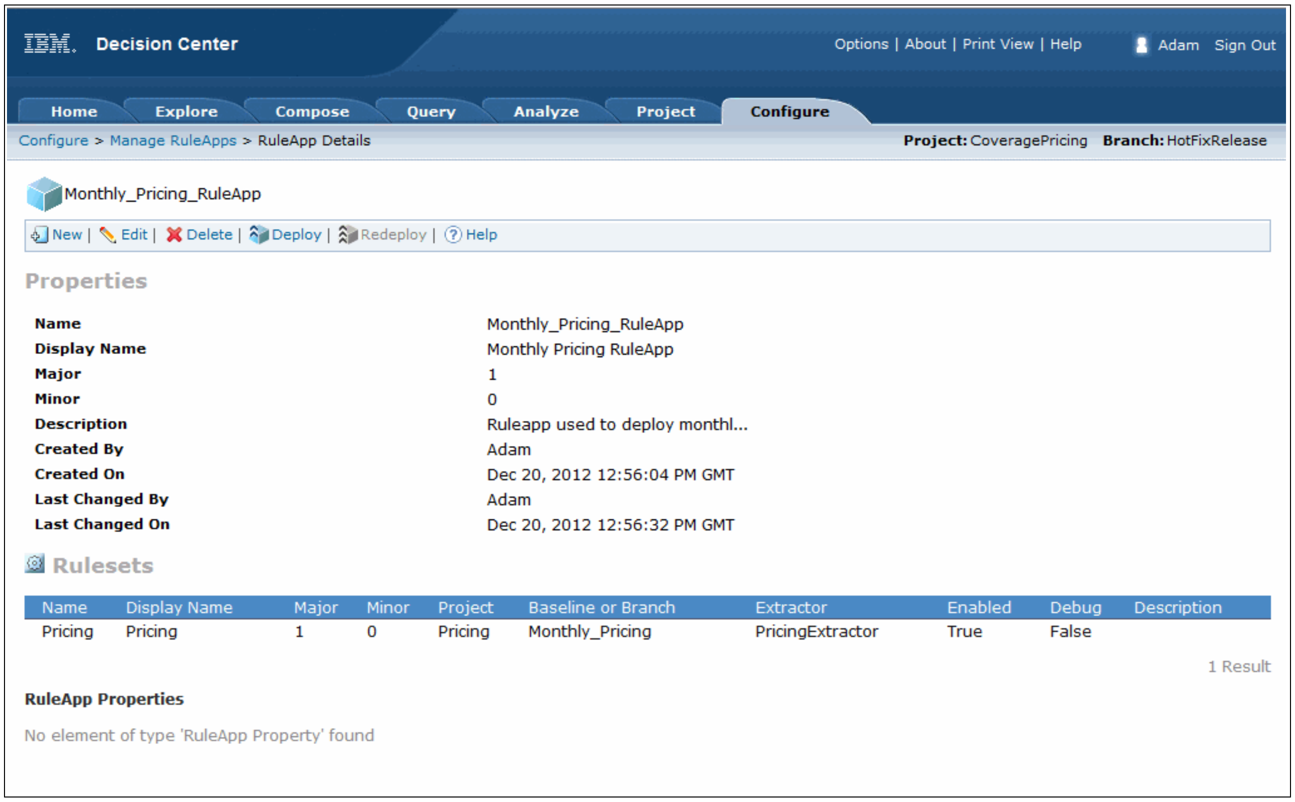
Figure 9-20 Selection of change activities in the Business Console

Having set up this structure, each Monthly pricing release can now be managed by independently performing the changes in the change activities and then, when they are merged back into the release, validation can be performed within the release.

Definition of Release Deployment RuleApp

Decision services are deployed to decision servers through a RuleApp. The RuleApp definition defines the projects that are included in the deployment, the branches from which the rules are extracted, and the versioning that is assigned to the various rule sets that are deployed.

For these monthly pricing releases, Adam creates a Monthly_Pricing_RuleApp that can be used to consistently deploy each monthly release. This RulesApp is established by using the Manage RuleApps capability of the Enterprise Console, as shown in Figure 9-21.


IBM. Decision Center
Options | About | Print View | Help
Adam Sign Out
Home
Explore
Compose
Query
Analyze
Project
Configure
Configure > Manage RuleApps > RuleApp Details
Project: CoveragePricing Branch: HotFixRelease
Monthly_Pricing_RuleApp
New | Edit | Delete | Deploy | Redeploy | Help

Properties

Name	Monthly_Pricing_RuleApp
Display Name	Monthly Pricing RuleApp
Major	1
Minor	0
Description	Ruleapp used to deploy monthl...
Created By	Adam
Created On	Dec 20, 2012 12:56:04 PM GMT
Last Changed By	Adam
Last Changed On	Dec 20, 2012 12:56:32 PM GMT

Rulesets

Name	Display Name	Major	Minor	Project	Baseline or Branch	Extractor	Enabled	Debug	Description
Pricing	Pricing	1	0	Pricing	Monthly_Pricing	PricingExtractor	True	False	

1 Result

RuleApp Properties

No element of type 'RuleApp Property' found

Figure 9-21 Monthly Pricing RuleApp definition

This RulesApp now allows each new monthly release to be deployed consistently to the appropriate Decision Server environment and can also be used to generate a configuration item that can be managed in the IT Operational Configuration Management solution in use by the organization.

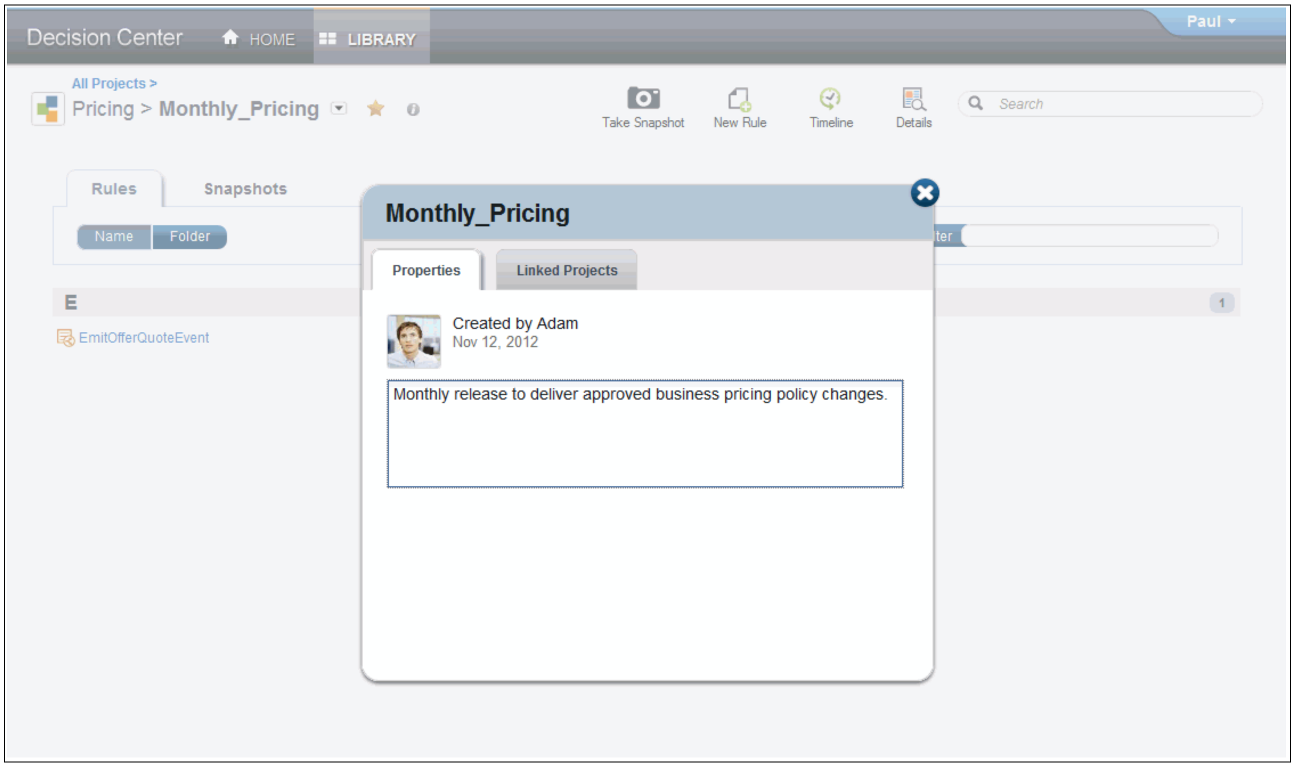
9.4.2 Defining release goals

This section describes the steps that are required to define the goals for the release.

Release goals definition

The next task for Paul is to define the goals for the release. This task can be undertaken in Decision Center Business console. IBM Operational Decision Manager provides many ways of communicating the goals to the release participants.

As the Monthly_Pricing release is reused from one month to the next, the description should reflect the overall goals and purpose of the release rather than the goals of a specific monthly release, as shown in Figure 9-22.



Created by Adam
Nov 12, 2012

Monthly release to deliver approved business pricing policy changes.

Figure 9-22 Release Goal Definition in Project details

This information is then available to any users that are examining this information in the root project of the release.

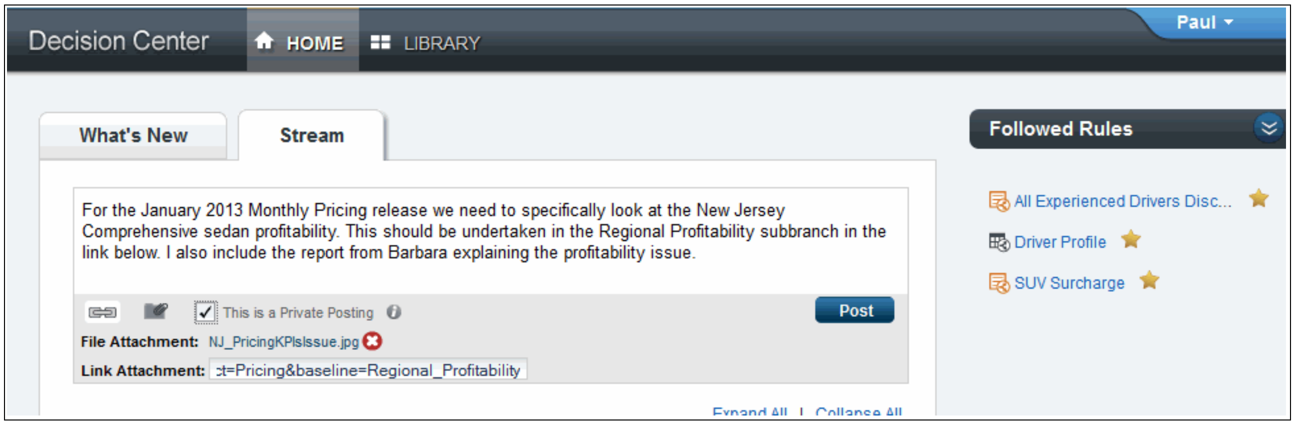
This approach can also be used in each of the Change Activity branches to define the purpose of that change activity.

Because you want to reuse this release from one month to the next, the goals for each individual monthly release are best defined by using the activity streams features.

Collaboration using activity streams

IBM Operation Decision Manager also supports activity streams, which are free format postings that can be used to communicate requirements to the rest of the team.

In Figure 9-23, Paul defines the requirements for the change activity by using the stream.



For the January 2013 Monthly Pricing release we need to specifically look at the New Jersey Comprehensive sedan profitability. This should be undertaken in the Regional Profitability subbranch in the link below. I also include the report from Barbara explaining the profitability issue.

✓ This is a Private Posting

Post

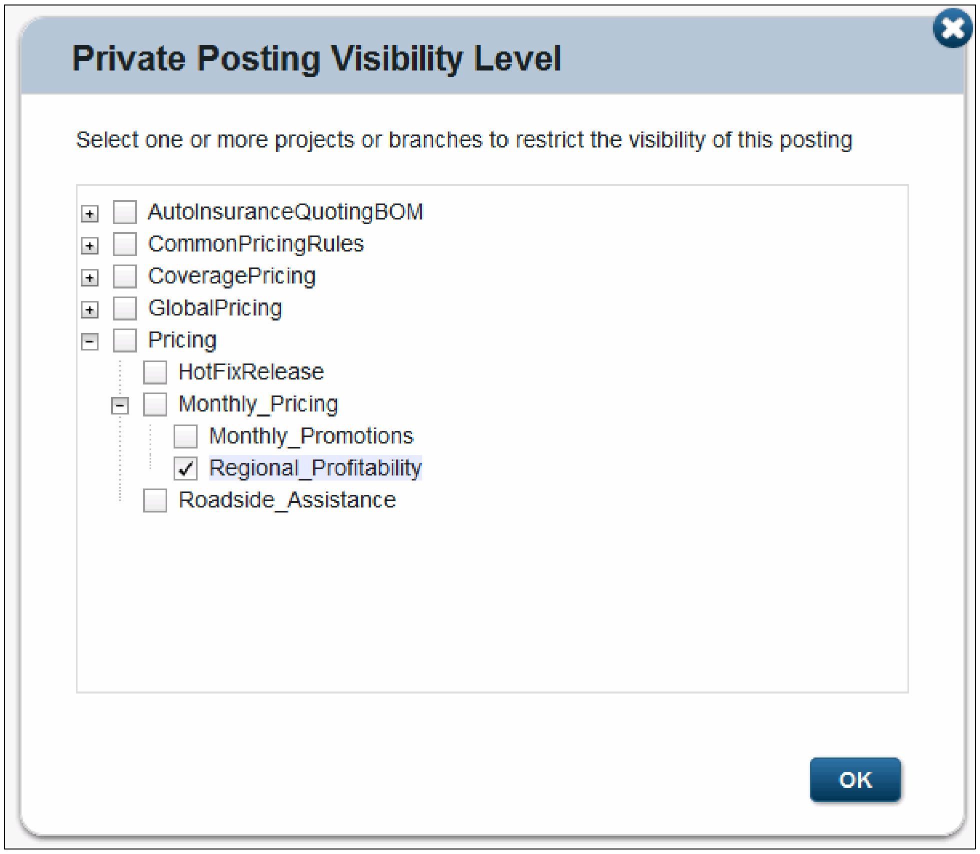
File Attachment: NJ_PricingKPIsIssue.jpg

Link Attachment: ɔt=Pricing&baseline=Regional_Profitability

All Experienced Drivers Disc...
Driver Profile
SUV Surcharge

Figure 9-23 Activity stream that shows the definition of requirements

By scoping this change activity as a private posting, it is picked up only by those participants that are following the appropriate selected branches, as shown in Figure 9-24.



Private Posting Visibility Level

Select one or more projects or branches to restrict the visibility of this posting

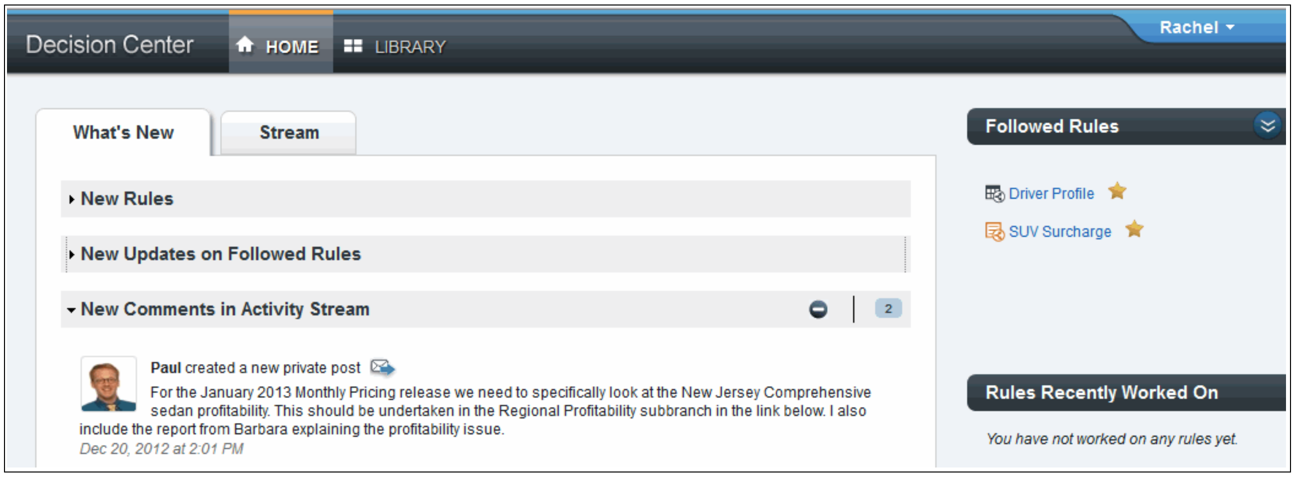
- AutoInsuranceQuotingBOM
- CommonPricingRules
- CoveragePricing
- GlobalPricing
- Pricing
  - HotFixRelease
  - Monthly_Pricing
    - Monthly_Promotions
    - ✓ Regional_Profitability
  - Roadside_Assistance

OK

Figure 9-24 Private Posting Visibility

Requirements notification

When Rachel, the New Jersey regional manager logs in to the Business Console, she receives a notification immediately in her What's New tab. See Figure 9-25.



Paul created a new private post

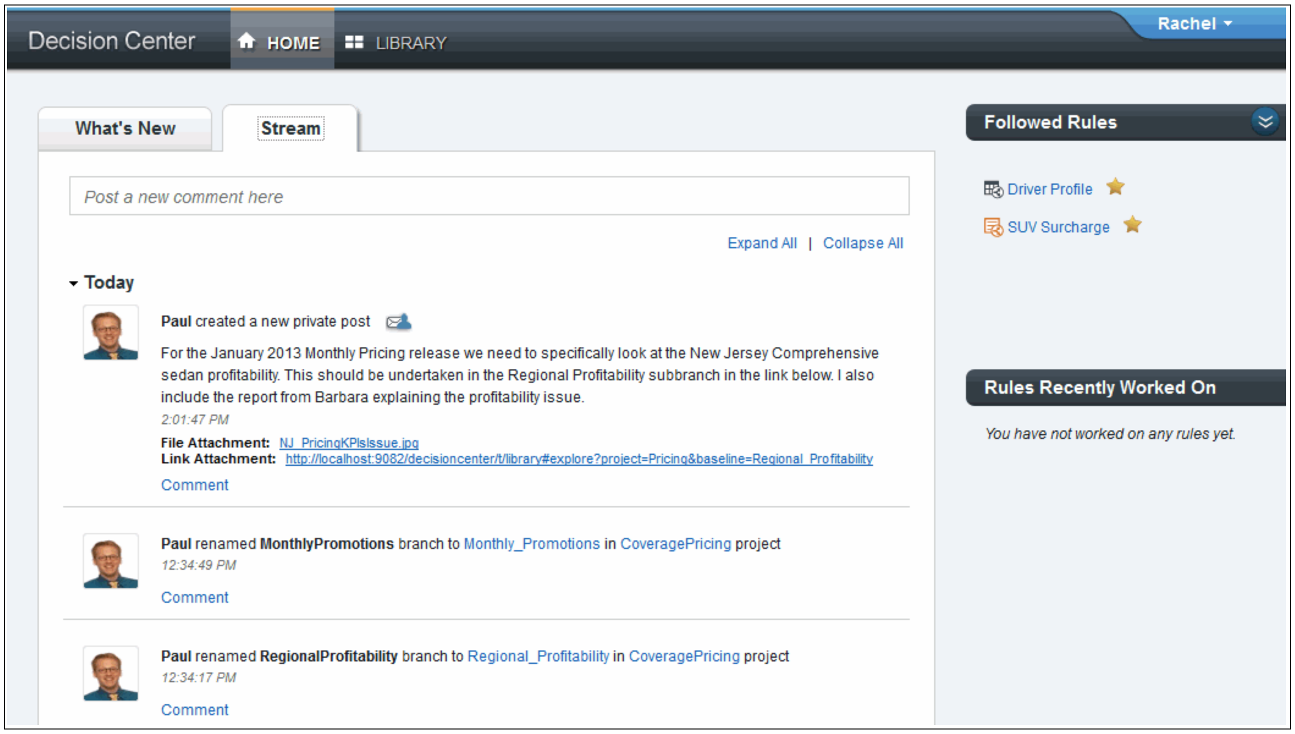
For the January 2013 Monthly Pricing release we need to specifically look at the New Jersey Comprehensive sedan profitability. This should be undertaken in the Regional Profitability subbranch in the link below. I also include the report from Barbara explaining the profitability issue.

Dec 20, 2012 at 2:01 PM

You have not worked on any rules yet.

Figure 9-25 What's New tab showing Regional Profitability requirements

By following the link to the post, she can see the details of the activity stream posting. See Figure 9-26.



Paul created a new private post

For the January 2013 Monthly Pricing release we need to specifically look at the New Jersey Comprehensive sedan profitability. This should be undertaken in the Regional Profitability subbranch in the link below. I also include the report from Barbara explaining the profitability issue.

2:01:47 PM

File Attachment: NJ_PricingKPIsIssue.jpg
Link Attachment: http://localhost:9082/decisioncenter/t/library#explore?project=Pricing&baseline=Regional_Profitability

Paul renamed MonthlyPromotions branch to Monthly_Promotions in CoveragePricing project

12:34:49 PM

Paul renamed RegionalProfitability branch to Regional_Profitability in CoveragePricing project

12:34:17 PM

You have not worked on any rules yet.

Figure 9-26 Viewing the activity stream posting

From here, Rachel can view the report or navigate to any links within the posting. The stream also includes events such as creation or modification of release and change activity branches, which can be used to navigate straight to the branch in which the change activities are to be carried out, as shown in Figure 9-27.

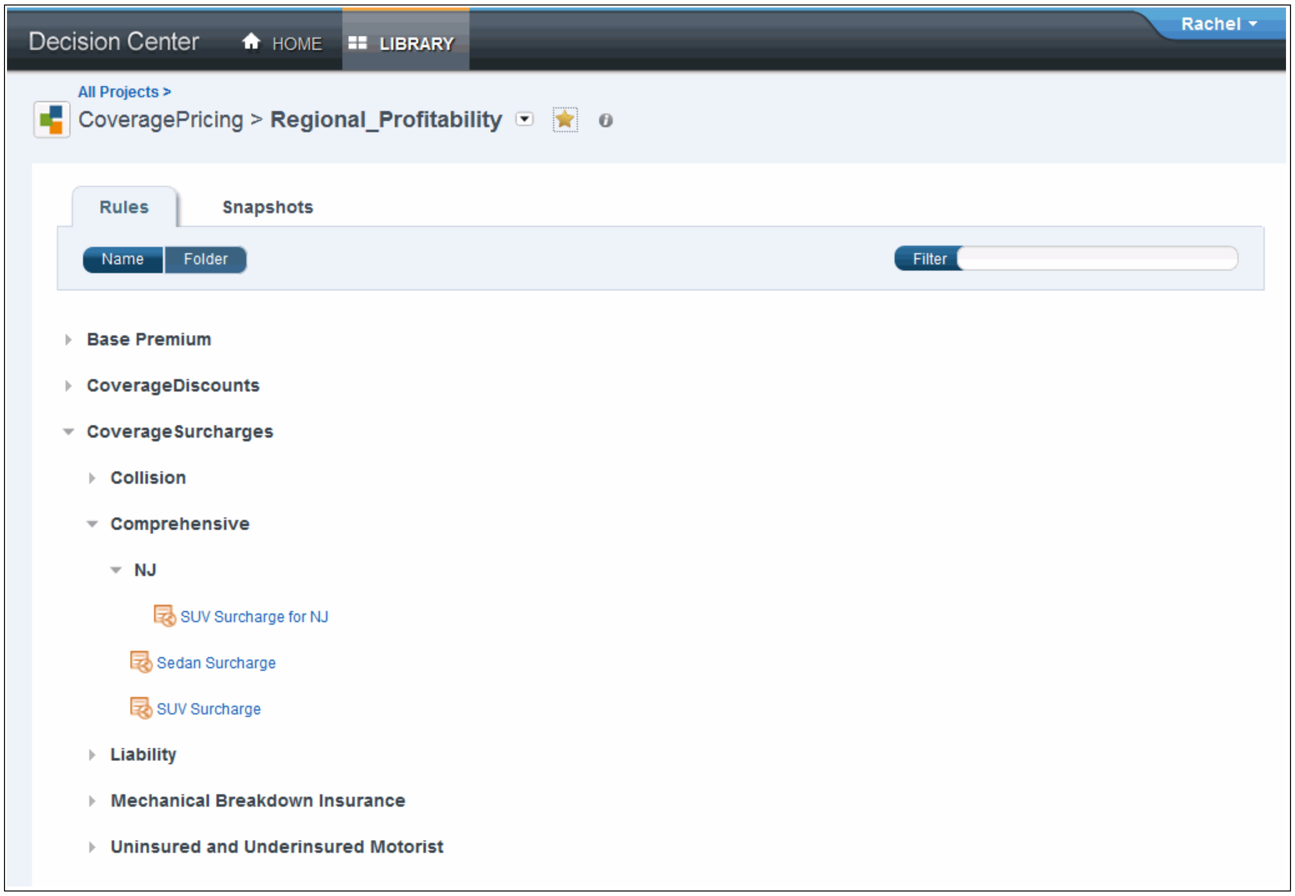


Figure 9-27 Change activity branch of Coverage Pricing project

At this point in the monthly release lifecycle, Rachel now has a good understanding of what her changes must cover and where they should be undertaken.

9.4.3 Templated rule changes

The Regional_Profitability change activity is intended to allow regional managers to update their local pricing policies to deal with current local requirements. The Regional managers performing these updates do not have a great deal of experience with rules, so their changes are guided by the usage of rule templates.

This section shows how decision center rule templates can be used to constrain the changes that are made by the regional managers and thus minimize the validation activities that are required to ensure that the changes are implemented correctly.

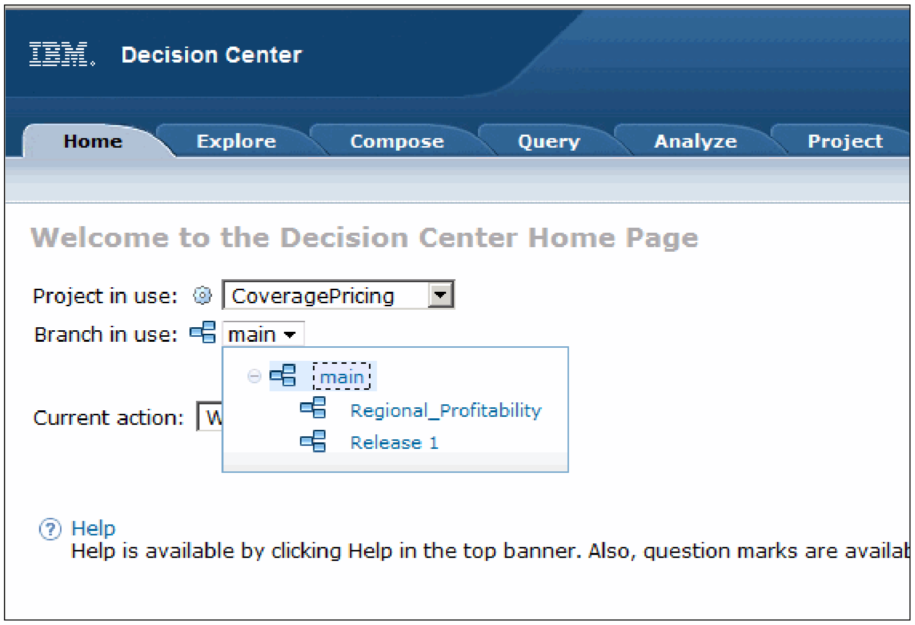
“Requirements notification” on page 124 showed how Rachel the regional rule manager identifies the requirements for her changes and the branch in which the changes must be made.

The steps in the following sections show how Rachel can use the Decision Center Enterprise console templates.

Selecting the Regional_Profitability change activity branch for Coverage pricing

To accomplish this task, Rachel completes the following steps:

1. After she logs in to the Enterprise Console, Rachel selects the Regional_Profitability branch of the Coverage pricing project, as shown in Figure 9-28.



Welcome to the Decision Center Home Page

Project in use: CoveragePricing

Branch in use: main

main
Regional_Profitability
Release 1

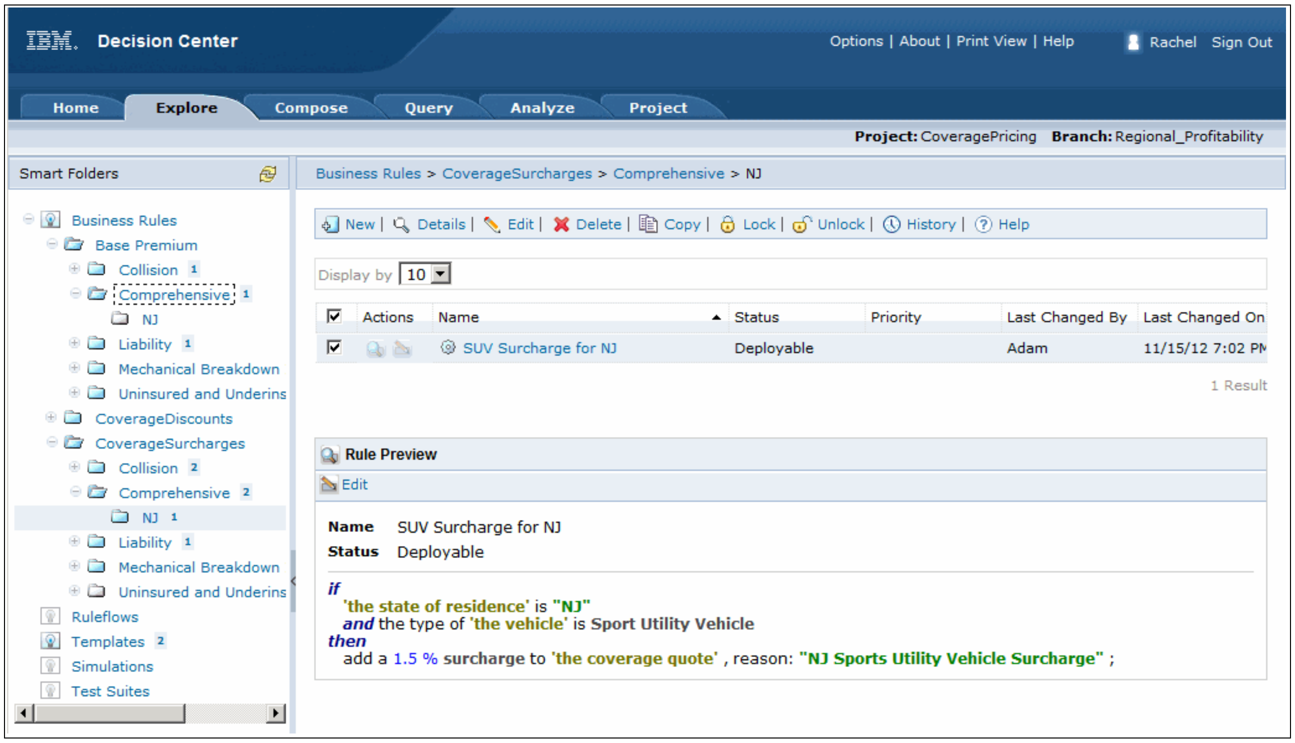
Current action: W

Help
Help is available by clicking Help in the top banner. Also, question marks are availab

Figure 9-28 Selecting the Regional_profitability change activity branch for the Coverage pricing project

Rachel does not have access to other branches, such as the Monthly Pricing Release or other change activities because of the access control permissions that were set up at the project and branch level.

2. After she selects the correct branch, she can see the rules for which she is authorized, as shown in Figure 9-29.



Project: CoveragePricing **Branch:** Regional_Profitability

Actions	Name	Status	Priority	Last Changed By	Last Changed On
	SUV Surcharge for NJ	Deployable		Adam	11/15/12 7:02 PM

1 Result

Name SUV Surcharge for NJ
Status Deployable

```
if
    'the state of residence' is "NJ"
    and the type of 'the vehicle' is Sport Utility Vehicle
then
    add a 1.5 % surcharge to 'the coverage quote' , reason: "NJ Sports Utility Vehicle Surcharge" ;
```

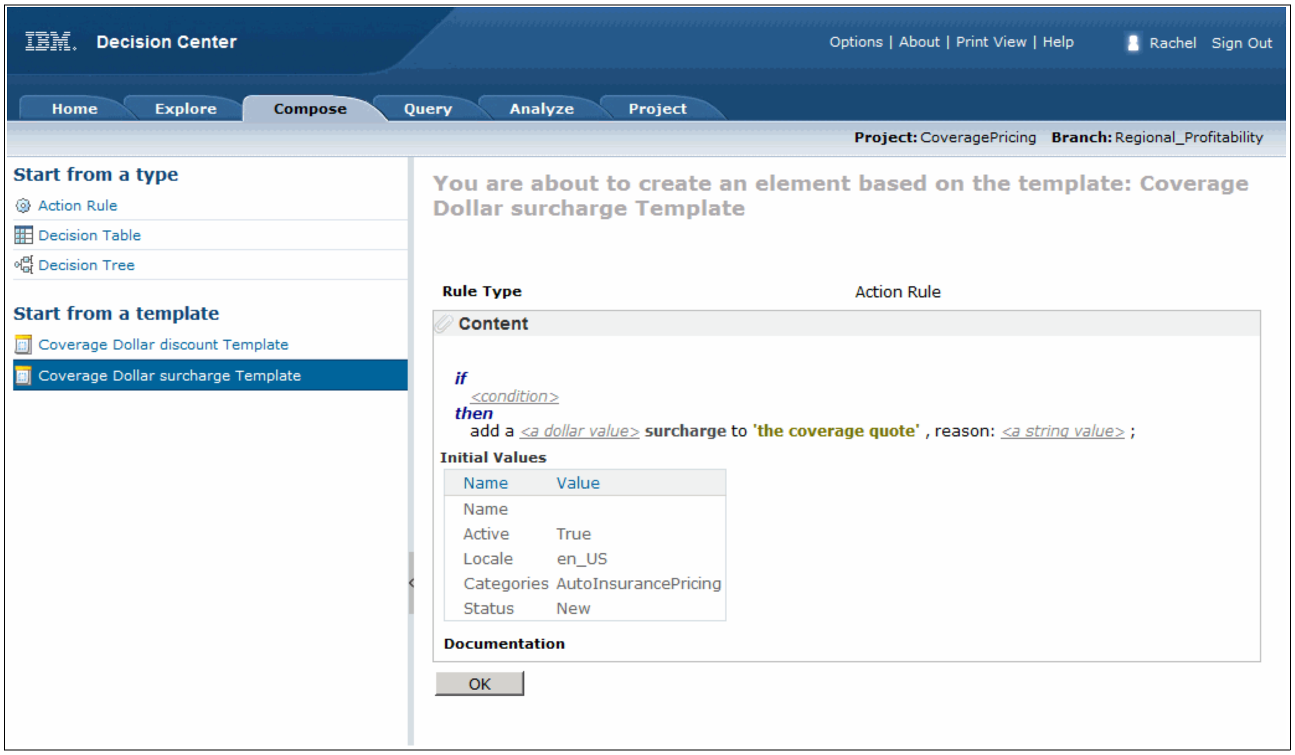
Figure 9-29 Explore tab in the Enterprise Console

Creating a rule using a rule template

Having decided that she must create a surcharge for NJ sedan cars, Rachel decides to create a rule. The usage of the template makes it easy for Rachel to write rules.

She completes the following steps:

1. Rachel selects **New** from the CoverageSurcharges / Comprehensive / NJ folder and she is offered a number of options, including some rule templates that are specifically designed for surcharges and discounts, as shown in Figure 9-30.



Project: CoveragePricing **Branch:** Regional_Profitability

Start from a type
- Action Rule
- Decision Table
- Decision Tree

Start from a template
- Coverage Dollar discount Template
- Coverage Dollar surcharge Template

You are about to create an element based on the template: Coverage Dollar surcharge Template

Rule Type	Action Rule

Content

```
if
  <condition>
then
  add a <a dollar value> surcharge to 'the coverage quote' , reason: <a string value> ;
```

Initial Values

Name	Value
Name	
Active	True
Locale	en_US
Categories	AutoInsurancePricing
Status	New

Documentation

OK

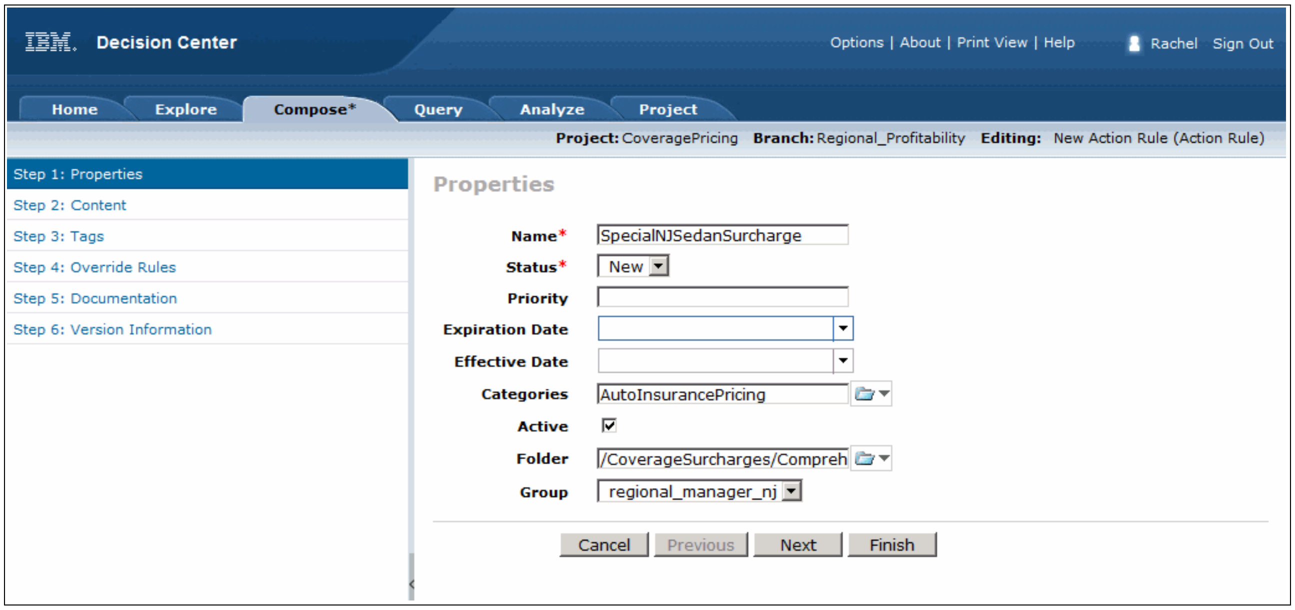
Figure 9-30 Coverage Dollar Surcharge template

2. Selecting the Coverage Dollar surcharge template, Rachel can see that she must provide three values:
   - The conditions under which the surcharge is to be levied.
   - The amount of the surcharge in dollars.
   - A reason that is passed to the customer if the surcharge is levied.
3. After she clicks **OK**, the rule template wizard takes Rachel through the steps for establishing the new rule.

Rule definition

Rachel completes the following steps to define rules:

1. She establishes key properties for the rules, as shown in Figure 9-31 on page 129.


Decision Center

Project: CoveragePricing Branch: Regional_Profitability Editing: New Action Rule (Action Rule)

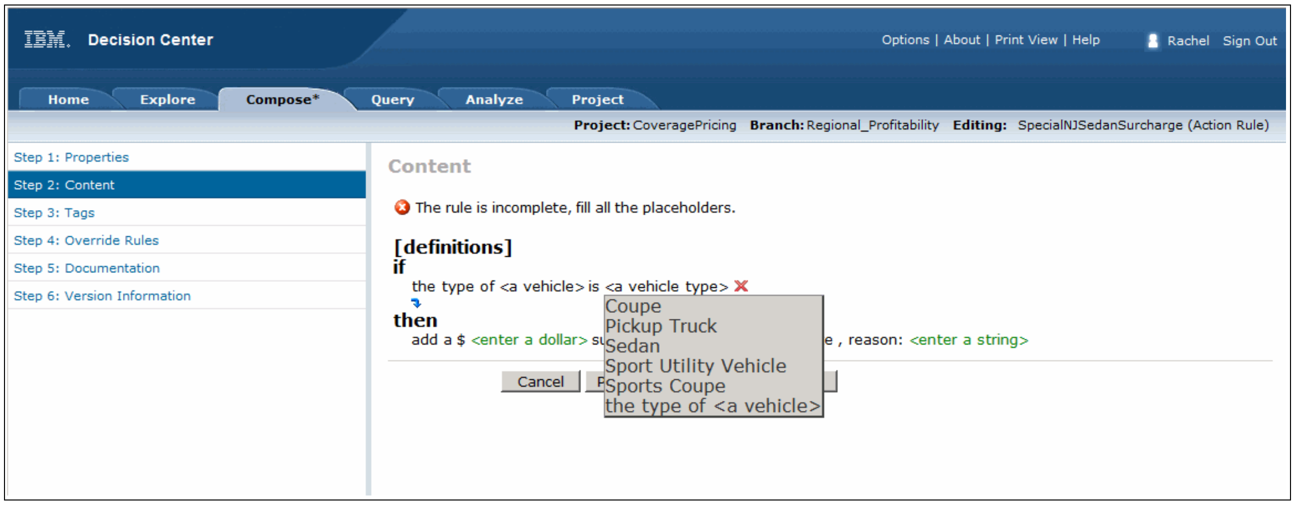
Properties

Name*	SpecialNJSedanSurcharge
Status*	New
Priority	
Expiration Date	
Effective Date	
Categories	AutoInsurancePricing
Active	✓
Folder	/CoverageSurcharges/Compreh
Group	regional_manager_nj

Figure 9-31 Step 1 - SpecialNJSedanSurcharge properties

In this step, the essential features and the name for the rules are defined. The status of the rule defaults to New. This status is an example of metadata that can be attached to rules that allow the individual rules in a release or change activity to be systematically managed. Validation activities can search based on this metadata, and the status field historically has been used as the basis for tracking the lifecycle of individual rule artifacts. In the past, the status of individual artifacts needed to be managed manually, but ideally this status should be managed automatically as part of the change or validation activity actions.

2. When Rachel selects a placeholder in the rule that is produced, she is offered options to select from that are drawn from the vocabulary and the constraints that are defined in the template. For the simple value fields, she just enters the appropriate numeric or textual value, as shown in Figure 9-32.


Decision Center

Project: CoveragePricing Branch: Regional_Profitability Editing: SpecialNJSedanSurcharge (Action Rule)

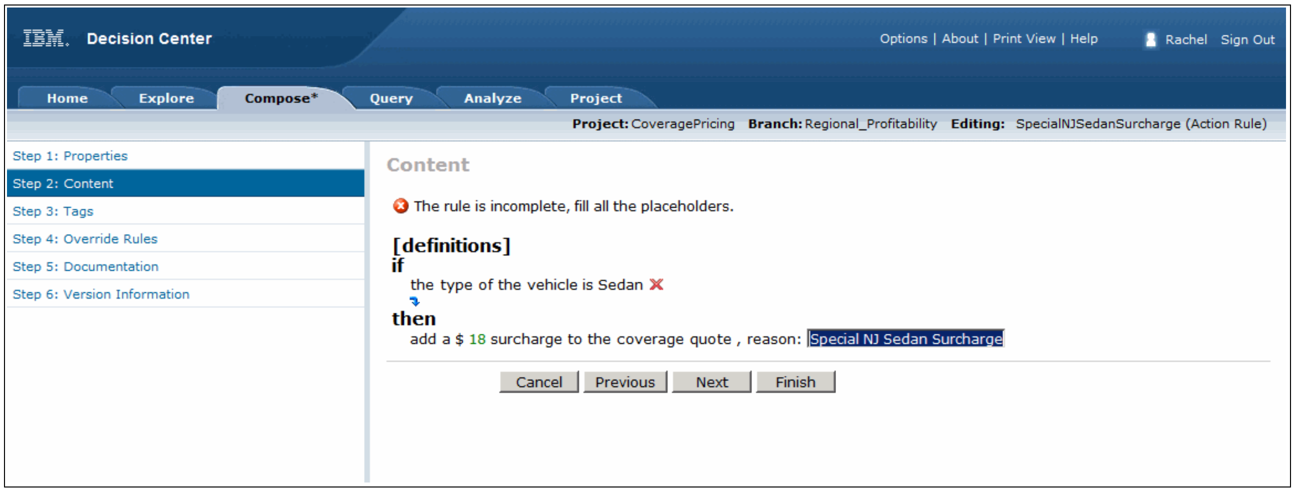
Content

The rule is incomplete, fill all the placeholders.

```
[definitions]
if
  the type of <a vehicle> is <a vehicle type>
then
  add a $ <enter a dollar> su                    e , reason: <enter a string>
```

Figure 9-32 Step 2 - using the auto completion to select from valid content

Figure 9-33 shows the results of the selection for vehicle type and that the dollar amount field is complete. Because there is still one field to complete (“reason”), an error is shown at the top of the rule.



Project: CoveragePricing **Branch:** Regional_Profitability **Editing:** SpecialNJSedanSurcharge (Action Rule)

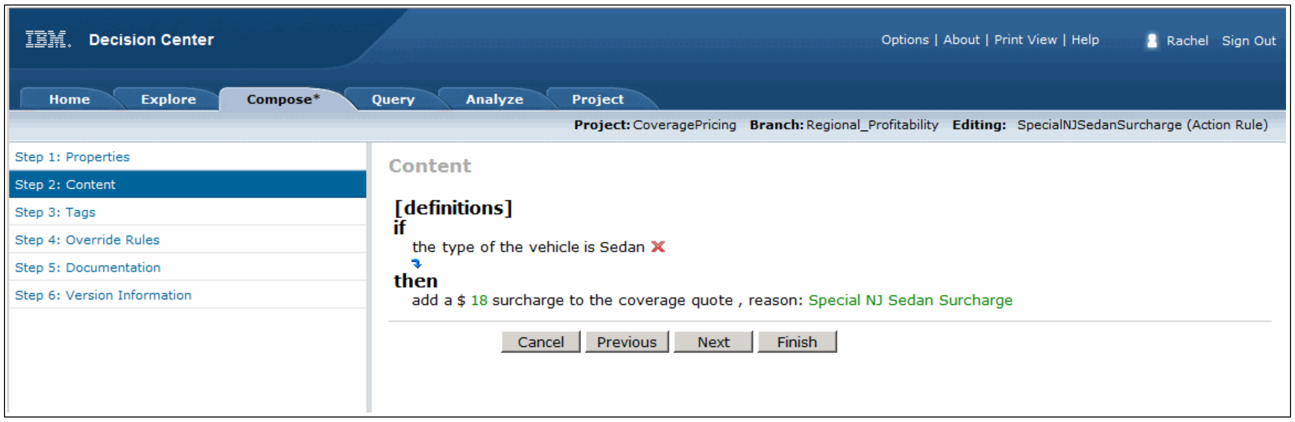
Content

The rule is incomplete, fill all the placeholders.

```
[definitions]
if
   the type of the vehicle is Sedan
then
   add a $ 18 surcharge to the coverage quote , reason: Special NJ Sedan Surcharge
```

Figure 9-33 Step 2 - completing values in the placeholders

3. When all the placeholders are completed, the errors disappear and Rachel can click **Next** to continue to document her changes or click **Finish** to complete the rule creation. See Figure 9-34.



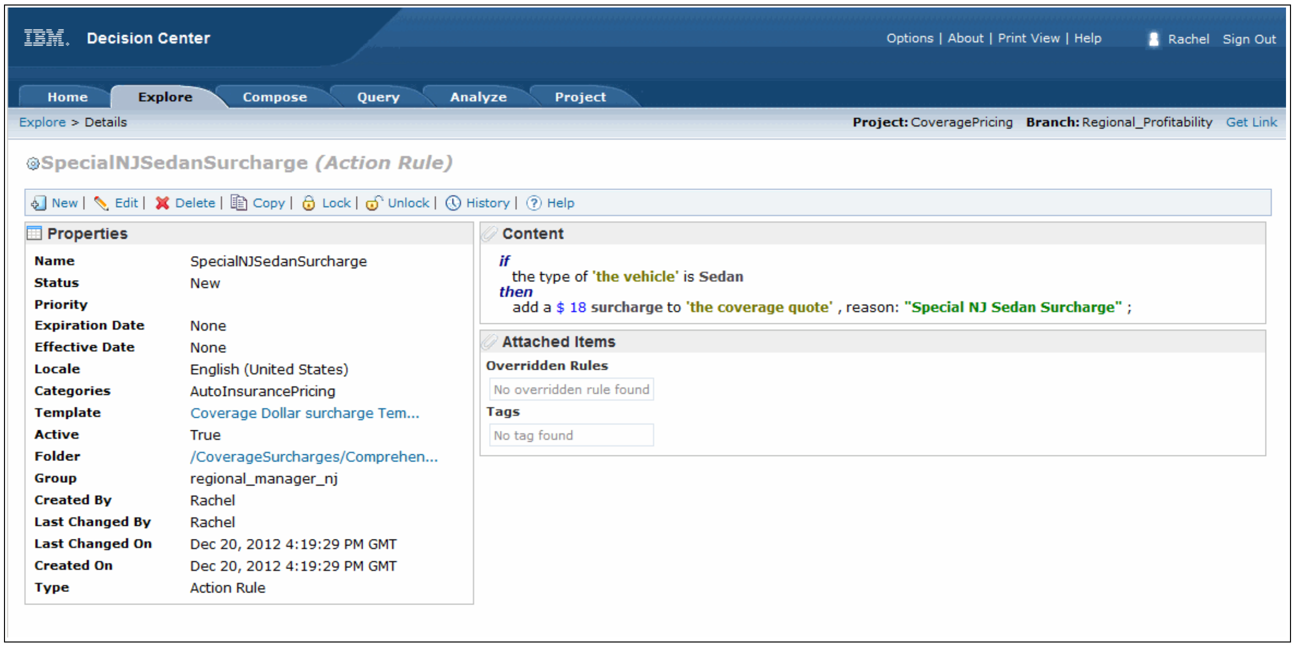
Project: CoveragePricing **Branch:** Regional_Profitability **Editing:** SpecialNJSedanSurcharge (Action Rule)

Content

```
[definitions]
if
   the type of the vehicle is Sedan
then
   add a $ 18 surcharge to the coverage quote , reason: Special NJ Sedan Surcharge
```

Figure 9-34 Step 3 - completing the rules content.

4. The remaining steps allow rule and version documentation to be added as well as more sophisticated features, such as tagging the rule with metadata, which allows it be filtered in the rule flows, or specifying other rules that this should replace. We do not cover these advanced features in this book. After you finish the rule, the final content can be displayed or edited again in the console, as shown in Figure 9-35.


IBM. Decision Center
Options | About | Print View | Help
Rachel Sign Out
Home
Explore
Compose
Query
Analyze
Project
Explore > Details
Project: CoveragePricing Branch: Regional_Profitability Get Link
SpecialNJSedanSurcharge (Action Rule)
New | Edit | Delete | Copy | Lock | Unlock | History | Help

Properties

Name	SpecialNJSedanSurcharge
Status	New
Priority	
Expiration Date	None
Effective Date	None
Locale	English (United States)
Categories	AutoInsurancePricing
Template	Coverage Dollar surcharge Tem...
Active	True
Folder	/CoverageSurcharges/Comprehen...
Group	regional_manager_nj
Created By	Rachel
Last Changed By	Rachel
Last Changed On	Dec 20, 2012 4:19:29 PM GMT
Created On	Dec 20, 2012 4:19:29 PM GMT
Type	Action Rule

Content

```
if
  the type of 'the vehicle' is Sedan
then
  add a $ 18 surcharge to 'the coverage quote' , reason: "Special NJ Sedan Surcharge" ;
```

Attached Items

Overridden Rules

No overridden rule found

Tags

No tag found

Figure 9-35 Completed SpecialNJSedanSurcharge rule

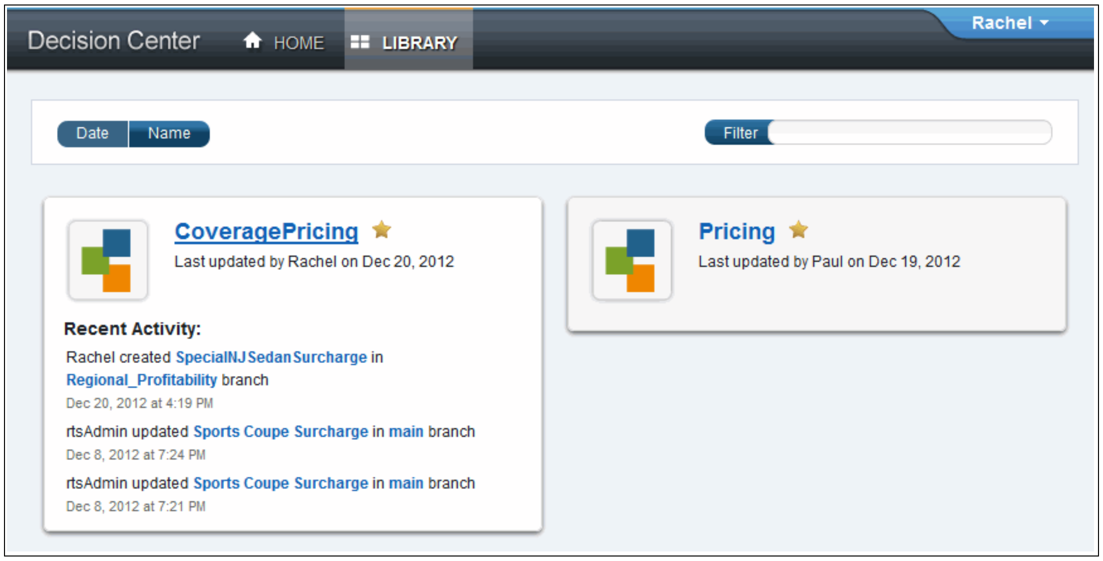
The next section looks at how the Business Console provides an alternative simpler way of authoring rules and decision tables that hide some of this complexity from business users such as Rachel.

9.4.4 Business Console rule modification

IBM Operational Decision Manager now provides a business user interface that is intended to make it easier for LOB users to find and modify rules and policies. This section provides an overview of how this business console can be used to support business users in undertaking their change activities.

Project navigation

After the user logs in to the business console, the Decision Center Library tab shows all projects that the user is authorized to access (as defined by the project permissions). See Figure 9-36.



CoveragePricing
Last updated by Rachel on Dec 20, 2012

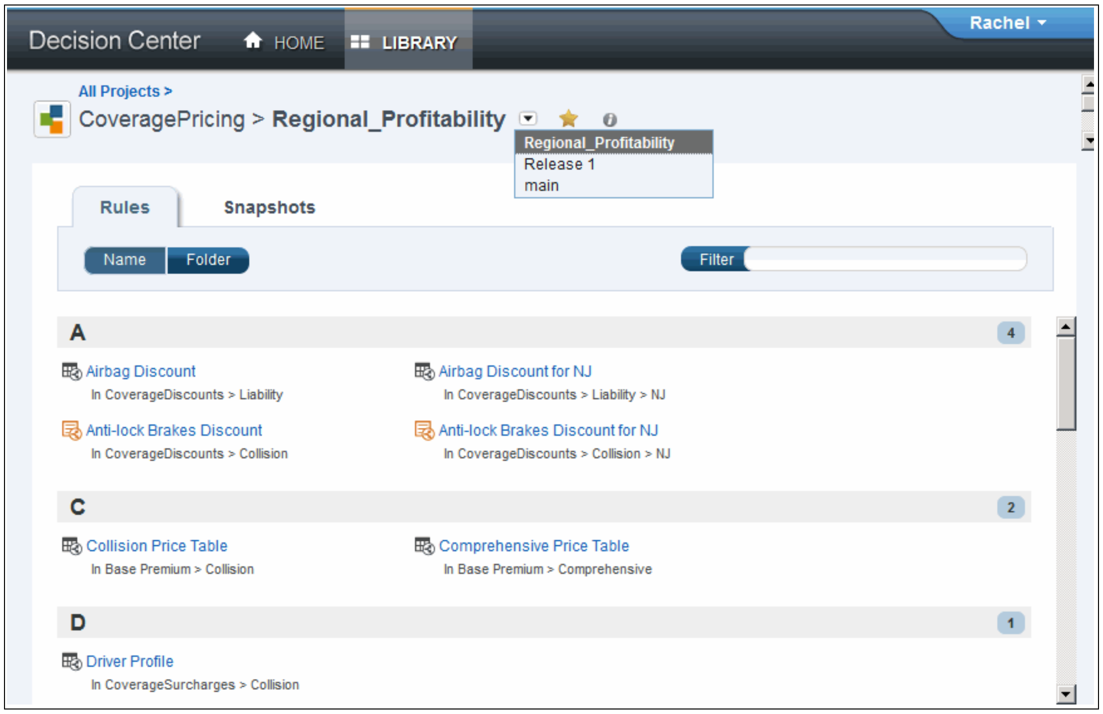
Pricing
Last updated by Paul on Dec 19, 2012

Recent Activity:
Rachel created SpecialNJSedanSurcharge in Regional_Profitability branch
Dec 20, 2012 at 4:19 PM
rtsAdmin updated Sports Coupe Surcharge in main branch
Dec 8, 2012 at 7:24 PM
rtsAdmin updated Sports Coupe Surcharge in main branch
Dec 8, 2012 at 7:21 PM

Figure 9-36 Business Console Library tab

The projects can be followed by selecting the star, which allows all changes to be visible in the activity stream for that user. Recent activity also shows what changes have been occurring in that project.

After a project is selected (in this case, CoveragePricing), the required branch can be selected, which shows the rules in alphabetical order. See Figure 9-37.



A 4
Airbag Discount
In CoverageDiscounts > Liability
Airbag Discount for NJ
In CoverageDiscounts > Liability > NJ
Anti-lock Brakes Discount
In CoverageDiscounts > Collision
Anti-lock Brakes Discount for NJ
In CoverageDiscounts > Collision > NJ

C 2
Collision Price Table
In Base Premium > Collision
Comprehensive Price Table
In Base Premium > Comprehensive

D 1
Driver Profile
In CoverageSurcharges > Collision

Figure 9-37 Project release navigation

Access control permissions apply to the individual releases as well as the individual rules. Rachel has access only to the Regional_Profitability change activity. By selecting the Folder view, rules can be navigated through the folder structure that is established for the project. See Figure 9-38.

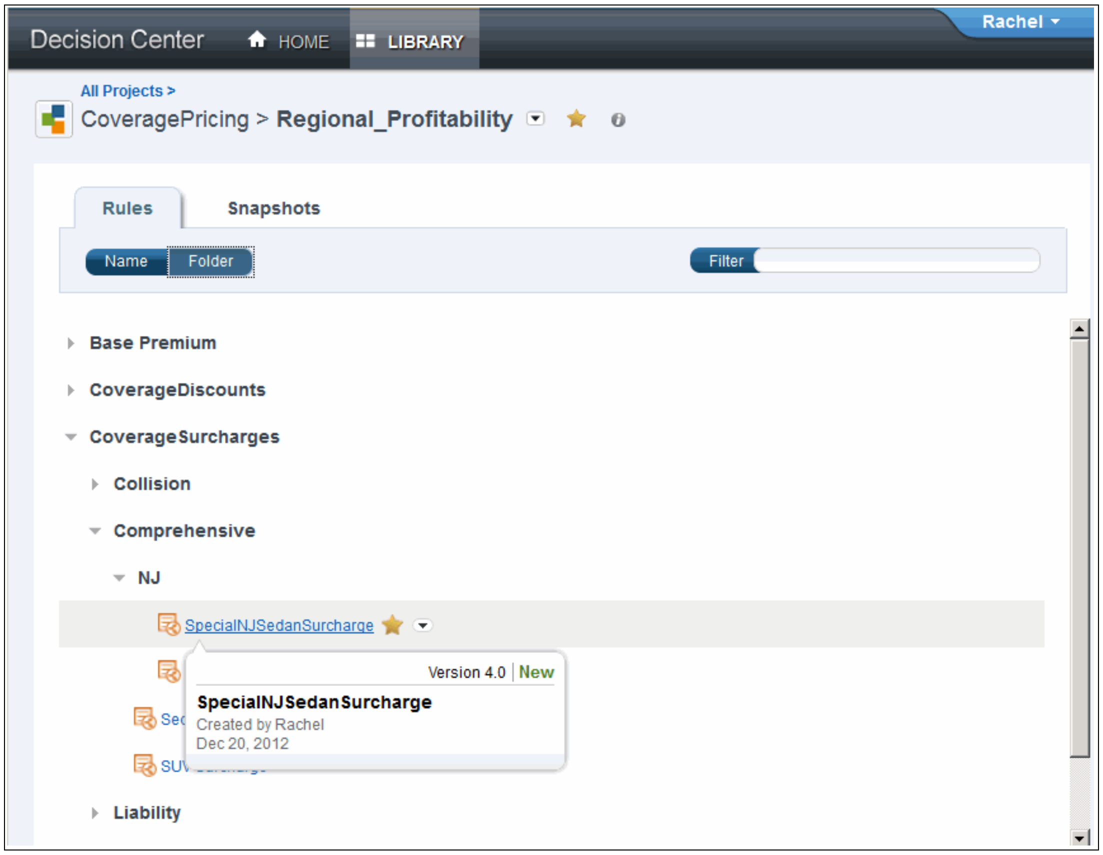
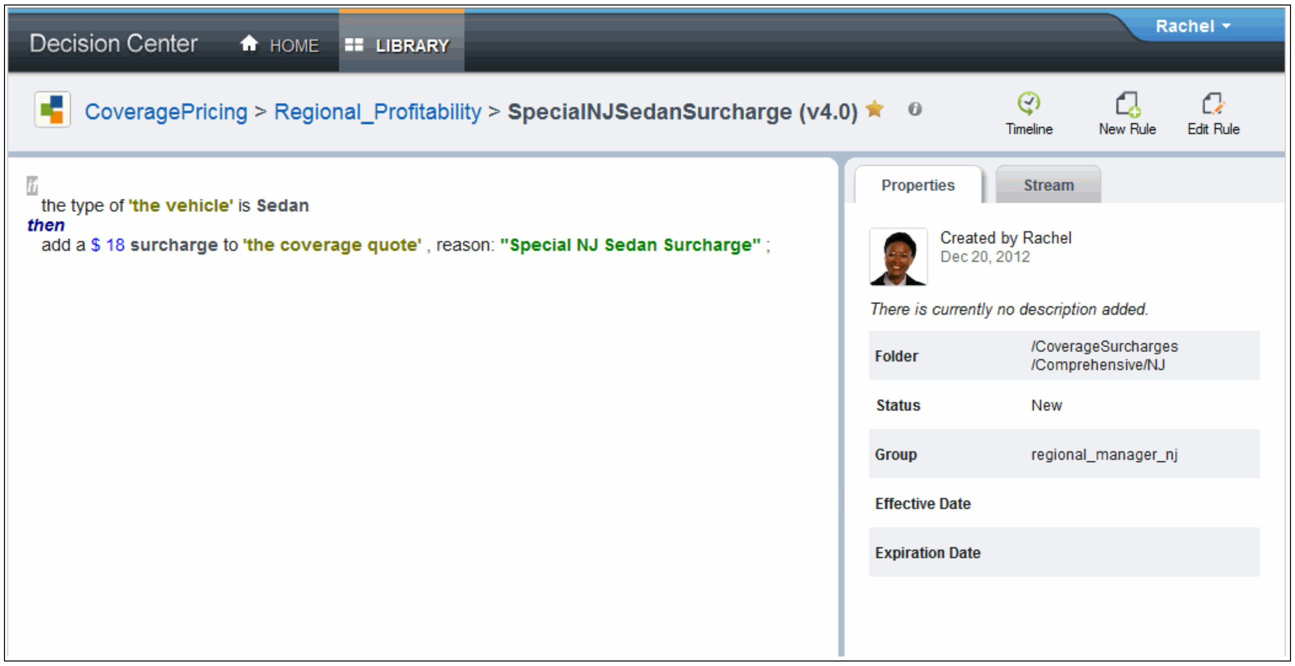


Figure 9-38 Folder-based project navigation

In this view, the folder-based access control permissions are also applied, allowing Rachel only to see the New Jersey regional folders and rules within them. By hovering over each of the rules, overview status information can be seen. In this case, the SpecialNJSedanSurcharge rule that Rachel created is visible.

Business rule editing

By selecting an individual rule, the content can be immediately seen together with options for editing the rule or creating a rule. See Figure 9-39.


Decision Center
HOME
LIBRARY
Rachel
CoveragePricing > Regional_Profitability > SpecialNJSedanSurcharge (v4.0)
Timeline
New Rule
Edit Rule
Properties
Stream

```
if
  the type of 'the vehicle' is Sedan
then
  add a $ 18 surcharge to 'the coverage quote' , reason: "Special NJ Sedan Surcharge" ;
```

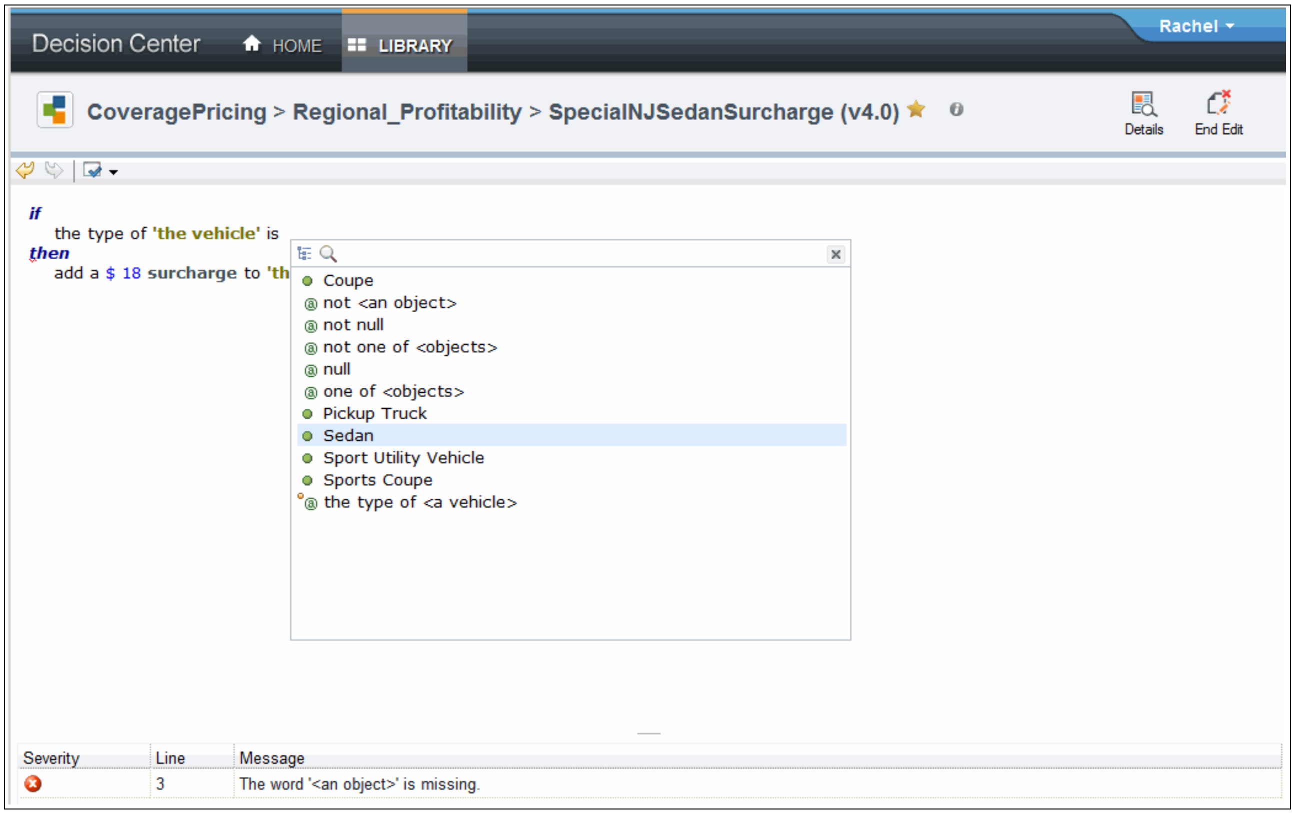
Created by Rachel
Dec 20, 2012

There is currently no description added.

Folder	/CoverageSurcharges /Comprehensive/NJ
Status	New
Group	regional_manager_nj
Effective Date	
Expiration Date	

Figure 9-39 Business Console rule view

By selecting the **Edit Rule** button, business users can modify the rule content by clicking the drop-down menu to select from available options. See Figure 9-40 on page 135.


Decision Center
HOME
LIBRARY
Rachel
CoveragePricing > Regional_Profitability > SpecialNJSedanSurcharge (v4.0)
Details
End Edit

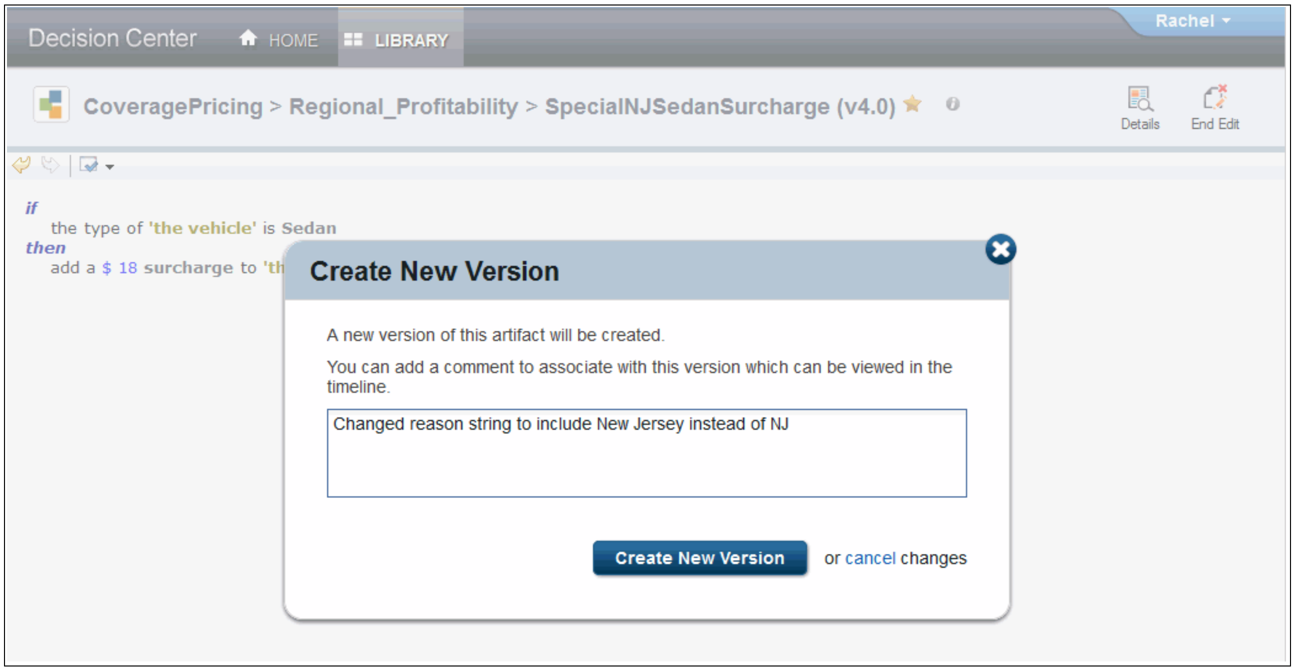
```
if
    the type of 'the vehicle' is
then
    add a $ 18 surcharge to 'th
```

- Coupe
- not <an object>
- not null
- not one of <objects>
- null
- one of <objects>
- Pickup Truck
- Sedan
- Sport Utility Vehicle
- Sports Coupe
- the type of <a vehicle>

Severity	Line	Message
✗	3	The word '<an object>' is missing.

Figure 9-40 Business Console rule editing

On completion of the changes, the user is prompted for a comment to document the changes that were made. See Figure 9-41.


Decision Center
HOME
LIBRARY
Rachel
CoveragePricing > Regional_Profitability > SpecialNJSedanSurcharge (v4.0)
Details
End Edit

```
if
    the type of 'the vehicle' is Sedan
then
    add a $ 18 surcharge to 'th
```

Create New Version

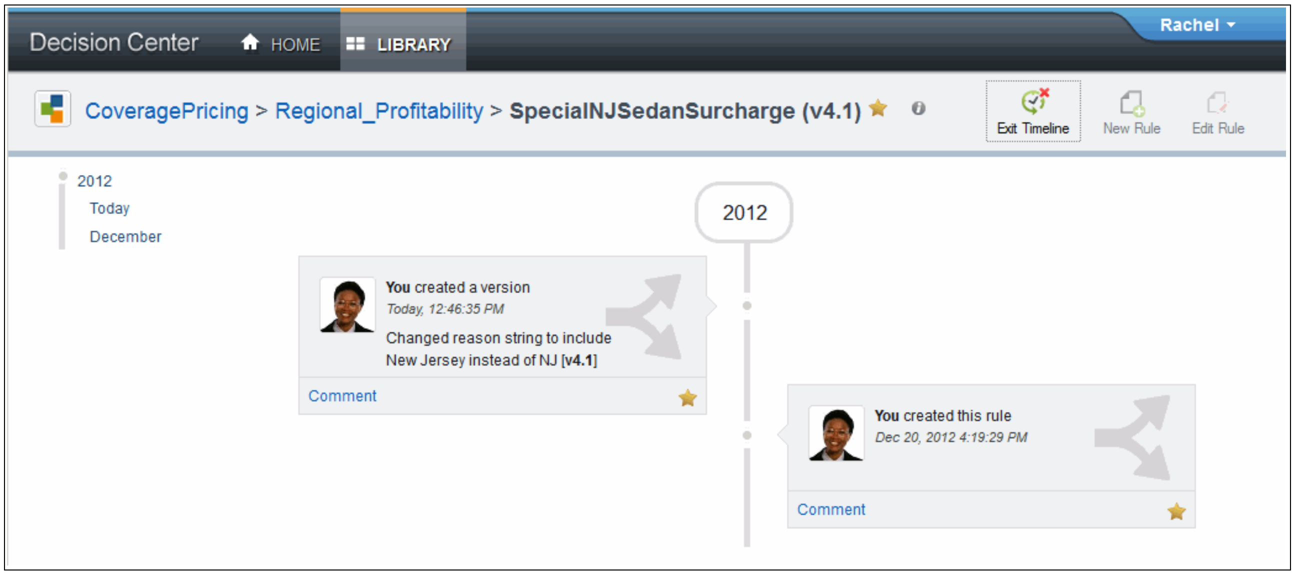
A new version of this artifact will be created.

You can add a comment to associate with this version which can be viewed in the timeline.

Changed reason string to include New Jersey instead of NJ

Figure 9-41 Business Console end edit

After the user creates the new version, the change is stored in the rule history within the change activity. See Figure 9-42.



You created a version
Today, 12:46:35 PM
Changed reason string to include New Jersey instead of NJ **[v4.1]**

You created this rule
Dec 20, 2012 4:19:29 PM

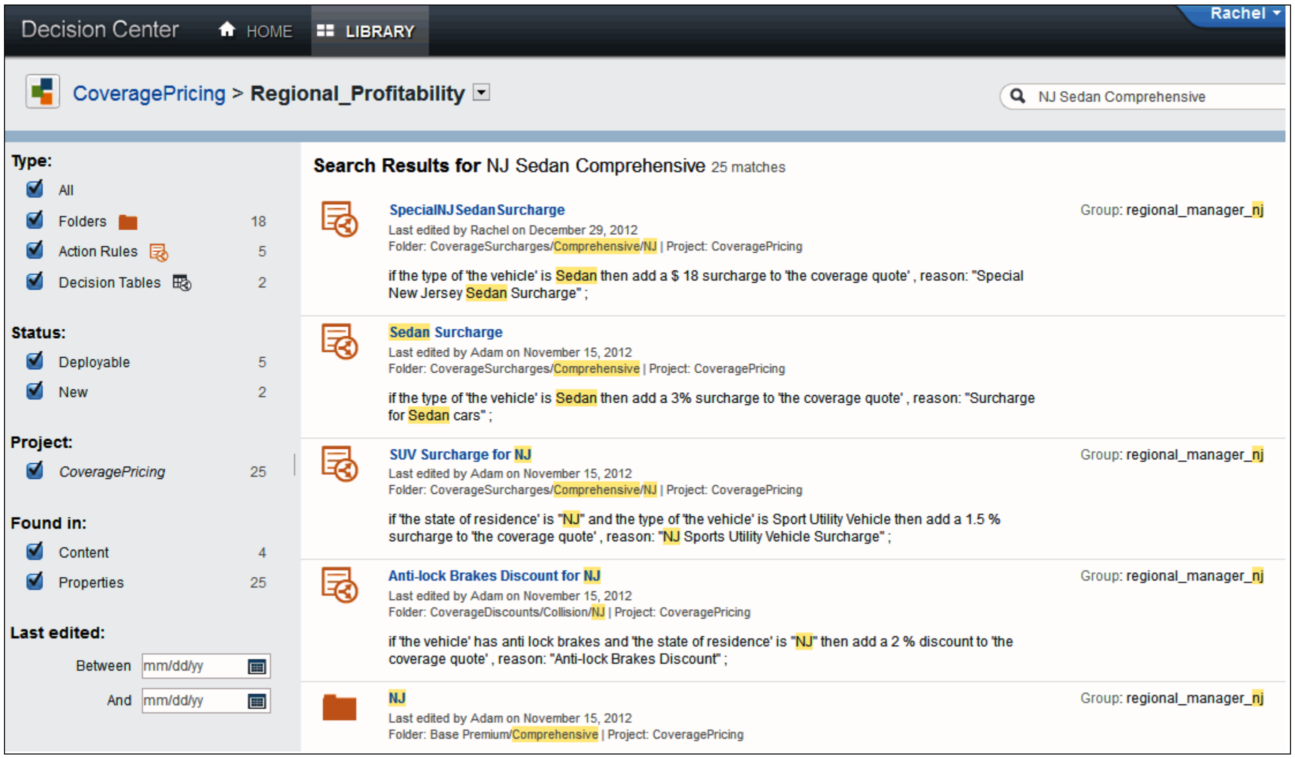
Figure 9-42 Business Console rule timeline

This timeline provides for collaboration between users on the changes that are being made to the rules and policies in a release or change activity.

Searching to find content

As well as navigating to find rules, the business console provides a text search function that can find rules either by the rule content or by metadata that is attached to the rules.

The search for “NJ Sedan Comprehensive” shows all rules and artifacts that contain these terms, as shown in Figure 9-43 on page 137.


Decision Center HOME LIBRARY Rachel

CoveragePricing > Regional_Profitability

NJ Sedan Comprehensive

Type:

☑	All	
☑	Folders	18
☑	Action Rules	5
☑	Decision Tables	2

Status:

☑	Deployable	5
☑	New	2

Project:

☑	*CoveragePricing*	25

Found in:

☑	Content	4
☑	Properties	25

Last edited:

Between mm/dd/yy

And mm/dd/yy

Search Results for NJ Sedan Comprehensive 25 matches

SpecialNJSedanSurcharge Group: regional_manager_nj
Last edited by Rachel on December 29, 2012
Folder: CoverageSurcharges/Comprehensive/NJ | Project: CoveragePricing
if the type of 'the vehicle' is Sedan then add a $ 18 surcharge to 'the coverage quote' , reason: "Special New Jersey Sedan Surcharge" ;

Sedan Surcharge
Last edited by Adam on November 15, 2012
Folder: CoverageSurcharges/Comprehensive | Project: CoveragePricing
if the type of 'the vehicle' is Sedan then add a 3% surcharge to 'the coverage quote' , reason: "Surcharge for Sedan cars" ;

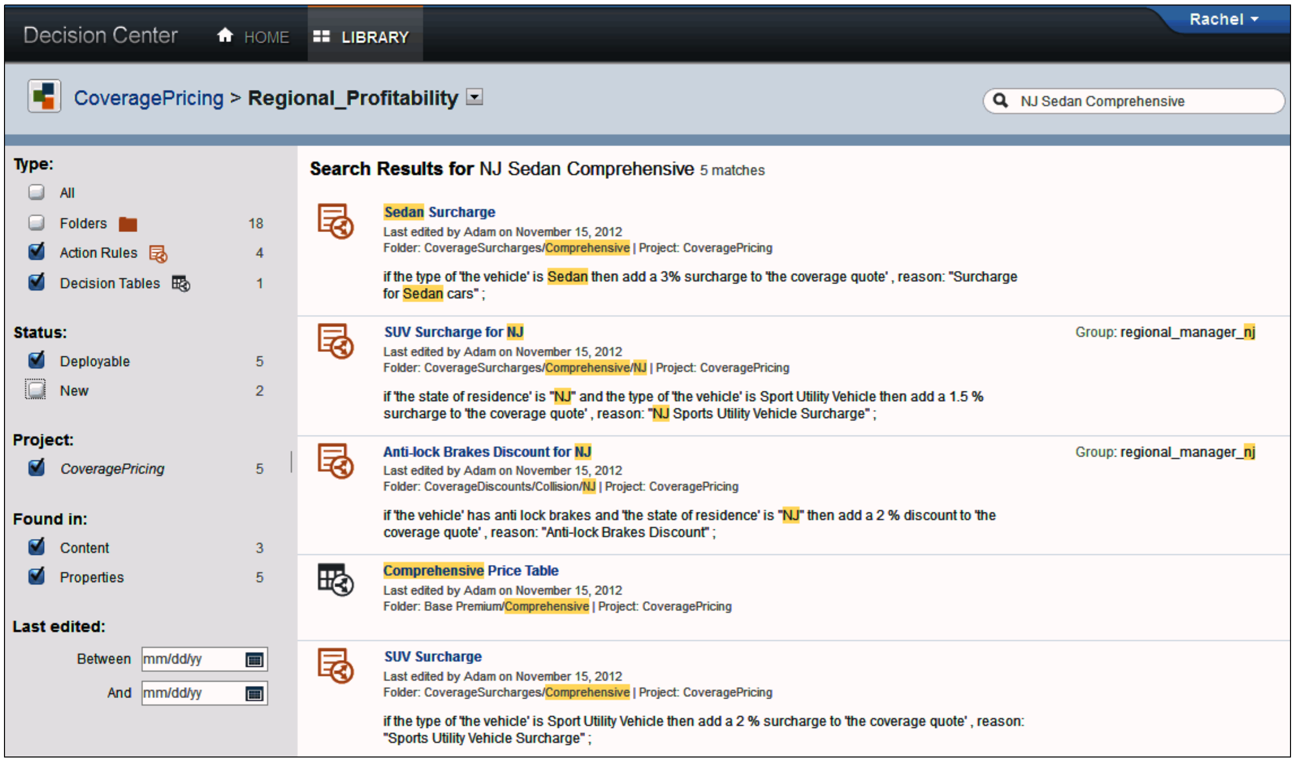
SUV Surcharge for NJ Group: regional_manager_nj
Last edited by Adam on November 15, 2012
Folder: CoverageSurcharges/Comprehensive/NJ | Project: CoveragePricing
if 'the state of residence' is "NJ" and the type of 'the vehicle' is Sport Utility Vehicle then add a 1.5 % surcharge to 'the coverage quote' , reason: "NJ Sports Utility Vehicle Surcharge" ;

Anti-lock Brakes Discount for NJ Group: regional_manager_nj
Last edited by Adam on November 15, 2012
Folder: CoverageDiscounts/Collision/NJ | Project: CoveragePricing
if 'the vehicle' has anti lock brakes and 'the state of residence' is "NJ" then add a 2 % discount to 'the coverage quote' , reason: "Anti-lock Brakes Discount" ;

NJ Group: regional_manager_nj
Last edited by Adam on November 15, 2012
Folder: Base Premium/Comprehensive | Project: CoveragePricing

Figure 9-43 Search results

The results can be refined by clearing options on the left side, as shown in Figure 9-44. In this way, the rules that are required can be quickly identified.


Decision Center HOME LIBRARY Rachel

CoveragePricing > Regional_Profitability

NJ Sedan Comprehensive

Type:

☐	All	
☐	Folders	18
☑	Action Rules	4
☑	Decision Tables	1

Status:

☑	Deployable	5
☐	New	2

Project:

☑	*CoveragePricing*	5

Found in:

☑	Content	3
☑	Properties	5

Last edited:

Between mm/dd/yy

And mm/dd/yy

Search Results for NJ Sedan Comprehensive 5 matches

Sedan Surcharge
Last edited by Adam on November 15, 2012
Folder: CoverageSurcharges/Comprehensive | Project: CoveragePricing
if the type of 'the vehicle' is Sedan then add a 3% surcharge to 'the coverage quote' , reason: "Surcharge for Sedan cars" ;

SUV Surcharge for NJ Group: regional_manager_nj
Last edited by Adam on November 15, 2012
Folder: CoverageSurcharges/Comprehensive/NJ | Project: CoveragePricing
if 'the state of residence' is "NJ" and the type of 'the vehicle' is Sport Utility Vehicle then add a 1.5 % surcharge to 'the coverage quote' , reason: "NJ Sports Utility Vehicle Surcharge" ;

Anti-lock Brakes Discount for NJ Group: regional_manager_nj
Last edited by Adam on November 15, 2012
Folder: CoverageDiscounts/Collision/NJ | Project: CoveragePricing
if 'the vehicle' has anti lock brakes and 'the state of residence' is "NJ" then add a 2 % discount to 'the coverage quote' , reason: "Anti-lock Brakes Discount" ;

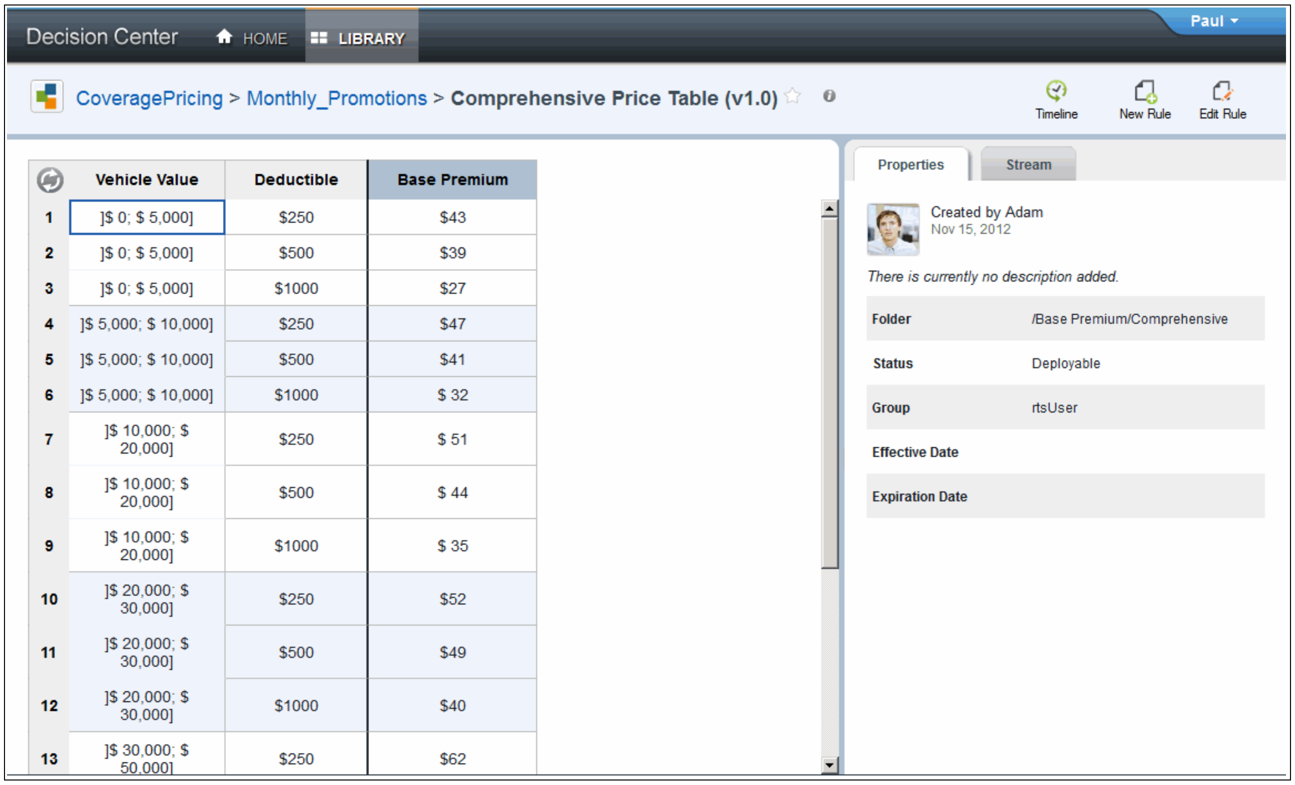
Comprehensive Price Table
Last edited by Adam on November 15, 2012
Folder: Base Premium/Comprehensive | Project: CoveragePricing

SUV Surcharge
Last edited by Adam on November 15, 2012
Folder: CoverageSurcharges/Comprehensive | Project: CoveragePricing
if the type of 'the vehicle' is Sport Utility Vehicle then add a 2 % surcharge to 'the coverage quote' , reason: "Sports Utility Vehicle Surcharge" ;

Figure 9-44 Refined search results

Decision table editing

One of the most popular approaches for business user rule definition is the decision table. In a decision table, the columns defining the conditions and action responses are usually defined by a rules expert, allowing the LOB user to concentrate on supplying the values in the table, as shown in Figure 9-45.


Decision Center
HOME
LIBRARY
Paul
CoveragePricing > Monthly_Promotions > Comprehensive Price Table (v1.0)
Timeline
New Rule
Edit Rule
Properties
Stream

	Vehicle Value	Deductible	Base Premium
1	]$ 0; $ 5,000]	$250	$43
2	]$ 0; $ 5,000]	$500	$39
3	]$ 0; $ 5,000]	$1000	$27
4	]$ 5,000; $ 10,000]	$250	$47
5	]$ 5,000; $ 10,000]	$500	$41
6	]$ 5,000; $ 10,000]	$1000	$ 32
7	]$ 10,000; $ 20,000]	$250	$ 51
8	]$ 10,000; $ 20,000]	$500	$ 44
9	]$ 10,000; $ 20,000]	$1000	$ 35
10	]$ 20,000; $ 30,000]	$250	$52
11	]$ 20,000; $ 30,000]	$500	$49
12	]$ 20,000; $ 30,000]	$1000	$40
13	]$ 30,000; $ 50,000]	$250	$62

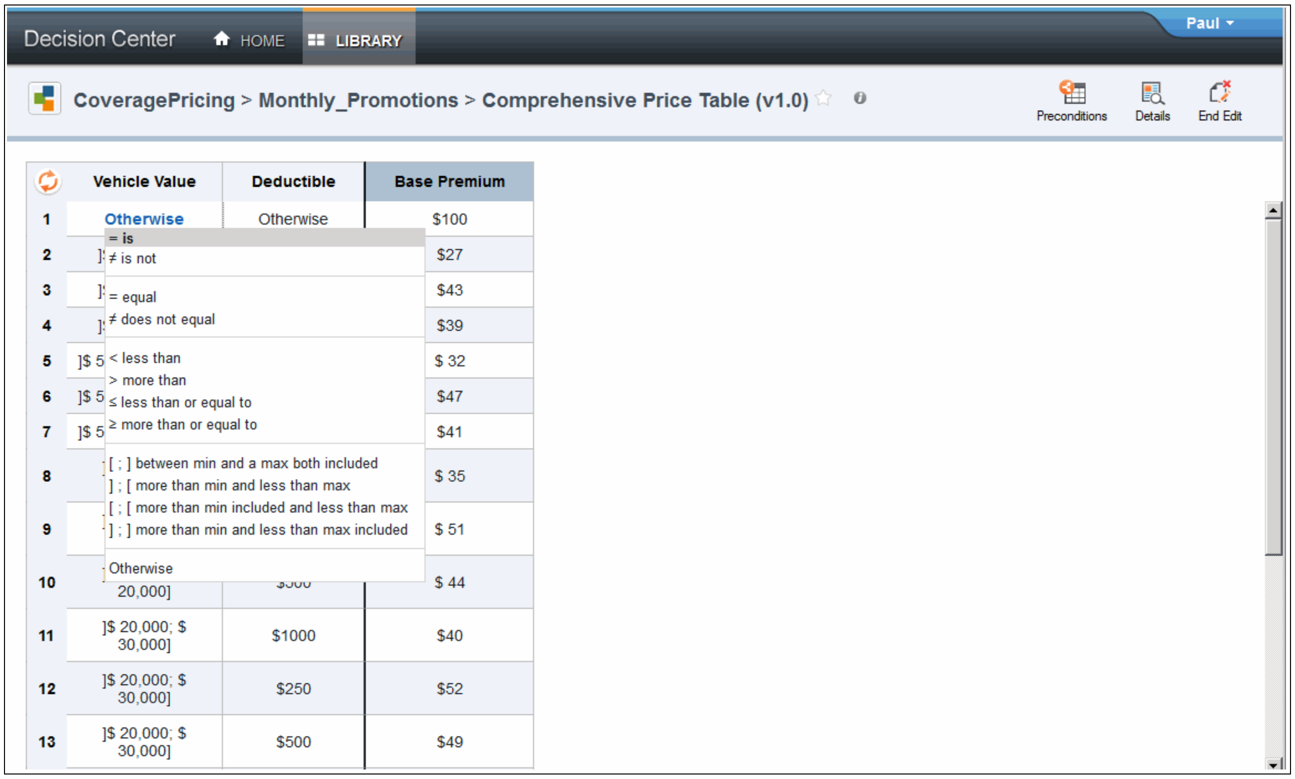
Created by Adam
Nov 15, 2012

There is currently no description added.

Folder	/Base Premium/Comprehensive
Status	Deployable
Group	rtsUser
Effective Date	
Expiration Date	

Figure 9-45 Business Console decision table

When editing a decision table, individual cells can be modified or new rows can be added with the editor prompting the user for the appropriate values in each cell. Because the column definitions are not normally modified and because there is validation that is built in that can check for overlapping or missing conditions, decision tables are an attractive mechanism for LOB users to maintain policies. See Figure 9-46.



CoveragePricing > Monthly_Promotions > Comprehensive Price Table (v1.0)

	Vehicle Value	Deductible	Base Premium
1	Otherwise	Otherwise	$100
2	]$		$27
3	]$		$43
4	]$		$39
5	]$ 5		$ 32
6	]$ 5		$47
7	]$ 5		$41
8			$ 35
9			$ 51
10	20,000]	$500	$ 44
11	]$ 20,000; $ 30,000]	$1000	$40
12	]$ 20,000; $ 30,000]	$250	$52
13	]$ 20,000; $ 30,000]	$500	$49

= is
≠ is not

= equal
≠ does not equal

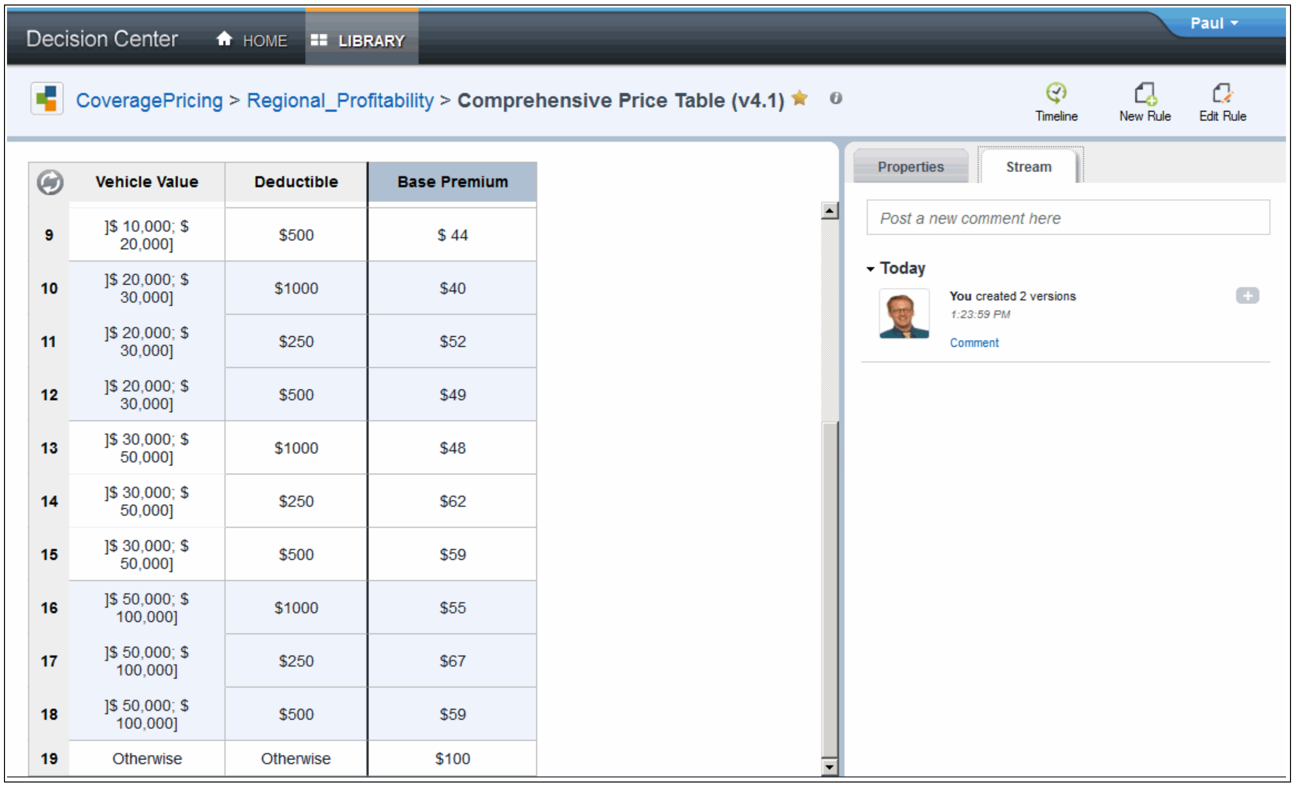
< less than
> more than
≤ less than or equal to
≥ more than or equal to

[;] between min and a max both included
] ; [more than min and less than max
[; [more than min included and less than max
] ;] more than min and less than max included

Otherwise

Figure 9-46 Business Console decision table editing

When the changes are completed in the Regional profitability change activity, the decision table contains the additional row that is shown at the bottom. See Figure 9-47.



CoveragePricing > Regional_Profitability > Comprehensive Price Table (v4.1)

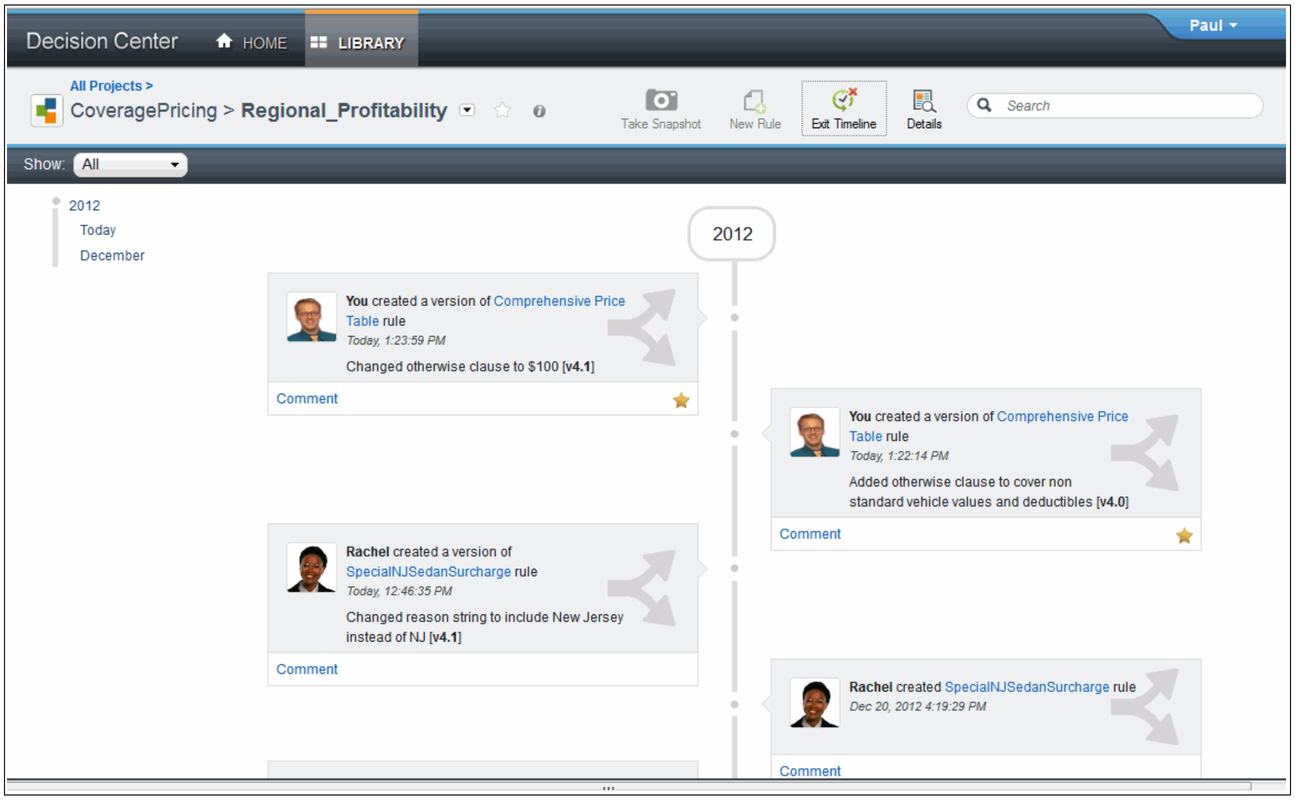
	Vehicle Value	Deductible	Base Premium
9	]$ 10,000; $ 20,000]	$500	$ 44
10	]$ 20,000; $ 30,000]	$1000	$40
11	]$ 20,000; $ 30,000]	$250	$52
12	]$ 20,000; $ 30,000]	$500	$49
13	]$ 30,000; $ 50,000]	$1000	$48
14	]$ 30,000; $ 50,000]	$250	$62
15	]$ 30,000; $ 50,000]	$500	$59
16	]$ 50,000; $ 100,000]	$1000	$55
17	]$ 50,000; $ 100,000]	$250	$67
18	]$ 50,000; $ 100,000]	$500	$59
19	Otherwise	Otherwise	$100

You created 2 versions
1:23:59 PM

Figure 9-47 Business Console decision table end edit

Timelines to show project-wide changes

As well as being able to show the individual change history of a rule, the business console allows all changes that are made to a branch to be seen in a timeline. This means that the changes within a change activity branch can be clearly seen, as shown in Figure 9-48.



You created a version of Comprehensive Price Table rule
Today, 1:23:59 PM
Changed otherwise clause to $100 **[v4.1]**

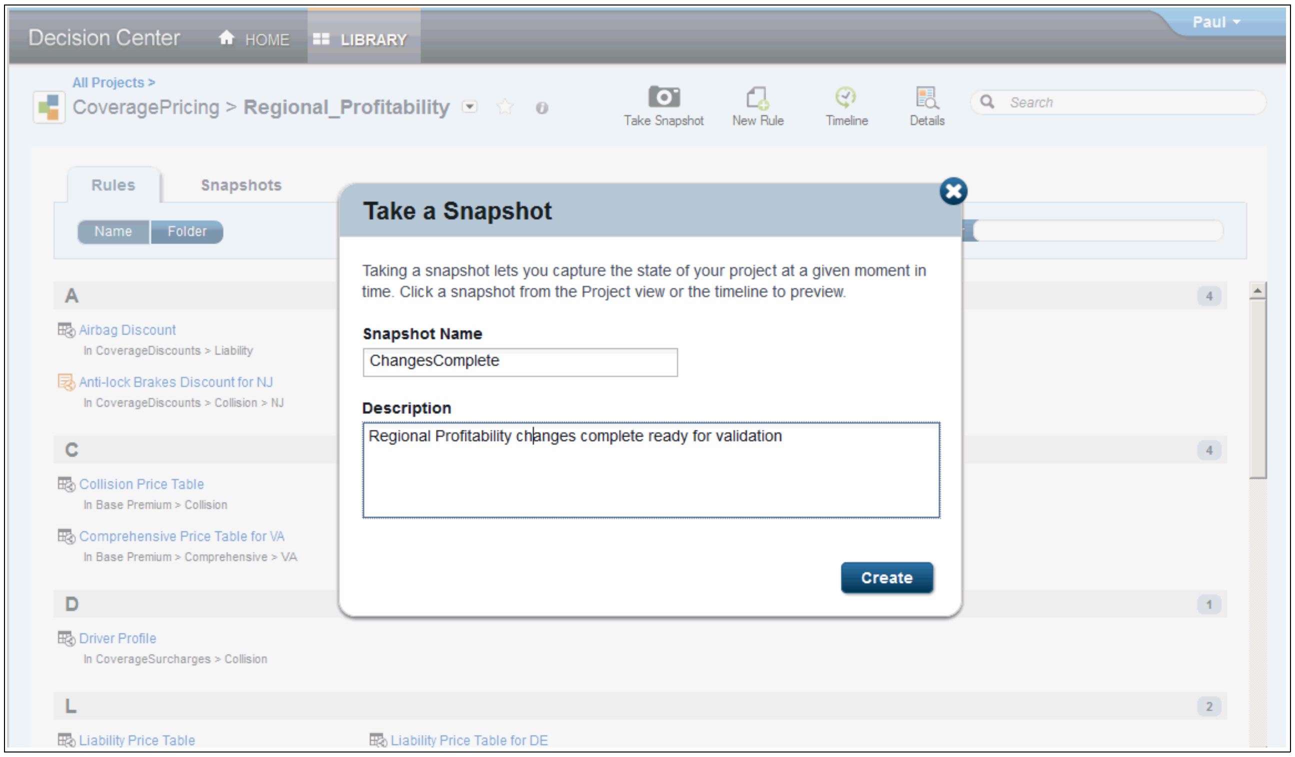
You created a version of Comprehensive Price Table rule
Today, 1:22:14 PM
Added otherwise clause to cover non standard vehicle values and deductibles **[v4.0]**

Rachel created a version of SpecialNJSedanSurcharge rule
Today, 12:46:35 PM
Changed reason string to include New Jersey instead of NJ **[v4.1]**

Rachel created SpecialNJSedanSurcharge rule
Dec 20, 2012 4:19:29 PM

Figure 9-48 Project timeline

This view provides the users that are collaborating within a change activity a clear way to see and comment on the changes undertaken. When all changes are complete, a snapshot may be taken and used to validate the activity or release, as shown in Figure 9-49.


Decision Center
HOME
LIBRARY
Paul
All Projects >
CoveragePricing > Regional_Profitability
Take Snapshot
New Rule
Timeline
Details
Search
Rules
Snapshots
Name
Folder
A
Airbag Discount
In CoverageDiscounts > Liability
Anti-lock Brakes Discount for NJ
In CoverageDiscounts > Collision > NJ
C
Collision Price Table
In Base Premium > Collision
Comprehensive Price Table for VA
In Base Premium > Comprehensive > VA
D
Driver Profile
In CoverageSurcharges > Collision
L
Liability Price Table
Liability Price Table for DE
Create

Take a Snapshot

Taking a snapshot lets you capture the state of your project at a given moment in time. Click a snapshot from the Project view or the timeline to preview.

Snapshot Name: ChangesComplete

Description: Regional Profitability changes complete ready for validation

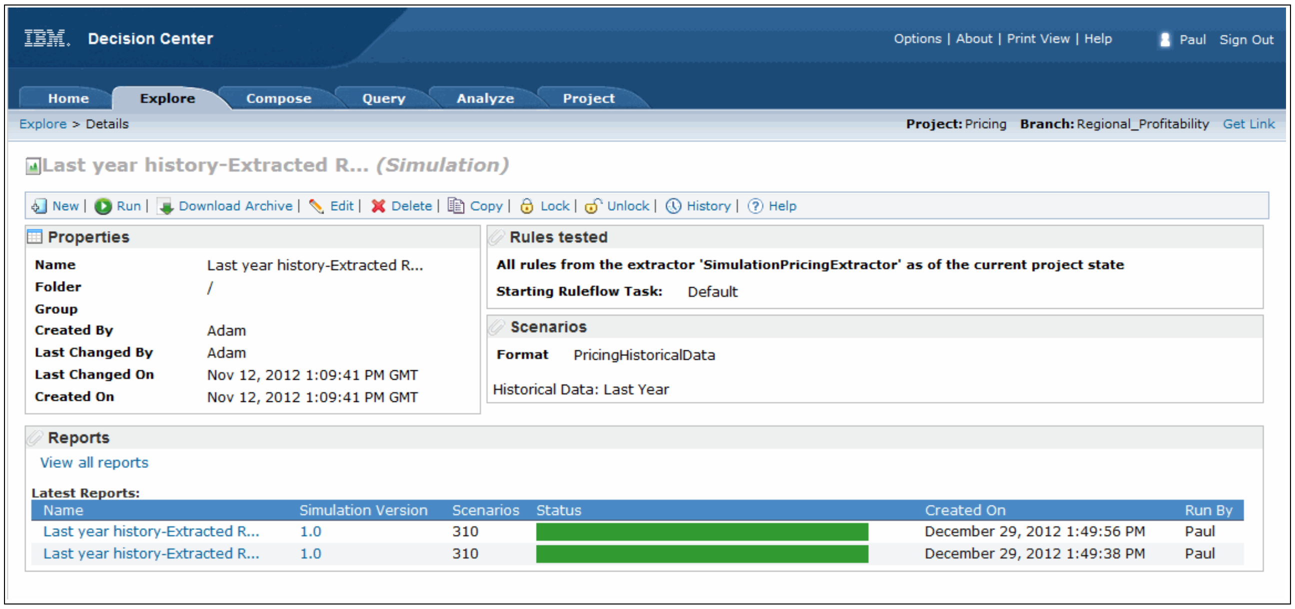
Figure 9-49 Snapshot definition

9.4.5 Simulation

IBM Operational Decision Manager provides a simulation capability that can be used to validate rules against predefined scenarios. Various different approaches can be adopted. This section shows how some predefined simulation scenarios and KPI metrics can be used to show how the changed rules can be shown to have improved the behavior.

Preparing for simulation

In a standard release, the simulations that assess how good the rules are would already be defined. These simulations are shown in Figure 9-50 on page 143 for the Pricing decision service, which takes historic information and runs a particular branch against that historic data. A typical definition is shown in Figure 9-50 on page 143.


IBM. Decision Center
Options | About | Print View | Help
Paul Sign Out
Home
Explore
Compose
Query
Analyze
Project
Explore > Details
Project: Pricing Branch: Regional_Profitability Get Link
Last year history-Extracted R... (Simulation)
New | Run | Download Archive | Edit | Delete | Copy | Lock | Unlock | History | Help

Properties

Name	Last year history-Extracted R...
Folder	/
Group	
Created By	Adam
Last Changed By	Adam
Last Changed On	Nov 12, 2012 1:09:41 PM GMT
Created On	Nov 12, 2012 1:09:41 PM GMT

Rules tested

All rules from the extractor 'SimulationPricingExtractor' as of the current project state

Starting Ruleflow Task: Default

Scenarios

Format PricingHistoricalData

Historical Data: Last Year

Reports

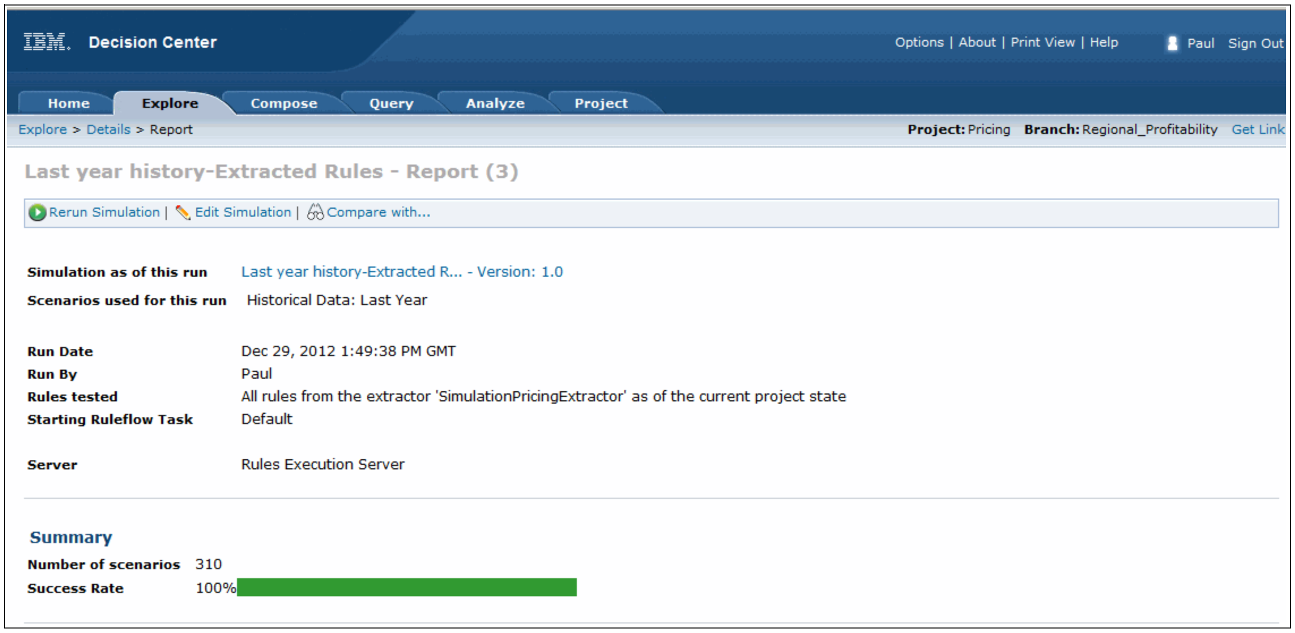
View all reports

Latest Reports:

Name	Simulation Version	Scenarios	Status	Created On	Run By
Last year history-Extracted R...	1.0	310	🟩	December 29, 2012 1:49:56 PM	Paul
Last year history-Extracted R...	1.0	310	🟩	December 29, 2012 1:49:38 PM	Paul

Figure 9-50 Simulation definition

Each of the reports for a simulation shows the scenarios that succeeded (matched the expected results) as well as the number that did not match the expected results. In addition, customized KPIs can be developed that allow graphs to be shown that can help business users interpret the simulation results. See Figure 9-51.


IBM. Decision Center
Options | About | Print View | Help
Paul Sign Out
Home
Explore
Compose
Query
Analyze
Project
Explore > Details > Report
Project: Pricing Branch: Regional_Profitability Get Link
Last year history-Extracted Rules - Report (3)
Rerun Simulation | Edit Simulation | Compare with...

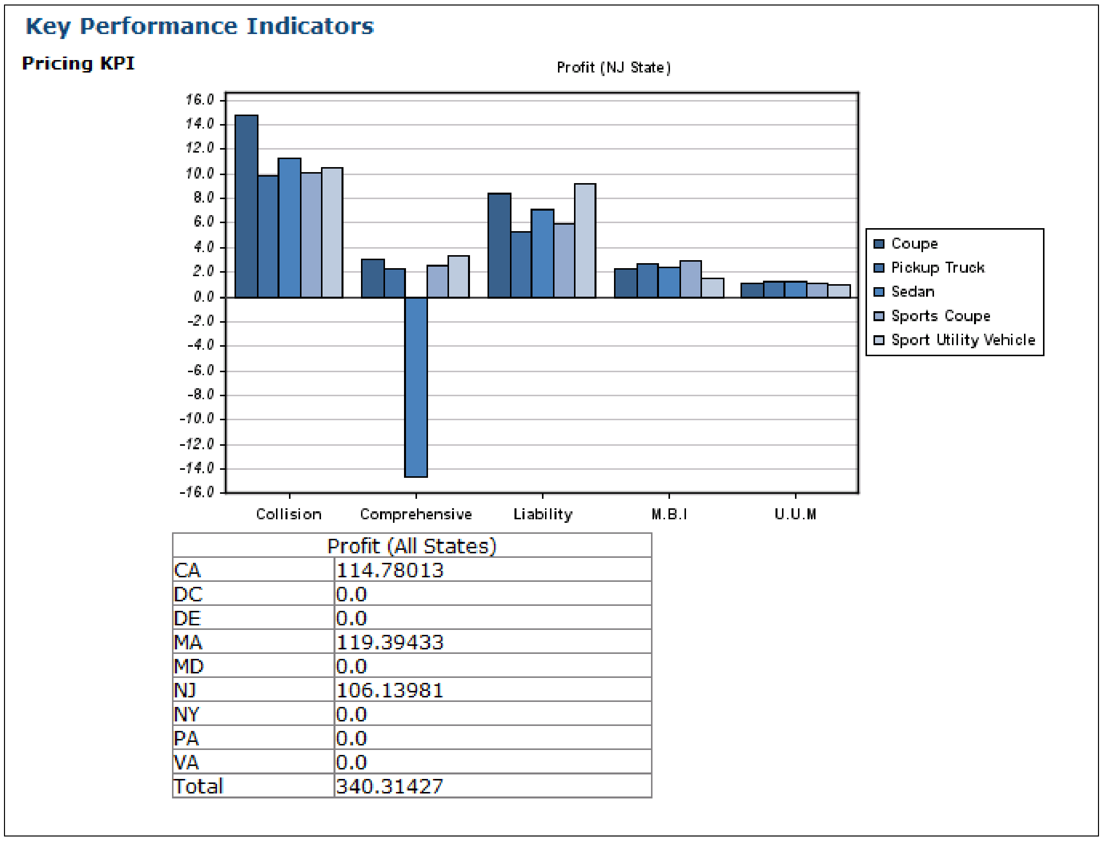
Simulation as of this run	Last year history-Extracted R... - Version: 1.0
Scenarios used for this run	Historical Data: Last Year
Run Date	Dec 29, 2012 1:49:38 PM GMT
Run By	Paul
Rules tested	All rules from the extractor 'SimulationPricingExtractor' as of the current project state
Starting Ruleflow Task	Default
Server	Rules Execution Server

Summary

Number of scenarios	310
Success Rate	100% 🟩

Figure 9-51 Simulation Summary

Figure 9-52 shows the results of the predefined KPIs for the simulation. The results provide a visual representation of the KPIs that visually show the issue we are trying to address with the rule changes.



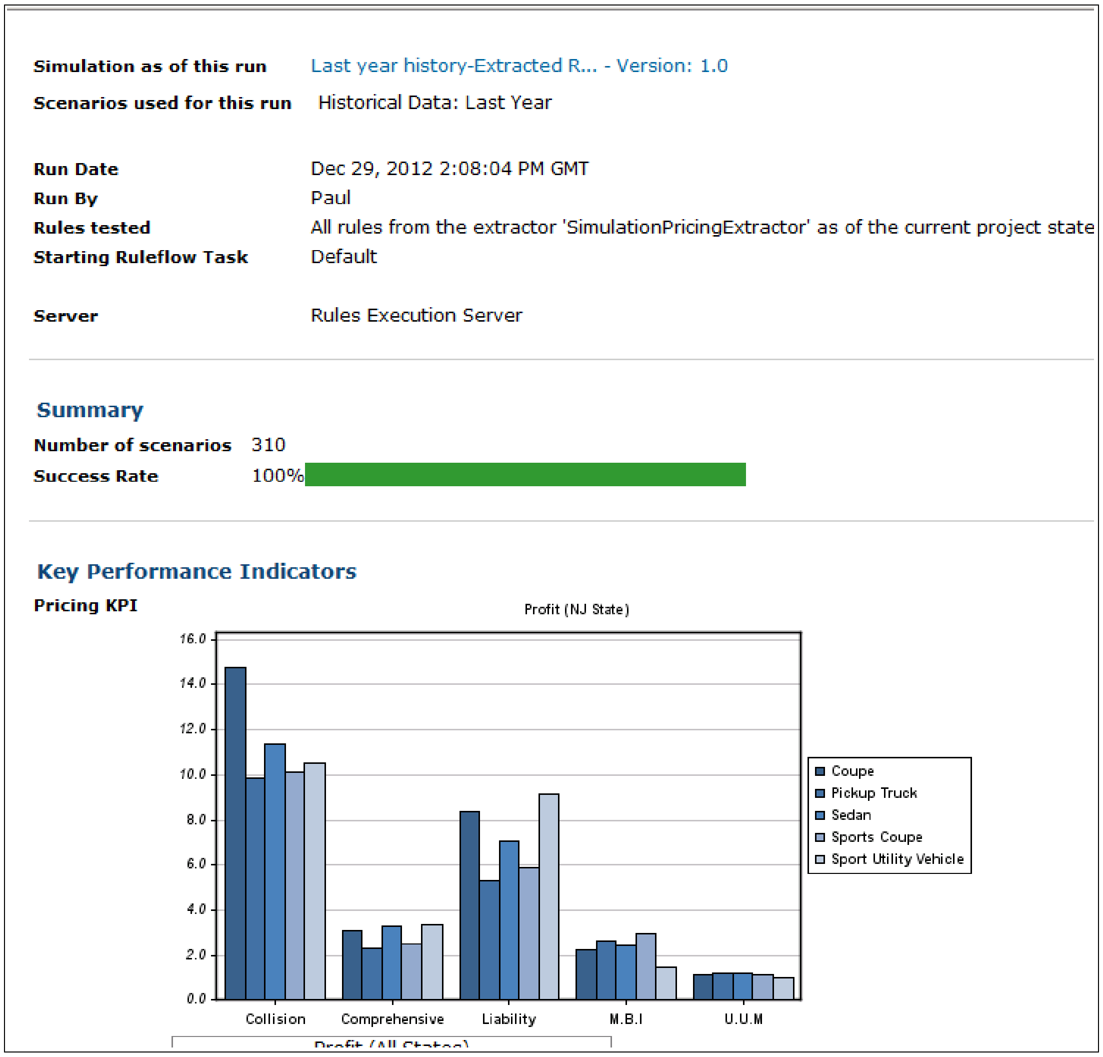
Profit (All States)	
CA	114.78013
DC	0.0
DE	0.0
MA	119.39433
MD	0.0
NJ	106.13981
NY	0.0
PA	0.0
VA	0.0
Total	340.31427

Figure 9-52 Simulation KPIs

After these initial reports are generated, the modified rules can be included in the simulation.

Rerunning the simulation with the changed rules

When the simulation is rerun, a new report is generated with modified KPIs, as shown in Figure 9-53.



Simulation as of this run	Last year history-Extracted R... - Version: 1.0
Scenarios used for this run	Historical Data: Last Year
Run Date	Dec 29, 2012 2:08:04 PM GMT
Run By	Paul
Rules tested	All rules from the extractor 'SimulationPricingExtractor' as of the current project state
Starting Ruleflow Task	Default
Server	Rules Execution Server

Summary

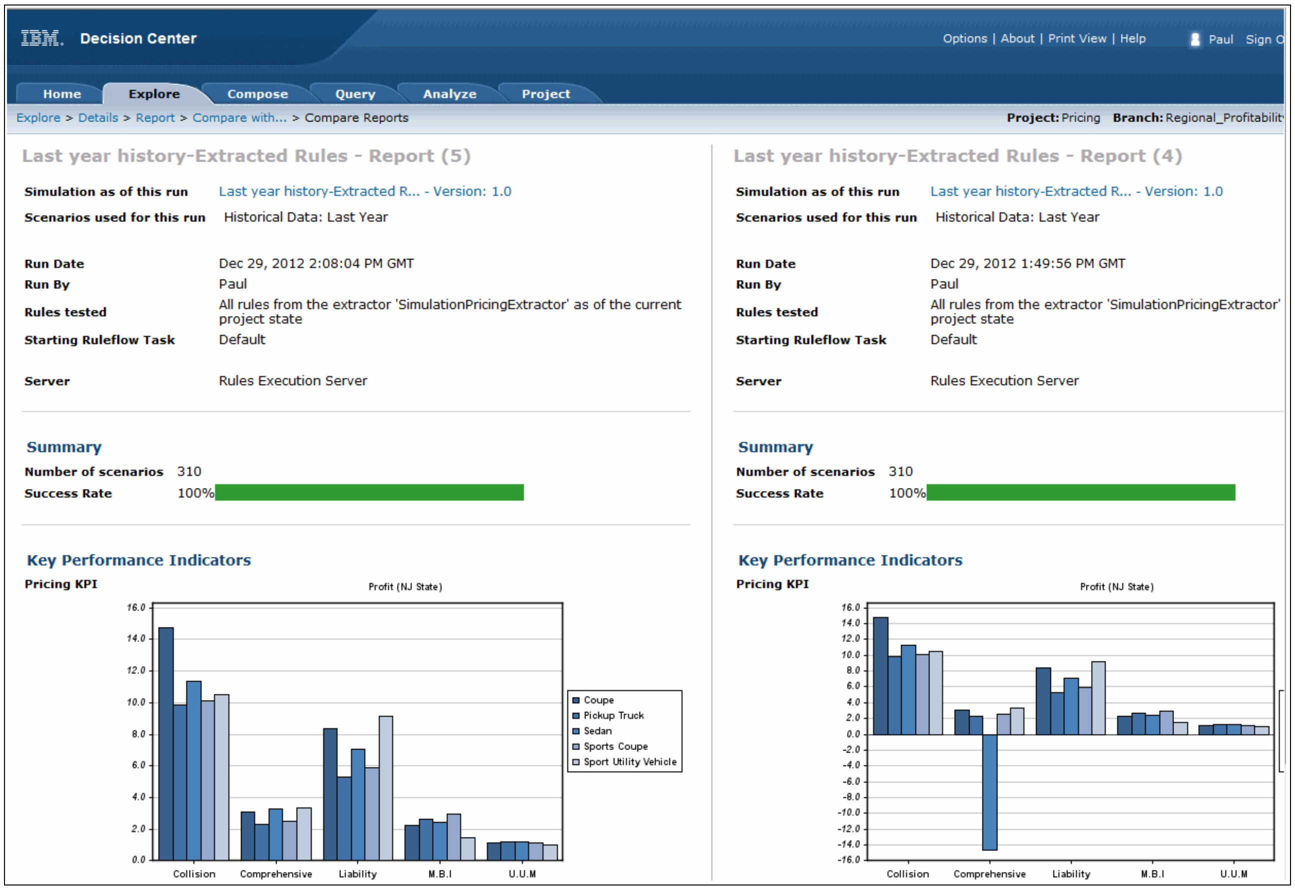
Number of scenarios	310
Success Rate	100%

Key Performance Indicators

Pricing KPI

Figure 9-53 Simulation with updated rules

These updated results show that all the simulation results were as expected and that the profitability KPIs for NJ Sedan Comprehensive were improved (as would be expected by increasing the price for these quotes). See Figure 9-54.



Explore > Details > Report > Compare with... > Compare Reports

Project: Pricing **Branch:** Regional_Profitabilit

Last year history-Extracted Rules - Report (5)

Simulation as of this run	Last year history-Extracted R... - Version: 1.0
Scenarios used for this run	Historical Data: Last Year
Run Date	Dec 29, 2012 2:08:04 PM GMT
Run By	Paul
Rules tested	All rules from the extractor 'SimulationPricingExtractor' as of the current project state
Starting Ruleflow Task	Default
Server	Rules Execution Server

Summary

Number of scenarios	310
Success Rate	100%

Key Performance Indicators

Pricing KPI

Last year history-Extracted Rules - Report (4)

Simulation as of this run	Last year history-Extracted R... - Version: 1.0
Scenarios used for this run	Historical Data: Last Year
Run Date	Dec 29, 2012 1:49:56 PM GMT
Run By	Paul
Rules tested	All rules from the extractor 'SimulationPricingExtractor' project state
Starting Ruleflow Task	Default
Server	Rules Execution Server

Summary

Number of scenarios	310
Success Rate	100%

Key Performance Indicators

Pricing KPI

Figure 9-54 Simulation side by side comparison

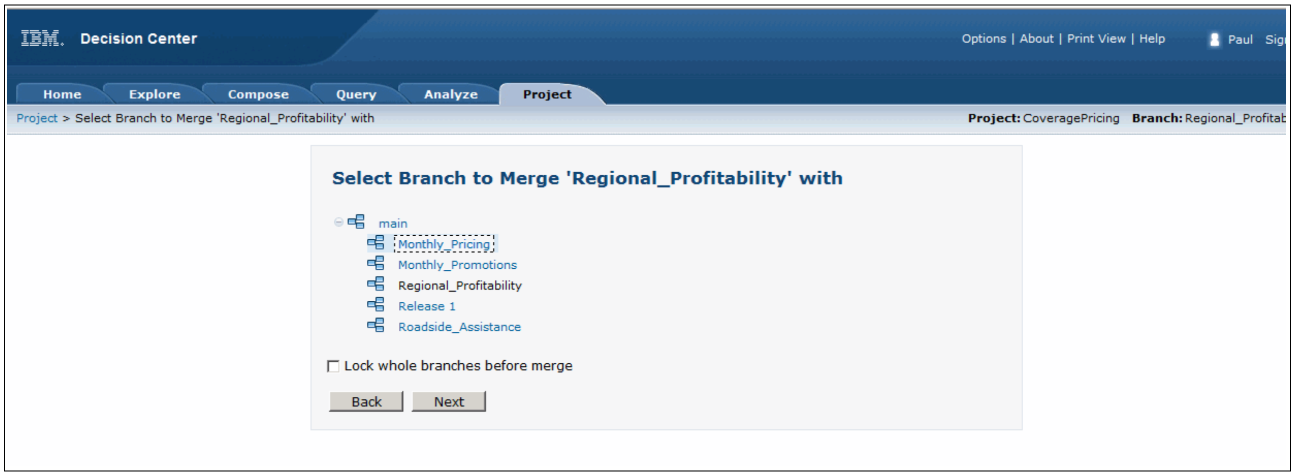
This side by side comparison allows the effectiveness of different releases to be quickly compared and thus validated.

9.4.6 Activity closure and testing

After the change activities are completed and approved, they are then ready to be merged into the release and the overall release is tested. Validation may occur at both the change activity and release branch. As review and simulation at the change activity level has already been described, this section concentrates on the testing at the release level.

Merging change activities into the release

The Regional_pricing change activity changes now must be merged into the monthly pricing release. This activity is undertaken by Paul, the release owner. In this case, the changes are in the CoveragePricing project, so it must be merged with the Monthly_Pricing release branch. See Figure 9-55.



Project > Select Branch to Merge 'Regional_Profitability' with

Project: CoveragePricing **Branch:** Regional_Profitab

Select Branch to Merge 'Regional_Profitability' with

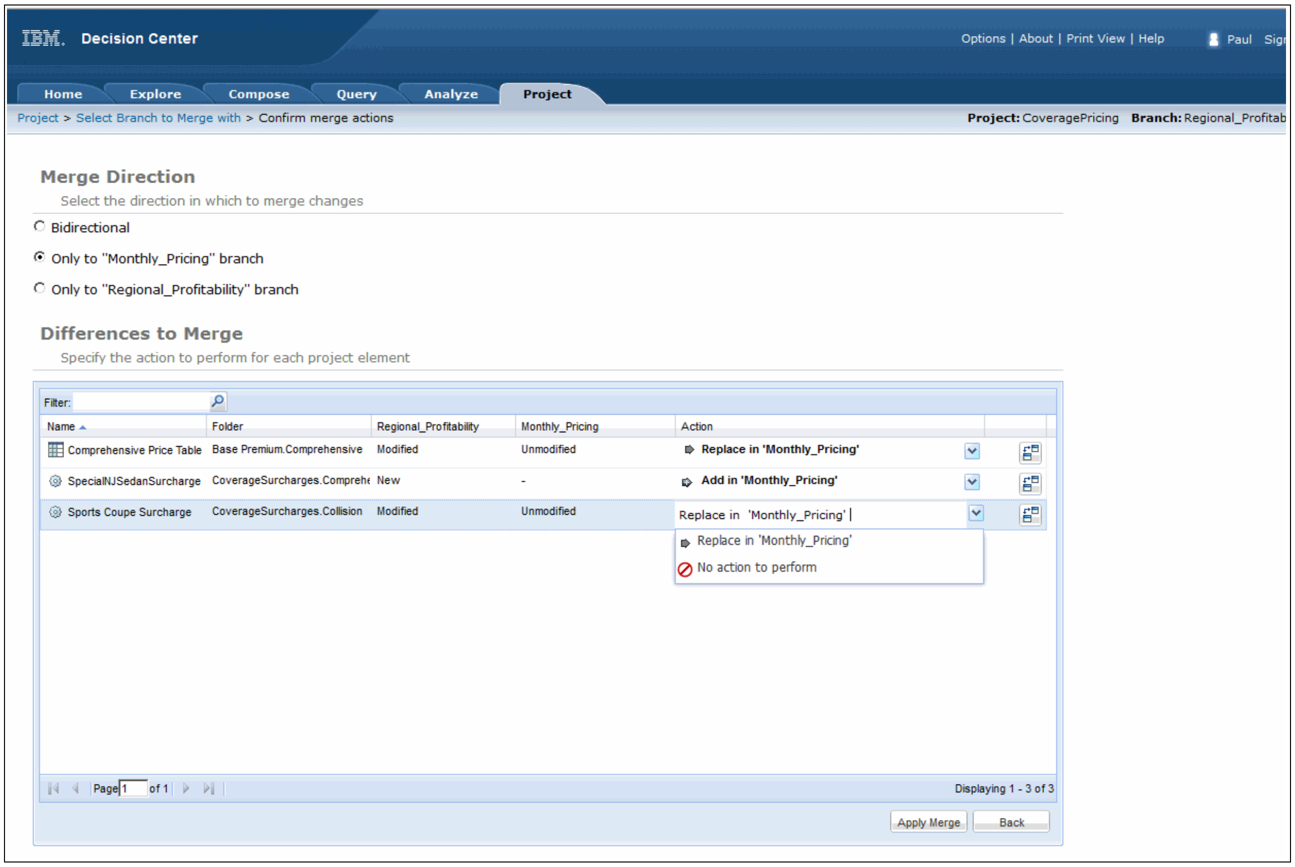
- main
  - Monthly_Pricing
  - Monthly_Promotions
  - Regional_Profitability
  - Release 1
  - Roadside_Assistance

☐ Lock whole branches before merge

Back Next

Figure 9-55 Regional profitability merge

Figure 9-56 shows the changes that were made in either branch that resolved any problems.



Project > Select Branch to Merge with > Confirm merge actions

Project: CoveragePricing **Branch:** Regional_Profitab

Merge Direction

Select the direction in which to merge changes

○ Bidirectional

● Only to "Monthly_Pricing" branch

○ Only to "Regional_Profitability" branch

Differences to Merge

Specify the action to perform for each project element

Filter:

Name	Folder	Regional_Profitability	Monthly_Pricing	Action
Comprehensive Price Table	Base Premium.Comprehensive	Modified	Unmodified	**Replace in 'Monthly_Pricing'**
SpecialNJSedanSurcharge	CoverageSurcharges.Comprehe	New	-	**Add in 'Monthly_Pricing'**
Sports Coupe Surcharge	CoverageSurcharges.Collision	Modified	Unmodified	Replace in 'Monthly_Pricing'

Replace in 'Monthly_Pricing'
No action to perform

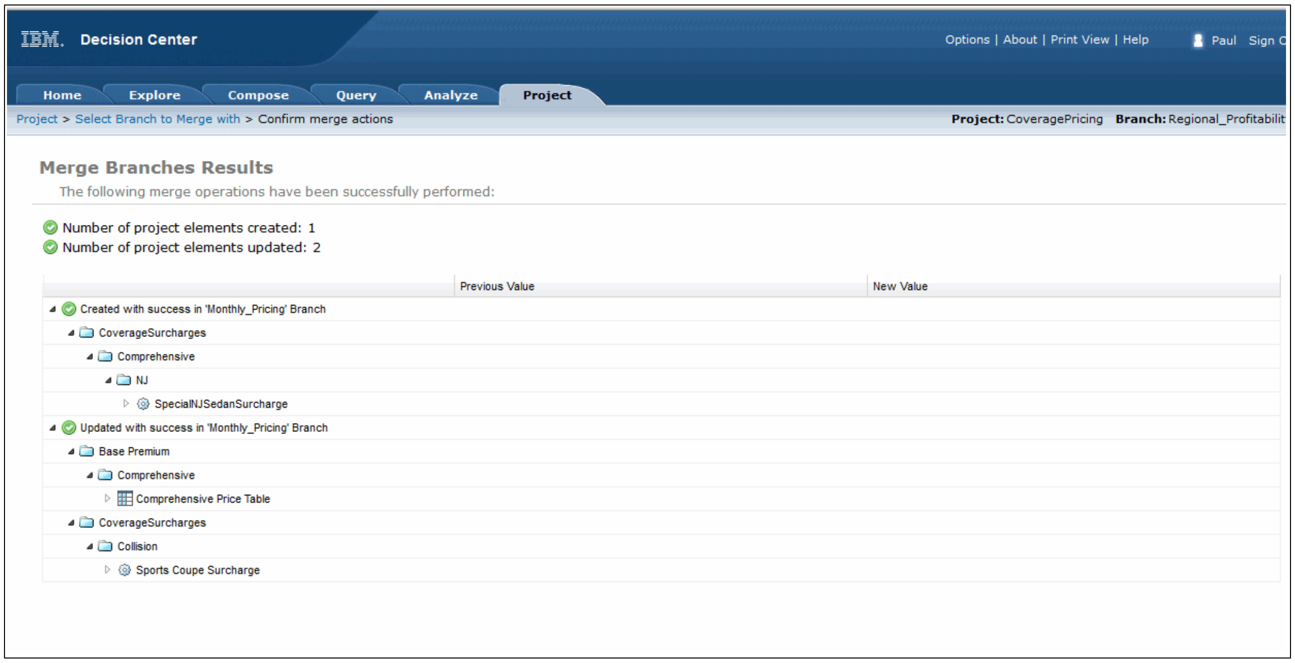
Page 1 of 1

Displaying 1 - 3 of 3

Apply Merge Back

Figure 9-56 Merge action selection

After the merge actions are selected, the merge can be applied. A summary is shown in Figure 9-57.


IBM. Decision Center
Options | About | Print View | Help
Paul Sign O
Home
Explore
Compose
Query
Analyze
Project
Project > Select Branch to Merge with > Confirm merge actions
Project: CoveragePricing Branch: Regional_Profitabilit

Merge Branches Results

The following merge operations have been successfully performed:

Number of project elements created: 1
Number of project elements updated: 2

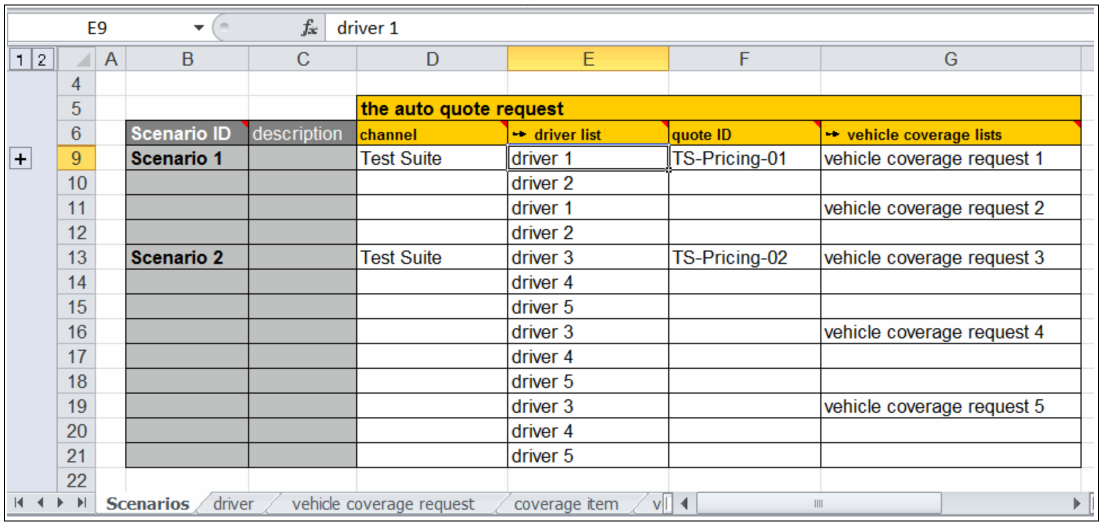
	Previous Value	New Value
Created with success in 'Monthly_Pricing' Branch		
CoverageSurcharges		
Comprehensive		
NJ		
SpecialNJSedanSurcharge		
Updated with success in 'Monthly_Pricing' Branch		
Base Premium		
Comprehensive		
Comprehensive Price Table		
CoverageSurcharges		
Collision		
Sports Coupe Surcharge		

Figure 9-57 Merge results

The change activity rules are now included in the monthly pricing release, and are ready for testing.

Testing the release

After the change activity rules are merged in to the release, overall release testing can be undertaken. Testing is similar to simulations, and can be undertaken by using test suites, which can be defined in spreadsheet templates that are generated directly from the rule set. These templates allow the input conditions that are required for each scenario to be quickly defined, as shown in Figure 9-58.

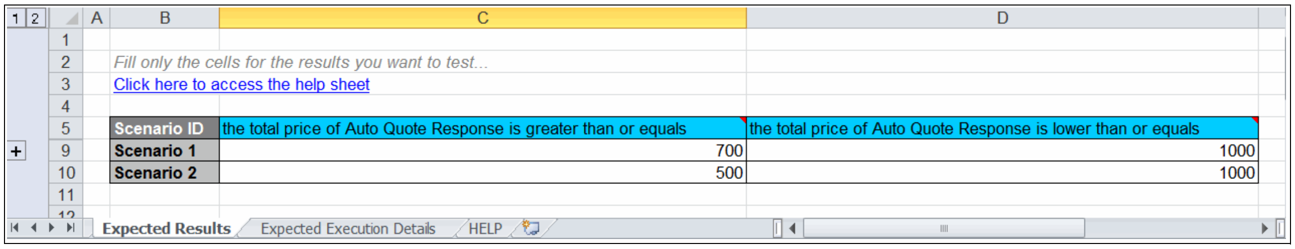


	A	B	C	D	E	F	G
4							
5				the auto quote request			
6		Scenario ID	description	channel	driver list	quote ID	vehicle coverage lists
9		Scenario 1		Test Suite	driver 1	TS-Pricing-01	vehicle coverage request 1
10					driver 2		
11					driver 1		vehicle coverage request 2
12					driver 2		
13		Scenario 2		Test Suite	driver 3	TS-Pricing-02	vehicle coverage request 3
14					driver 4		
15					driver 5		
16					driver 3		vehicle coverage request 4
17					driver 4		
18					driver 5		
19					driver 3		vehicle coverage request 5
20					driver 4		
21					driver 5		
22							

Figure 9-58 Test suite scenario definition

Each named column represents a value or a reference to another object that is defined in another worksheet. For example, driver 1 refers to the definition of a driver in the driver worksheet.

The test suite spreadsheet also allows the expected results to be defined, as shown in Figure 9-59.



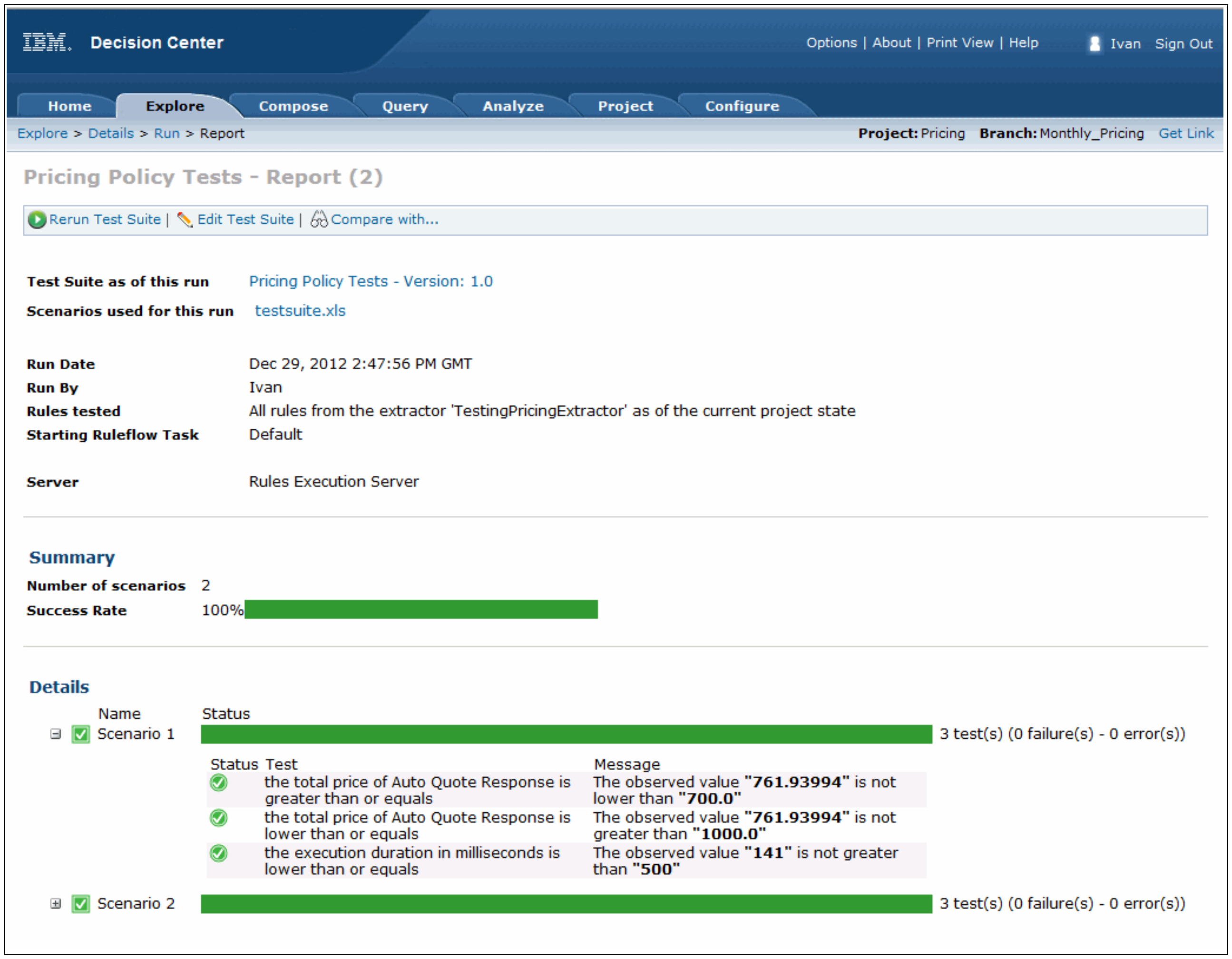
	A	B	C	D
1				
2		*Fill only the cells for the results you want to test...*		
3		Click here to access the help sheet		
4				
5		**Scenario ID**	the total price of Auto Quote Response is greater than or equals	the total price of Auto Quote Response is lower than or equals
9		**Scenario 1**	700	1000
10		**Scenario 2**	500	1000
11				

Expected Results
Expected Execution Details
HELP

Figure 9-59 Test suite expected results

The Expected Execution Details worksheet can also include performance criteria that can be tested to ensure that quality of service is met.

When the test suite is run, each scenario is checked against specific test criteria, as shown in Figure 9-60.



IBM. Decision Center
Options | About | Print View | Help
Ivan Sign Out
Home
Explore
Compose
Query
Analyze
Project
Configure
Explore > Details > Run > Report
Project: Pricing Branch: Monthly_Pricing Get Link
Rerun Test Suite | Edit Test Suite | Compare with...

Pricing Policy Tests - Report (2)

Test Suite as of this run	Pricing Policy Tests - Version: 1.0
Scenarios used for this run	testsuite.xls
Run Date	Dec 29, 2012 2:47:56 PM GMT
Run By	Ivan
Rules tested	All rules from the extractor 'TestingPricingExtractor' as of the current project state
Starting Ruleflow Task	Default
Server	Rules Execution Server

Summary

Number of scenarios	2
Success Rate	100%

Details

Name	Status
Scenario 1	3 test(s) (0 failure(s) - 0 error(s))
Scenario 2	3 test(s) (0 failure(s) - 0 error(s))

Scenario 1:

Status	Test	Message
✓	the total price of Auto Quote Response is greater than or equals	The observed value **"761.93994"** is not lower than **"700.0"**
✓	the total price of Auto Quote Response is lower than or equals	The observed value **"761.93994"** is not greater than **"1000.0"**
✓	the execution duration in milliseconds is lower than or equals	The observed value **"141"** is not greater than **"500"**

Figure 9-60 Test suite results

Although the example shown in Figure 9-60 on page 149 is simple, these test suites can be extended to provide the regression testing that is needed for these standard releases.

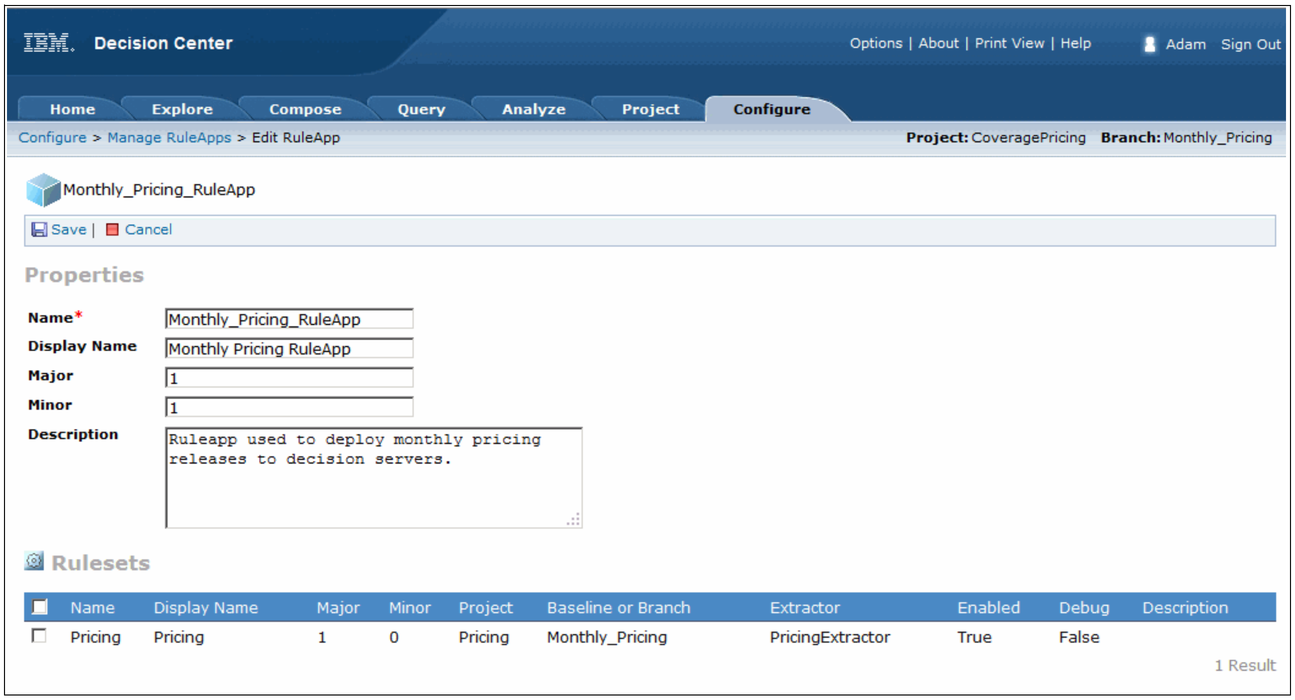
Each validation activity might test different groups of rule artifacts, so it might be appropriate to update the metadata (such as status) for these rule artifacts. Using smart queries and searches that are based on this metadata can then highlight which changed artifacts might not have been tested across the release, which then helps with the overall understanding of the quality of the testing and thus the rationale for approving a release for deployment.

9.4.7 Release baselining and deployment

Having completed the change and validation activities, the release is now ready to be built into a deployable RuleApp configuration item that may be deployed to staging or production environments. Although some release lifecycles might require a number of different RuleApps in order to cope with these different environments, this section describes the process for deploying to a production system. These steps are undertaken by Adam, the IT administrator.

RuleApp versioning

The first step is defining the versioning that is used for this release. The monthly pricing RuleApp must be modified to use the next minor version to be used for this release, as shown in Figure 9-61.



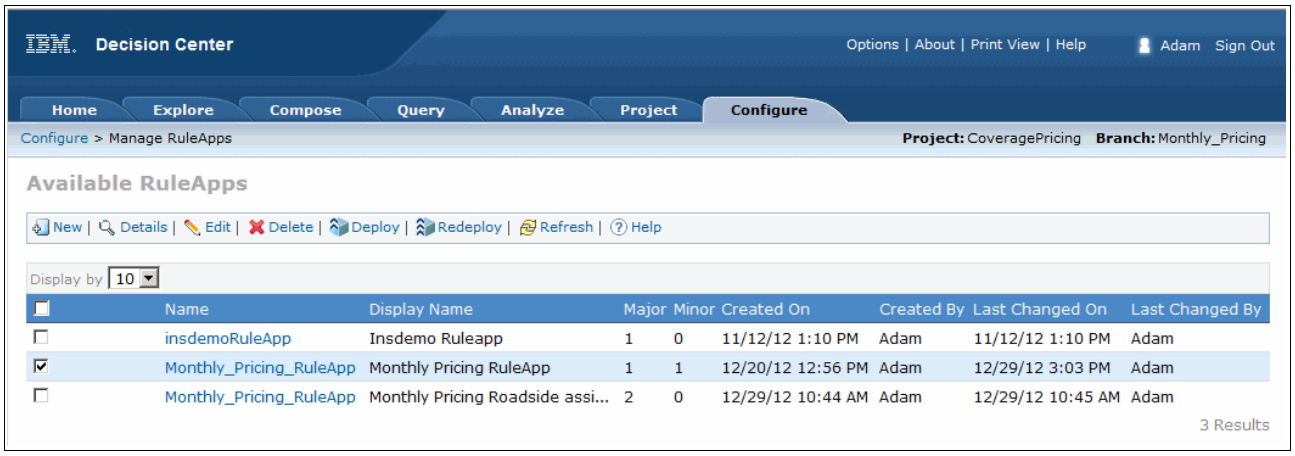
Name*	Monthly_Pricing_RuleApp
Display Name	Monthly Pricing RuleApp
Major	1
Minor	1
Description	Ruleapp used to deploy monthly pricing releases to decision servers.

Name	Display Name	Major	Minor	Project	Baseline or Branch	Extractor	Enabled	Debug	Description
Pricing	Pricing	1	0	Pricing	Monthly_Pricing	PricingExtractor	True	False	

1 Result

Figure 9-61 RuleApp minor version modification

In this case, the RuleApp version is defined as 1.1, resulting in the available RuleApps shown in Figure 9-62 on page 151.



IBM. Decision Center — Options | About | Print View | Help — Adam Sign Out

Home | Explore | Compose | Query | Analyze | Project | Configure

Configure > Manage RuleApps — **Project:** CoveragePricing **Branch:** Monthly_Pricing

Available RuleApps

New | Details | Edit | Delete | Deploy | Redeploy | Refresh | Help

Display by 10

	Name	Display Name	Major	Minor	Created On	Created By	Last Changed On	Last Changed By
☐	insdemoRuleApp	Insdemo Ruleapp	1	0	11/12/12 1:10 PM	Adam	11/12/12 1:10 PM	Adam
☑	Monthly_Pricing_RuleApp	Monthly Pricing RuleApp	1	1	12/20/12 12:56 PM	Adam	12/29/12 3:03 PM	Adam
☐	Monthly_Pricing_RuleApp	Monthly Pricing Roadside assi...	2	0	12/29/12 10:44 AM	Adam	12/29/12 10:45 AM	Adam

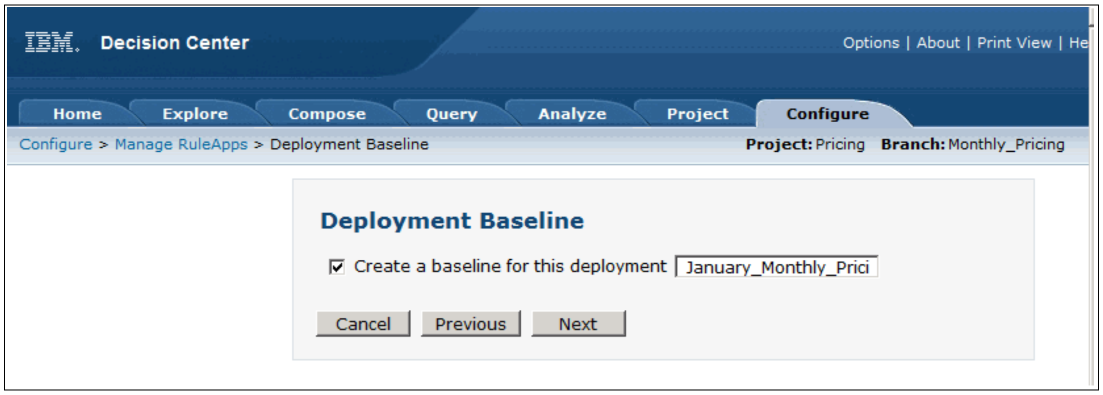
3 Results

Figure 9-62 Available RuleApps

From this RuleApp definition, the deployable artifact can now be generated clicking the **Deploy** link.

Generating the deployable RuleApp

The `deploy` command performs a number of steps, the first of which creates (optionally) a deployment baseline. This ties the rules in the RuleApp to their corresponding versions in the Decision Center, allowing traceability to the source of each rule that was run at run time. See Figure 9-63.



IBM. Decision Center — Options | About | Print View | He

Home | Explore | Compose | Query | Analyze | Project | Configure

Configure > Manage RuleApps > Deployment Baseline — **Project:** Pricing **Branch:** Monthly_Pricing

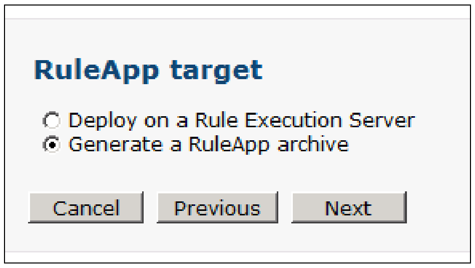
Deployment Baseline

☑ Create a baseline for this deployment: January_Monthly_Prici

Cancel | Previous | Next

Figure 9-63 Deployment baseline

The next step is to select the deployment target. For staging and test environments, it is typical to deploy straight to the appropriate Rule Execution Server, but in this case, a RuleApp archive is generated that can be deployed or placed as a configuration item in the IT configuration management system. See Figure 9-64.



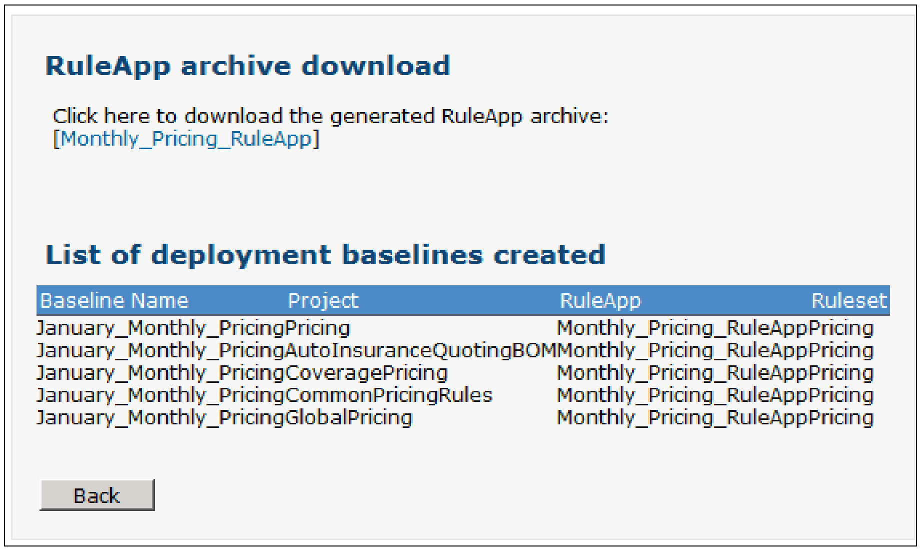
RuleApp target

○ Deploy on a Rule Execution Server
● Generate a RuleApp archive

Cancel | Previous | Next

Figure 9-64 RuleApp target

The Decision Center then builds and generates the RuleApp archive and lists the deployment baselines that are created for each of the projects in the decision service. The Monthly_Pricing_RuleApp can then be downloaded and placed in the configuration management system. See Figure 9-65.



RuleApp archive download

Click here to download the generated RuleApp archive:
[Monthly_Pricing_RuleApp]

List of deployment baselines created

Baseline Name	Project	RuleApp	Ruleset
January_Monthly_Pricing	Pricing	Monthly_Pricing_RuleApp	Pricing
January_Monthly_Pricing	AutoInsuranceQuotingBOM	Monthly_Pricing_RuleApp	Pricing
January_Monthly_Pricing	CoveragePricing	Monthly_Pricing_RuleApp	Pricing
January_Monthly_Pricing	CommonPricingRules	Monthly_Pricing_RuleApp	Pricing
January_Monthly_Pricing	GlobalPricing	Monthly_Pricing_RuleApp	Pricing

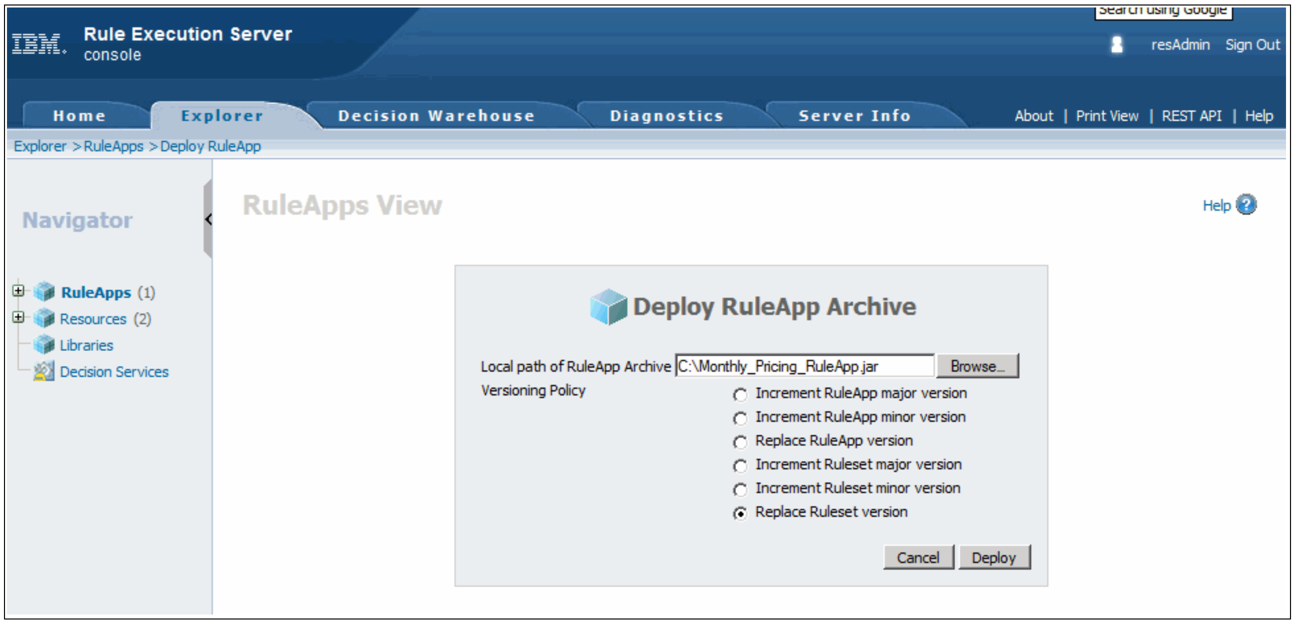
Figure 9-65 RuleApp archive generated

User Acceptance Testing

Now, the release is deployed to a User Acceptance Testing (UAT) environment, allowing Terry the QA engineer to ensure that the new release works correctly in an environment that is identical to the production environment. This deployment can be undertaken directly from Decision Center or the RuleApp archive that was generated in “Generating the deployable RuleApp” on page 151 can be deployed on to the UAT decision server. The governance and approvals of this acceptance testing should take place in the context of the UAT environment testing process. Any defects that are identified means that the release must be reworked and a new baseline be generated.

Deployment to the production system

The final stage of deployment to production is normally undertaken by using scripts. This section shows a manual deployment to the rule execution server. The RuleApp archive generated earlier is deployed to the rule execution server, as shown in Figure 9-66.

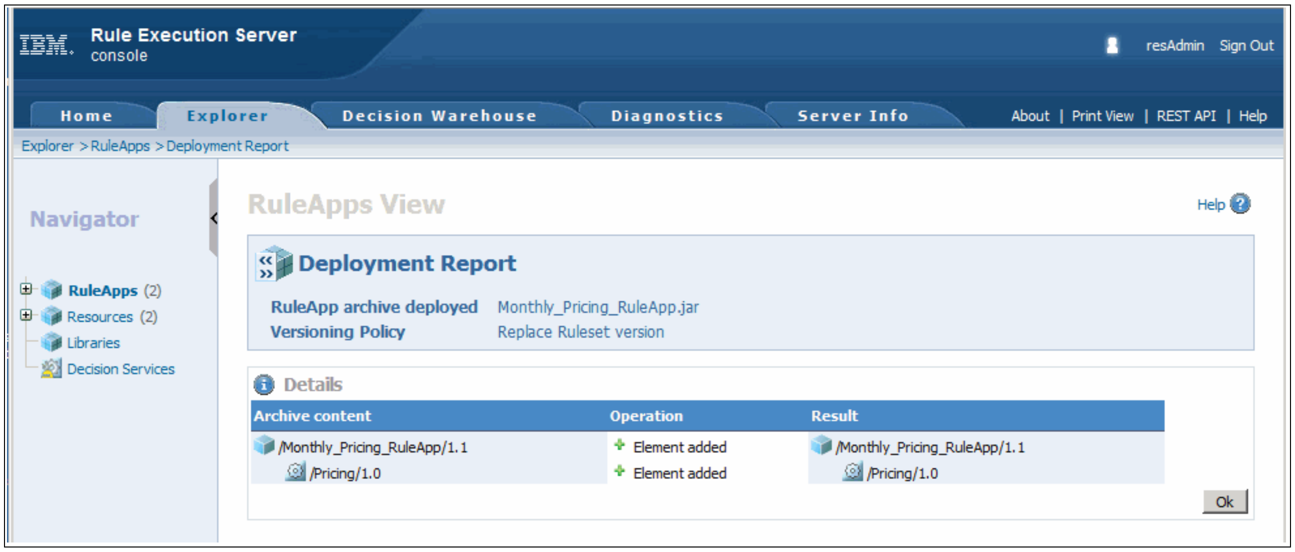


Local path of RuleApp Archive: C:\Monthly_Pricing_RuleApp.jar

Versioning Policy:
- ( ) Increment RuleApp major version
- ( ) Increment RuleApp minor version
- ( ) Replace RuleApp version
- ( ) Increment Ruleset major version
- ( ) Increment Ruleset minor version
- [✓ Replace Ruleset version]

Figure 9-66 Rule Execution Server deployment

Because the versions to be used in the RuleApp definition are configured, a replacement rule set version versioning policy is used. This means that if the version that is deployed already exists (as might be the case for an emergency fix), the rules in this release overwrite the existing rules. The rules are then deployed to the Rule Execution server by using the versioning that is defined, as shown in Figure 9-67.

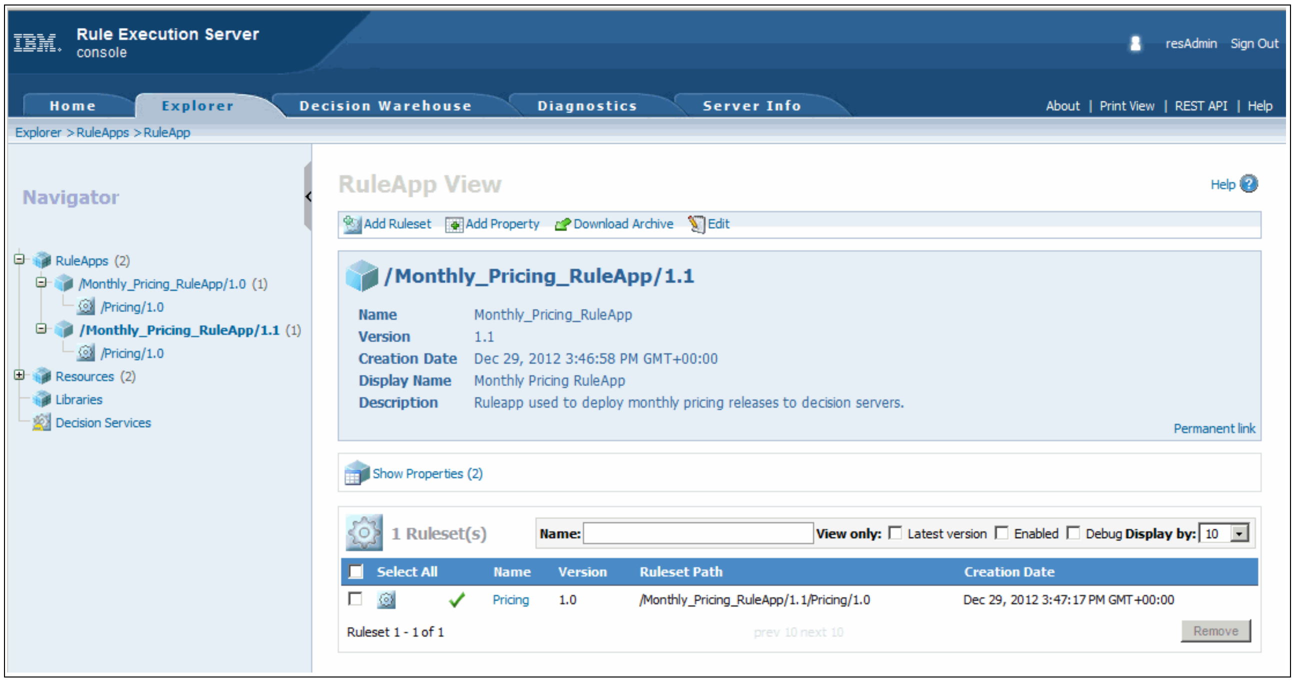


RuleApp archive deployed	Monthly_Pricing_RuleApp.jar
Versioning Policy	Replace Ruleset version

Archive content	Operation	Result
/Monthly_Pricing_RuleApp/1.1	Element added	/Monthly_Pricing_RuleApp/1.1
/Pricing/1.0	Element added	/Pricing/1.0

Figure 9-67 Rule Execution Server deployment complete

This release is then available concurrently with any previous monthly pricing releases that might still be needed to support decisions in long running processes, as shown in Figure 9-68 on page 154.



/Monthly_Pricing_RuleApp/1.1

Name	Monthly_Pricing_RuleApp
Version	1.1
Creation Date	Dec 29, 2012 3:46:58 PM GMT+00:00
Display Name	Monthly Pricing RuleApp
Description	Ruleapp used to deploy monthly pricing releases to decision servers.

Select All	Name	Version	Ruleset Path	Creation Date
✓	Pricing	1.0	/Monthly_Pricing_RuleApp/1.1/Pricing/1.0	Dec 29, 2012 3:47:17 PM GMT+00:00

Figure 9-68 Rule Execution Server Explorer showing the deployed RuleApps

9.5 Technical Change Process example

This section examines how a normal or major release is set up that requires changes to be made that affect both the business rules and the technical solutions that invoke the rules. This follows the types of activity that are identified in the Technical Change Process in the Chapter 8, “Processes” on page 95.

In this case, we describe how the new information that is needed to support the roadside assistance initiative is included in to the decision service. This section concentrates on the definition of a technical release and how this can be tied to the software development tools used by the solutions.

The section focuses on activities that are undertaken in the Rule Designer environment where Ivan can specify changes to the Execution Object Model (XOM) and Ruleset Signature, thus allowing the new information to be passed consistently from the new solution to the decision service release. Having completed these IT-centric changes, integration testing then occurs. This testing is not described in detail in this book, as it depends greatly on the integration techniques that is being used between the solution and decision service. Following this testing, the scenario shows how the new release with the modified information model can then be synchronized with Decision Center to allow the new information model to be used by the business in this and future releases.

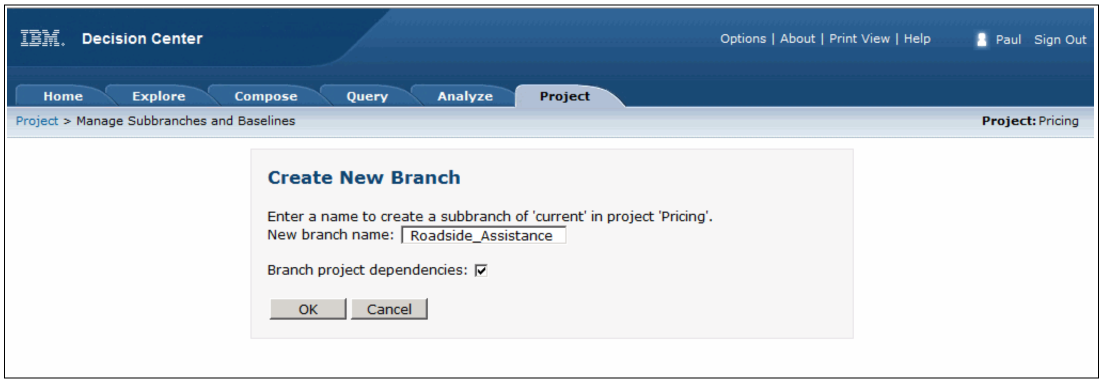
9.5.1 Technical release definition

This section describes how to define a technical release.

Creating the release

In the technical release lifecycle, many of the activities take place within the Eclipse Rule Designer environment. For this reason, the changes are managed in a single Roadside_Assistance branch rather than creating subbranches for individual change activities.

The branch is first created from the starting release by using the manage subbranches and baselines view of the project tab. In this case, the current / trunk branch is chosen as the basis for the branch from the Pricing project. See Figure 9-69.



Project > Manage Subbranches and Baselines

Project: Pricing

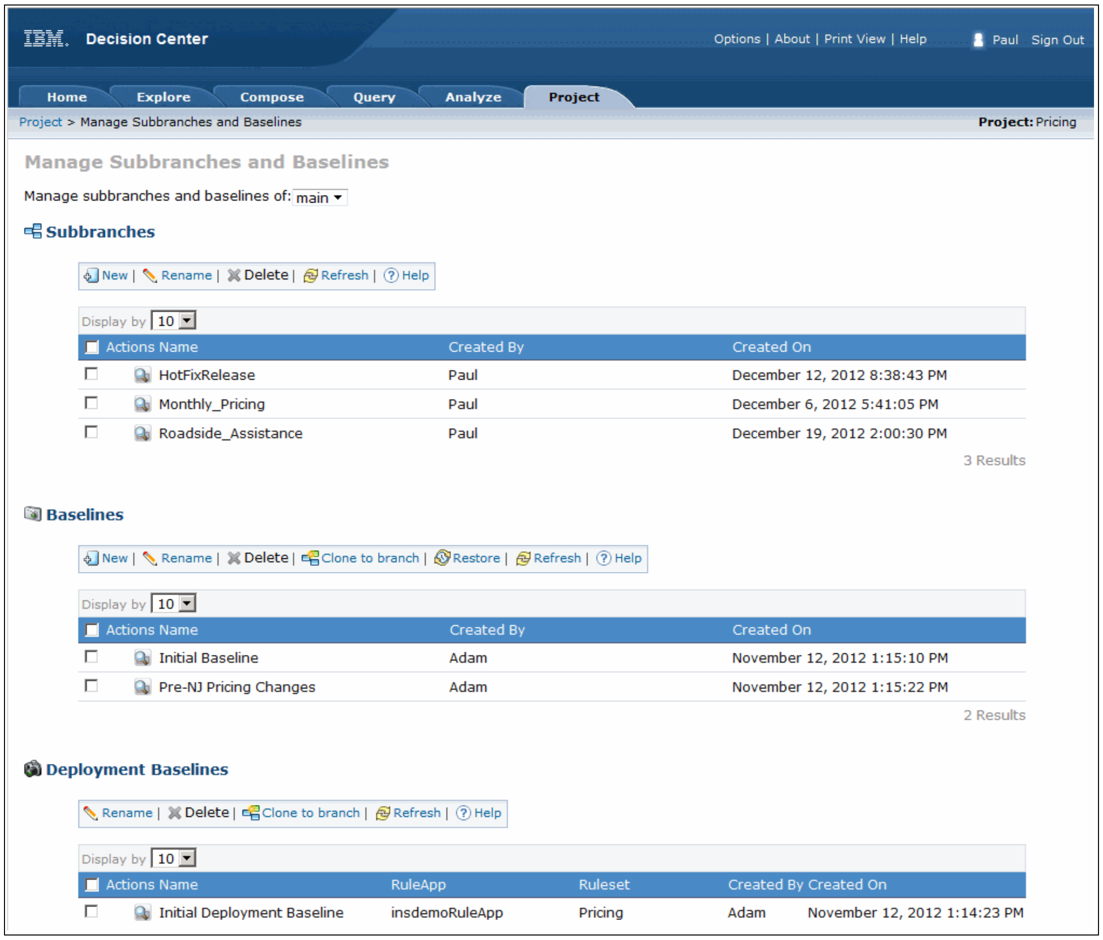
Create New Branch

Enter a name to create a subbranch of 'current' in project 'Pricing'.
New branch name: Roadside_Assistance

Branch project dependencies: ✓

Figure 9-69 Roadside Assistance release creation

This branch is then visible to those users with the appropriate permissions. See Figure 9-70.


IBM. Decision Center
Options | About | Print View | Help
Paul Sign Out
Home
Explore
Compose
Query
Analyze
Project

Project > Manage Subbranches and Baselines   **Project:** Pricing

Manage Subbranches and Baselines

Manage subbranches and baselines of: main

Subbranches

New | Rename | Delete | Refresh | Help
Display by 10

Actions Name	Created By	Created On
HotFixRelease	Paul	December 12, 2012 8:38:43 PM
Monthly_Pricing	Paul	December 6, 2012 5:41:05 PM
Roadside_Assistance	Paul	December 19, 2012 2:00:30 PM

3 Results

Baselines

New | Rename | Delete | Clone to branch | Restore | Refresh | Help
Display by 10

Actions Name	Created By	Created On
Initial Baseline	Adam	November 12, 2012 1:15:10 PM
Pre-NJ Pricing Changes	Adam	November 12, 2012 1:15:22 PM

2 Results

Deployment Baselines

Actions Name	RuleApp	Ruleset	Created By	Created On
Initial Deployment Baseline	insdemoRuleApp	Pricing	Adam	November 12, 2012 1:14:23 PM

Figure 9-70 Pricing branches after adding roadside assistance

Establishing the release workspace

In order for the IT group to work on the release, they must bring the release into an Eclipse workspace. This workspace is where IT changes can be made and synchronized with the source code management system. As this release involves IT changes, the interfaces, Java code, and schemas that are used in all these changes are part of a new branch in the source code management system, so it is recommended that this release adopts a naming or identification system that can be tied to the major release that is being created.

In this example, Ivan establishes a new blank workspace called Roadside_Assistance. He then must go through and synchronize the projects that he is going to work on. He starts with the lowest project in the dependency hierarchy and uses the Rule Project from Decision Center wizard to import it. See Figure 9-71.

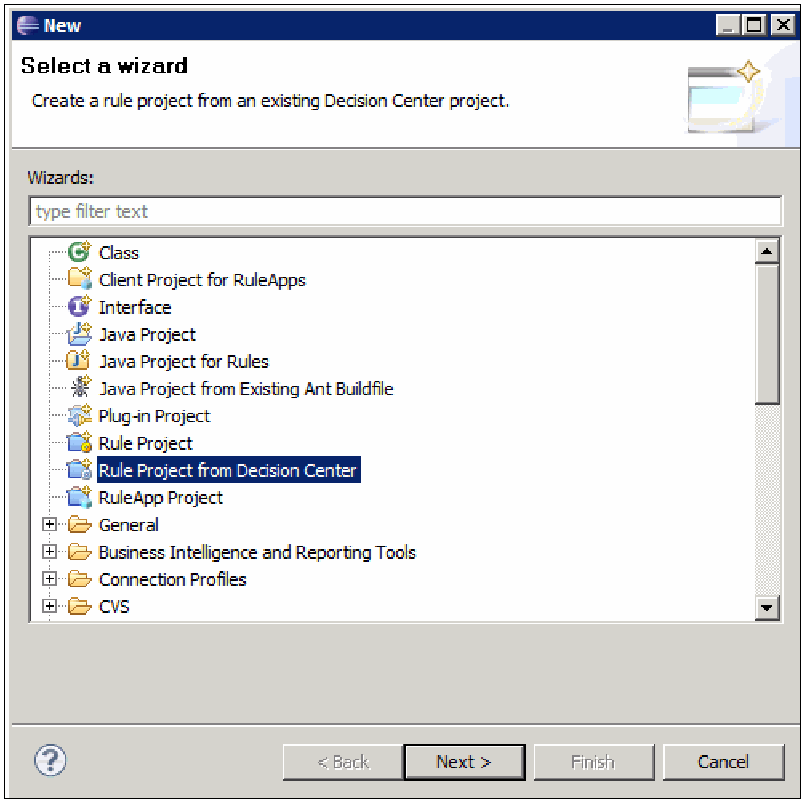
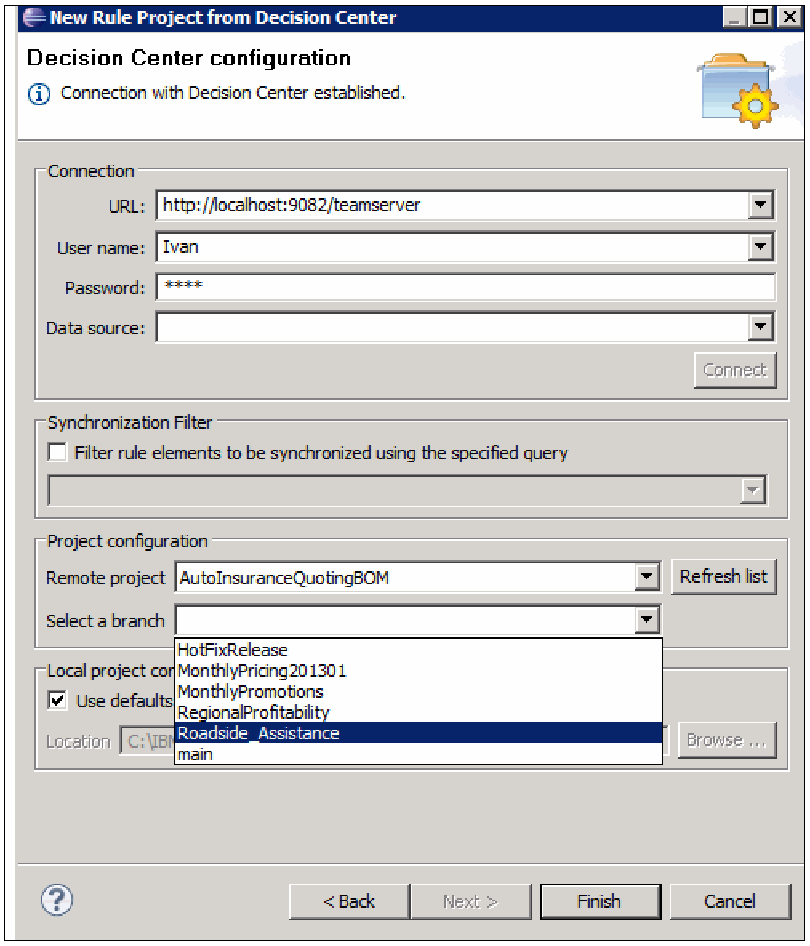

New
Select a wizard
Create a rule project from an existing Decision Center project.
Wizards:
type filter text
Class
Client Project for RuleApps
Interface
Java Project
Java Project for Rules
Java Project from Existing Ant Buildfile
Plug-in Project
Rule Project
Rule Project from Decision Center
RuleApp Project
General
Business Intelligence and Reporting Tools
Connection Profiles
CVS
< Back
Next >
Finish
Cancel

Figure 9-71 Rule project import from Decision Center

In Figure 9-72, Ivan selects the connection to Decision Center, the project he wants to import, and the Roadside_Assistance branch that he is going to implement the changes in.


New Rule Project from Decision Center

Decision Center configuration

Connection with Decision Center established.

Connection

URL: http://localhost:9082/teamserver

User name: Ivan

Password: ****

Data source:

Synchronization Filter

Filter rule elements to be synchronized using the specified query

Project configuration

Remote project: AutoInsuranceQuotingBOM

Select a branch:

HotFixRelease
MonthlyPricing201301
MonthlyPromotions
RegionalProfitability
Roadside_Assistance
main

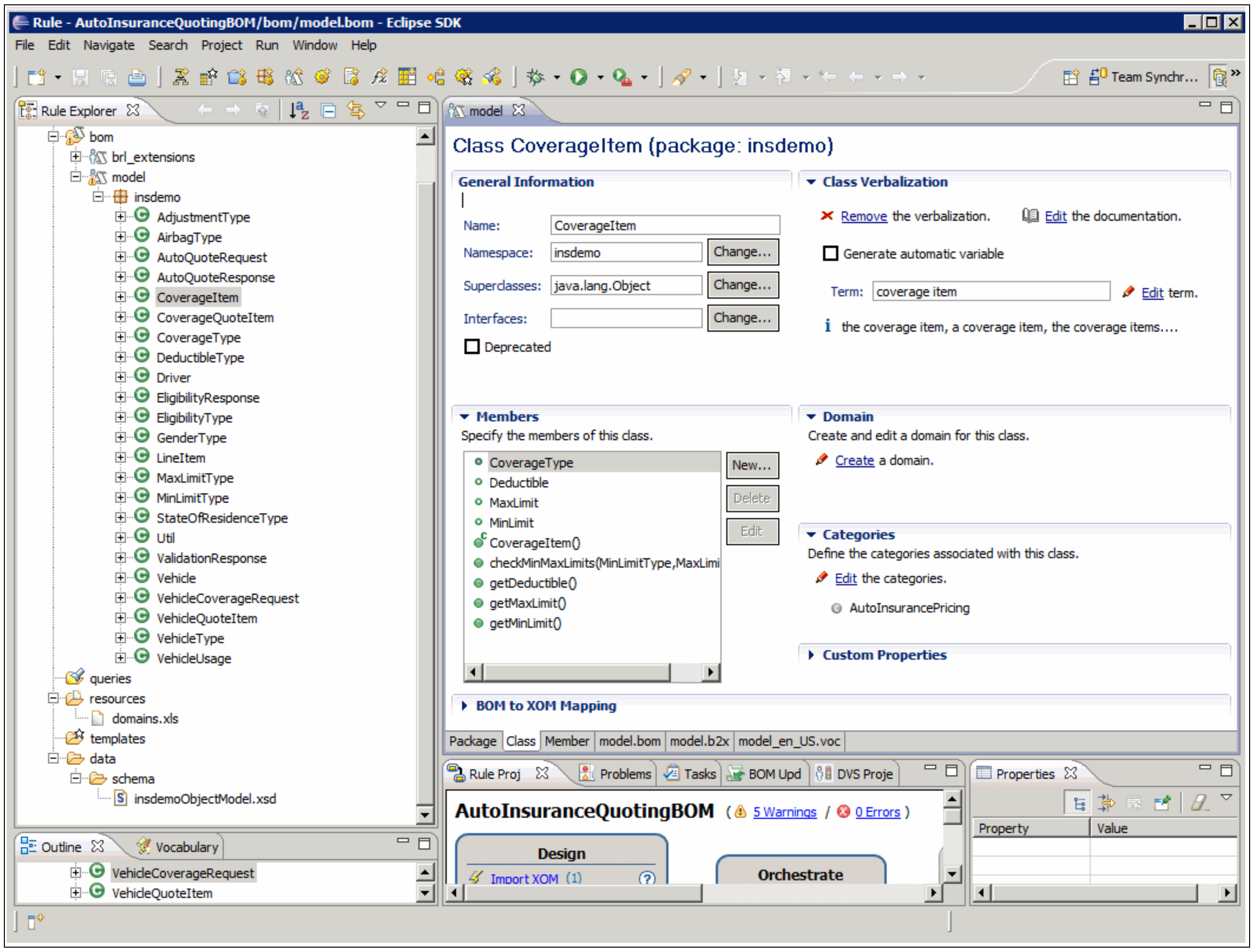
Local project con

Use defaults

Location C:\IBI

Figure 9-72 Import project branch

After the branch is selected, the contents of the project are synchronized in to the workspace, as shown in Figure 9-73.



Class CoverageItem (package: insdemo)

General Information

Name: CoverageItem

Namespace: insdemo [Change...]

Superclasses: java.lang.Object [Change...]

Interfaces: [Change...]

☐ Deprecated

Class Verbalization

Remove the verbalization. Edit the documentation.

☐ Generate automatic variable

Term: coverage item Edit term.

the coverage item, a coverage item, the coverage items....

Members

Specify the members of this class.

- CoverageType
- Deductible
- MaxLimit
- MinLimit
- CoverageItem()
- checkMinMaxLimits(MinLimitType,MaxLimi
- getDeductible()
- getMaxLimit()
- getMinLimit()

[New...] [Delete] [Edit]

Domain

Create and edit a domain for this class.

Create a domain.

Categories

Define the categories associated with this class.

Edit the categories.

AutoInsurancePricing

Custom Properties

BOM to XOM Mapping

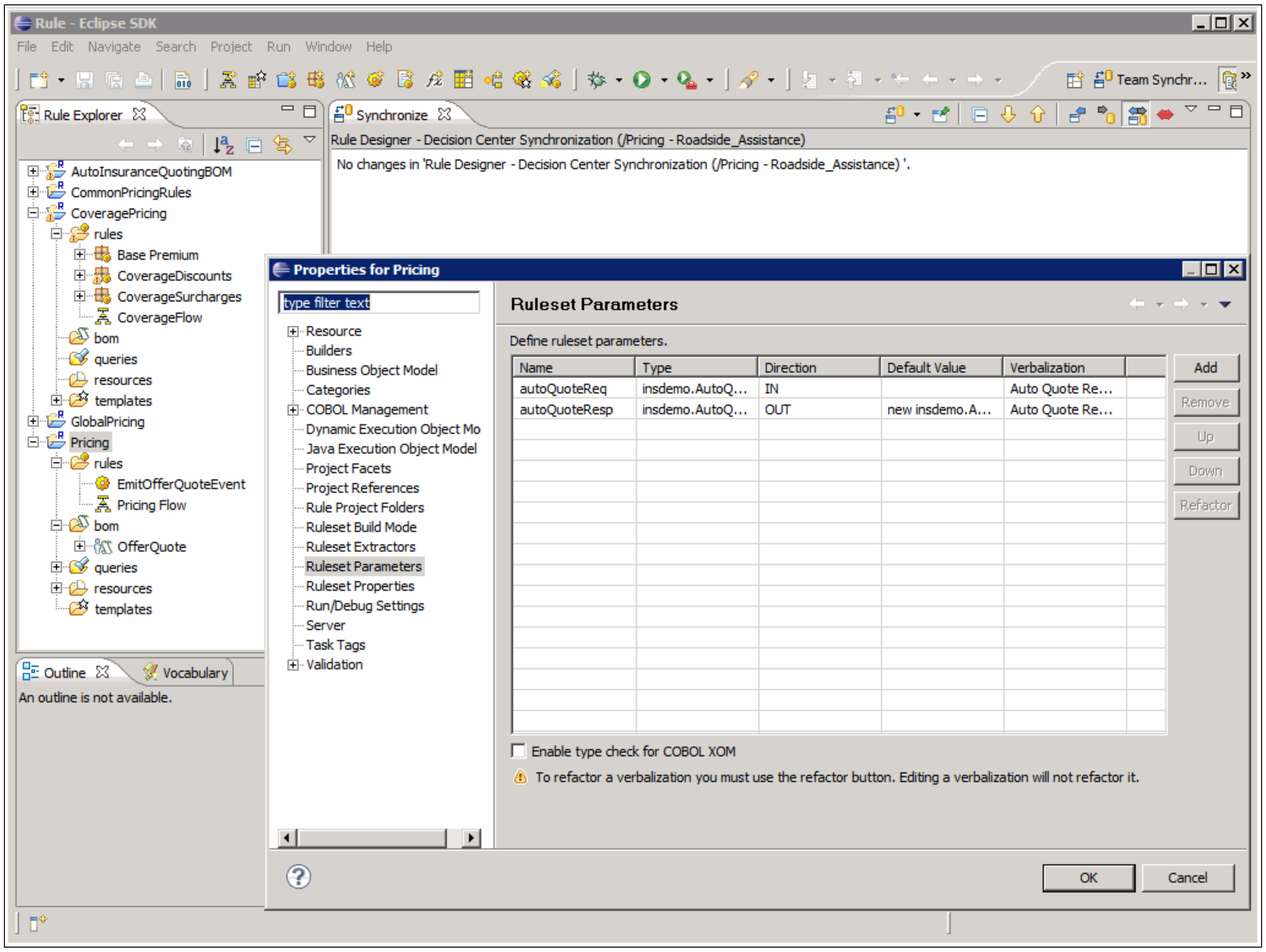
AutoInsuranceQuotingBOM (5 Warnings / 0 Errors)

Figure 9-73 Rule Designer workspace with imported project

Figure 9-73 shows the current information that is made available about any type of coverage, which could be modified to define the new characteristics that are used in roadside assistance. This is, however, an architectural choice, as described in “Completing the synchronization in to the workspace”.

Completing the synchronization in to the workspace

The process that is described in “Establishing the release workspace” on page 156 is then repeated for each of the projects in the pricing decision service hierarchy. This results in a workspace that looks like Figure 9-74.



Rule Designer - Decision Center Synchronization (/Pricing - Roadside_Assistance)

No changes in 'Rule Designer - Decision Center Synchronization (/Pricing - Roadside_Assistance) '.

An outline is not available.

Ruleset Parameters

Define ruleset parameters.

Name	Type	Direction	Default Value	Verbalization
autoQuoteReq	insdemo.AutoQ...	IN		Auto Quote Re...
autoQuoteResp	insdemo.AutoQ...	OUT	new insdemo.A...	Auto Quote Re...

☐ Enable type check for COBOL XOM

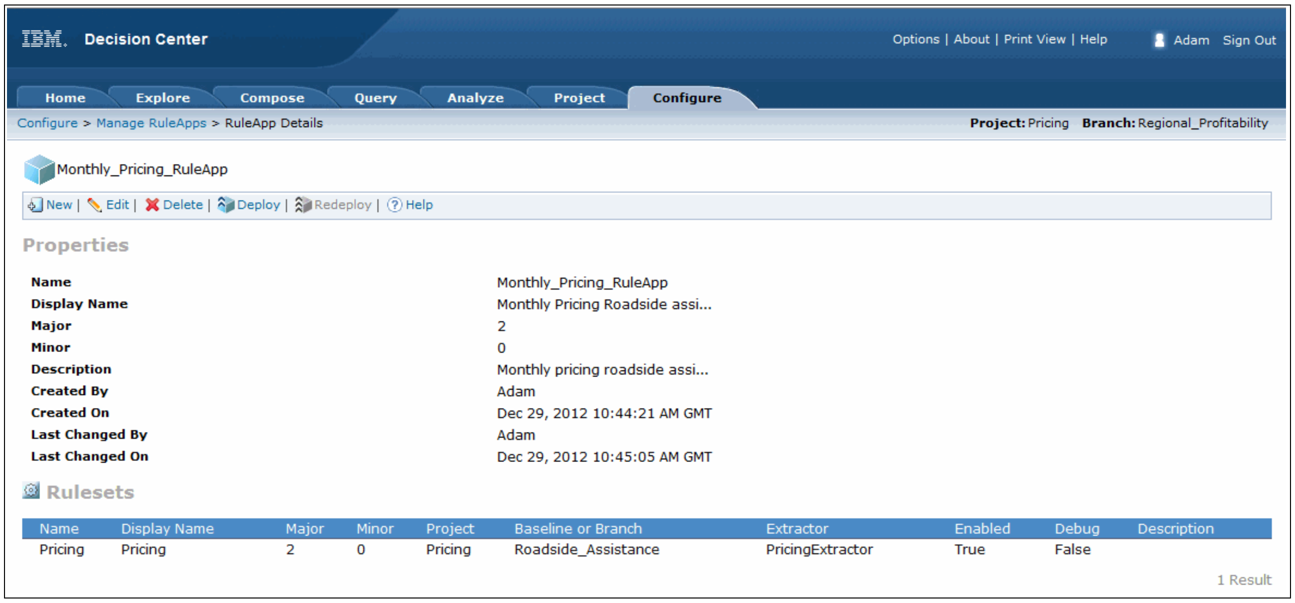
To refactor a verbalization you must use the refactor button. Editing a verbalization will not refactor it.

Figure 9-74 Roadside assistance workspace synchronized with release

This final view shows the workspace project structure that can be synchronized and managed in the IT source code management system together with any other IT artifacts that must be in place to integrate the rules into the solution. In particular, this view now shows the input and output parameters that are included to the decision service, which represents another option for passing the level of roadside assistance in to the pricing rules. “Changing the rule set parameters” on page 167 describes how to change these parameters.

Deployment RuleApp definition

The last step in the release definition is to define how this release is deployed in to the production (or staging) environments. This step is undertaken through a RuleApp definition that defines the versioning and rule set paths to be used by the solutions when you invoke this release. The RuleApp also provides the means to define the branch on which the rules are taken from. A second RuleApp is configured for the RoadsideAssistance release, as shown in Figure 9-75.


IBM. Decision Center
Options | About | Print View | Help
Adam Sign Out
Home
Explore
Compose
Query
Analyze
Project
Configure
Configure > Manage RuleApps > RuleApp Details
Project: Pricing Branch: Regional_Profitability
Monthly_Pricing_RuleApp
New | Edit | Delete | Deploy | Redeploy | Help
Properties
Rulesets
1 Result

Name	Monthly_Pricing_RuleApp
Display Name	Monthly Pricing Roadside assi...
Major	2
Minor	0
Description	Monthly pricing roadside assi...
Created By	Adam
Created On	Dec 29, 2012 10:44:21 AM GMT
Last Changed By	Adam
Last Changed On	Dec 29, 2012 10:45:05 AM GMT

Name	Display Name	Major	Minor	Project	Baseline or Branch	Extractor	Enabled	Debug	Description
Pricing	Pricing	2	0	Pricing	Roadside_Assistance	PricingExtractor	True	False	

Figure 9-75 Roadside Assistance release RuleApp

Although the Pricing branch specifies the Roadside_Assistance branch, the RuleApp name and rule set name are the same as for the monthly pricing RuleApp. In this case, the major versions are increased to reflect the new major version and to allow the solutions to be specific about which release of the decision service you want to integrate with. You can then specify that they work only with 2.* releases of the decision service by specifying this setting in the rule set path that is used to identify the decision to be invoked.

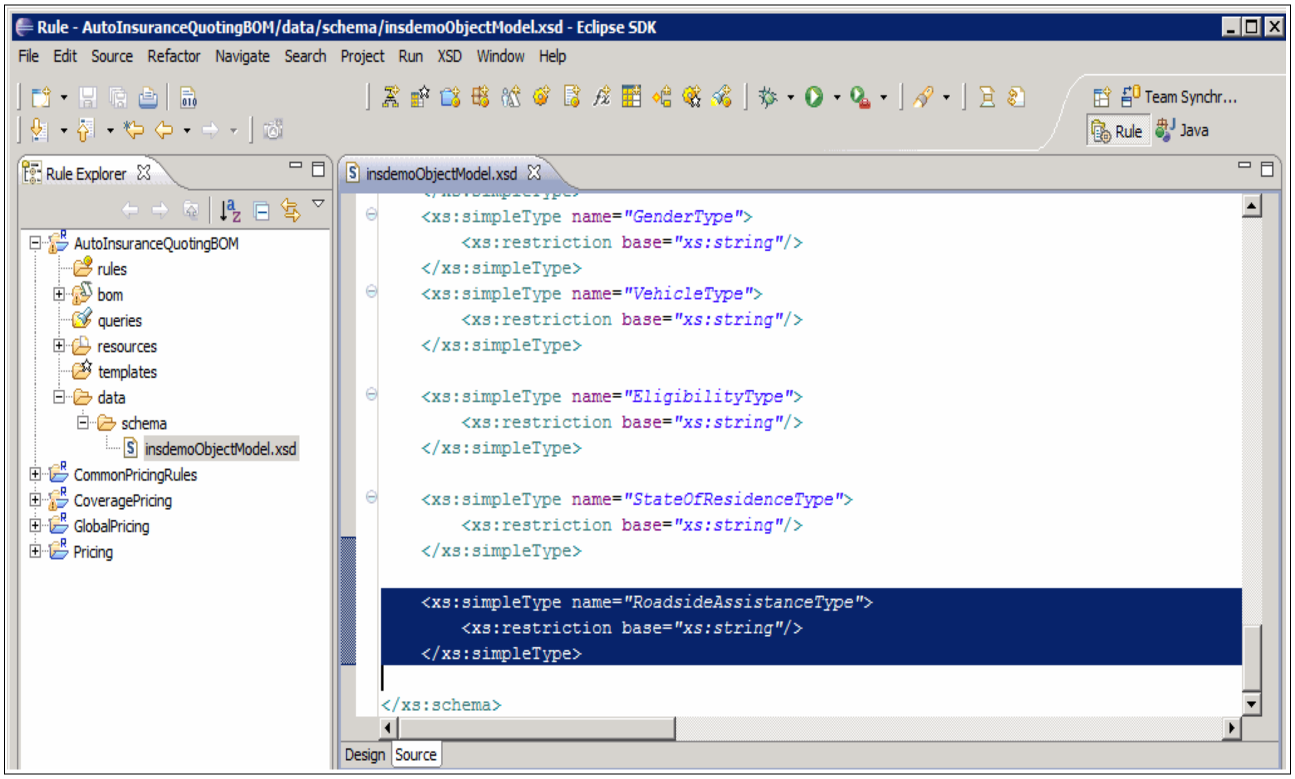
9.5.2 Execution Object Model change activity

A common requirement when you make changes to a decision service is the need to use new types of information within the rules. Although the inclusion of this new information in a decision must be taken into account in the signature, a precursor of this task is that Operational Decision Manager recognizes the types that the information is provided in. This is defined in the Execution Object Model (XOM). In addition to the changes to the XOM to support the signature, the Business Object Model that defines the way that the rules refer to this information must be extended. This section reviews how these rules change for our technical scenario, where we introduce a new data type that represents, home, regional, or nation-wide roadside assistance coverage.

Modifying the Execution Object Model

In the sample scenario, the XOM in use is based on a schema. Into this schema, we add a new type, thus making it compatible with existing definitions. Normal software versioning and configuration management practices must be employed here to make sure that the applications that use the new roadside assistance release use this new version of the schema.

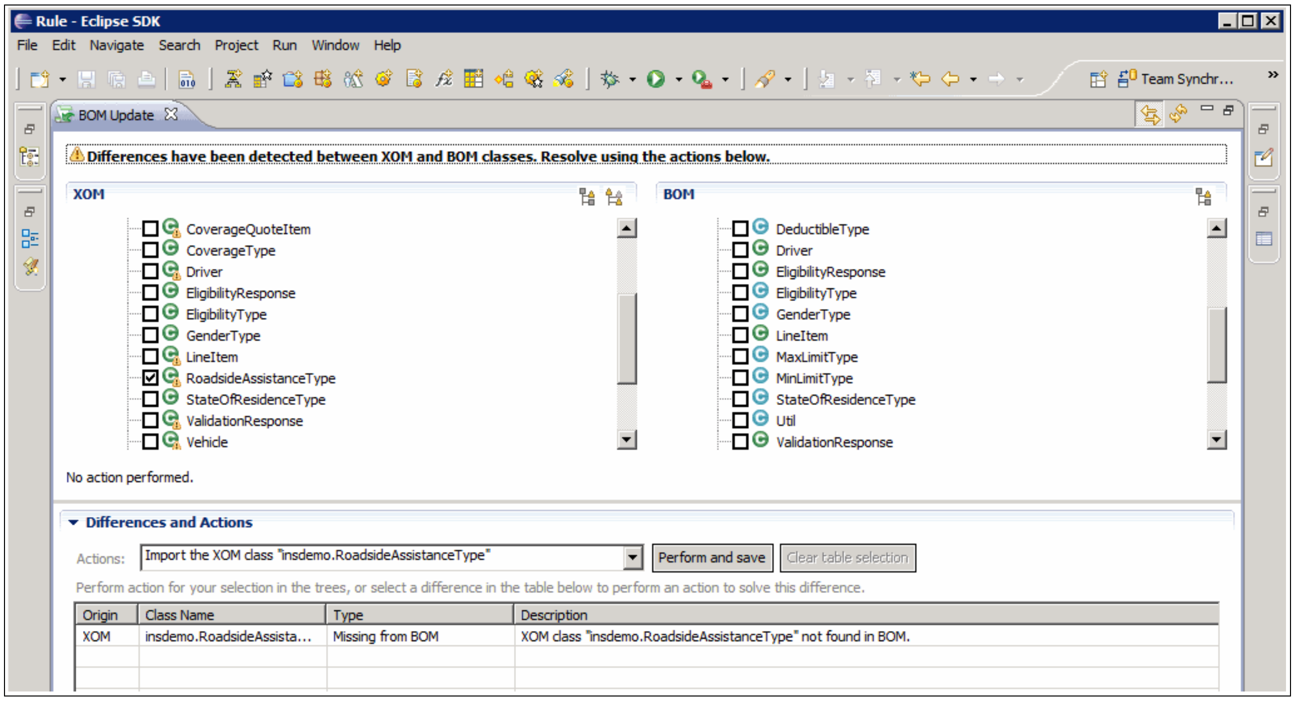
This task can be undertaken by Ivan using the Eclipse workspace that he set up previously. See Figure 9-76.



```
<xs:simpleType name="GenderType">
    <xs:restriction base="xs:string"/>
</xs:simpleType>
<xs:simpleType name="VehicleType">
    <xs:restriction base="xs:string"/>
</xs:simpleType>

<xs:simpleType name="EligibilityType">
    <xs:restriction base="xs:string"/>
</xs:simpleType>

<xs:simpleType name="StateOfResidenceType">
    <xs:restriction base="xs:string"/>
</xs:simpleType>

<xs:simpleType name="RoadsideAssistanceType">
    <xs:restriction base="xs:string"/>
</xs:simpleType>

</xs:schema>
```

Figure 9-76   XOM update to reflect new types

After the XOM is updated, it is possible to use a BOM update to make the RoadsideAssistanceType available within the vocabulary. By selecting the new RoadsideAssistanceType in the XOM, the BOM update can produce an equivalent BOM type, as shown in Figure 9-77.



Differences have been detected between XOM and BOM classes. Resolve using the actions below.

XOM
- [ ] CoverageQuoteItem
- [ ] CoverageType
- [ ] Driver
- [ ] EligibilityResponse
- [ ] EligibilityType
- [ ] GenderType
- [ ] LineItem
- [x] RoadsideAssistanceType
- [ ] StateOfResidenceType
- [ ] ValidationResponse
- [ ] Vehicle

BOM
- [ ] DeductibleType
- [ ] Driver
- [ ] EligibilityResponse
- [ ] EligibilityType
- [ ] GenderType
- [ ] LineItem
- [ ] MaxLimitType
- [ ] MinLimitType
- [ ] StateOfResidenceType
- [ ] Util
- [ ] ValidationResponse

No action performed.

Differences and Actions

Actions: Import the XOM class "insdemo.RoadsideAssistanceType"

Perform action for your selection in the trees, or select a difference in the table below to perform an action to solve this difference.

Origin	Class Name	Type	Description
XOM	insdemo.RoadsideAssista...	Missing from BOM	XOM class "insdemo.RoadsideAssistanceType" not found in BOM.

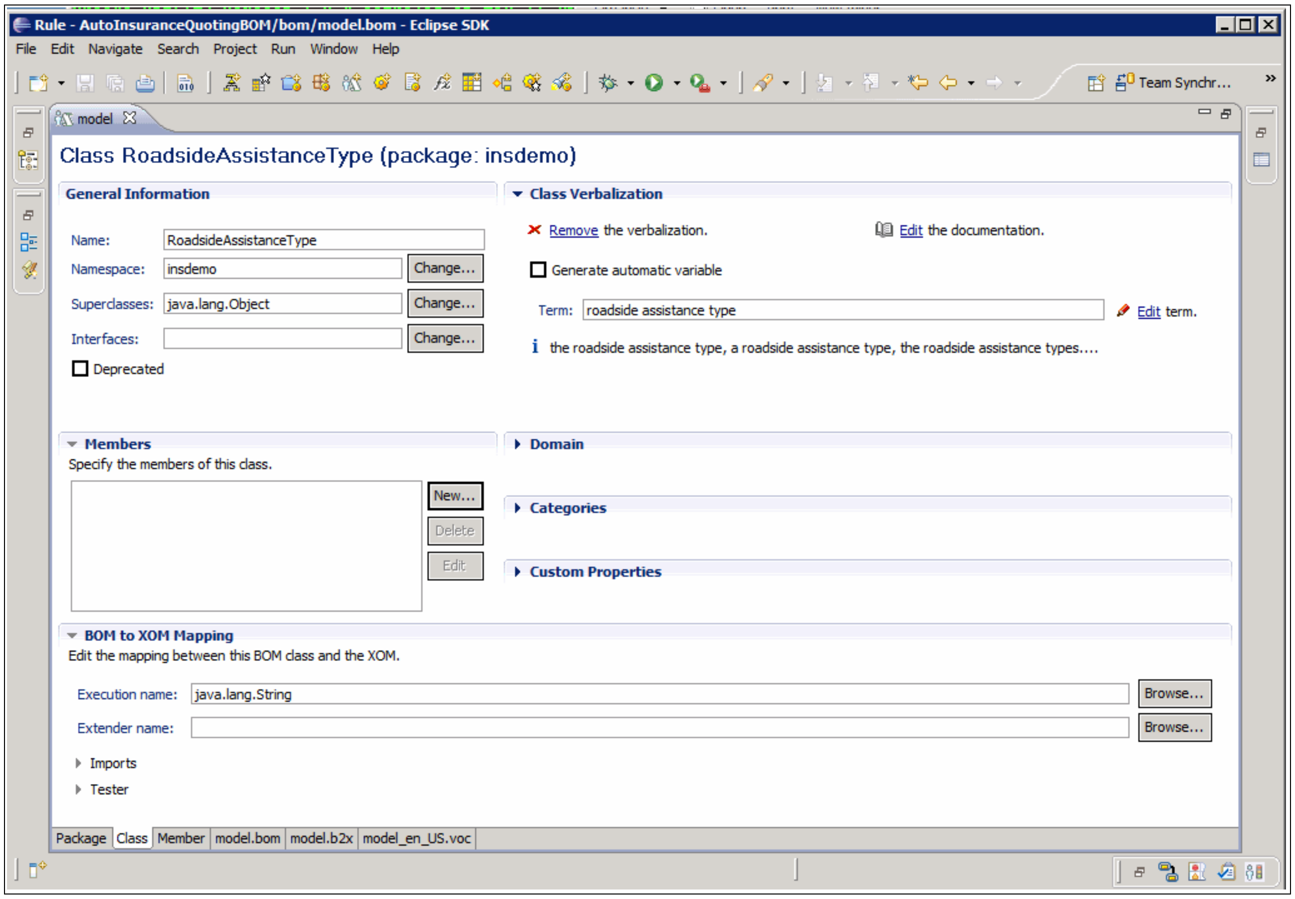
Figure 9-77 BOM update

By selecting **Perform and save**, the new class is generated in the BOM.

Defining the Business Object Model vocabulary

There is now a class entry in the BOM. You now must define how it is verbalized and the values that the rules should use.

The entry that is generated by default from an XML simple type contains a number of operations and is based on an XML string. By replacing the RoadsideAssistanceType with a String Execution type, you can define, in the BOM, the values that you must support and how they appear in the rules. See Figure 9-78.


Rule - AutoInsuranceQuotingBOM/bom/model.bom - Eclipse SDK

Class RoadsideAssistanceType (package: insdemo)

General Information

Name: RoadsideAssistanceType
Namespace: insdemo
Superclasses: java.lang.Object
Interfaces:
☐ Deprecated

Class Verbalization

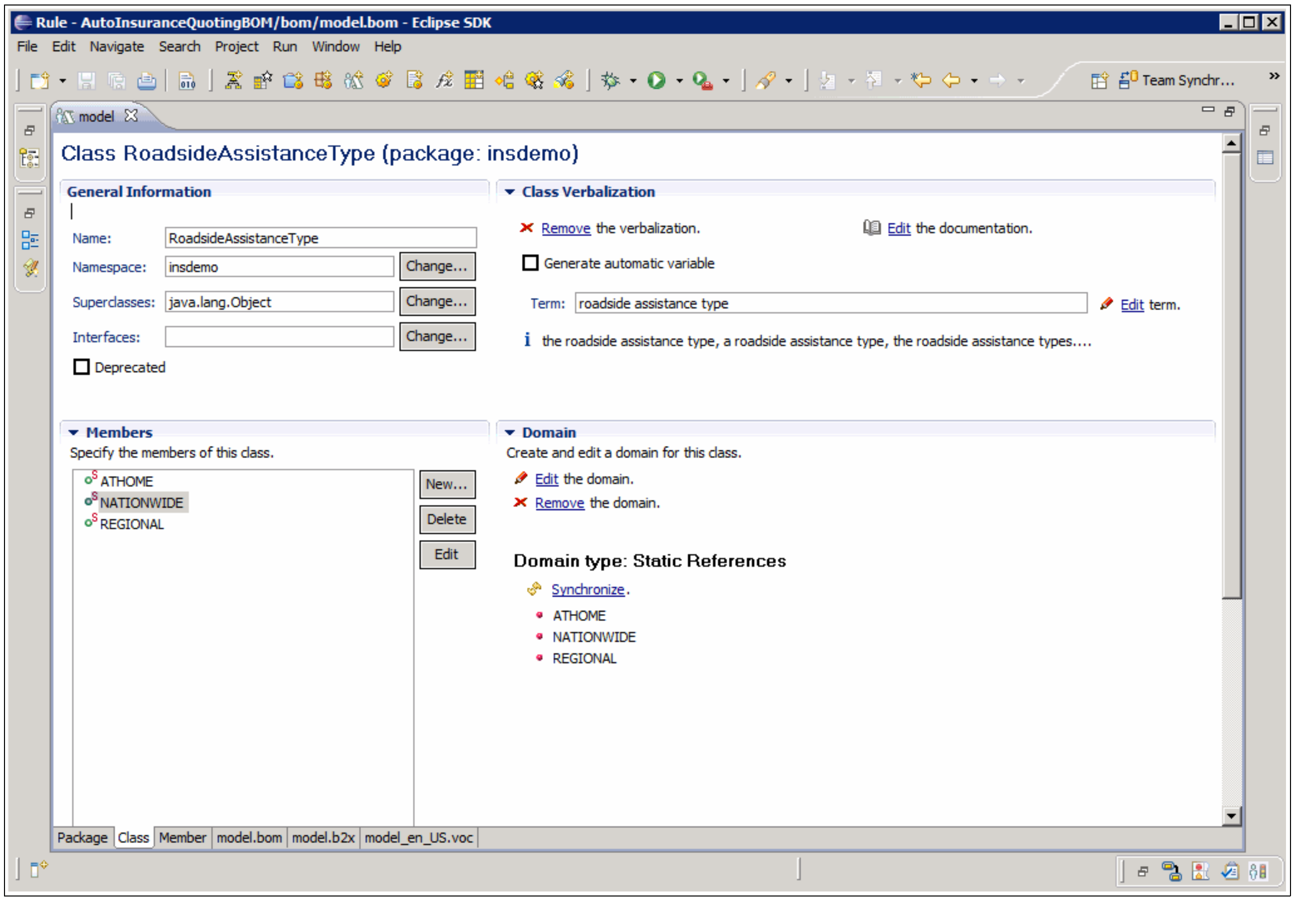
Remove the verbalization. Edit the documentation.
☐ Generate automatic variable
Term: roadside assistance type
the roadside assistance type, a roadside assistance type, the roadside assistance types....

Members
Specify the members of this class.

BOM to XOM Mapping
Edit the mapping between this BOM class and the XOM.
Execution name: java.lang.String
Extender name:

Figure 9-78 Roadside Assistance BOM entry

Now that the basic type is set up, you must identify static members to represent each of the values that you support. Each value entry is set up as a static readonly member of the RoadsideAssistanceType. Each of these members is then included in the domain that represents the set of values that are possible. See Figure 9-79.


Rule - AutoInsuranceQuotingBOM/bom/model.bom - Eclipse SDK

Class RoadsideAssistanceType (package: insdemo)

General Information

Name: RoadsideAssistanceType
Namespace: insdemo
Superclasses: java.lang.Object
Interfaces:
Deprecated

Class Verbalization

Remove the verbalization. Edit the documentation.
Generate automatic variable
Term: roadside assistance type Edit term.
the roadside assistance type, a roadside assistance type, the roadside assistance types....

Members

Specify the members of this class.

ATHOME
NATIONWIDE
REGIONAL

Domain

Create and edit a domain for this class.
Edit the domain.
Remove the domain.

Domain type: Static References

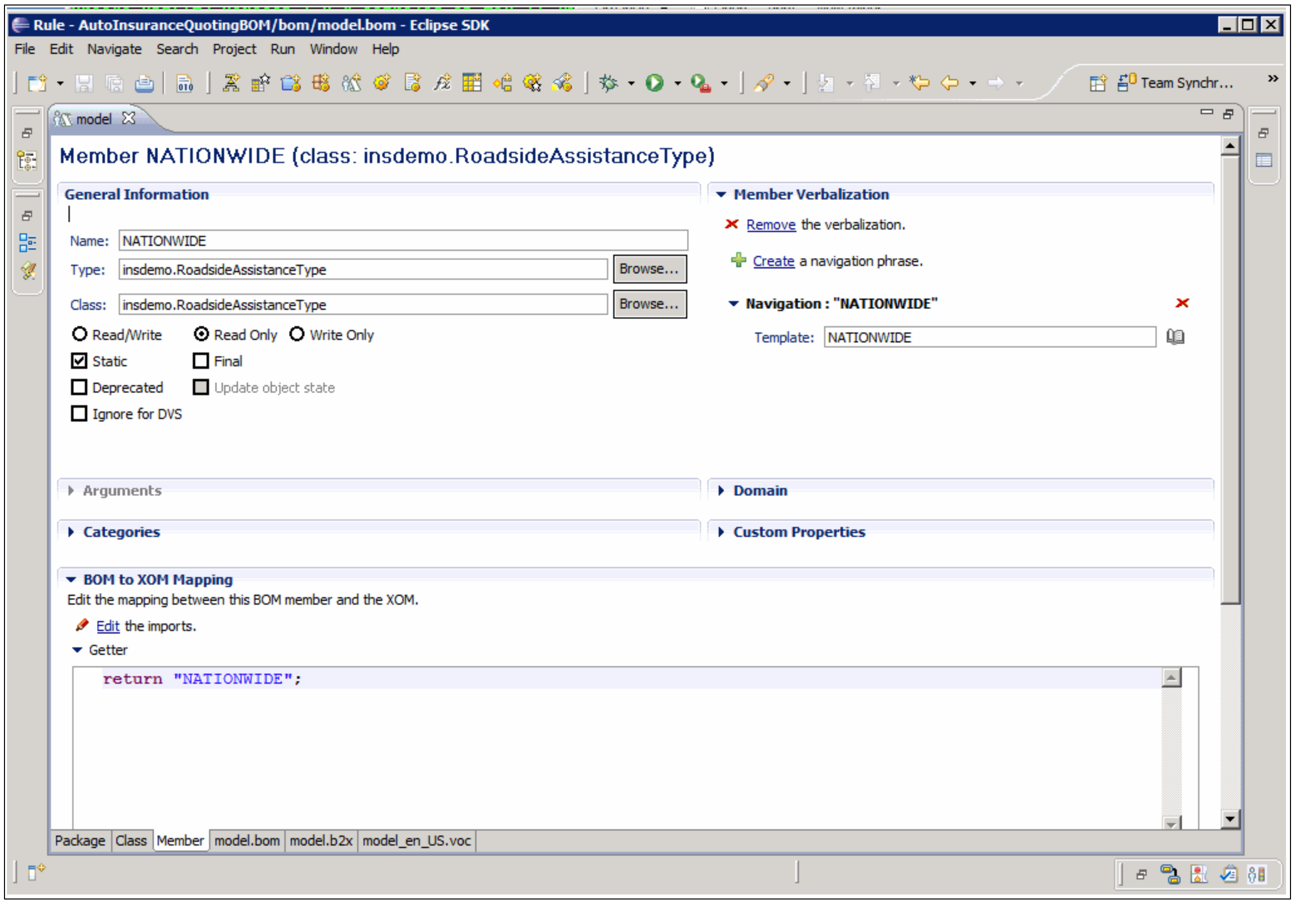
Synchronize.

- ATHOME
- NATIONWIDE
- REGIONAL

File Edit Navigate Search Project Run Window Help
Team Synchr...
model
Change...
New...
Delete
Edit
Package Class Member model.bom model.b2x model_en_US.voc

Figure 9-79 Roadside Assistance domain

Each of these values can then be individually verbalized and mapped to the value that is present in the string when it is passed to the XOM, as shown in Figure 9-80.


Rule - AutoInsuranceQuotingBOM/bom/model.bom - Eclipse SDK
File Edit Navigate Search Project Run Window Help
Team Synchr...
model

Member NATIONWIDE (class: insdemo.RoadsideAssistanceType)

General Information

Name: NATIONWIDE

Type: insdemo.RoadsideAssistanceType Browse...

Class: insdemo.RoadsideAssistanceType Browse...

Read/Write [✓ Read Only] Write Only

[✓ Static] Final

Deprecated Update object state

Ignore for DVS

Member Verbalization

Remove the verbalization.

Create a navigation phrase.

Navigation : "NATIONWIDE"

Template: NATIONWIDE

Arguments

Domain

Categories

Custom Properties

BOM to XOM Mapping

Edit the mapping between this BOM member and the XOM.

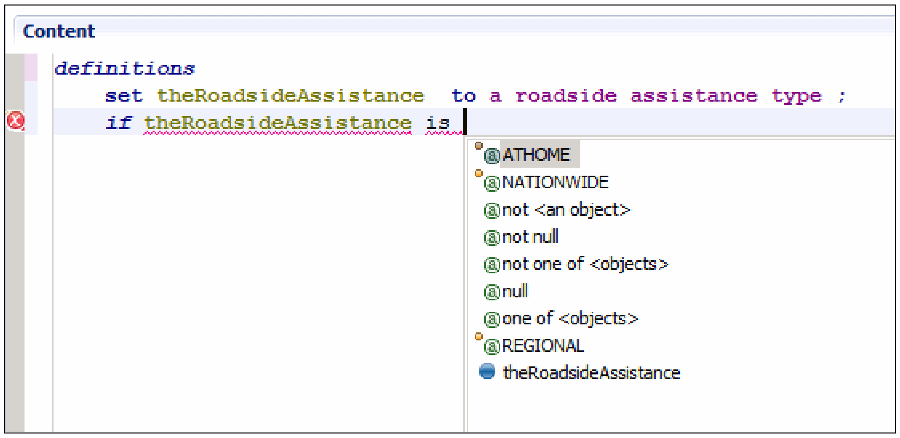
Edit the imports.

Getter

```
return "NATIONWIDE";
```

Figure 9-80 Roadside Assistance member

This action allows rules to be written that can refer to the RoadsideAssistanceType and use these domain values when referenced, as shown in Figure 9-81.



Content

```
definitions
    set theRoadsideAssistance  to a roadside assistance type ;
    if theRoadsideAssistance is
```

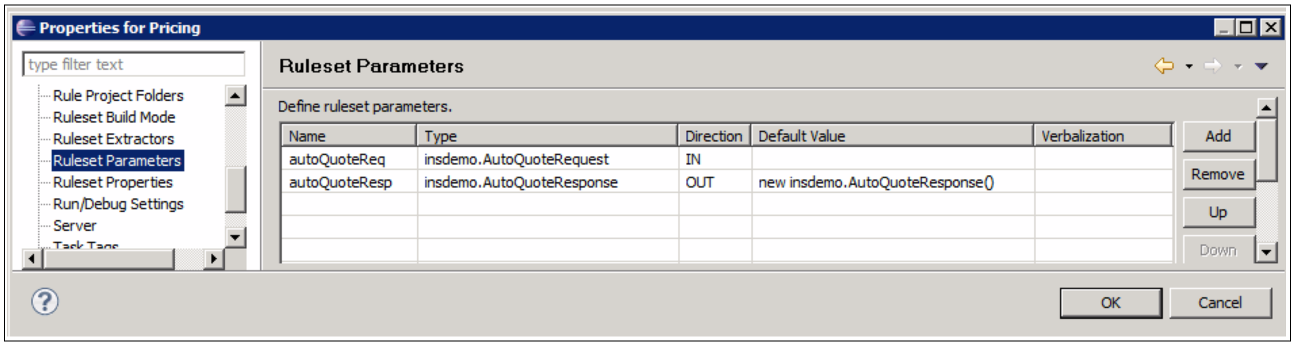
Figure 9-81 Roadside Assistance test rule

9.5.3 Ruleset Signature change activity

The previous section provided new types and verbalization of those types so that the rules in the decision service could be expressed clearly. This section shows how this information is made available to the rules from the applications.

Changing the rule set parameters

When you define or modify a rule set signature, changes must be made in the Eclipse based Rule Developer environment by the rule developers under the guidance of the IT architect. These changes are made in the Pricing project that defines the rule set and consist of the input and output rule set parameters, as shown in Figure 9-82.

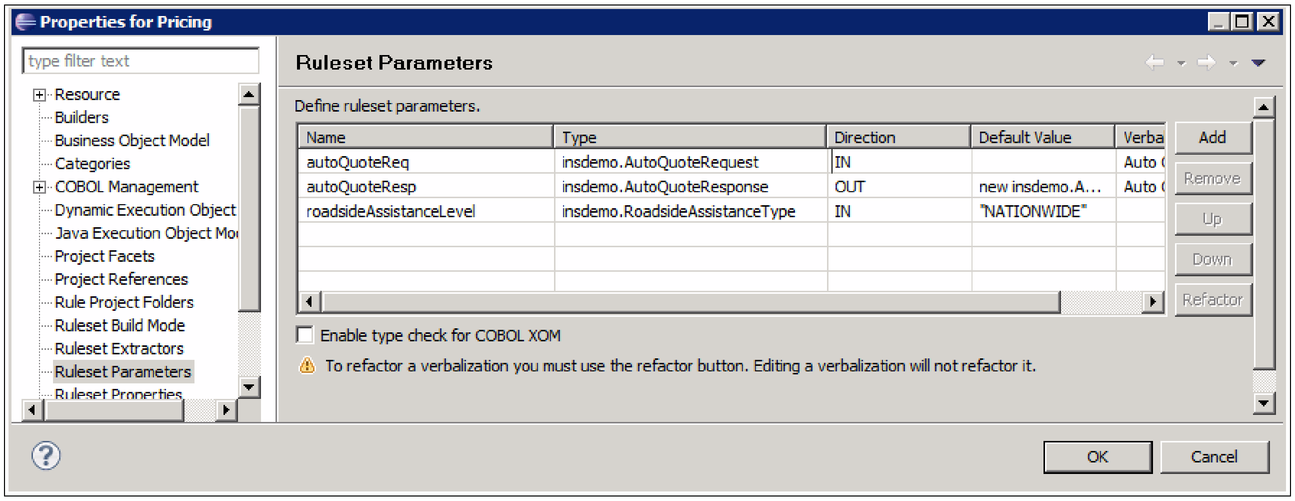


Define ruleset parameters.

Name	Type	Direction	Default Value	Verbalization
autoQuoteReq	insdemo.AutoQuoteRequest	IN		
autoQuoteResp	insdemo.AutoQuoteResponse	OUT	new insdemo.AutoQuoteResponse()	

Figure 9-82   Definition of rule set signature through Pricing project rule set parameters

Typical changes might be to add further parameters or change the types of the parameters. In this example, a single object was used for the input and out parameters. This approach might actually make this interface quite fragile and require a XOM change to accommodate the additional information. Rather than compounding this complexity, add a new parameter to this rule set that will hold the type of RoadsideAssistance that is required. This action minimizes the impact of this change on the existing rules. See Figure 9-83.



Define ruleset parameters.

Name	Type	Direction	Default Value	Verba
autoQuoteReq	insdemo.AutoQuoteRequest	IN		Auto (
autoQuoteResp	insdemo.AutoQuoteResponse	OUT	new insdemo.A...	Auto (
roadsideAssistanceLevel	insdemo.RoadsideAssistanceType	IN	"NATIONWIDE"	

☐ Enable type check for COBOL XOM

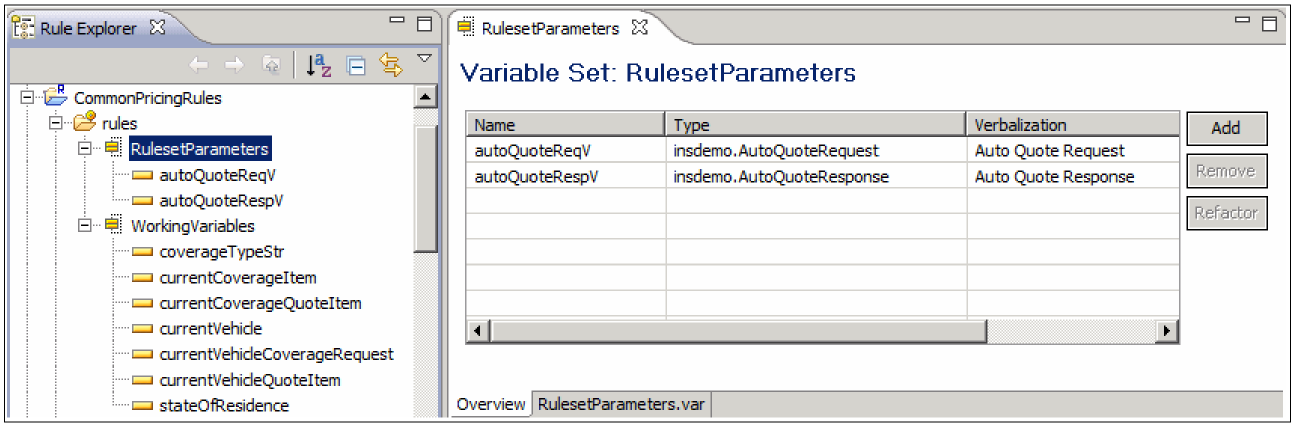
⚠ To refactor a verbalization you must use the refactor button. Editing a verbalization will not refactor it.

Figure 9-83   Roadside Assistance parameter

Defining variables that are visible to the rules

The main rule flow for the Pricing decision service is started every time that a request is made for a pricing decision. This rule flow starts the CoveragePricing sub rule flow, which is defined in the coverage pricing project. The global adjustments rule task then invokes rules from packages in the Global pricing project. This means that both Pricing projects need visibility of these new parameters and new variables.

Having changed the information in the signature, this information must be made available to the rules as variables. These are maintained in the CommonPricingRules project to ensure that they are visible to all projects in the Pricing rule set hierarchy. See Figure 9-84.

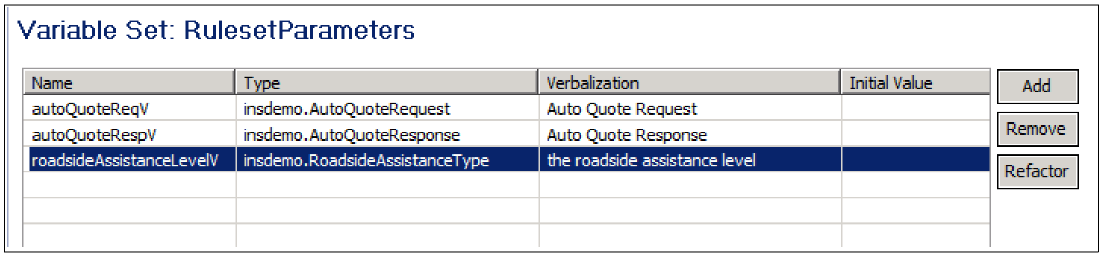


Variable Set: RulesetParameters

Name	Type	Verbalization
autoQuoteReqV	insdemo.AutoQuoteRequest	Auto Quote Request
autoQuoteRespV	insdemo.AutoQuoteResponse	Auto Quote Response

Figure 9-84 Rule set variables that are defined in CommonPricingRules to ensure visibility to all rule projects

As well as defining the variables that are visible to the rules, ensuring that the types match those of the signature, the rule set parameters must be mapped to these variables at a suitable point in the rule flow. These parameters then need a new variable that is visible to the rules, as shown in Figure 9-85.

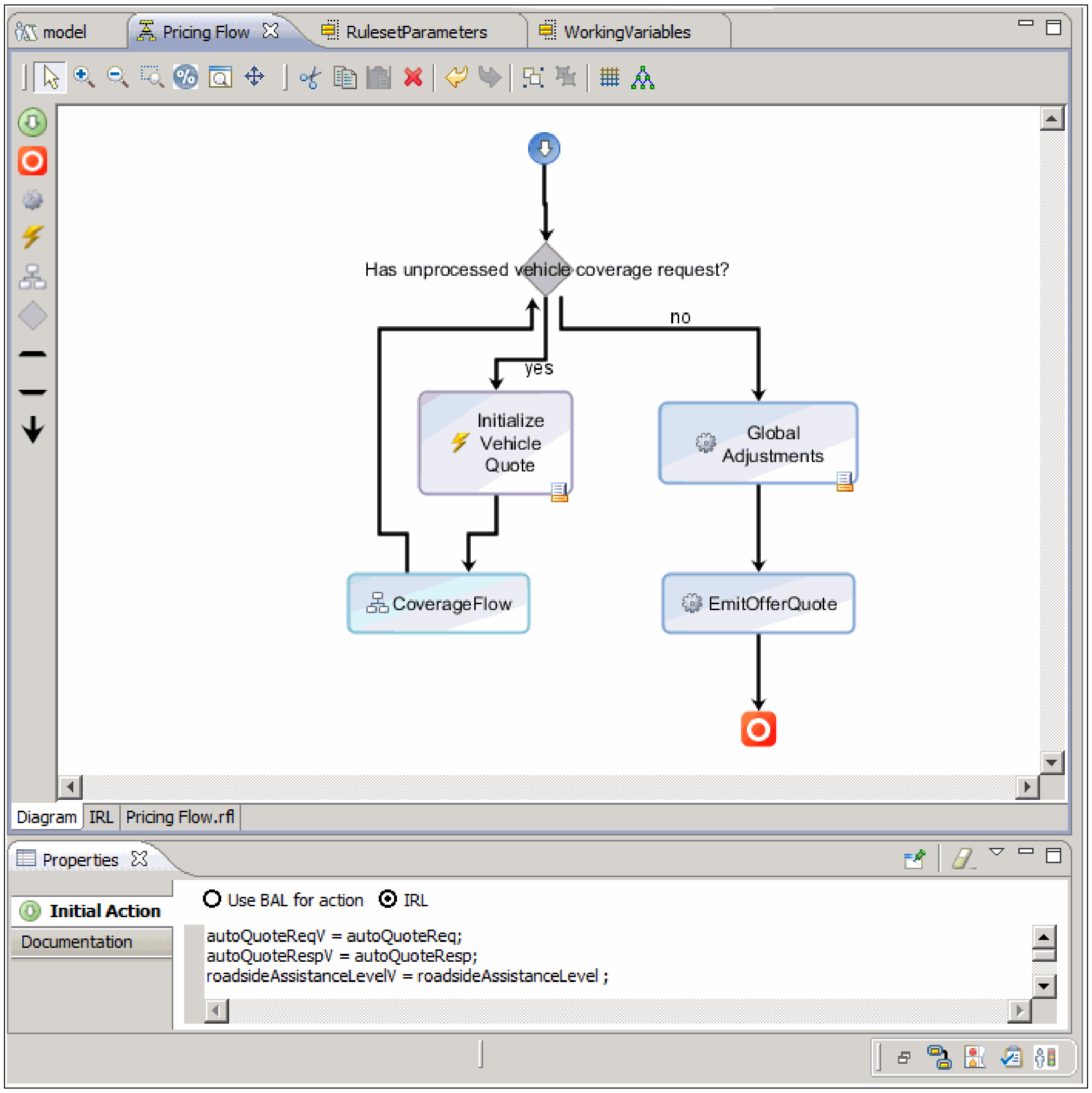


Variable Set: RulesetParameters

Name	Type	Verbalization	Initial Value
autoQuoteReqV	insdemo.AutoQuoteRequest	Auto Quote Request	
autoQuoteRespV	insdemo.AutoQuoteResponse	Auto Quote Response	
roadsideAssistanceLevelV	insdemo.RoadsideAssistanceType	the roadside assistance level	

Figure 9-85 Roadside assistance level variable

These variables then must be initialized from the parameters in the initial task of the pricing rule flow, as shown in Figure 9-86.



( ) Use BAL for action (●) IRL

```
autoQuoteReqV = autoQuoteReq;
autoQuoteRespV = autoQuoteResp;
roadsideAssistanceLevelV = roadsideAssistanceLevel ;
```

Figure 9-86 Pricing rule flow initial action - initialization of variables from rule set parameters

This completes the changes that are required if a rule set signature must change in a release. After these changes are undertaken, the rules can now use the additional information that is available in the rule set parameter variables.

The current client solutions might not be compatible with the decision service signature. If new optional parameters are added, previous solutions still can invoke this rule set. If mandatory parameters are added or parameters are removed, or the types of any parameter are changed, then existing clients cannot invoke the decision service any more.

In this case, consider creating a new version of the decision service signature. This task can be accomplished by creating a new Pricing_V2 project that is a replica of the existing Pricing project, except for the modified signature. This setup then allows existing systems to invoke the earlier signature and new systems to invoke the Pricing_V2 rule set.

After you perform integration testing, the schemas and Java projects that are used in the XOM and signature are frozen in the source code management system, allowing the rule development to continue according to a typical business release lifecycle.

9.5.4 Rule refactoring and synchronization

The final step in this cycle is to synchronize these changes with the new release in Decision Center. In practice, there are many integration testing and other validation activities that should take place in the development environment before synchronization takes place, but these activities are beyond the scope of this book. The configuration management of the rule designer artifacts, such as the XOM and BOM, might be better managed in the development source code management system to ensure that they are maintained in step with the solutions and applications that they must integrate with.

Rule refactoring

In some cases where changes impact the vocabulary or existing variables, rules that were previously correct must be refactored to remove errors. This activity varies with the extent of the changes. Tools are available in Rule Designer that help you with this refactoring as the vocabulary changes are made. When all refactoring is performed and the existing rules have no further errors, the technical changes can be synchronized back in to Decision Center.

Synchronization with Decision Center

With all the technical changes now complete, the rule projects may now be resynchronized with Decision Center, including the following projects:

- AutoInsuranceQuotingBOM: Contains the new XOM and BOM.
- CommonPricingRules: Contains the new variable.
- Pricing: Contains the new rule set parameters and mapping to the variables in the pricing rule flow.

The synchronization differences are shown in the synchronization view, which may then be published to Decision Center, as shown in Figure 9-87.

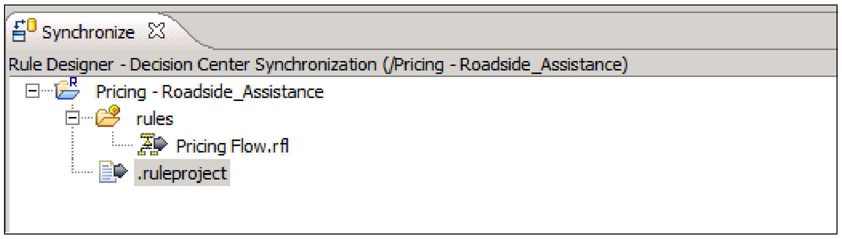
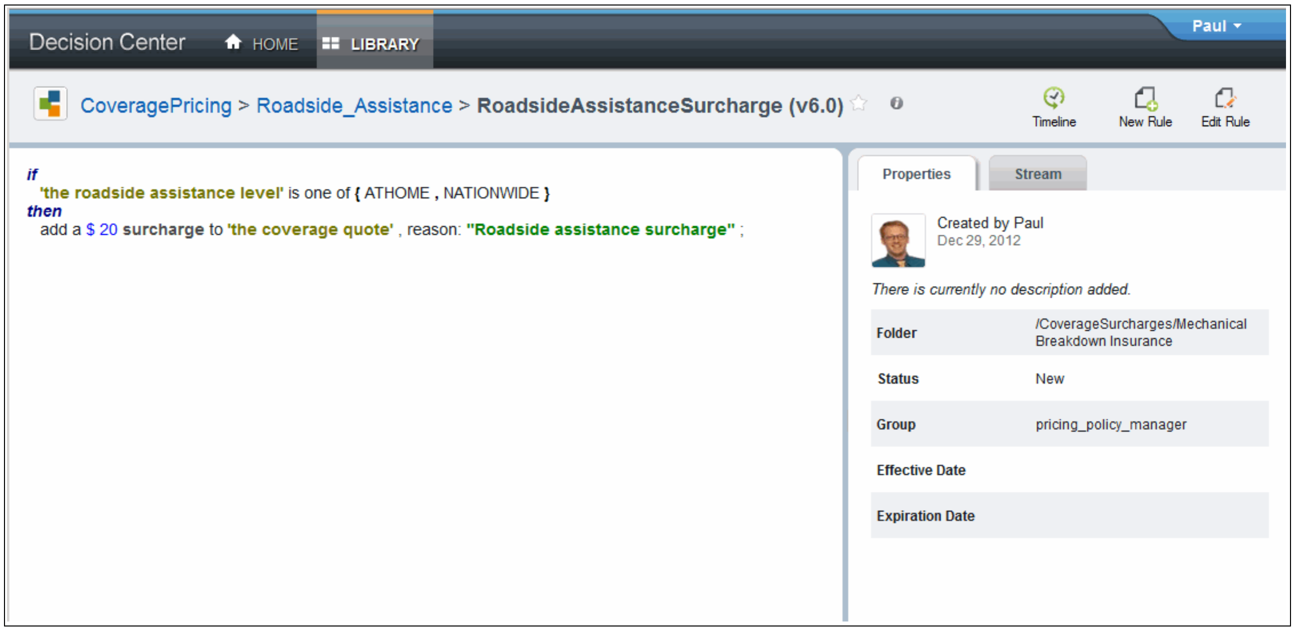

Synchronize
Rule Designer - Decision Center Synchronization (/Pricing - Roadside_Assistance)
Pricing - Roadside_Assistance
rules
Pricing Flow.rfl
.ruleproject

Figure 9-87 Pricing project synchronization

This synchronization then makes the updated vocabularies and signature variables visible to the business rule authors.

Editing rules in Decision Center using the new vocabulary

The rule authoring from this point is identical to that shown in the Business change scenario earlier, except that the new vocabulary can be used in the rules, as shown in Figure 9-88.


Decision Center
HOME
LIBRARY
Paul
CoveragePricing > Roadside_Assistance > RoadsideAssistanceSurcharge (v6.0)
Timeline
New Rule
Edit Rule
Properties
Stream

```
if
  'the roadside assistance level' is one of { ATHOME , NATIONWIDE }
then
  add a $ 20 surcharge to 'the coverage quote' , reason: "Roadside assistance surcharge" ;
```

Created by Paul
Dec 29, 2012

There is currently no description added.

Folder	/CoverageSurcharges/Mechanical Breakdown Insurance
Status	New
Group	pricing_policy_manager
Effective Date	
Expiration Date	

Figure 9-88 Roadside Assistance surcharge

Although the levels of testing and validation that are required for a new major release are more extensive, the essential steps for the rule authoring and deployment of the technical release are similar to the Business release lifecycle.

10

Conclusion

This book focused on the governance of operational decisions and shown how it fits within the enterprise context and its relationships to other industry governance frameworks.

It then explained the differences between traditional organizations and those organizations that use decision management technology to provide more visibility of operational decisions. Because decision changes now require more collaboration between IT and business and the typical IT governance frameworks are not sufficient, we needed to provide some guidance to help organizations with decision governance.

One of the stepping stones to help organizations with this topic is to describe the governance roles and responsibilities. The introduction of decision governance in an organization requires changes, and we want to reiterate that a governance role does not necessarily equate to a unique position in the organization.

Using IBM Operational Decision Manager as the basis for this guidance, this book then described some concepts or approaches:

- Decisions are composed of multiple rules, and it is the decisions that we want to manage, and not the individual rules.
- Decision changes must be managed and governed to deliver the agile decision improvement that is required by the business.
- Decision changes should be managed as releases that are composed of change activities and validation activities to provide a framework for decision governance.

Suggestions for good practices and guidance around the structure of decision services in IBM Operational Decision Manager were also described:

- Rule projects are used to compartmentalize projects to support different roles. An example of how these projects relate to support functional dependencies was shown.
- Rule packages and folders are used to group rules that are logically related so that it is easier for an author to locate the rules they need to work with.

© Copyright IBM Corp. 2013. All rights reserved.

The guidance around project structure and package organization is also the starting point to provide guidance around user groups, project security, and permissions. The main idea is that although it is possible to manage group permissions at the folder level, it is easier and more manageable to deal with permissions at a project level.

Chapter 5, “Release management ” on page 49 provided a description about how decision change management should be managed through the usage of releases. A release is composed of change activities and validation activities. We showed how branching and merging works and how it can be used to support the different types of releases (standard, test fix, and normal). We also showed how the release has a lifecycle and what to expect at each of the steps in this lifecycle.

We then described in more detail what change activities and validation activities are and provide a simple classification scheme for them.

Chapter 7, “Deployment” on page 89, described how decision services can be deployed to production. It also provided some samples of the business processes that must be documented in your organizations so that communication of the responsibilities to each actor is clear for everyone. These processes are samples, but can be used by your organization as a starting point for the development of your own processes.

Chapter 9, “Realization of the insurance scenario” on page 101 goes into a detailed example about how everything that was covered in the book up to this point applies by presenting a real example using IBM Operational Decision Manager. This example included specifics about how the decision service projects are organized, the roles that are defined in this specific example, and specifics about the details of a Pure Business change process example, as well as a technical process change example.

10.1 Governance of decision change management

This book aimed to review the goals and processes for governing operational decisions in an enterprise scalable way. It showed the following items:

- How to define the purpose and objectives of a decision service within the business process or solution that it supports. This forms the basis of the business decision making capability that you want to improve over time with your governance processes.
- How to identify clear goals and KPIs for measuring the performance of this decision and determining how to optimize the rules within the context of the business solution.
- How the different roles within an organization collaborate when you manage decision change and how you structure your decision service to allow their various responsibilities to be supported by appropriate tools.
- How you manage changes to your decision services through release lifecycles that support authorized (or planned) changes to be realized effectively and deployed safely to production systems within the context of your existing enterprise governance solutions.
- How you can use typical release process patterns to plan and manage the important change and validation activities that are needed to govern the release lifecycle.
- How you can use IBM Operational Decision Manager to support these release process patterns and thus deliver enterprise scalable governance of operational decision making.

These governance processes allow an organization to extend its decision improvement capabilities from a single decision management project to enterprise governance or program-based approach by delivering the following items:

- Focused goals for introducing and verifying decision changes
- Improved decision making ability over time that is related to business performance
- Reduced costs by effective use of IT and expert resource
- Fast time to value through clear visibility of goals and responsibilities

10.2 Where do you go from here

This book provides some ideas and recommended practices for extending the governance of operational decisions to your enterprise, but it cannot define the particular solution that is needed for your organization. Maturity and experience build over time, and it is important to capture and build on this experience within your organization.

Starting from your first projects, use these *pioneers* as champions to support the next projects. After a few projects are running, develop a community for sharing experiences and good practices across the organization.

When the community has a critical mass or there is sufficient demand for enterprise decision governance, establish a Center of Excellence (CoE). The funding of the time that is required by participants comes out of the project budgets (for example, some of the person's time is used to participate in the CoE on project time). Eventually, there might be a need to fund a specific role to lead the CoE or to promote champions to the next level as *internal consultants* or to participate in an enterprise decision governance across the various projects.

Whichever approach is adopted, the key goal is to adapt the ideas that are described in this book so that they fit with your processes and evolve over time to deliver the enterprise governance that you require for your organization.

Acronyms

This appendix provides the meanings of and descriptions for the acronyms that are used in this book, as shown in Table A-1.

Table A-1 Acronyms

Acronym	Meaning	Description
ADM	Architecture Development Method	The ADM describes a method for developing an enterprise architecture, and forms the core of TOGAF.
BOM	Business Object Model	The BOM is the basis of the vocabulary that is used in business rules.
CCB	Change Control Board	A committee that makes decisions regarding whether proposed changes to decisions should be implemented.
COBIT	Control Objectives for Information and Related Technology	A framework that was created by ISACA to provide IT governance and management in an organization.
DVS	Decision Validation Service	Part of the IBM Operational Decision Manager offering that is used to test and simulate rule sets in Rule Designer and Decision Center.
ITIL	Information Technology Infrastructure Library	A set of practices for IT service management that focus on aligning IT services with the needs of a business.
RES	Rule Execution Server	Part of the IBM Operational Decision Manager offering that provides a reliable and scalable execution environment for your business rule application.
SCCS	Source Code Control System	A system that allows you to track and control changes to source code documents.

© Copyright IBM Corp. 2013. All rights reserved.

Acronym	Meaning	Description
SIT	System Integration Testing	Testing that focuses on a decision service's coexistence with other systems
SLA	Service Level Agreement	A contractual agreement on the level of service to be provided by a service provider to a customer.
SOA	Service-oriented architecture	An approach and a methodology to design software solutions that are based on interoperable services.
TOGAF	The Open Group Architecture Framework	A framework for enterprise information architecture (designing, planning, implementing, and governing).
UAT	User Acceptance Testing	Testing that is performed by a user to confirm that a decision service meets the requirements.
XOM	eXecutable Object Model	The XOM is the model that is used to run the rules in IBM Operational Decision Manager.

Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this book.

IBM Redbooks

The following IBM Redbooks publications provide additional information about the topic in this document. Note that some publications referenced in this list might be available in softcopy only.

- *Building an Application with Decisions, Processes, and Insight*, TIPS0940
- *Decision Automation on IBM System z with IBM WebSphere Operation Decision Management V7.5*, TIPS0960
- *Flexible Decision Automation for Your zEnterprise with Business Rules and Events*, SG24-8014
- *Implementing an Advanced Application Using Processes, Rules, Events, and Reports*, SG24-8065
- *Making Better Decisions Using IBM WebSphere Operational Decision Management*, REDP-4836

You can search for, view, download or order these documents and other Redbooks, Redpapers, Web Docs, draft and additional materials, at the following website:

ibm.com/redbooks

Other publications

These publications are also relevant as further information sources:

- Boyer, et al, *Agile Business Rules Development*, Springer, 2011, ISBN 3642190405
- *Business Decision Maturity Model (BDMM)*, found at:

  http://www.kpiusa.com/index.php/The-Decision-Model/business-decision-maturity-model-bdmm.html
- *Business Rules Governance and Management*, found at:

  http://www.primatek.ca/blog/?dlname=Primatek-BusinessRulesGovernanceManagement¬WhitePaper.pdf
- *CMMI for Development, Version 1.3 (CMU/SEI-2010-TR-033)*, found at:

  http://www.sei.cmu.edu/library/abstracts/reports/10tr033.cfm
- *COBIT 5: A Business Framework for the Governance and Management of Enterprise IT*, found at:

  http://www.isaca.org/COBIT/Pages/default.aspx
- *COBIT 5: Enabling Processes*, found at:

  http://www.isaca.org/COBIT/Pages/COBIT-5-Enabling-Processes-product-page.aspx

© Copyright IBM Corp. 2013. All rights reserved.

- Office of Government Commerce (OGC), *The Official Introduction to the ITIL Service Lifecycle*, TSO, 2007, ISBN 0113310617
- Office of Government Commerce (OGC), *ITIL Service Transition*, TSO, 2011, ISBN 9780113313068
- *The Open Group Architecture Framework (TOGAF)*, found at:

  http://www.opengroup.org/togaf/
- *SOA Policy*, found at:

  http://www.ibm.com/developerworks/webservices/library/ws-soa-policy/index.html/
- von Halle, *Business Rules Applied: Building Better Systems Using the Business Rules Approach*, John Wiley & Sons, 2002, ISBN 9780471412939
- von Halle, et al, *The Decision Model: A Business Logic Framework Linking Business and Technology*, Taylor & Francis, 2009, ISBN 1420082817

Online resources

These websites are also relevant as further information sources:

- ABRD website guidance on governance:

  http://epf.eclipse.org/bp/abrd_1.5.1_20100928/process.abrd.base/deliveryprocesses/abrd_governance_BF9D60E1.html?nodeId=464d376e
- IBM Operational Decision Manager Version 8.0.1 Information Center

  http://pic.dhe.ibm.com/infocenter/dmanager/v8r0m1/index.jsp?topic=%2Fcom.ibm.help.doc%2Finfocenter_homepage.html
- IBM WebSphere Operational Decision Management Version 8.0 Information Center, found at:

  http://pic.dhe.ibm.com/infocenter/dmanager/v8r0m1/index.jsp
- Operational Decision Management

  http://www-01.ibm.com/software/decision-management/operational-decision-management/

Help from IBM

IBM Support and downloads

ibm.com/support

IBM Global Services

ibm.com/services

IBM Redbooks

Governing Operational Decisions in an Enterprise Scalable Way

(0.2"spine)
0.17"<->0.473"
90<->249 pages

Governing Operational Decisions in an Enterprise Scalable Way

IBM Operational Decision Manager support for release management

Decision governance overview and organization

Decision release management lifecycle

This IBM Redbooks publication presents decision governance topics from a theoretical discussion perspective and then goes on to make links to the practical aspects of applying these concepts by using the IBM Operational Decision Manager platform.

This book explores enterprise governance context to clarify the bigger picture for how governance is carried out across the enterprise.

You will also find this book valuable if you are using or considering the usage of an operational decision management system (or business rules management system (BRMS)) in your organization. You might be following a standard such as the The Open Group Architecture Framework (TOGAF) Architecture Development Method (ADM) and decided to use a decision management system that lets the business people take control of the business decisions that are made by the technology systems in their organization.

This book also describes Control Objectives for Information and Related Technology (COBIT), which provides a comprehensive framework that assists enterprises in achieving their objectives for the governance and management of enterprise IT.

Another topic of great importance that this book covers is the relationship to ITIL, a public framework that describes best practices in IT Service Management. Of the five stages of the ITIL lifecycle, this book focuses on the objectives and processes of the Service Transition stage.

INTERNATIONAL TECHNICAL SUPPORT ORGANIZATION

BUILDING TECHNICAL INFORMATION BASED ON PRACTICAL EXPERIENCE

IBM Redbooks are developed by the IBM International Technical Support Organization. Experts from IBM, Customers and Partners from around the world create timely technical information based on realistic scenarios. Specific recommendations are provided to help you implement IT solutions more effectively in your environment.

For more information:
ibm.com/redbooks

SG24-8127-00 ISBN 0738438030