Changing the rules of business™

ILOG JRules™ 6.7

Getting Started

Contents

Getting Started

This section introduces the main steps needed to implement a Business Rule Management System (BRMS) with ILOG JRules.

- *Introducing ILOG JRules*

  Describes the main features and purpose of the JRules Business Rule Management System (BRMS).

- *ILOG JRules Architecture Guidelines*

  Provides guidelines for architects who implement the BRMS.

- *Tutorials and Demos*

  Provides hands-on material to help you get started with JRules.

- *Using this Online Documentation*

  Describes the interactive help mechanisms available to you, and how to use this online help.

- *Technical Support and Resources*

  Provides information on how to obtain help and technical support from ILOG worldwide.

Introducing ILOG JRules

Describes the main features and purpose of the JRules Business Rule Management System (BRMS).

In this Section

Related Sections

Role of a Business Rule Management System (BRMS)

ILOG JRules is a Business Rule Management System (BRMS) that allows both business and IT users to manage the rules that drive business.

As companies rely more and more on information technology (IT) to manage their business, IT departments need to develop more complex applications and simultaneously accommodate an increasing rate of change in the applications they support.

Often, the implementation of the company's business policy within these applications becomes too complex, voluminous, and fast changing for a traditional software architecture. When this happens, an enterprise Business Rule Management System (BRMS) provides solutions to make this management more efficient, both for developers and for the business users of the applications.

With a BRMS, developers and architects can extract the business logic from the traditional code of an application. When business policies are hard-coded into an enterprise application, the process of updating the system requires specialist programming staff, puts the stability of the system at risk, and can take a long time. By externalizing the business logic from a business application with business rules, IT users can develop and run the business logic independently of the application.

A complete implementation of a BRMS can go even further and enable business users to manage business policies directly, with limited dependence on the IT department. The degree of dependence can range from limited review by business users of policies implemented by IT, to complete control over the specification, creation, testing, and deployment of the policy by business users.

From Business Policy to Business Rules

Business rules are an expression of business policy in a form that is comprehensible to business users and executable by a rule engine. From a business perspective, a business rule is a precise statement that describes, constrains, or controls some aspect of your business. From the IT perspective, business rules are a package of executable business policy statements that can be invoked from an application.

A business policy may be expressed as several business rules. Business rules formalize a business policy into a series of "if-then" statements. Here is an example of the kind of business policy you may be familiar with, and that could be expressed as business rules:

Customers who spend a lot of money in a single transaction should be upgraded.

The process of capturing rules consists of formalizing the vocabulary required to express the policy as a conceptual object model, and representing the logic of the business policy as if-then statements.

Once this has been done, the above business policy may be implemented with the following business rule in JRules:

If
  the customer's category is Gold
  and the value of the customer's shopping cart is more than $1500
Then
  change the customer's category to Platinum

In this form, the business logic can be packaged as an executable ruleset and called from the application code as a single entity. Therefore changes to the business policy do not require changes to the application code.

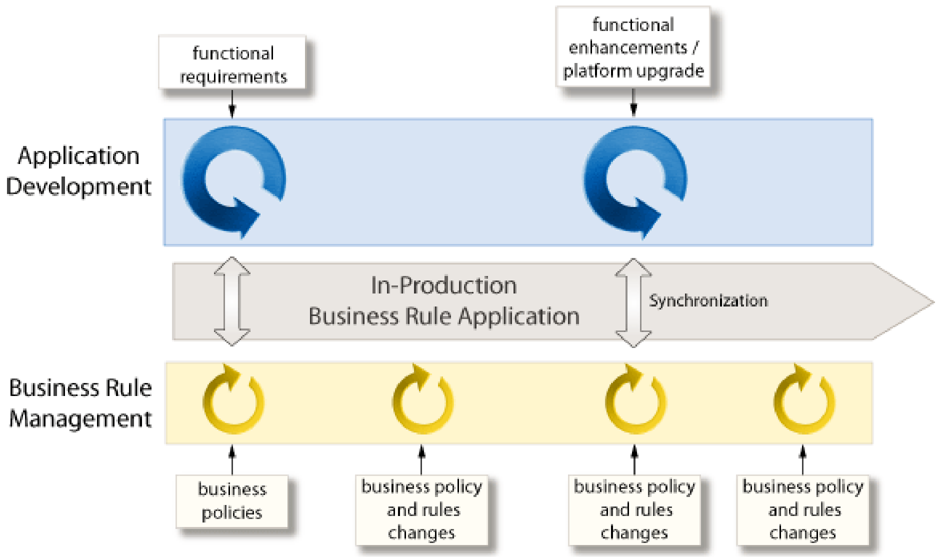
Once an application is adapted to use business rules to implement business policy, the rules representing the policy can be written by developers using developer tools, or by more business-oriented users such as business analysts or the business users themselves (referred to as policy managers in this documentation).

Synchronized Business and IT Cycles

JRules provides an environment for designing, developing, and deploying business rule applications. A business rule application is any application that uses the facilities of a BRMS to externalize business rules out of application code. The IT cycle consists in developing and maintaining this infrastructure. Once the infrastructure is set up, distributed business teams can start collaborating through a Web-based environment to create and maintain business rules.

In a BRMS, the business rule management and application development life cycles can evolve in parallel. Business policies can evolve as required, without putting an extra load on the development of the application. Each time the application evolves, the business policy implementation synchronizes with the application.

With this separation, business policy and application architecture can be managed asynchronously. For example, application developers may work in a semi-annual cycle in which a new application version is developed every six months in response to changing application infrastructure (for example, JDK and RDBMS versions) and additional core business requirements. At the same time, policy managers can work on a weekly cycle in which a new version of the business policy is delivered in response to an evolving market, new customers, or changing regulatory environment.


functional requirements
functional enhancements / platform upgrade
Application Development
In-Production Business Rule Application
Synchronization
Business Rule Management
business policies
business policy and rules changes
business policy and rules changes
business policy and rules changes

In addition to working on different timelines, IT and business users expect to work with different tools, reflecting their different skill sets and views of the application.

For example, developers are accustomed to the world of Java code. They use source code management systems to work simultaneously on separate copies of a project without interfering with each other.

Business users, on the other hand, do not concern themselves with the details of application development, but are interested in defining, testing, and managing well-defined decisions that implement business policy represented as business rules within the application. They therefore require tools that simplify organizing, searching, and authoring rules within the context of the overall policy.

With developers working in their environment at their own pace and business users doing the same, the work of both can be synchronized and merged easily.

Finally, both IT and business users require access to a rule execution environment in order to deploy rules. For developers, rule deployment is part of application testing. For business users, rule deployment is part of the specification, validation, and rollout to production of new and changed policies.

Related Sections

JRules BRMS Concept of Operation

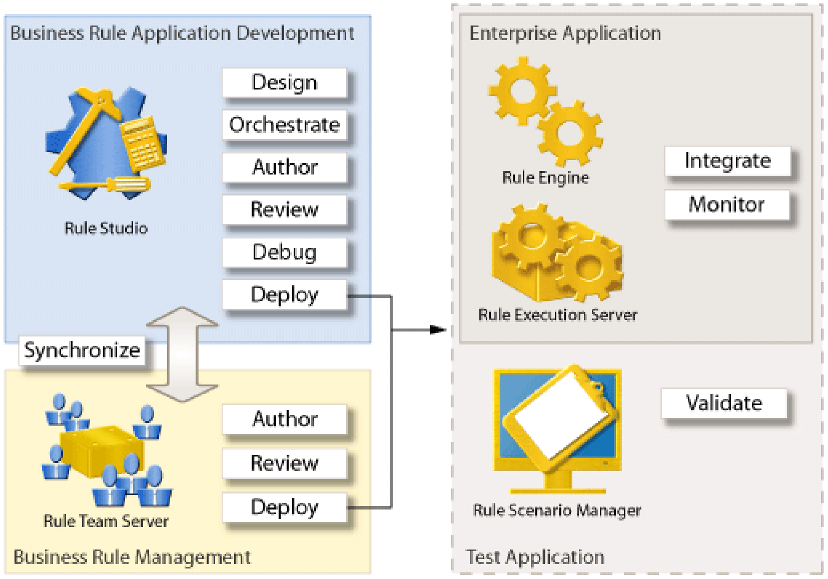
Business Rule Application Development

Integration into an Enterprise Application

Business Rule Management for Business Users

JRules BRMS Concept of Operation

ILOG JRules is a collection of modules that, while operating in different environments, work together to provide a comprehensive Business Rule Management System.



There are three broad areas in the implementation of JRules as your enterprise BRMS. For each of these areas, JRules provides dedicated modules aimed at specific user roles to perform a number of activities:

◆ **Business rule application development**

IT users work with the Rule Studio module in Eclipse for design, Java development, and rule project development. See *Business Rule Application Development*.

◆ **Integration, validation and deployment into the enterprise application**

IT users have access to the Rule Execution Server module to monitor deployed rulesets. In addition, they can use the Rule Scenario Manager framework to create testing and simulation solutions for IT and/or business users to validate the correctness and effectiveness of rulesets. See *Integration into an Enterprise Application*.

◆ **Business rule management enabled for business users**

Business users work with a Web-based module, Rule Team Server, to maintain business rules, both during application development and after the application is deployed to production. See *Business Rule Management for Business Users*.

Related Sections

Architecture of a Business Rule Application

Business Rule Application Development

Integration into an Enterprise Application

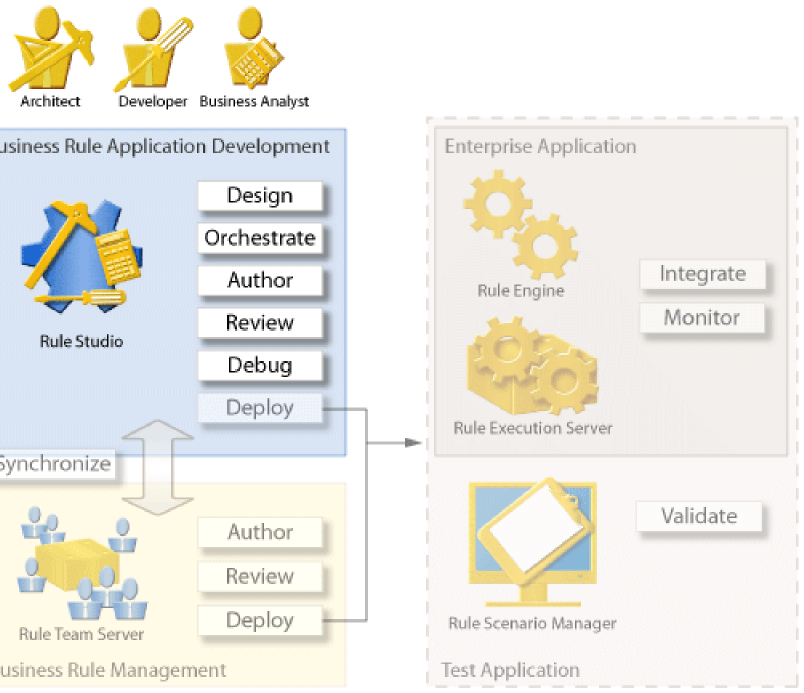
Business Rule Management for Business Users

Business Rule Application Development

Business rule application development comprises:

- designing the business rule application
- orchestrating its execution
- authoring business rules
- reviewing the rule artifacts
- debugging the business rule application

Developers and architects first design a data model, often working with business analysts as a source for models and requirements. They develop a rule project, write business rules, and integrate rule execution into a production application. The business logic implemented as a rule project can now be maintained independently of the application.



The following table describes the module you need to develop a business rule application.

Table 1 *Business Rule Application Development Module*

Module	**Description**
Rule Studio	Rule Studio is the development environment for business rule applications. It is integrated into Eclipse. Developers can take advantage of this integration to develop their Java projects along with rule projects. Developers and architects can therefore use Rule Studio to integrate and deploy the business rule application into their enterprise client application. Architects, developers and business analysts use Rule Studio to design the business rule application, author, review, and debug business rules. Rule Studio also has tools for keeping the rules synchronized with Rule Team Server.

The following table describes the user roles involved in developing a business rule application.

Table 2 *Business Rule Application Development Roles*

Role	Description
Architect	Architects are responsible for managing the overall deployment organization of the rules, and making sure that the execution of rulesets is optimized. In the context of business rule application development, architects are responsible for: ◆ mapping logical packages in Rule Team Server to physical packages in Rule Studio ◆ defining the rulesets in a business rule application: the elements they contain, the static or dynamic rule selection mechanisms required, the orchestration of rule execution with ruleflow
Developer	Developers are responsible for the development, debugging, and deployment of business rule applications. They are familiar with object models, APIs, and the development environment (J2EE application servers or J2SE). In the context of business rule application development, developers are responsible for: ◆ creating an implementation for the business rule vocabulary ◆ testing and debugging rule execution in Rule Studio ◆ tuning rules ◆ writing complex rules that business users cannot write Although the developer role can overlap with the business analyst role, the business analyst typically performs some limited tasks in both Rule Studio and Rule Team Server, whereas the developer mainly works in Rule Studio and uses the integration and extension APIs provided in JRules.
Business Analyst	Business analysts act as the bridge between business and IT departments, by smoothing the way as corporate policy moves from design to integration inside a software application. They translate the policy into a formal specification acceptable to developers, and validate the formal specification with policy managers. In the context of business rule application development, business analysts are responsible for: ◆ defining the vocabulary used in rules ◆ writing and organizing business logic into business rules that can be maintained by the policy managers Depending on their level of technical knowledge, business analysts may perform tasks that are currently described as developer tasks. However, generally business analysts do not write code.

The following table describes the activities involved in developing a business rule application.

Table 3 *Business Rule Application Development Activities*

Activity	Description
Design	Developers and architects create a rule project in Rule Studio. A rule project is a type of Eclipse project dedicated to the development of business rule applications. Business analysts work with business users and developers to define the vocabulary used in the business policy. This vocabulary is based on a business object model. Developers implement the business object model as a Java or XML implementation model. In some cases, the Java or XML object model exists already, and can be imported into JRules to generate the business object model and the vocabulary. Business analysts and developers also define and implement the additional rule properties required for managing and tracking business rules, using rule model extensions. Rule Studio provides editors for the business object model and the rule model extensions.
Orchestrate	Developers and architects define how they want the rules to be executed. In this rule project, they define the organization of the business rules into packages and their persistence in a Source Code Control (SCC) system. They also define a high-level flow of execution for the business rules.
Author	Developers and architects set up the tools to help rule authors write rules. To make business rule editing easier for business users, they may, for example, write some complex business rules, create business rule templates, and define business rule vocabulary categories.
Review	Developers use Rule Studio to perform static analysis on rules and make sure the initial set of rules they developed is consistent.
Debug	Developers use Rule Studio to test and debug rule execution in a sandbox.

Related Sections

Designing the Rule Project

Orchestrating Ruleset Execution

Authoring and Reviewing Rules

Creating Rule Projects

Writing Rules

Orchestrating Ruleset Execution

Querying and Reporting

Running and Debugging

Integrating Application Data

Customizing JRules

Rule Studio User Interface

Rule Languages

Related Samples and Tutorials

Tutorial: Defining a Vocabulary

Tutorial: Creating Business Rules

Tutorial: Editing Decision Tables

Tutorial: Creating Your First Ruleflow

Tutorial: Debugging a Ruleset

Rule-Based Programming

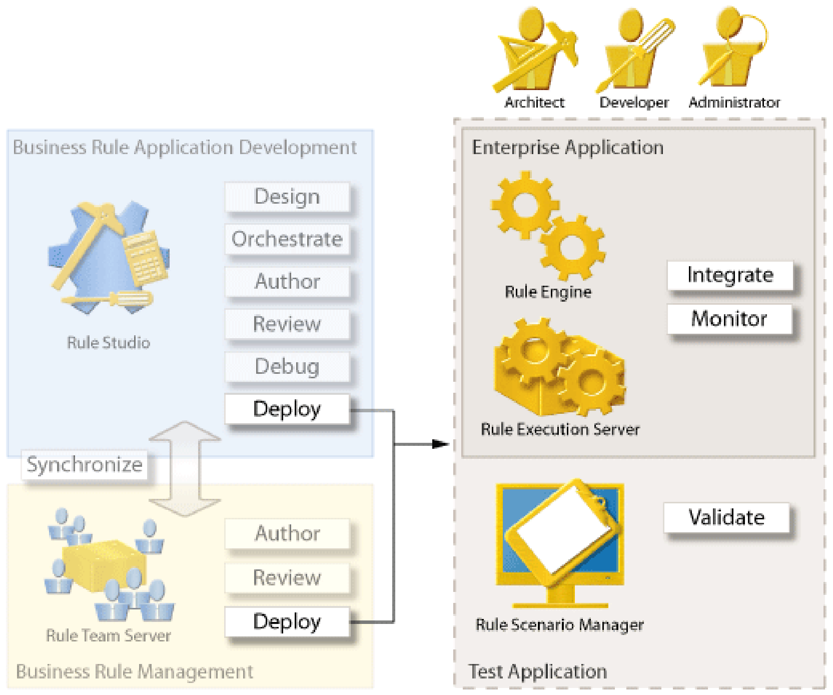
Rule Studio Business Rule Management Extensions

Rule Studio Authoring Extensions

Integration into an Enterprise Application

Integrating a business rule application into your enterprise environment consists in writing the integration code that calls the execution of business rules from your application, and deploying the rules to the execution environment.

Developers in charge of quality can also deploy the business rules onto a test server and use a solution created with Rule Scenario Manager to validate that the business rules comply with the required implementation logic. Once the business rule application is deployed, an administrator can monitor execution using the Rule Execution Server console.



The following table describes the modules you need to integrate business rule execution into an enterprise application, and validate the rules on a test application.

Table 4 *Validation and Integration Modules*

Module	Description
Rule Engine	The rule engine is a component in which you isolate and apply the business logic (the rules) of an application. A rule engine is used to answer questions such as "can we offer our client a mortgage?" and execute actions based on the answer. You provide the data (here, the client and the mortgage) and rules (a description of cases when the client can be offered a mortgage, and what to do in each case), and the rule engine gives an answer to the question and executes corresponding actions. Architects and developers may also use the low-level engine API in advanced integration scenarios. They can use many of the features of the rule engine (selection of execution algorithms, runtime rule selection mechanisms, performance tuning) from Rule Studio.
Rule Execution Server	Rule Execution Server is a managed, monitored execution environment for deployed business rules. Rule Execution Server handles the creation, pooling and management of ruleset instances in order to make invocation of a decision from application code as simple as possible. For deployment purposes, individual rulesets are packaged into a RuleApp. Rule Execution Server provides a management console, from which you can deploy, manage, and monitor RuleApps.
Rule Scenario Manager	Rule Scenario Manager is a framework that enables developers to provide business rule testing and simulation solutions to other developers or policy managers.

The following table describes the user roles involved in integrating business rule execution into an enterprise application.

Table 5 *Validation and Integration Roles*

Role	Description
Administrator	System administrators are responsible for ensuring a smoothly running environment, and have the ability to restore a particular application state. In the context of integration into an enterprise application, system administrators are responsible for: ◆ Installing and configuring Rule Execution Server ◆ Deploying the applications ◆ Redeploying rulesets as changes are made ◆ Managing user access to the execution servers
Developer	Developers are responsible for the development, debugging, and deployment of business rule applications. They are familiar with object models, APIs, and the development environment (J2EE application servers or J2SE). In the context of integration into an enterprise application, developers are responsible for writing the invocation code for rule execution, using either Rule Execution Server or the rule engine.
Architect	Architects are responsible for ensuring the overall deployment organization of the rules, and that the execution of rulesets is optimized. In the context of integration into an enterprise application, architects are responsible for ensuring coherent ruleset deployment across a number of business rule applications, and defining reuse of rules.

The following table describes the activities for integration into an enterprise application.

Table 6 *Validation and Integration Activities*

Activity	Description
Deploy	Administrators and developers make a ruleset available to a rule engine from Rule Studio or Rule Team Server. Developers deploy rules from Rule Studio following two main scenarios: hot deployment when they want an immediate availability of rules for testing purposes, or staged deployment when they want to deploy to a controlled production environment Administrators may also deploy rules from Rule Team Server or the Rule Execution Server console.
Validate	Developers use Rule Scenario Manager to create a business rule testing and simulation environment for use by rule developers, business analysts and policy managers. Depending on the individual user's needs, developers can use different Rule Scenario Manager services. For example, rule developers may create Excel spreadsheets that can be filled in by business analysts or policy managers with test data, which the rule developers then use to execute test suites. An example of a testing and simulation solution that can be built with Rule Scenario Manager is the Rule Scenario Manager Console.
Integrate	Developers and architects package the contents of a rule project into an executable archive, called a ruleset, and write the code to run the rule engine on this ruleset. They may integrate rule execution into an embedded rule engine, or implement rule execution as a decision service within Rule Execution Server.
Monitor	An administrator uses the Rule Execution Server Console, Ant scripts, or enterprise management tools (such as HP Tivoli or IBM OpenView) and JMX MBeans to monitor the execution of rulesets within Rule Execution Server. They can create backup versions of rulesets and revert to a previous version if necessary, and gather statistics on performance.

Related Sections

Integrating Rule Execution into an Embedded Rule Engine

Deploying Rulesets to a Managed Rule Execution Environment

Installing JRules Modules

Executing Rules

Deploying Rules

Integrating Application Data

Managing Rule Execution Server

Rule Execution Server Console Online Help

Testing Rules with Rule Scenario Manager

Rule Scenario Manager Console Online Help

Related Samples and Tutorials

Tutorial: Executing a Hosted Transparent Decision Service

Tutorial: Debugging a Remote Rule Execution Server Application

Tutorial: Creating a Web Application to Invoke JRules in BEA WebLogic Workshop

Tutorial: Creating a Web Application to Invoke JRules on IBM Rational Application Developer

Tutorial: RuleApp Management

Tutorial: Debugging a Remote Rule Execution Server Application

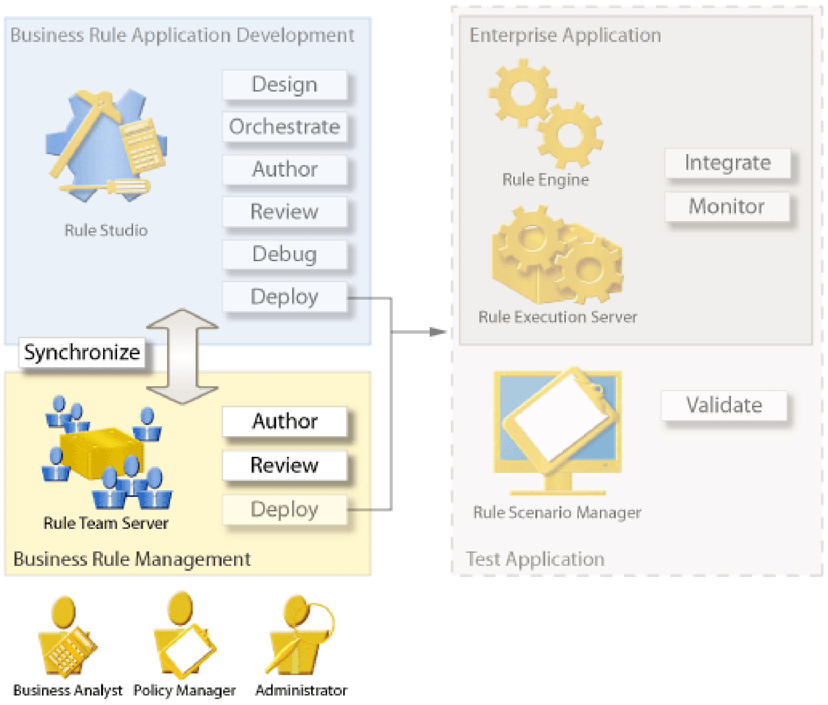
Rule Engine Integration

Rule Execution Server Integration

Rule Scenario Manager Demonstration

Business Rule Management for Business Users

Developers publish rule projects to Rule Team Server. Rule Team Server enables business analysts and policy managers to maintain the business rules, and report on these rules. A business analyst or policy manager with administrator privileges can then deploy the latest version of the business rules to the application from Rule Team Server to the Rule Execution Server, either directly, or through a staged process in cooperation with a Quality Assurance (QA) department and system administrator.



The following table describes the module you need to enable business rule management for business users.

Table 7 *Business Rule Management Modules*

Module	**Description**
	Rule Team Server is a scalable rule management server and repository with a collaborative Web environment for authoring, managing, validating, and deploying business rules. Rule Team Server is integrated into a J2EE application server to provide a central storage system for business rules and their metadata. Policy managers use Rule Team Server to work collaboratively on rule artifacts. Access to the rule projects and artifacts is controlled by the administrator using permission management. Rule Team Server provides history and versioning services that support auditing and rollback of rule artifacts.

The following table describes the user roles involved in enabling business rule management for business users.

Table 8 *Business Rule Management Roles*

Role	**Description**
	Policy managers are experts in business logic and the owners of business policy within an organization, they manage these corporate policies. Policy managers are responsible for writing or enforcing the business policies of an enterprise. Policy managers participate in the initial process of extracting business rules from existing policy (documents, databases, and so on). This process is carried out with the business analyst whose aim is to determine the vocabulary elements that are necessary to be able to write the rules that implement the policy. Policy managers work in Rule Team Server and are responsible for: ◆ creating and updating rules ◆ reviewing how the execution of rules is orchestrated in a ruleflow ◆ reporting on the status of the business policy

Table 8 *Business Rule Management Roles*

Role	Description
Administrator	System administrators are responsible for ensuring a smoothly running environment, and have the ability to restore a particular application state. In the context of business rule management, system administrators are responsible for: ◆ installing and configuring Rule Team Server ◆ managing user access to the rule management servers
Business Analyst	Business analysts translate the policy into a formal specification acceptable to developers, and validate the formal specification with policy managers. In the context of business rule management, business analysts are responsible for synchronizing rule project content between Rule Studio and Rule Team Server.

The following table describes the activities involved in enabling business rule management for business users.

Table 9 *Business Rule Management Activities*

Activity	Description
Synchronize	When the business rule application is ready to be made available to business analysts and policy managers, developers publish rule projects to Rule Team Server, a Web-based business rule management environment for business users. Rule projects are then accessible to business users in a multi-user, access-controlled Web application. Synchronization is two-way, so developers can update their working copy of the rule project with the contents of the Rule Team Server repository as changed by policy managers and business analysts, at any time.
Author	Policy managers and business analysts create and edit business rules in Rule Team Server. Their work is saved in a repository database that handles versioning, history, and multi-user access.
Review	Policy managers and business analysts use baselines to create a snapshot of the rules they worked on before deploying them, write queries on the business rules in their rule project, and create reports on the results of the queries. Policy managers can see the result of their work, or review the changes made by others, by searching the business rules in their rule project.

Related Sections

Enabling Business Rule Management for Business Users

Storing and Sharing Rules

Rule Team Server Online Help

Flash Demo: Rule Team Server Tour

Flash Demo: Synchronizing Between Rule Studio and Rule Team Server

Related Tutorials and Samples

Rule and Decision Table Editing Tutorial

Views and Queries Tutorial

Baseline Management Tutorial

Rule Analysis Tutorial

Rule Team Server Business Rule Management Extensions

Rule Team Server Authoring Extensions

ILOG JRules Architecture Guidelines

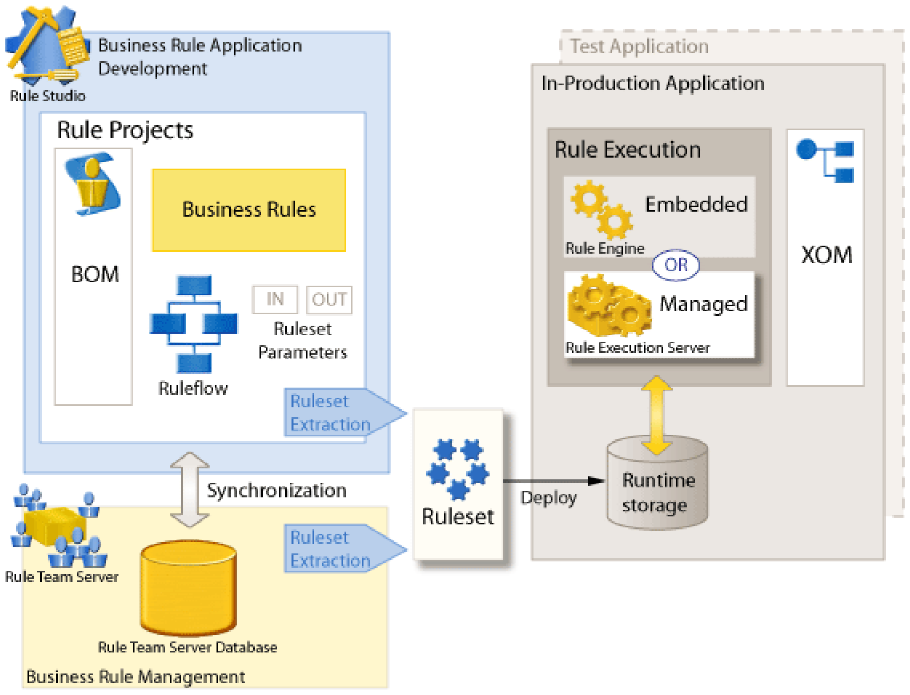
Provides guidelines for architects who implement the BRMS. Before you read this section, make sure you are familiar with the main features and purpose of the JRules BRMS.

In this Section

Related Sections

Architecture of a Business Rule Application

JRules BRMS allows you to manage business rules independently from the application logic, and to enable business users to manage these business rules.



Business rule application development consists in putting in place the infrastructure necessary for editing rules, and producing one or several rulesets.

The ruleset is a standalone and executable container that corresponds to a decision. Rules are atomic expressions of policies, and the automation of these policies and their application to events can generate decisions. Rules pertaining to a given decision are organized for execution and stored in the ruleset.

Using Rule Studio, you develop rule projects from which you extract rulesets.

A BRMS enables the business logic to be written in business terms, rather than in programming language terms. To this end, you create a Business Object Model (BOM) that describes the data on which your decision is based, and assign it a natural language vocabulary. All elements of the decision can then be written in terms of this vocabulary.

In order for your application to take advantage of this decision written in terms of the vocabulary, there must be a contract between the application and ruleset. In the simplest case this contract consists of two primary elements:

- A mapping from the BOM to your application environment's Java objects or XML schemas. These objects or schemas are called the Execution Object Model (XOM). The mapping allows data that your application manipulates in the form of native Java objects or XML to be reasoned upon by the ruleset written in terms of the BOM. In the simplest case, the BOM can be the same as the XOM. In more complex cases, however, you need to provide an explicit BOM to XOM mapping.
- A set of input and output parameters describing the data you pass to the ruleset and the data you expect to receive back in terms of the BOM. Through the BOM to XOM mapping, these data elements are provided from and to your application in the form of the XOM, but reasoned on by the rules in terms of the vocabulary and BOM.

Once the ruleset contract has been established, you have several options for invoking the ruleset from your application:

- You can host the ruleset on Rule Execution Server. In this case you can invoke the ruleset from your application code using either a plain Java interface, or local EJBs. Or, you may invoke the ruleset from a remote Java Virtual Machine (JVM) using remote EJBs or asynchronously using message driven beans. JRules provides prewritten components for all four of these options. In addition, JRules includes tools that make it easy to expose a ruleset as a Web Service. Hosting the decision in Rule Execution Server is the preferred option as it gives you maximum flexibility in managing the rulesets.
- You can host the ruleset as a transparent decision service, for invocation through Web service protocols.
- You can embed the invocation into your application and interact directly with the rule engine. In this case, the ruleset must be packaged as a standalone ruleset archive.
- You can deploy the ruleset to a test environment to perform test and simulation using Rule Scenario Manager.

To put business users in control of the business logic, you publish rule projects to Rule Team Server. Rule Team Server is a JRules module that provides a database for storing rule projects, and a Web-based interface to enable business users to edit and manage rule project contents. A user with the appropriate rights may also extract rulesets from Rule Team Server and deploy them to your application.

Related Concepts

Rule Project

Business Object Model (BOM)

Overview: Ways of Expressing a Business Policy

Overview: Designing Rulesets

Rulesets and Ruleset Archives

Ruleset Parameters

Ruleflows

Execution Object Model (XOM)

Related Tasks

Setting Up a Rule Project

Working With BOM Entries

Defining Categories

Creating Ruleset Parameters

Working with Ruleflows

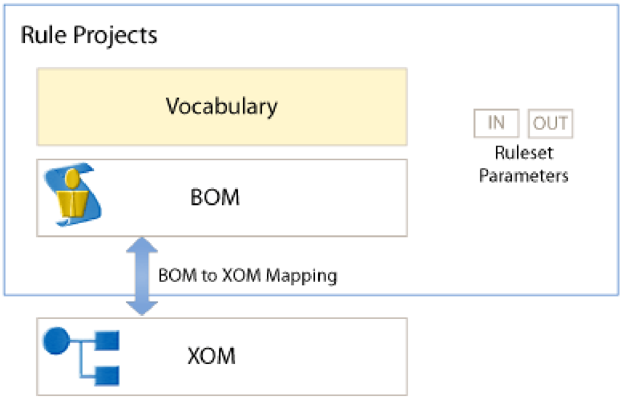
Extracting Artifacts to be Put in a Ruleset

Related Reference

Rule Languages

Designing the Rule Project

In Rule Studio, rule projects that contain the business logic are managed separately from Java projects. Rule projects contain all artifacts related to rule writing and ruleset creation. Rule projects can reference each other like Java projects. For example, this is a good way to share a common model between several projects implementing different decisions, or to manage commonly-used rules in a project. A project that references another can see and use all elements from this project. A ruleset can come from a single project, or from a top-level rule project that references other rule projects.

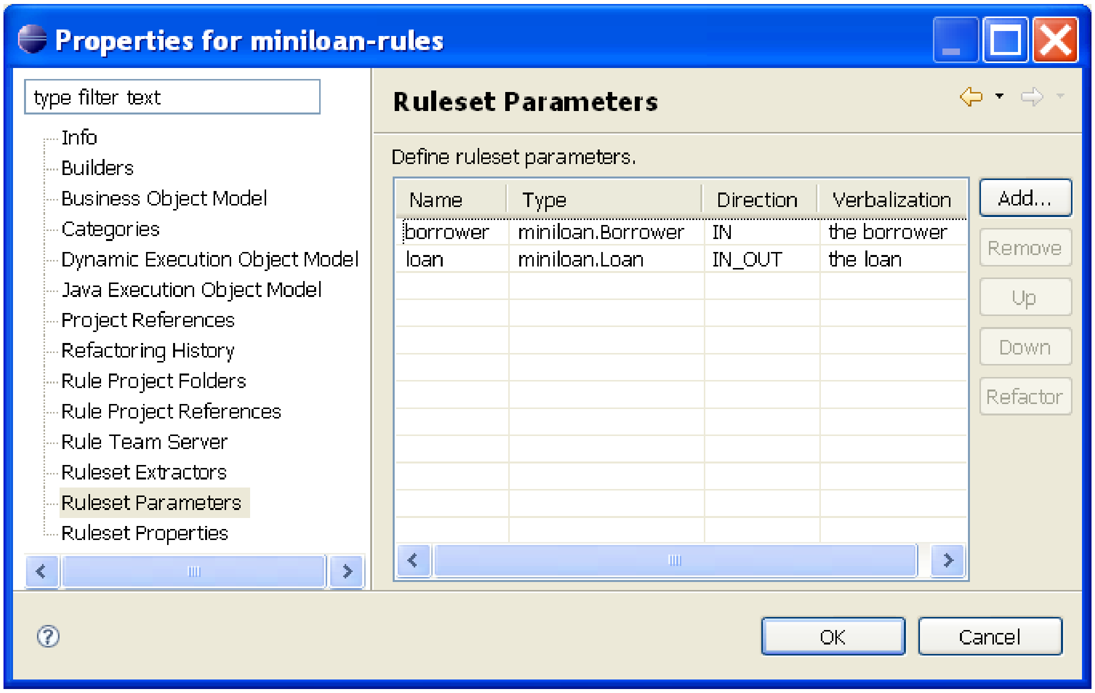


Rule Studio provides rule project templates that set a predefined structure for the rule project contents.

Specifying the Objects to be Manipulated by the Ruleset

Ruleset parameters specify the objects to be manipulated by the ruleset. They also define the interface between the business rules and the calling client. They specify the types (XML or Java) expected as input, and the types produced as output by the ruleset. The input data defines the data that is available to the rule engine for processing, while the output data contains the results of rule processing.

For example, if your decision applies to a borrower and a loan, and as a result provides information about the status of the loan, you define two parameters: `borrower` and `loan`. By providing a verbalization to the ruleset parameters, you make them available in the vocabulary for editing business rules.



Ruleset Parameters

Define ruleset parameters.

Name	Type	Direction	Verbalization
borrower	miniloan.Borrower	IN	the borrower
loan	miniloan.Loan	IN_OUT	the loan

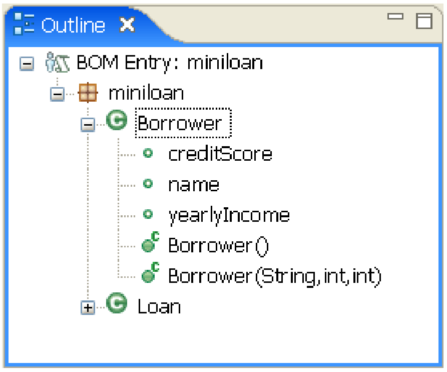
Defining the ruleset parameters involves close collaboration between business users (including business analysts) and software architects. The business users must have the data required by the business policy available, while the architects must ensure that the data can be retrieved efficiently and must understand the required inputs for invoking the ruleset, and the data available after execution is complete.

A change to the ruleset parameters can be disruptive to calling applications as it may require client code changes to make additional input data available or to process additional output conclusions. However, version management capabilities, such as those provided by Rule Execution Server, can help to manage changes in rules while maintaining compatibility with existing clients.

Defining the Entities Upon Which you Write Rules

In order to write business rules, you first need to set up the model upon which they apply. You define this model as a *Business Object Model (BOM)* in Rule Studio. The BOM is a logical model of your business domain.

For example, if policy managers in a bank want to write business rules about how loans are granted to candidate borrowers, the BOM would contain a class `Borrower` with a `creditScore` attribute, and a class `Loan`.

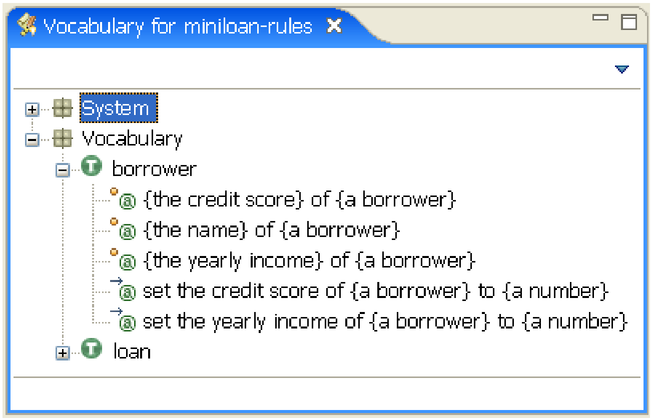


```
BOM Entry: miniloan
  miniloan
    Borrower
      creditScore
      name
      yearlyIncome
      Borrower()
      Borrower(String,int,int)
    Loan
```

Creating a Natural Language Layer on Top of the Business Object Model

To make the BOM understandable to business users, you verbalize it, that is, you attach natural-language terms and phrases to the classes and methods in the BOM. The complete set of terms and phrases available to write rules is called the *vocabulary*. Think of the vocabulary as a near natural language verbalization of the formal concepts and data elements in your business model. The vocabulary enables business users to write business rules in a near-natural language.

For example, if the BOM contains an attribute `creditScore` on class `Borrower`, the vocabulary will contain two phrases `{the credit score} of {a borrower}` and `set the credit score of {a borrower} to {a number}`.



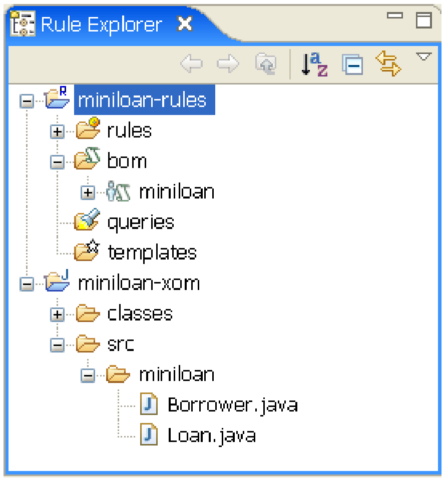
```
System
Vocabulary
  borrower
    {the credit score} of {a borrower}
    {the name} of {a borrower}
    {the yearly income} of {a borrower}
    set the credit score of {a borrower} to {a number}
    set the yearly income of {a borrower} to {a number}
  loan
```

Implementing the Physical Model Upon Which the Rules Execute

So that the vocabulary has an executable base, the BOM must be mapped to a physical data model. This data model is called the *Execution Object Model (XOM)*. Every BOM element used in a rule must have a corresponding XOM implementation, but it does not have to be a simple one-to-one mapping. You can create more complex BOM to XOM mappings, such as combining information from multiple XOM properties into one BOM element.

You can implement the XOM as a Java object model, or an XML schema (XSD). If your XOM is a Java object model, you can work on it as a Java project in Rule Studio. If your XOM is an XML schema, JRules can process schema-typed XML data using business rules without code generation. This dynamic deployment option is loosely coupled, because only the XML Schema is shared between the client and the rules, and is useful for applications that process XML. You can also use the JRules to .NET tool to develop a mapping to a .NET CLR (Common Language Runtime).

For example, the two classes `Borrower` and `Loan` in the BOM are implemented as Java classes in the XOM.



Related Concepts

Rule Project

Business Object Model (BOM)

Rulesets and Ruleset Archives

Ruleset Parameters

Execution Object Model (XOM)

Related Tasks

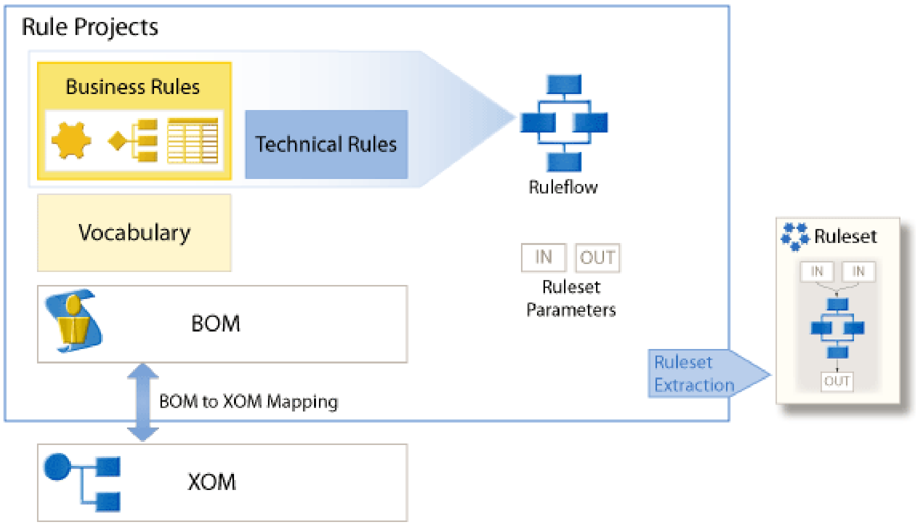
Setting Up a Rule Project

Working With BOM Entries

Creating Ruleset Parameters

Orchestrating Ruleset Execution

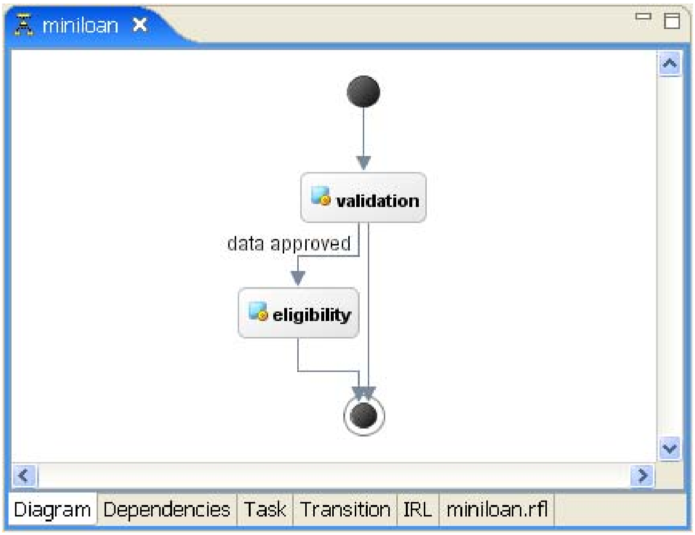
Within a rule project you organize rule artifacts into rule packages. Any application may have several aspects: business function (pricing, scoring), geography, product, responsibility, and so on. As a best practice, you should use a simple rule package structure to capture one of these aspects, and use custom rule properties to map other aspects. Rule packages are also a useful way of preparing ruleflow tasks. Each rule package may correspond to a task.



Defining the Ruleset Execution Flow

The *ruleflow* defines the execution order of rule artifacts within the context of a larger decision. To the calling application, the ruleset is invoked to make a single business decision. However, that business decision may break down into a series of smaller decisions. There may also be a routing logic that determines the appropriate sequence of smaller decisions from a set of possible paths through the ruleset. Using a ruleflow, you identify the smaller decisions as tasks and the routing logic as transitions. The transitions between tasks can have a transition condition that must be true in order to move to the next task.

For example, if your decision is made of two activities, one for validating the information about the borrower and the requested loan, and another one for deciding whether the borrower is eligible for the loan, you can create two rule packages called `validation` and `eligibility`. You store the rules and other artifacts related to each activity in the corresponding rule package. When you create the ruleflow, you create two rule tasks, `validation` and `eligibility`, by dragging and dropping the packages into the ruleflow diagram. Between the two rule tasks, you define the transitions so that the eligibility rules are only executed when the data processed by the validation rules has been approved.



With properties set on a rule task, you can specify execution information, such as the algorithm to be used in the rule engine for executing the task.

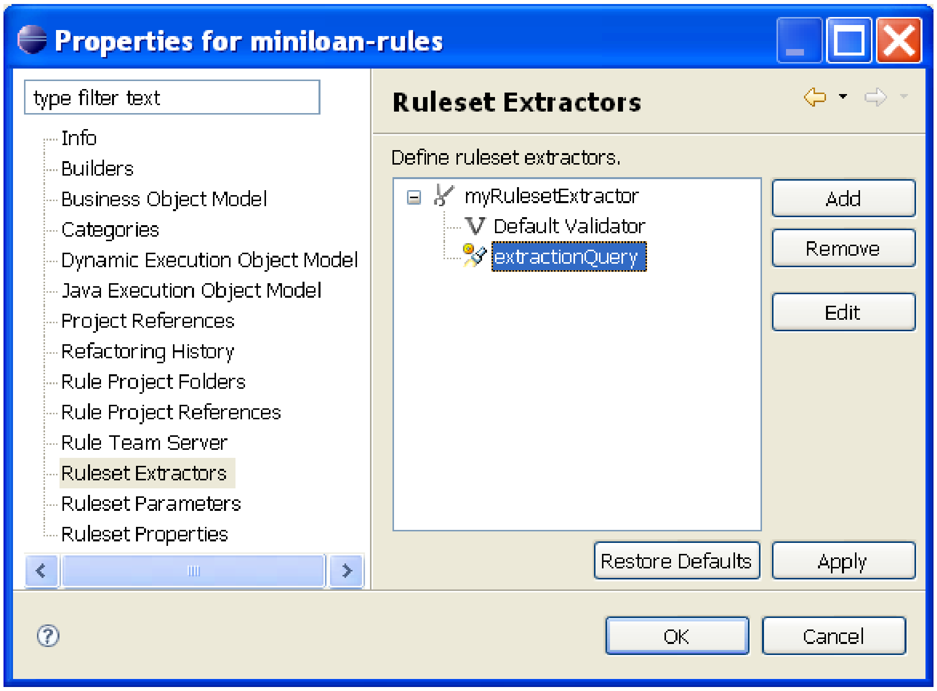
Selecting and Extracting the Rule Artifacts for a Ruleset

Ruleset extraction uses queries to select rule artifacts for a ruleset.

For example, you can write a query that finds all business rules that have the status `defined`, as follows:

```
Find all business rules
   such that the status of each business rule is defined
```

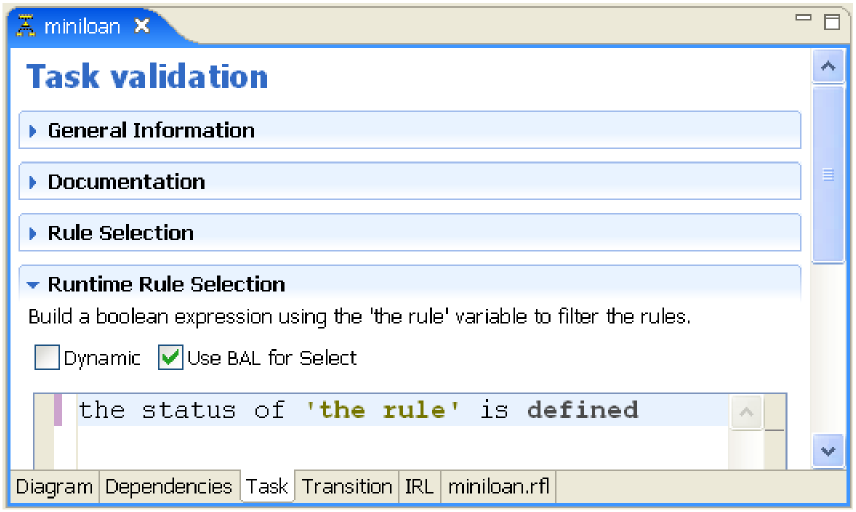
Then you use this query to define a ruleset extractor. When you extract the ruleset from the rule project, only the rules that have the status `defined` will be included in the ruleset.



When you extract a ruleset you create a ruleset archive. Each ruleset archive represents a complete decision implemented as rules in IRL, the BOM, and BOM to XOM mapping. In addition, if the XOM for the project is an XML schema, this is packaged into the ruleset archive.

Depending on your design, you decide how many rulesets you need to extract from one rule project. For example, if you know you can handle large rulesets, you can extract a single ruleset and specify mechanisms to dynamically select the rules of a ruleset at runtime. You specify this *runtime rule selection* in the ruleflow.

For example, you write the following runtime rule selection statement to make sure only the rules with the status `defined` are treated by the rule engine. These rules are in the ruleset however, and may be used in another rule task.


miniloan

Task validation

General Information

Documentation

Rule Selection

Runtime Rule Selection

Build a boolean expression using the 'the rule' variable to filter the rules.

Dynamic [✓ Use BAL for Select]

```
the status of 'the rule' is defined
```

Diagram | Dependencies | Task | Transition | IRL | miniloan.rfl

If you need to make sure the ruleset is small, for example to minimize parsing time or memory footprint, you can define several ruleset extractors in order to produce several rulesets from the same rule project.

Related Concepts

Overview: Designing Rulesets

Ruleflows

Related Tasks

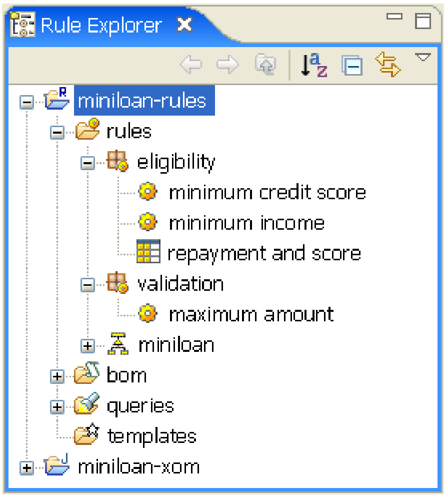
Working with Ruleflows

Extracting Artifacts to be Put in a Ruleset

Authoring and Reviewing Rules

Rule projects enable you to provide a language to business users that enables them to understand and manipulate business rules easily. At the same time, you need to make sure this language remains unambiguous and executable.

The following graphic shows a rule project containing different types of rule artifact.



Expressing Business Rules

The *Business Action Language (BAL)* provides a syntax for expressing business rules. The BAL defines a simple if-then syntax (IF some set of conditions are true THEN a set of actions should be taken), and provides constructs for expressing business rule conditions and actions.

With the vocabulary and the BAL, you can create three types of business rule artifact in Rule Studio or Rule Team Server:

- **Business rules** are sentence-like statements expressing a set of conditions followed by the actions to be taken if the conditions are found to be true.
- **Decision tables** represent the decision logic as a table. Each row in a decision table corresponds to a business rule.
- **Decision trees** represent the decision logic graphically. In a decision tree, branches of the tree represent conditions, and the leaves of the tree represent the actions to be taken if the actions are found to be true.

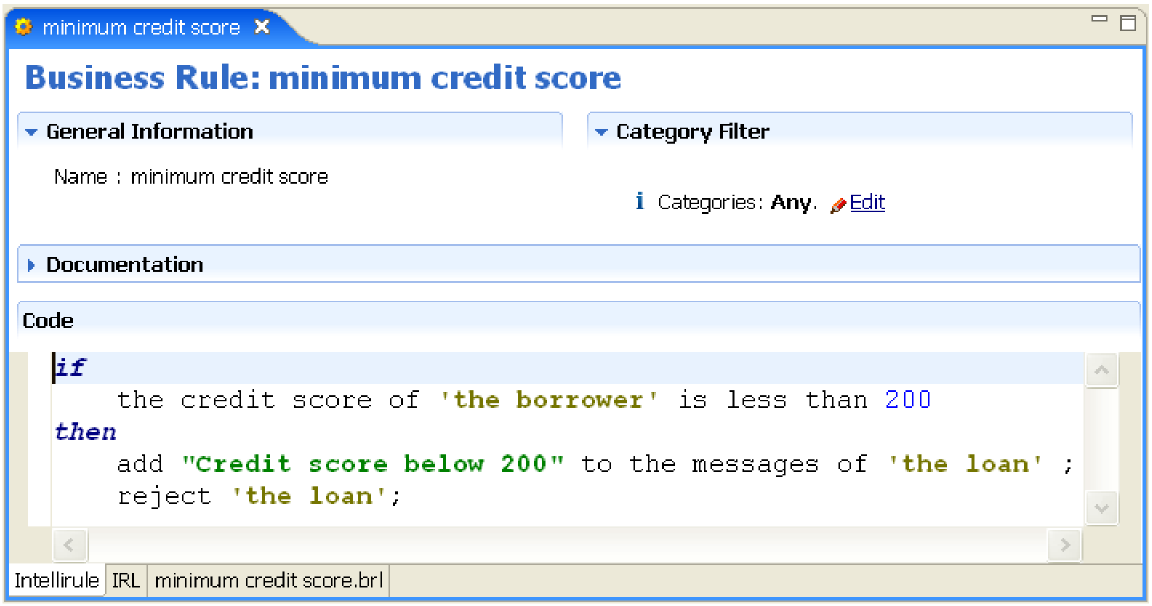
If you want to write more complex business rules that business users will not edit, you can use technical rules. Technical rules do not use the vocabulary; they apply directly to BOM elements. You use IRL to write them. Technical rules can be useful if you want to write rule actions that handle loops or exceptions, or use Java constructs that are not available in the BOM to XOM mapping.

Understanding How Business Rules are Parsed and Executed

The *ILOG Rule Language (IRL)* is the executable form of the rule artifacts. In the ruleset, all rule artifacts are translated to the ILOG Rule Language. For example, each row of a decision table is generated as a rule in IRL. Each action column corresponds to an IRL action in the corresponding rule.

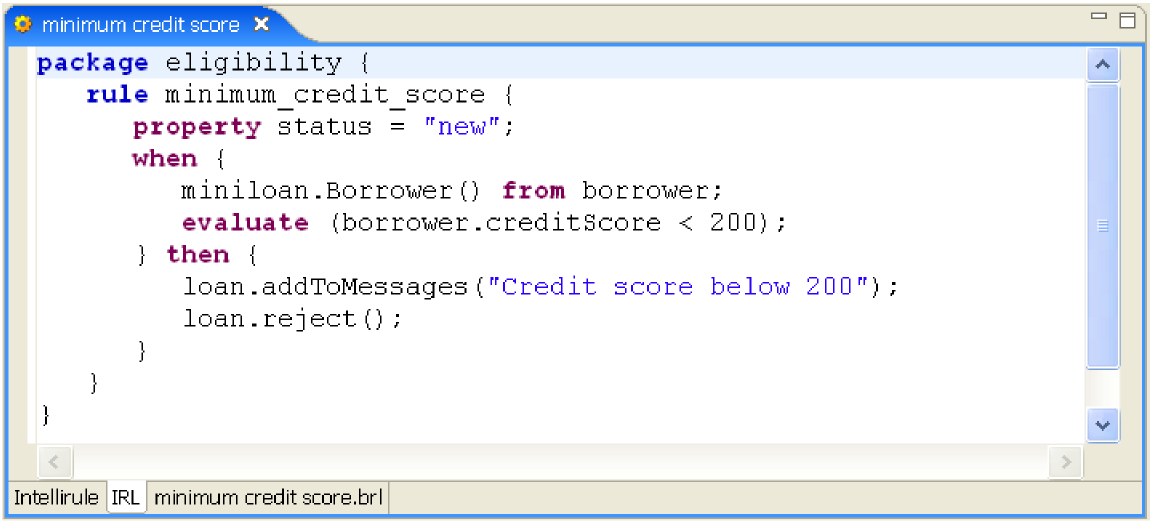
The BAL is set up to automatically translate to IRL. However if you extend it, you need to make sure you cover this translation for your new constructs.

For example, you write the following business rule using BAL.



```
if
    the credit score of 'the borrower' is less than 200
then
    add "Credit score below 200" to the messages of 'the loan' ;
    reject 'the loan';
```

The executable form of this business rule is IRL code. You can view this code by clicking the IRL tab in the rule editor.



```
package eligibility {
    rule minimum_credit_score {
        property status = "new";
        when {
            miniloan.Borrower() from borrower;
            evaluate (borrower.creditScore < 200);
        } then {
            loan.addToMessages("Credit score below 200");
            loan.reject();
        }
    }
}
```

At runtime, the rule engine parses the ruleset and uses BOM to XOM mapping to access the XOM. The business rule application can then invoke the ruleset, and provide data in the form of Java objects (if a Java XOM was specified), or XML documents (if an XSD XOM was specified), against which the rules execute.

The rule engine makes calls to the XOM Java methods in the course of evaluating and executing the rule. In this example, the rule engine calls the Java method `java.util.Collection.add` for the BOM method `addToMessages` because this method has an explicit BOM to XOM mapping defined as `this.messages.add(argument)`, `messages` being a BOM attribute of type `Collection`. For the `reject` BOM method, the rule engine calls the Java method `reject` directly.

Related Concepts

Overview: Ways of Expressing a Business Policy

Execution Object Model (XOM)

Related Tasks

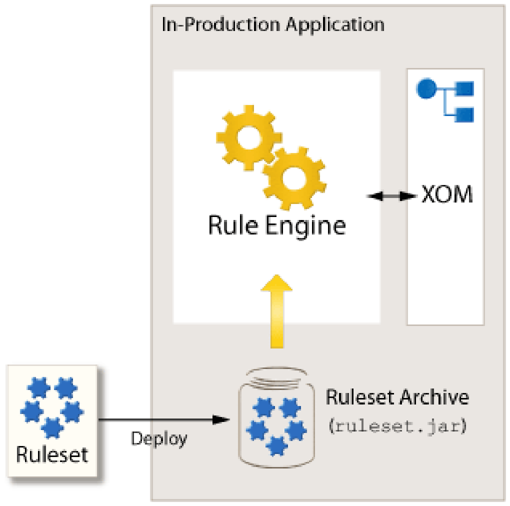
Working With BOM Entries

Related Reference

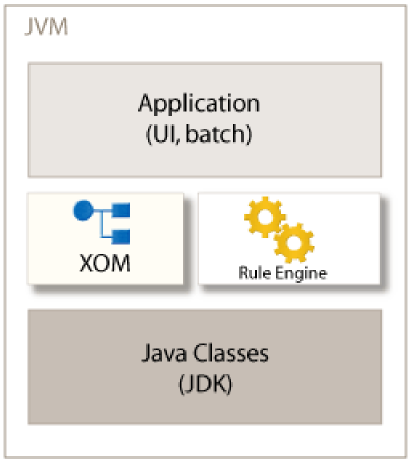
Rule Languages

Integrating Rule Execution into an Embedded Rule Engine

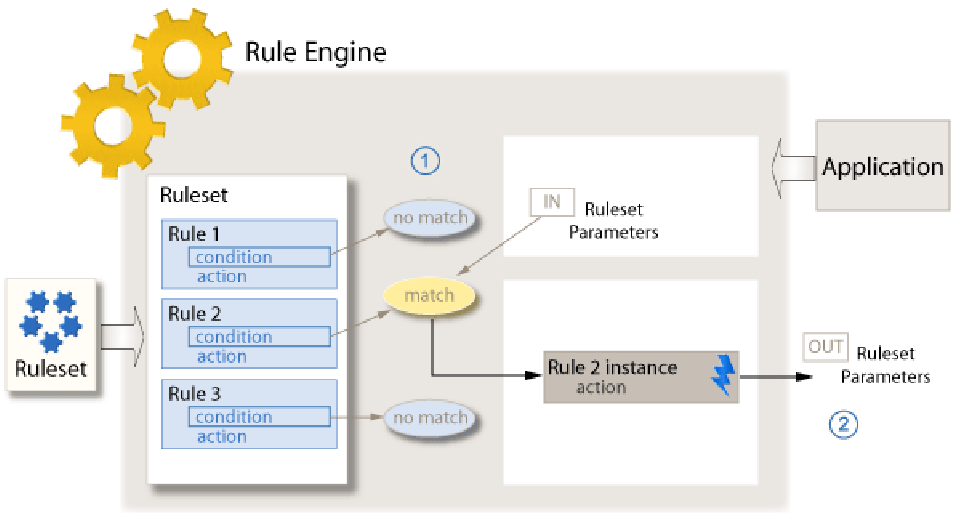
When you execute rules in an embedded environment, you deploy and execute a ruleset archive on a single Java Virtual Machine (JVM).



You include the rule engine in the class path of the application. In the application, you use the JRules API to load a ruleset, instantiate a rule engine, and execute the rule engine against objects passed as parameters.



The rule engine is a component for executing rules. It uses a ruleset and a group of objects or ruleset parameters to evaluate what rules need to be fired. Firing a rule means executing the action part of the rule in the context of the objects that have satisfied its conditions.



Step 1 of this graphic shows that the rule engine is based on pattern matching: it matches rule conditions with a group of objects or ruleset parameters.

In step 2, the rule engine creates a rule instance for each match between a rule and an object or ruleset parameter. Depending on the algorithm used by the rule engine, or execution mode, the way the rule instances are fired varies.

Related Concepts

Rule Engine

Rulesets and Ruleset Archives

Engine Execution Modes

Related Tasks

Executing a Ruleset Using a Native Rule Engine

Deploying Rulesets to a Managed Rule Execution Environment

If you need simultaneous execution of several rulesets, automatic ruleset updates, or transaction management, you must use a managed rule execution environment.

Rule Execution Server is the module for executing rules in a managed environment. It runs on a J2EE application server. Rule Execution Server can be deployed as a centralized service, executing multiple rulesets on the requests of multiple clients. It provides a variety of execution components to allow developers to easily integrate rule execution into enterprise applications.

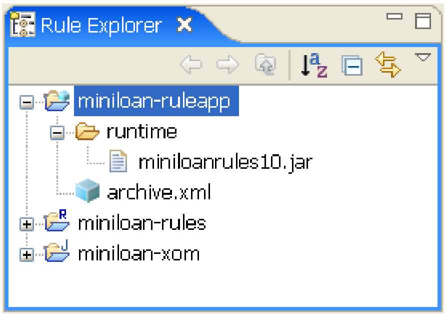
Rule Execution Server is based around a modular architecture that can be deployed as a set of J2SE compliant Plain Old Java Objects (POJOs), hosted using Apache Tomcat, or run within a full J2EE compliant application server. It provides Web-based management, management through JMX tools, logging, and debugging integration.

Rule Execution Server provides a packaging of the rule engine as a Java Connector Architecture (J2C) resource adapter. The Execution Unit (XU) resource adapter implements the J2C interactions between the application server and the rule engine.

Packaging Rulesets for Deployment

In order to be deployed, rulesets need to be packaged into *RuleApps*. A RuleApp is a deployable management unit. It can contain one or more rulesets. Each ruleset in a RuleApp can be invoked using a ruleset path. The ruleset path is the entry point for clients to access the business logic encapsulated in a RuleApp.

In Rule Studio, you create a RuleApp project to specify the rulesets contained in a RuleApp.



You can also create RuleApps using the Rule Execution Server Console, Rule Team Server, or an Ant task.

Managing Rule Engines and Rulesets

In a managed rule execution environment, rule engines are made available by an Execution Unit (XU). The XU is a resource adapter that manages rule engines and loads rulesets, and passes data between the application and the rule engine. The XU creates several rule engine instances and pools them.

The XU handles the low-level details of ruleset execution, such as pooling or multithreaded processing. It also provides management access to its resources and manages hot deployment, that is, it receives notification of new versions of the ruleset in the RuleApp storage (persistence layer). Rule Execution Server includes an implementation of the XU that can be used in any J2SE-compliant JVM.

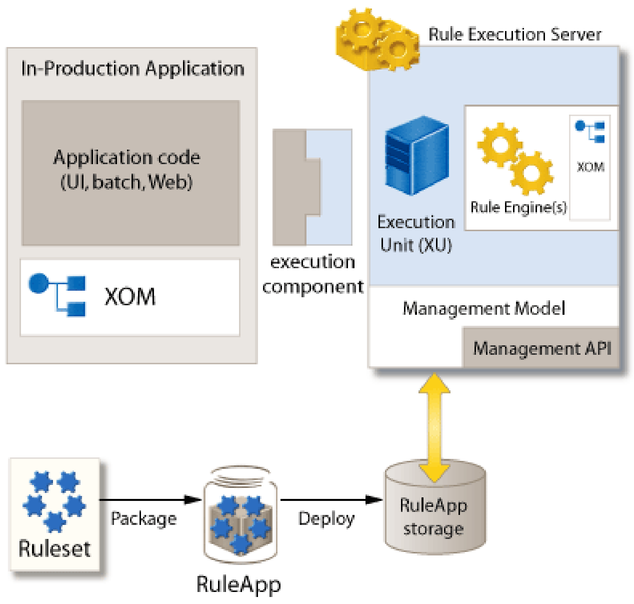
Making the XOM Accessible to the Rule Engine

The XOM is deployed to the enterprise application and made accessible to the rule engine. The way the rule engine accesses the XOM is different depending on whether it is a Java XOM or an XML XOM. If your XOM is based on Java classes, you package it into your application and deploy it along with the application. At runtime, your application classloader makes the XOM objects available to the rule execution environment. If your XOM is based on XML Schema, you deploy the XSD or WSDL to your client application. The XML Schema also packaged into the ruleset archive and therefore deployed with the RuleApp.

Choosing the Right Architecture for Managed Execution

While choosing the right architecture for managed rule execution, you decide whether you execute rules on J2SE, on an application server, or as a Web service. When defining the software architecture of your application, and fitting rule execution in this architecture, you will need to make some trade-offs.

For example, to avoid frequent network usage during rule execution, you may want to instantiate the engine close to the data it is processing, in as many nodes of the software architecture as you can. You may also want to exploit the independence of decision services and use an architecture based on a centralized Service-Oriented Architecture (SOA), in which case you need to make a trade-off between manageability and performance.

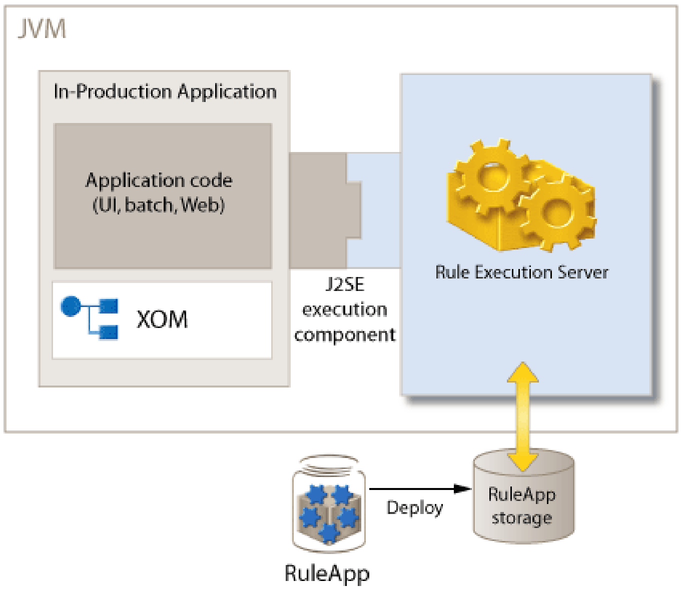


You can choose from a wide variety of integration schemes to deploy JRules. The implementation of the business service is embodied in the business rules for the service, allowing you to deploy the same decision service as a Plain Old Java Object (POJO) within one application and as a J2EE Java Message Service (JMS) message-driven bean (MDB) within another application.

ILOG has also developed a number of connectors to third-party Business Process Management (BPM) systems and SOA development tools, including the leading tools from IBM, BEA, and Oracle. See `http://www.ilog.com/solutions/business/bpm/` for more information.

Deploying Rulesets to a J2SE Environment

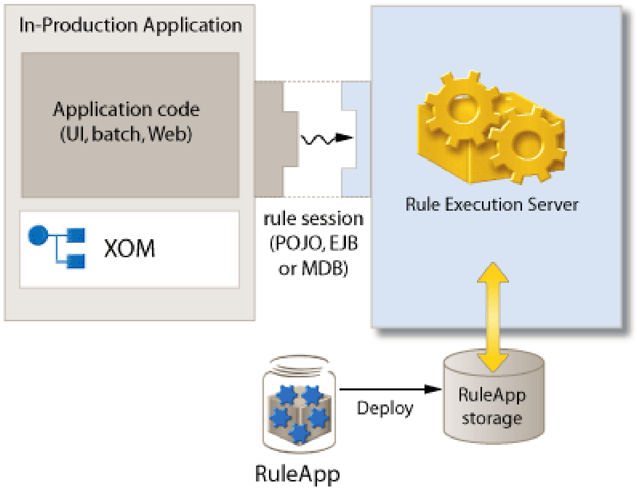
When you deploy a ruleset to a J2SE environment, your application and Rule Execution Server must run on the same JVM. You develop an execution component that handles J2SE requests between the application and Rule Execution Server.



For applications that are Web-based it is better to have the rule engine in the same JVM as the Servlet container. This sometimes minimizes the need for expensive object serialization between the JVM hosting the Servlet container and the JVM hosting the rule engine.

Deploying Rulesets to an Application Server (J2EE)

When you deploy a ruleset to an application server, the application can be remote, in which case you use an EJB or MBD rule session to handle requests between the application and Rule Execution Server. If you application is local, you use a POJO rule session or use the local interface to an EJB.


In-Production Application
Application code (UI, batch, Web)
XOM
rule session (POJO, EJB or MDB)
Rule Execution Server
Deploy
RuleApp storage
RuleApp

Synchronous invocation of a decision service from a remote client can be accomplished using the stateful or stateless EJB. Synchronous invocation of a decision service from a local client can be accomplished using the local interfaces of the stateful or stateless EJB, or using the POJOs. EJBs are useful for their remote client access capabilities as well as support for declarative transaction and security descriptors. POJOs are useful for their simpler packaging and deployment, and for use outside the EJB container.

Applications that require asynchronous invocations of decision services may use the Message-Driven Bean (MDB), which provides a scalable means to invoke rulesets where high-latency or high peak-load is expected.

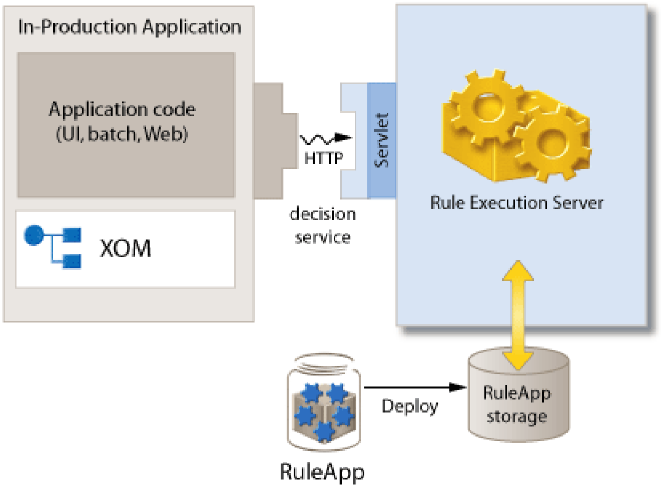
In general, you must tune the J2EE application server for the deployed application and the expected peak or average load. This includes settings like:

- Number of request threads
- Size of EJB pools
- Size of JCA pools
- Using native I/O or pure Java I/O
- Pool reclamation policy
- Data replication strategy for clustered deployments

Refer to the documentation of your application server for specific information related to your environment.

Deploying Rulesets as a Web Service

When you deploy a ruleset as a Web service, your application is typically remote and requests to Rule Execution Server are done through HTTP. You develop a transparent decision service to handle requests between the application and Rule Execution Server. JRules provides tools for generating transparent decision services.



You can implement the servlet with a monitored transparent decision service implemented with the Java API for XML Web services (JAX-WS 2.0), see `http://java.sun.com/webservices/jaxws/`, or with a hosted transparent decision service provided by JRules.

JRules provides tools that can help with key SOA governance challenges, such as controlling access to the implementation rules for a service, providing an audit trail for all implementation changes, business level reporting, as well as runtime monitoring of the rulesets that implement a transparent decision service.

Related Concepts

Rule Execution Server

Transparent Decision Services

Execution Unit (XU)

Execution

Related Tasks

Executing a Ruleset Using Rule Execution Server

Creating a Web Service or Monitored Transparent Decision Service Project for RuleApps

Tracing Ruleset Execution

Enabling Business Rule Management for Business Users

To enable business users to create and edit business rules, you can publish rule projects to Rule Team Server, a Web-based module for storing, managing, and editing business rules. After the rule project has evolved in Rule Team Server, you can synchronize its contents back into Rule Studio.

Simplifying Rule Editing for Business Users

When you have a large vocabulary, *categories* let you filter the terms and phrases that are available to you when you edit rules in Rule Studio or Rule Team Server. Only those vocabulary terms and phrases in a category are visible in a business rule with that category assigned.

You can also use *business rule templates* to guide policy managers to write rules properly and create many similar rules efficiently. A business rule template is a partly completed business rule that can be used to create a series of rules with the same structure. If you find that you are using templates to manage many rules with shared conditions and actions and are only modifying values, you may consider managing them as a decision table.

Adding Custom Properties to Rule Artifacts

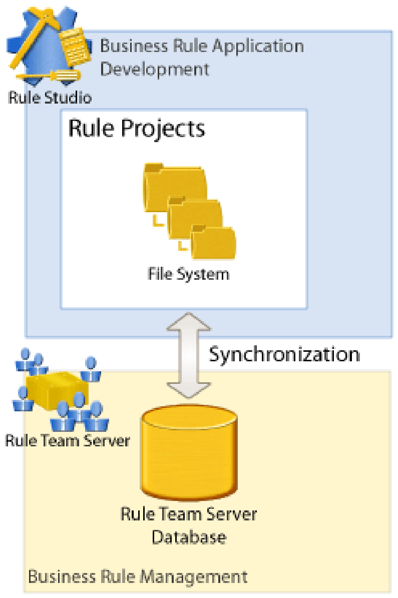
To add properties to existing types of rule artifact, you extend the rule model. BAL and rule model extensions can be set up once and then deployed on both Rule Studio and Rule Team Server.

For example, you could create a property `geography` to identify the country for which a rule is applicable.

Synchronizing Work Between Business and IT Users

Developers store and access business rules using ILOG Rule Studio for .NET. Rule Studio stores business rules and other business rule artifacts (such as business object models) in a file system. You can integrate this file system into a Source Code Control (SCC) system for file sharing, conflict resolution, and version management. In an SCC system, each developer works on a copy of the rule projects in their Eclipse workspace, and a copy is kept in the SCC system. All interaction with the SCC system is handled by Eclipse.

Business users store and access business rules using Rule Team Server. Rule Team Server stores rule projects in a database repository. The Rule Team Server repository handles multi-user concurrency and versioning. In Rule Team Server, all users collaborate within a shared workspace, and use locking to control access to resources currently being edited.



In Rule Studio, you can synchronize the content of rule projects within Rule Team Server. Synchronization provides for:

- creating a Rule Team Server project from an existing Rule Studio project
- creating a Rule Studio project that is a copy of an existing Rule Team Server project.
- updating the Rule Studio copy of a project that exists in both Rule Studio and Rule Team Server with changed and new content from Rule Team Server.
- publishing changes in the Rule Studio copy of a project that exists in both Rule Studio and Rule Team Server to the Rule Team Server copy.

The way you implement business rule management with Rule Studio and Rule Team Server depends on who the owners of the business policy are, and whether you use the Rule Team Server repository as production storage.

- **Developer-centric rules**

  The software development cycle and business policy cycle are managed together in Rule Studio. Projects are published to Rule Team Server for review and verification deployment. Business users are given access to the rules in Rule Team Server, but they are not yet empowered to make changes. When a project is put into production, the Rule Studio project is published to Rule Team Server. When business users identify needs for rule changes, they work with developers to have the changes implemented in Rule Studio and the updates are published to Rule Team Server where the business users have an up-to-date copy of the rules. Versions are maintained in Rule Team Server this way, but change is driven from Rule Studio.

Since all rule authoring and management is performed from Rule Studio, this approach is not appropriate for projects requiring users not familiar with Java and IDEs. It also does not provide for any role-based permission management for rules.

◆ **Separated business and IT cycles**

Developers use, almost uniquely, Rule Studio. Policy managers are responsible for the business policy life cycle in Rule Team Server. The Rule Team Server repository is deployed to production.

Business users can make rule updates in Rule Team Server. Developers carry out initial rule development and templates for new rules in Rule Studio. When the project is stable, but before it is put into production, they publish the rule project to Rule Team Server for business user review. Business users are able to make updates and create new rules based on templates. The developers synchronize any rule changes back to Rule Studio and deploy from there. Rule changes may be made either in Rule Studio or Rule Team Server and are synchronized.

Post-production updates are driven by Rule Team Server. Once the application is in production, not only can business users make rule changes, they can also deploy rule changes from Rule Team Server to a test server where they can test the changed rules with the existing application before having IT complete the deployment to production.

Related Concepts

Categories

Business Rule Templates

Rule Project Storage

Rule Project Sharing and Synchronizing

Guidelines for Sharing Repositories Between User Profiles

Related Samples and Tutorials

Permissions Tutorial

Baseline Management Tutorial

Tutorials and Demos

The following sections list all the tutorials and flash demos in JRules, who they are designed for, and what their objectives are. Apart from the Quick Start Tutorial, which uses all the JRules, all other tutorials are grouped by the JRules module required to run the tutorial.

Quick Start Tutorial and Demos

The Quick Start tutorial provides an introduction to business rules and Business Rule Management Systems, followed by a series of tasks that describe each stage in the creation of a JRules-powered application. It is aimed at project architects and developers.

Each task is illustrated by a short flash demo, which you can watch or download from `http://www.ilog.com/products/jrules/quickstart/demos`.

To get started with JRules, see *Welcome to the Quick Start Tutorial*.

Rule Studio Tutorials

This table lists the tutorials that are provided to help you to get started with the rule development features available in Rule Studio

Note: *Tutorials for Rule Studio are delivered so that they compile and run in JDK 1.4 and 5.0. However, if you compile with JDK 5.0 some warning messages may appear. This is an expected behavior. If you want to remove these warnings, do the following:*

1. *In Rule Studio, on the **Window** menu, click **Preferences**.*
2. *On the left pane of the **Preferences** dialog, click **Java** > **Compiler**.*
3. *On the Compiler page, in the **Compiler compliance level field**, select* `1.4`.
4. *Click **Apply**.*
5. *Rule Studio prompts you to rebuild your projects. Click **Yes**.*

Table 1 *Tutorials for Developers and Business Analysts in Rule Studio*

Tutorial	**Audience**	**Objectives**	
		Concepts learned	**Tasks achieved**
Tutorial: Defining a Vocabulary	◆ Developer ◆ Business Analyst	◆ Business Object Model (BOM) ◆ Execution Object Model (XOM) ◆ Vocabulary layers	◆ Create a rule project ◆ Create a BOM entry ◆ Verbalize the BOM for rule editing ◆ Extend the BOM
Tutorial: Creating Business Rules	◆ Developer ◆ Business Analyst	◆ Basic rule programming principles ◆ Use of Categories ◆ Else part of a rule	◆ Create a rule ◆ Create and use rule templates ◆ Set up categories ◆ Use an enumerated domain
Tutorial: Editing Decision Tables	◆ Business Analyst	◆ Decision tables ◆ Decision table editing errors and warnings ◆ Overlapping interval data	◆ Create a decision table ◆ Simulate an error ◆ Import data from Excel ◆ Change a decision table's formatting

Table 1 *Tutorials for Developers and Business Analysts in Rule Studio*

Tutorial	Audience	Objectives	
		Concepts learned	**Tasks achieved**
Tutorial: Creating Your First Ruleflow	◆ Developer	◆ Elements of a ruleflow ◆ Rule engine algorithms ◆ Transition conditions	◆ Create and edit a ruleflow ◆ Define transitions ◆ Define a transition condition on a ruleset parameter
Tutorial: Debugging a Ruleset	◆ Developer	◆ Ruleset archive ◆ Execution order ◆ Update object state property ◆ Ruleflow debugging	◆ Extract a ruleset archive ◆ Launch a debug session ◆ Set a breakpoint
Tutorial: Debugging a Remote Rule Execution Server Application	◆ Developer	◆ RuleApps ◆ Rule Execution Server	◆ Extract a RuleApp and configure a target server from Rule Studio ◆ Execute a ruleset on different application servers ◆ Debug rulesets deployed on Rule Execution Server, from Rule Studio ◆ Update the ruleset version to simulate "hot deployment"

Rule Execution Server Tutorials

This table lists the tutorials that are provided to help you get started with the integration and RuleApp management features of Rule Execution Server.

Table 2 *Tutorials for Developers and System Administrators using Rule Execution Server*

Tutorial	**Audience**	**Objectives**	
		Concepts learned	**Tasks learned**
Tutorial: RuleApp Management	◆ Developer ◆ Administrator	◆ RuleApp management ◆ RuleApp versioning mechanisms	◆ How to use the Rule Execution Server Console ◆ How to use Rule Execution Server Ant tasks
Tutorial: Executing a Hosted Transparent Decision Service	◆ Developer ◆ Business Analyst	◆ Transparent decision service	◆ How to expose a ruleset as a transparent decision service and how to call it using different clients

Table 2 Tutorials for Developers and System Administrators using Rule Execution Server

Tutorial	Audience	Objectives	
		Concepts learned	Tasks learned
Tutorial: Creating a Web Application to Invoke JRules in BEA WebLogic Workshop	◆ Developer	◆ Developing for Rule Execution Server in BEA WebLogic Workshop	◆ Integrate Rule Execution Server into WebLogic Workshop ◆ Manually deploy Rule Execution Server stacks in the WebLogic Server ◆ Import the execution components JAR into Workshop ◆ Add a reference description of the JRules EJBs in a Web application ◆ Deploy a RuleApp to Rule Execution Server ◆ Execute a ruleset on Rule Execution Server

Table 2 *Tutorials for Developers and System Administrators using Rule Execution Server*

Tutorial	**Audience**	**Objectives**	
		Concepts learned	**Tasks learned**
Tutorial: Creating a Web Application to Invoke JRules on IBM Rational Application Developer	◆ Developer	◆ Developing for Rule Execution Server in IBM Rational Software Development Platform	◆ Integrate Rule Execution Server into Rational Software Development Platform ◆ Create a server configuration project for a test server or a full implementation of Rational Software Development Platform ◆ Add a data source to the server configuration ◆ Configure the Rule Execution Server Console with the data source ◆ Create a new J2EE application ◆ Import the Rule Execution Server modules to the project ◆ Create a Web project and object model ◆ Add an EJB reference to your Web application ◆ Deploy a RuleApp to Rule Execution Server ◆ Execute a ruleset on Rule Execution Server

Rule Team Server Tutorials

This table lists the tutorials that are provided to help you get started with the rule editing and management features of Rule Team Server.

Table 3 *Tutorials for Business Analysts, and Policy Managers in Rule Team Server*

Tutorial	Audience	Objectives	
		Concepts learned	Tasks learned
Rule and Decision Table Editing Tutorial	◆ Policy Manager ◆ Business Analyst	◆ Business rules ◆ Decision tables	◆ Create and edit a business rule ◆ Edit a decision table
Views and Queries Tutorial	◆ Policy Manager ◆ Business Analyst	◆ Queries ◆ Smart views	◆ Create a new query to extract required rules ◆ Creating a new smart view to display the required rules
Permissions Tutorial	◆ Administrator	◆ Permissions ◆ Groups	◆ Implement fine-grained project security in Rule Team.
Baseline Management Tutorial	◆ Business Analyst	◆ Project baselines ◆ Deployment baselines ◆ Dependent projects	◆ Create, consult, and restore a baseline ◆ Update a deployment baseline
Rule Analysis Tutorial	◆ Policy Manager ◆ Business Analyst	◆ Rule analysis ◆ Semantic queries	◆ Set rule analysis options ◆ Perform checks on query results

Rule Scenario Manager Tutorial

This table lists the tutorial that is provided to help you get started with Rule Scenario Manager.

Table 4 *Tutorials in Rule Scenario Manager*

Tutorial	**Audience**	**Objectives**	
		Concepts learned	**Tasks learned**
Tutorial	◆ Developer ◆ Business Analyst	◆ Scenario ◆ Scenario suite ◆ Simulation	◆ Test and simulate business rules ◆ Create a scenario

Flash Demos

This table lists the flash demos that are provided with JRules:

Table 5 *Flash Demos for Developers and Business Analysts in Rule Studio, Rule Execution Server, and Rule Team Server*

Demos	**Audience**	**Objectives**	
		Concepts learned	**Tasks learned**
Flash Demo: Synchronizing Between Rule Studio and Rule Team Server	◆ Developers ◆ Business Analysts	◆ Publishing ◆ Synchronization	◆ Publish an existing rule project from Rule Studio to Rule Team Server ◆ Synchronize changes and handle conflicts.
Flash Demo: Rule Team Server Tour	◆ Developers ◆ Business Analysts	◆ The Explore tab. ◆ The Compose tab. ◆ The Query tab.	◆ Navigation ◆ Orientation.
Flash Demo: Deploying Rules to Rule Execution Server	◆ Developers ◆ Business Analysts	◆ Rule Execution Server Configurations ◆ RuleApp projects ◆ RuleApp deployment	◆ Create a RuleApp project ◆ Export and deploy a RuleApp

Using this Online Documentation

Describes how to use this online documentation, which consists of a collection topics that are organized by subject. You can access each topic directly from the table of contents in the left pane, from a link in another Help topic, or from the results of a search.

In this Section

Navigating the Online Help

Using the Search

Related Sections

Introducing ILOG JRules

ILOG JRules Architecture Guidelines

Tutorials and Demos

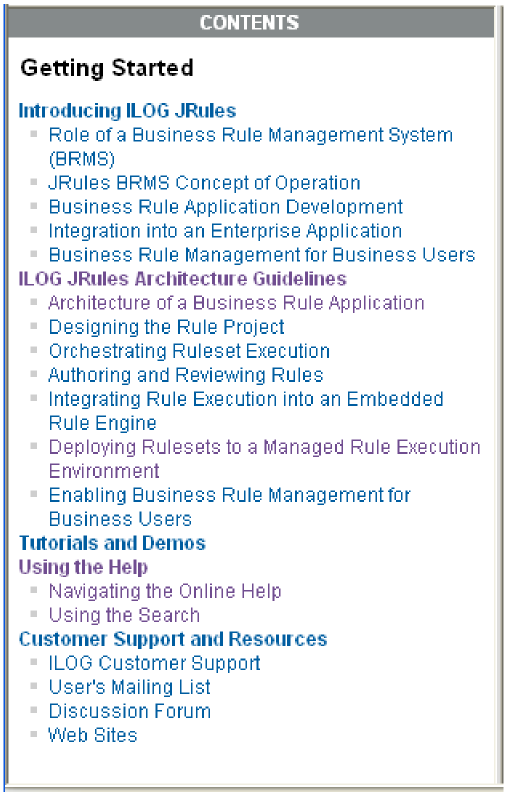
Technical Support and Resources

Navigating the Online Help

This topic describes how to use this online help.

The ILOG online documentation pages are made up of three frames.

The **left frame** displays a table of contents:



CONTENTS

Getting Started

Introducing ILOG JRules
- Role of a Business Rule Management System (BRMS)
- JRules BRMS Concept of Operation
- Business Rule Application Development
- Integration into an Enterprise Application
- Business Rule Management for Business Users

ILOG JRules Architecture Guidelines
- Architecture of a Business Rule Application
- Designing the Rule Project
- Orchestrating Ruleset Execution
- Authoring and Reviewing Rules
- Integrating Rule Execution into an Embedded Rule Engine
- Deploying Rulesets to a Managed Rule Execution Environment
- Enabling Business Rule Management for Business Users

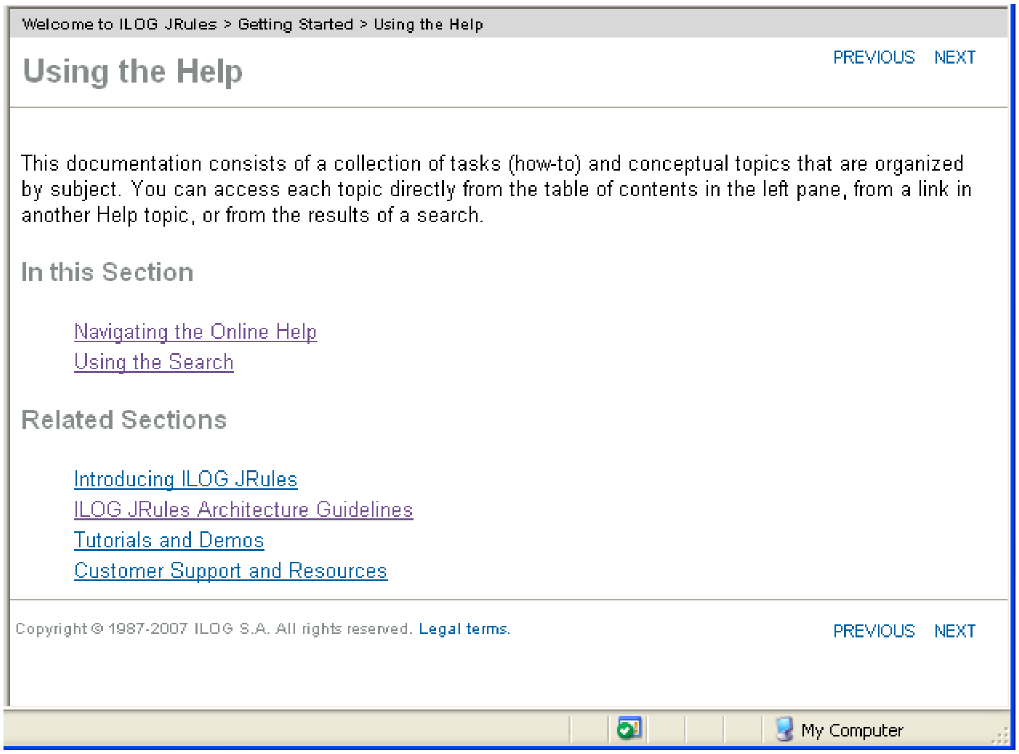
Tutorials and Demos

Using the Help
- Navigating the Online Help
- Using the Search

Customer Support and Resources
- ILOG Customer Support
- User's Mailing List
- Discussion Forum
- Web Sites

If you are on the Home page, the table of contents lists all the manuals supplied in the documentation set. If you are on another page, it displays the table of contents for the manual that you are consulting. To select a manual or topic, click on the manual or topic title in the left frame.

The **central frame** displays the content of the topic you have selected:



Welcome to ILOG JRules > Getting Started > Using the Help

PREVIOUS NEXT

Using the Help

This documentation consists of a collection of tasks (how-to) and conceptual topics that are organized by subject. You can access each topic directly from the table of contents in the left pane, from a link in another Help topic, or from the results of a search.

In this Section

Navigating the Online Help
Using the Search

Related Sections

Introducing ILOG JRules
ILOG JRules Architecture Guidelines
Tutorials and Demos
Customer Support and Resources

Copyright © 1987-2007 ILOG S.A. All rights reserved. Legal terms.

PREVIOUS NEXT

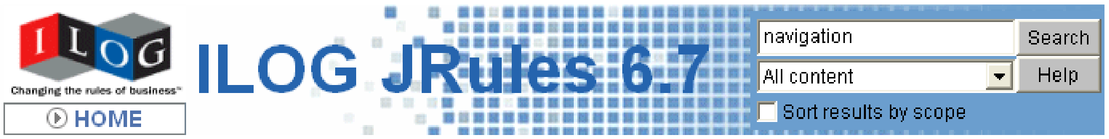
From here, you can navigate within the current manual using the **Previous** and **Next** links, at either the top or the bottom of the page. You can also use the table of contents shown in the left frame.

The "bread crumb" across the top of the page tells you where you are. For example, the bread crumb for the page shown above is:

Welcome to ILOG JRules > Getting Started > Using the Help

You can use the bread crumb to navigate to a different part of the current manual or to return to the documentation Home page. For example, from the current page, you could click Getting Started to return to the top level page of this manual, or click Welcome to ILOG JRules to return to the Home page.

The **top frame** provides you with a Home button to return to the documentation Home page:



It also provides you with a search function, to look for specific topics. See *Using the Search* for more details.

In addition, you can use the browser Back button to return sequentially to previous pages you have viewed, and the Forward button to return to the page you just left.

> ***Note:*** *If JavaScript is not enabled in your browser, the table of contents does not synchronize automatically when you click a link to another manual. In this case, click the name of the manual in the bread crumb at the top of the central frame. Doing this will update the table of contents of the manual and reload its first page.*

Browser Compatibility

The ILOG online documentation is designed to function properly on:

◆ Netscape 6.2.x and 7.0 on Windows, Solaris, and Linux.

◆ Mozilla Firefox 1.1 to 2.0 on Windows, Solaris, and Linux.

◆ Internet Explorer 6 and 7 on Windows.

Related Tasks

Using the Search

Using the Search

The search applet is based on the Lucene Project from Apache (see `http://lucene.apache.org/java/docs/index.html`). Lucene accepts “*” and “?” as wildcards, but they must not be at the start of the string. For more information, see *Search Engine Query Syntax*, or click **Help** in the Java applet.

To deactivate active content warnings in IE, select the option **Allow active content to run in files on My Computer** from **Tools** > **Internet Options** > **Advanced** > **Security**.

To activate the Java applet in Mozilla Firefox, click **Tools** > **Options**, click **Content**, and then ensure that the **Enable Java** and **Enable JavaScript** check boxes are selected.

There is also a FAQ on the Lucene site for further information. See `http://wiki.apache.org/jakarta-lucene/LuceneFAQ`.

The search applet has been customized for JRules to:

- Provide an option to sort the search results by scope. To use this feature, select the check box labelled **Search results by scope**, enter your search string, and then click **Search**. If the **Search results by scope** check box is left unchecked, the results will be sorted by relevance.
- Apply the search on a single section. To search a single section, select the name of the section you want to search in the drop-down menu in the applet, enter your search string, and then click **Search**.
- Exclude the JRules API from the search. To search all of the sections with the exclusion of the API, select **All content except the JRules API** in the drop-down menu in the applet, enter your search string, and then click **Search**.

By default, the scope of the search is set to include all of the sections, that is, **All content** in the drop-down menu. The search results are listed in order of relevance.

***Tip:** Do not forget that you can find and highlight words on the current Web page you are viewing by pressing **Ctrl+f**. Simply type the word to look for and press **Enter**.*

Related Sections

Navigating the Online Help

Technical Support and Resources

Contains information on how to obtain help and technical support from ILOG worldwide, should you encounter any problems using ILOG JRules.

In this Section

ILOG Technical Support

For technical support, contact your local distributor or, if you are a direct ILOG customer, contact one of the offices listed below according to your location. The ILOG technical support you depend on is defined in the Exhibit A of your maintenance agreement.

Access to technical support is allowed for owners of an ongoing maintenance contract number. To access the technical support pages on our Web sites you will need the maintenance contract number and the name of the person this contract was sent to in your company.

The use of e-mail is encouraged to obtain faster service.

Countries	Telephone	e-mail Address	Web Site
The Americas	1 877 ILOG TECH (1 877 456 4832) (Toll free) or +1 408 991 7080 6 am to 6 pm Pacific time	`jrules-support@ilog.com`	`http://support.ilog.com`
France	0 800 09 27 91 (Numero vert) 9 am to 6 pm local time	`jrules-support@ilog.fr`	`http://support.ilog.fr`
United Kingdom	+44 1344 661 630 8 am to 5 pm local time	`jrules-support@ilog.co.uk`	`http://support.ilog.fr`
Spain	+34 902 170 295 9 am to 6 pm local time	`jrules-support@ilog.es`	`http://support.ilog.fr`
Germany	+49 6172 40 60 33 9 am to 6 pm local time	`jrules-support@ilog.de`	`http://support.ilog.fr`
Europe - Middle East - Africa	+33 1 49 08 35 04 9 am to 6 pm Paris time	`jrules-support@ilog.fr`	`http://support.ilog.fr`
Asia	+65 6773 0626 9 am to 6 pm Singapore time	`jrules-support@ilog.com.sg`	`http://support.ilog.com.sg`
Japan	+81 3 5211 5770 9 am to 6 pm local time	`jrules-support@ilog.co.jp`	`http://support.ilog.com.sg`

Related Sections

User’s Mailing List

Discussion Forum

Web Sites

User’s Mailing List

The electronic mailing list `jrules-list@ilog.fr` is the best way for you to receive the latest news about JRules events such as beta tests, new releases, new technical resources and update releases. Subscription to this list is subject to an ongoing maintenance contract.

To subscribe to this list you must go the customer support Web site and navigate to the JRules product support pages in the Products section. The User's List link enables you access a page where you can subscribe to the JRules mailing list.

Related Sections

ILOG Technical Support

Discussion Forum

Web Sites

Discussion Forum

ILOG now provides a discussion forum to enable JRules users to share information, exchange ideas and swap tips: the ILOG Discussion Forum.

To access the forum you will need to sign in to the forums from your `myILOG` Support account.

Connect to the web site corresponding to your location (see *ILOG Technical Support*) and fill in the fields in the right column. Enter your e-mail address and password. Click the **Discussion Forum** link in the left column of the JRules homepage.

These online forums are for user-to-user discussions. They are not an official customer support channel for ILOG. If you require direct assistance, or prefer to contact ILOG Support staff directly, please contact ILOG Support.

Related Sections

ILOG Technical Support

User's Mailing List

Web Sites

Web Sites

To obtain more information about ILOG products, connect to the Web site corresponding to your location.

- Americas—`http://support.ilog.com`
- Asia Pacific—`http://support.ilog.com.sg`
- Europe, Africa, and Middle East—`http://support.ilog.fr`

Related Sections

ILOG Technical Support

User’s Mailing List

Discussion Forum