IBM WebSphere ILOG Business Rule Management Systems: The Case for Architects and Developers
November 2009

WebSphere® software

IBM WebSphere ILOG Business Rule Management Systems: The Case for Architects and Developers

Brett Stineman

Product Marketing, Business Rules Management (WebSphere),

Application and Integration Middleware Software,

IBM Software Group

Contents

Introduction

In a world where business agility—the ability to quickly and efficiently adapt to changing markets, competitive actions, regulatory and legislative mandates, or other challenges—is a hallmark of successful organizations, information technology plays a critical role. In fact, the role of IT systems has become intertwined with how organizations operate, enabling significant productivity and efficiency improvements.

But at the same time, these improvements have come at a cost. For many organizations, their operational systems are a black box to the people who direct business strategies and policies. They entrust the implementation of policies within IT systems to their technical development staff, and they frequently expect the implementation to be handled more rapidly than can be feasibly accomplished. This frequently results in frustration for both the line-of-business (LOB) groups and IT: LOB wants better visibility of the logic that drives their systems, along with the ability to make changes as needed; IT wants to maintain control over systems, while being able to hand off certain maintenance aspects of these systems tied to the business domain and knowledge, so they can focus their resources on adding new functionality and building improved systems.

The focus of this paper is to explain how a business rule management system (BRMS) can help organizations become more agile, while meeting the needs of both LOB and IT. For enterprise architects and application developers, this paper will provide an overview of how the IBM® WebSphere® ILOG BRMS offerings can help them build highly flexible solutions that provide their LOB constituents with improved access, insight and direction over the decision logic which drives the behavior of critical business systems.

What is a Business Rule Management System?

A Business Rule Management System (BRMS) is an integrated application development, maintenance and execution platform. It enables organizations to define, deploy, monitor and maintain highly variable and complex decision logic used by operational systems. This decision logic is also referred to as “business rules”—you can think of conditional statements used in activities such as pricing, underwriting, eligibility approvals, product recommendations, and determinations of straight-through processing vs. manual intervention (for example, insurance claims). Business rules are typically found inside application code in the form of if-then-else statements, or they might be defined in process models. They can also be stored in documentation (for example, procedural manuals)—they may even be undocumented, in the minds of the subject matter experts who have to be involved in specific operational situations.

A BRMS allows decision logic to be centralized and managed as an enterprise asset, so it can be easily understood, maintained and reused across the organization. By externalizing rules from application code—and other places where rules exist—business experts can be more closely involved in the definition, maintenance and management of decision logic, reducing the amount of time and effort required to update production systems, and increasing the organization’s ability to respond to changes in the business environment.

A BRMS includes three primary components:

- **Rule Repository:** allows rules to be managed separately from the systems that utilize them, making it easier to understand and update decision logic, along with consolidating rules from disparate applications and information silos so that they can shared and reused across the organization.
- **User Tools:** provides both IT and LOB with functionality for defining and managing decision logic, while also supporting collaboration between them for application development and maintenance. BRMS tools should offer specific environments and capabilities to meet the needs of various technical and non-technical roles involved in the rule management life cycle.
- **Runtime engine:** enables production systems to access and execute decision logic managed within the BRMS; the “rule engine” allows complex and inter-related rules to be executed based on specific process or transaction context, using a combination of data inputs, sets of applicable rules and execution algorithms in order to provide outputs back to the invocation from the requesting system.

By implementing a BRMS, an organization is able to effectively deal with the problems associated with traditional embedded decision logic. The organization gains visibility and access to business rules, along with the ability to more easily define and automate decisions in operational systems. In addition, a BRMS allows systems to behave with more intelligence, precision and consistency, determining highly customized decisions if the situation allows, while also enforcing standardization if required (for example, regulatory compliance requirements).

Roles Involved in Business Rule Management

Applications constructed around a BRMS place the ability to author, test and manage the rules that implement business policies directly into the hands of LOB personnel. Policy changes can be driven exclusively by business imperatives, and thus evolve independently of the application development cycle that too frequently constrains change in information-intensive businesses. This also frees application architects and developers to focus on areas where they can provide value through their expertise, such as implementing updated architectures for their enterprise applications, improving data management and enterprise application integration, and creating improved IT systems, rather than spending significant time and resources on repetitive activities to implement changes to business policies.

In order to provide these benefits, a BRMS must satisfy the divergent requirements of the different stakeholders involved in creating and maintaining rule-based systems:

- **Architect:** responsible for assuring that the application design meets long-term business needs for functionality, efficiency and performance.
- **Developer:** creates and tests the application, as well as iteratively adding functionality to the application as required. (Developers also build test scenarios, some of which can be used by LOB in testing or simulating rule changes and some of which are used to run comprehensive regression tests prior to deploying each version of a rule application into production.)
- **Business Analyst:** responsible for modeling the business domain and business rule vocabulary. (Business analysts generally act as the bridge between the business and IT departments, translating business policies into a formal specification acceptable to developers, and validating the formal specification with policy managers.)
- **Policy Manager:** LOB subject matter expert responsible for translating business policies into detailed business rules, and managing those rules on an ongoing basis. (Policy managers, either on their own or in conjunction with business analysts, also test rules for correctness and their overall business impact.)
- **System Administrator:** manages and monitors applications in production to ensure applications achieve performance and availability goals.

Separation of Application and Rule Management Life Cycles

Who is responsible for managing changes to rules, and how frequently can changes occur? The answer depends on where you work within the organization. Business policy and decision automation within the enterprise is generally driven by two competing sets of imperatives, which can be broadly characterized as the *business rule life cycle* and the *application development cycle*. This is illustrated in Figure 1 below. In the top half of the diagram, application releases are driven by major requirements and functionality changes. These releases are driven by architects, developers and business analysts following a traditional application development cycle of requirement specification, analysis, design, development, testing and deployment.

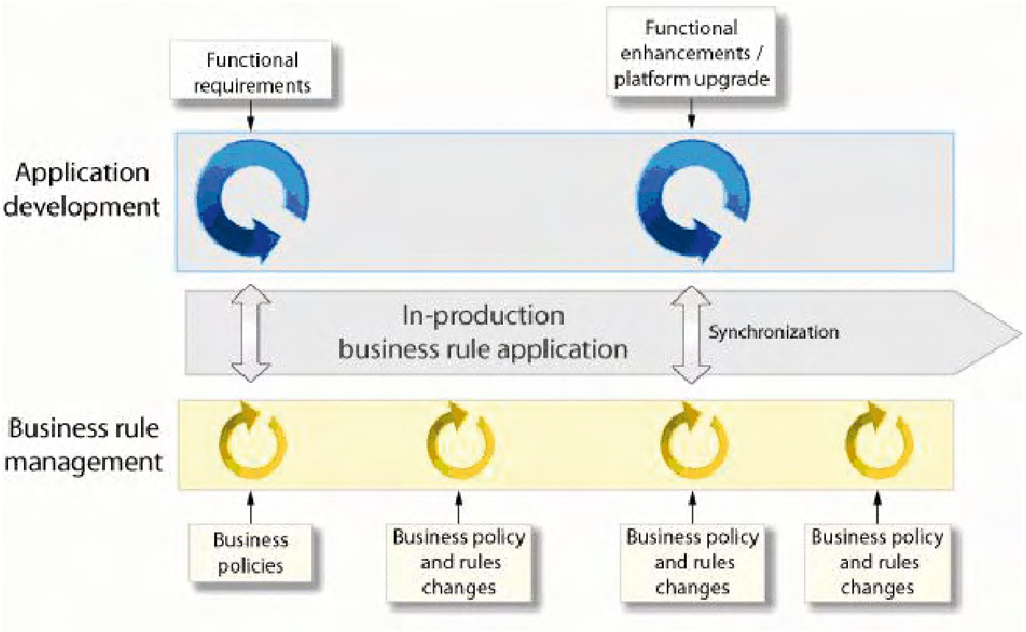

Functional requirements
Functional enhancements / platform upgrade
Application development
In-production business rule application
Synchronization
Business rule management
Business policies
Business policy and rules changes
Business policy and rules changes
Business policy and rules changes

Figure 1: Application Development and Business Rule Life Cycles

However, in most situations, business rules change according to different imperatives. They are driven by business policy changes that represent variants or extensions of the established functional base of an application's current release. For example, an application may require new or updated rules to establish specialized discounting for a new customer or territory, ensure adherence to a contractual obligation, or launch a new promotional program. These types of changes are shown in the lower half of the diagram, and their implementation assumes a stable application on which the policy is elaborated, requiring a shorter, more focused cycle of authoring and testing by the policy manager and more rapid deployment to production. Depending on the needs of the organization and the testing procedures for changes, the business rule life cycle can take as long as a few months (when rule changes are aggregated and deployed at set intervals) or as little as a couple of hours (for a single, high-priority rule change) to complete. In both cases, rule changes can occur asynchronously to the application development cycle.

In some cases, the authoring and maintenance of business rules require an extension of the rule vocabulary and the underlying object model upon which the rules are structured. Depending on the extent and scope of this type of change, additions may fall into either the application development or rule management category.

The result of all this is that business rules "live" simultaneously in both the application development world and the business rule life cycle, and that policy managers and application architects, analysts and developers need specialized tools that recognize and accommodate this reality.

Introducing the WebSphere ILOG BRMS

Now that the basic concepts behind a BRMS have been established, the remainder of this paper will explain how IBM enables organizations to build agile, flexible rule-based applications through its WebSphere ILOG BRMS product family, which is comprised of two principal BRMS offerings: IBM WebSphere ILOG JRules BRMS and IBM WebSphere ILOG Rules for .NET BRMS. As can be seen in the following diagram, each BRMS has platform-specific products as well as a shared set of products and capabilities that can be leveraged by both BRMS offerings.

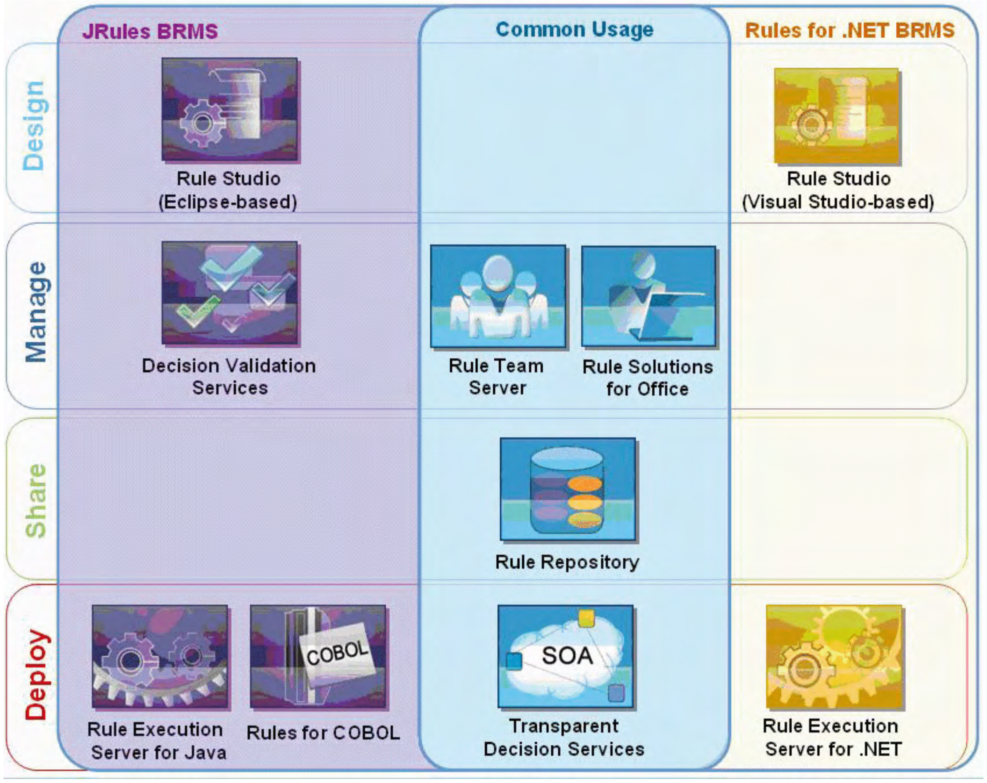

JRules BRMS
Common Usage
Rules for .NET BRMS
Design
Manage
Share
Deploy
Rule Studio (Eclipse-based)
Rule Studio (Visual Studio-based)
Decision Validation Services
Rule Team Server
Rule Solutions for Office
Rule Repository
Rule Execution Server for Java
COBOL
Rules for COBOL
SOA
Transparent Decision Services
Rule Execution Server for .NET

Figure 2: IBM WebSphere ILOG BRMS Offerings

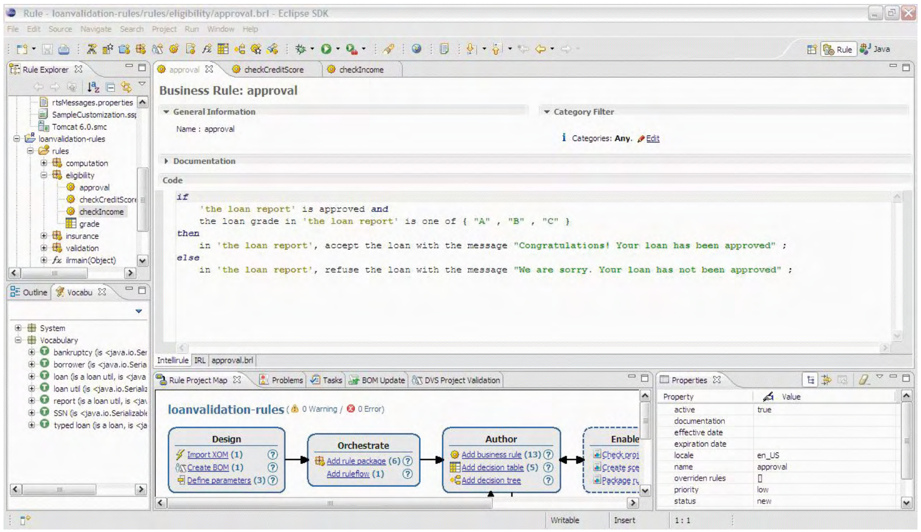
The combination of functionality in each BRMS provides rich rule application development and maintenance capabilities integrated into standard software development tools for technical users, and specialized rule authoring and management tools for business policy managers working in the rule management life cycle. In addition, system administrators have tooling designed for their specific needs in managing and monitoring deployed production rule applications.

BRMS Development Environments

Both JRules and Rules for .NET provide a rule application development environment called Rule Studio. The JRules technical environment is an Eclipse-based integrated development environment (IDE), while Rules for .NET's development environment is integrated into Microsoft Visual Studio. Both IDEs permit IT professionals to apply familiar skills and corporate best practices to construct state-of-the-art business rule applications. Each version of Rule Studio provides a "One-Stop" rule development environment: all the artifacts and operations needed to create and maintain a rule application are included. From within Rule Studio, a developer can:

- Create a logical business object model (BOM) for the application, and map it to a customized, domain-specific rule vocabulary
- Associate the BOM to an execution model of Java or C#/VB.NET classes and XML schemas
- Create a metadata model for rules (application-specific data fields beyond standard metadata; for example, custom rule status properties)
- Create business rules in a natural language syntax, which can be expressed in one or a more localized versions (for example, English or Spanish)
- Create rules in the form of decision tables and decision trees
- Create technical rules in a platform-specific syntax
- Specify packaging of rules into executable rule sets (corresponding broadly to a single policy-driven decision within the application.)
- Separate rules in a rule set into tasks, and specify a rule flow to orchestrate the execution of these tasks
- Create default applications that invoke the rule sets for test purposes

- Deploy rule sets to their respective execution server environments for test or production



Business Rule: approval

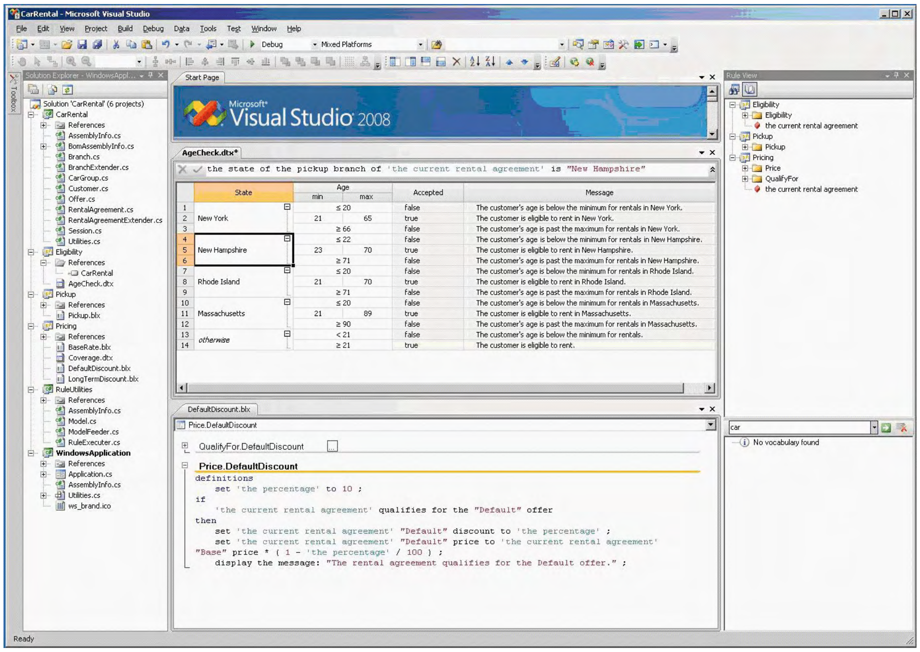
Name : approval

Categories: Any.

```
if
    'the loan report' is approved and
    the loan grade in 'the loan report' is one of { "A" , "B" , "C" }
then
    in 'the loan report', accept the loan with the message "Congratulations! Your loan has been approved" ;
else
    in 'the loan report', refuse the loan with the message "We are sorry. Your loan has not been approved" ;
```

Property	Value
active	true
documentation	
effective date	
expiration date	
locale	en_US
name	approval
overriden rules	
priority	low
status	new

Figure 3: Rule Studio (JRules, Eclipse-based)



AgeCheck.dtx*

```
the state of the pickup branch of 'the current rental agreement' is "New Hampshire"
```

	State	Age		Accepted	Message
		min	max		
1	New York	≤ 20		false	The customer's age is below the minimum for rentals in New York.
2		21	65	true	The customer is eligible to rent in New York.
3		≥ 66		false	The customer's age is past the maximum for rentals in New York.
4	New Hampshire	≤ 22		false	The customer's age is below the minimum for rentals in New Hampshire.
5		23	70	true	The customer is eligible to rent in New Hampshire.
6		≥ 71		false	The customer's age is past the maximum for rentals in New Hampshire.
7	Rhode Island	≤ 20		false	The customer's age is below the minimum for rentals in Rhode Island.
8		21	70	true	The customer is eligible to rent in Rhode Island.
9		≥ 71		false	The customer's age is past the maximum for rentals in Rhode Island.
10	Massachusetts	≤ 20		false	The customer's age is below the minimum for rentals in Massachusetts.
11		21	89	true	The customer is eligible to rent in Massachusetts.
12		≥ 90		false	The customer's age is past the maximum for rentals in Massachusetts.
13	*otherwise*	< 21		false	The customer's age is below the minimum for rentals.
14		≥ 21		true	The customer is eligible to rent.

DefaultDiscount.blx

Price.DefaultDiscount

QualifyFor.DefaultDiscount

Price.DefaultDiscount

```
definitions
    set 'the percentage' to 10 ;
if
    'the current rental agreement' qualifies for the "Default" offer
then
    set 'the current rental agreement' "Default" discount to 'the percentage' ;
    set 'the current rental agreement' "Default" price to 'the current rental agreement'
"Base" price * ( 1 - 'the percentage' / 100 ) ;
    display the message: "The rental agreement qualifies for the Default offer." ;
```

CarRental - Microsoft Visual Studio
Solution Explorer
Solution 'CarRental' (6 projects)
Start Page
Microsoft Visual Studio 2008
Rule View
Eligibility
the current rental agreement
Pickup
Pricing
Price
QualifyFor
car
No vocabulary found
Ready

Figure 4: Rule Studio (Rules for .NET, Visual Studio-based)

Within both BRMS offerings, rules are written in reference to a Business Object Model (BOM). This is an abstract, object-oriented representation of the information model for the application or enterprise, along with natural language-like verbalizations of the classes and members. For projects that need to integrate rules into an existing application with Java/.NET classes or XML schemas that already define its data structures, Rule Studio can import execution models, along with their classes and schemas, and create a corresponding BOM and verbalizations in a few simple steps.

Rule Studio does not insist that the execution model and BOM be identical images of each other. The "BOM to execution" modeling layer (B2X) permits divergence between the abstract business object model and the concrete execution model. Business rules can be written in the natural language syntax verbalization of a BOM that represents the closest policy analogy to the way business users think about their problem space and still be executable on an execution model that meets the needs of the application architecture, even when the BOM and execution model evolve somewhat differently.

As testing is a key part of application development, Rule Studio provides the ability to integrate with standard, platform-specific testing tools such as JUnit and NUnit. When a test fails and the developer needs to "dig into the application" to find out what went wrong, Rule Studio for Java™ provides integrated co-debugging of rules and Java code that lets the developer launch the application or a remotely running instance of the application in debug mode, and use the standard Eclipse debugging facilities to set breakpoints in Java code. The developer can examine Java's memory while simultaneously setting breakpoints in rules, and examine the rule execution agenda and working memory (both Java debugging information and JRules rule debugging information are displayed in the same Eclipse debug perspective). For Rule Studio for .NET, a trace utility is available as an add-in function for debugging.

Both versions of Rule Studio provide integration with source code control systems that support their respective IDE platforms, allowing rule-based applications to be securely stored, shared, branched and versioned.

Finally, both versions of Rule Studio can publish rule projects to IBM WebSphere ILOG Rule Team Server (the Web-based, business-user rule management environment) or update local copies of a rule project from Rule Team Server. Developers can also access prior versions of rule projects stored as baselines within Rule Team Server.

Rule-based application development is the foundation for providing organizations with new levels of flexibility in managing rules independently from the business systems that they support. Rule Studio delivers comprehensive rule application development through a full-featured set of tools integrated within developers' preferred IDEs.

BRMS Rule Management Environments

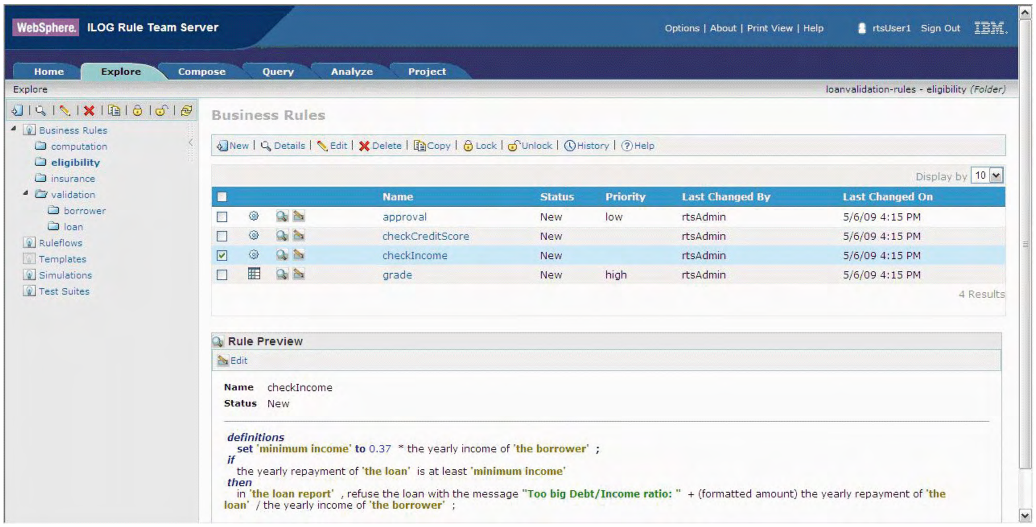
The rule authoring and management environment for business analysts and policy managers is called IBM WebSphere ILOG Rule Team Server (RTS), a thin-client Web-based environment with a scalable, high-performance enterprise rule repository. RTS enables business policy managers to take charge of authoring, managing and validating dynamic business policies.

The repository provides the BRMS with a central "source of truth", addressing the specific needs of rule-based business policy management:

- It is designed to store multiple independent or dependent rule projects and their histories, scaling to large numbers of concurrent users working on the same or different projects and managing hundreds of thousands of individual rule artifacts.
- Every project is a complete, self-contained entity containing the project BOM, vocabulary and all the rule artifacts. As artifacts evolve with an application, the repository maintains all prior versions of each artifact in an accessible and browsable format, providing a complete audit trail of policy implementations. The repository also maintains "baselines" of rule project states, allowing any previous project state for which a baseline has been created to be recalled for examination. The current project state can be "rolled back" to any previous baseline.
- The repository supports automatic rule-level locking, as well as user-managed persistent locks.
- The repository is fully relational and supports all major database vendors. It is accessed under transactional protection, giving the repository the full benefit of the protection offered by enterprise relational databases.
- Both Java and .NET rule projects can be stored and managed in the repository.

Policy managers author, modify, manage and deploy rules from the repository through the RTS environment. This Web-based application gives policy managers a comprehensive rule editing, management and deployment facility. Key technical features of the environment include:

- The RTS application is a clusterable J2EE application ready for deployment in a self-contained enterprise application archive. Prebuilt deployments for major application server vendors and versions are provided.
- As a J2EE application, RTS participates in role-based J2EE authentication and authorization. Integrating permissions from an enterprise directory service is handled by standard application server-level integration. Each RTS user can be provided different levels of access and control of the projects and rules within the repository.
- RTS defines and enforces fine-grained permissions on rule artifacts using either a permissive or restrictive default model.
- From within RTS, policy managers can author, modify and organize rules, rule templates, decision tables and decision trees. In addition, they can create and browse baselines, and query for, report on and deploy rules. The full history of each rule is accessible to policy managers, and facilities are provided for visually comparing two versions of a rule or decision table. User-definable views allow policy managers to see rules in the organization that best suit the task immediately at hand, while preserving full navigability of the overall project repository.
- RTS has a rich application programming interface (API) for the rule repository, and also provides predefined extension points for customization of RTS functionality. Rebranding of RTS with customer logos is supported.



☐	Name	Status	Priority	Last Changed By	Last Changed On
☐	approval	New	low	rtsAdmin	5/6/09 4:15 PM
☐	checkCreditScore	New		rtsAdmin	5/6/09 4:15 PM
☑	checkIncome	New		rtsAdmin	5/6/09 4:15 PM
☐	grade	New	high	rtsAdmin	5/6/09 4:15 PM

4 Results

Name checkIncome
Status New

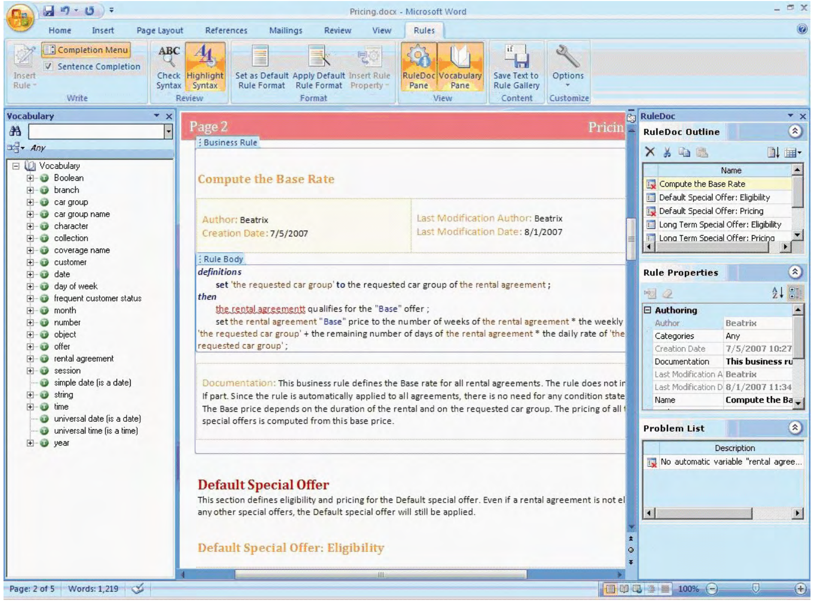
```
definitions
  set 'minimum income' to 0.37  * the yearly income of 'the borrower'  ;
if
  the yearly repayment of 'the loan'  is at least 'minimum income'
then
  in 'the loan report'  , refuse the loan with the message "Too big Debt/Income ratio: "  + (formatted amount) the yearly repayment of 'the
loan'  / the yearly income of 'the borrower'  ;
```

Figure 5: Rule Team Server

The synchronization capability between both the JRules and Rules for .NET versions of Rule Studio and RTS allows developers and policy managers to maintain separate views of business rules in accordance with the needs of their respective responsibilities, while ensuring that both sides have access to the latest version of a rule project. For example, once the initial version of a rule application goes into production, policy managers can change and add to the policies represented as rules in the application as new policy initiatives dictate, while developers work independently on a new version of the entire application that may expand its overall scope or update its core functionality. Developers can synchronize with the repository as needed until the new version is ready for rollout, at which time the developers publish an entirely new application and set of rules that incorporates all ongoing changes from the production rule set.

IBM WebSphere ILOG Rule Solutions for Office is an alternate rule authoring and editing option for non-technical participants in the rule management life cycle. Rule Solutions for Office allows rules to be maintained in the familiar Microsoft Office Word and Excel desktop applications, while at the same time enforcing the underlying application requirements of the systems that will use these rules in the production environment.

By loading a lightweight plug-in for Microsoft Office (2007), business users can work with file-based "ruledocs" in a guided manner, leveraging rule editors with auto-completion assistance from the IBM ILOG Intellirule technology. Since the ruledoc contains the business object model for the associated project of the rules, users have access to customized business vocabulary, and rules are automatically validated for correct logic and syntax.



Compute the Base Rate

Author: Beatrix
Creation Date: 7/5/2007

Last Modification Author: Beatrix
Last Modification Date: 8/1/2007

```
definitions
    set 'the requested car group' to the requested car group of the rental agreement ;
then
    the rental agreementt qualifies for the "Base" offer ;
    set the rental agreement "Base" price to the number of weeks of the rental agreement * the weekly
'the requested car group' + the remaining number of days of the rental agreement * the daily rate of 'the
requested car group' ;
```

Documentation: This business rule defines the Base rate for all rental agreements. The rule does not ir
If part. Since the rule is automatically applied to all agreements, there is no need for any condition state
The Base price depends on the duration of the rental and on the requested car group. The pricing of all t
special offers is computed from this base price.

Default Special Offer

This section defines eligibility and pricing for the Default special offer. Even if a rental agreement is not el
any other special offers, the Default special offer will still be applied.

Default Special Offer: Eligibility

Compute the Base Rate
Default Special Offer: Eligibility
Default Special Offer: Pricing
Long Term Special Offer: Eligibility
Long Term Special Offer: Pricing

Authoring	
Author	Beatrix
Categories	Any
Creation Date	7/5/2007 10:27
Documentation	This business ru
Last Modification A	Beatrix
Last Modification D	8/1/2007 11:34
Name	Compute the Ba

No automatic variable "rental agree...

Figure 6: Rule Solutions for Office (Microsoft Word-based Ruledoc)

Rule Solutions for Office extends and facilitates rule management throughout the enterprise by providing the following benefits:

- Specialized toolbars are incorporated within Microsoft Office Word and Excel, adding viewing panes and lists that facilitate rule authoring and validation.
- Structured rules and unstructured associated information can be combined together for documentation purposes.
- Ruledocs can be downloaded from the BRMS and edited offline (ruledocs are created using RTS or Rule Studio for .NET, which allows specific rules or entire rulesets in a project to be extracted from the repository in a ruledoc format).
- Ruledocs contain full rule project metadata, allowing them to be synchronized with the BRMS repository—completed rule changes can be easily uploaded to the repository through RTS for testing, simulation, versioning and deployment.

From the perspective of the architect or developer of a rule application, providing support for rule authoring and management to non-developers is a key function of a BRMS, enabling technical and non-technical personnel to collaborate and close the loop between the application development cycle and the business rule life cycle.

Decision Validation Services: Integrated Rule Testing and Simulation for Developers and Business Policy Managers

Of course, there is more to both application development and rule management than just authoring and synchronization. Rules must be tested, both within the context of application development and as business policy implementations. IBM WebSphere ILOG Decision Validation Services (DVS) is an optional product offering for the JRules BRMS, providing integrated test, simulation and audit capabilities that apply realistic scenarios to determine the effects of business rule changes on production systems:

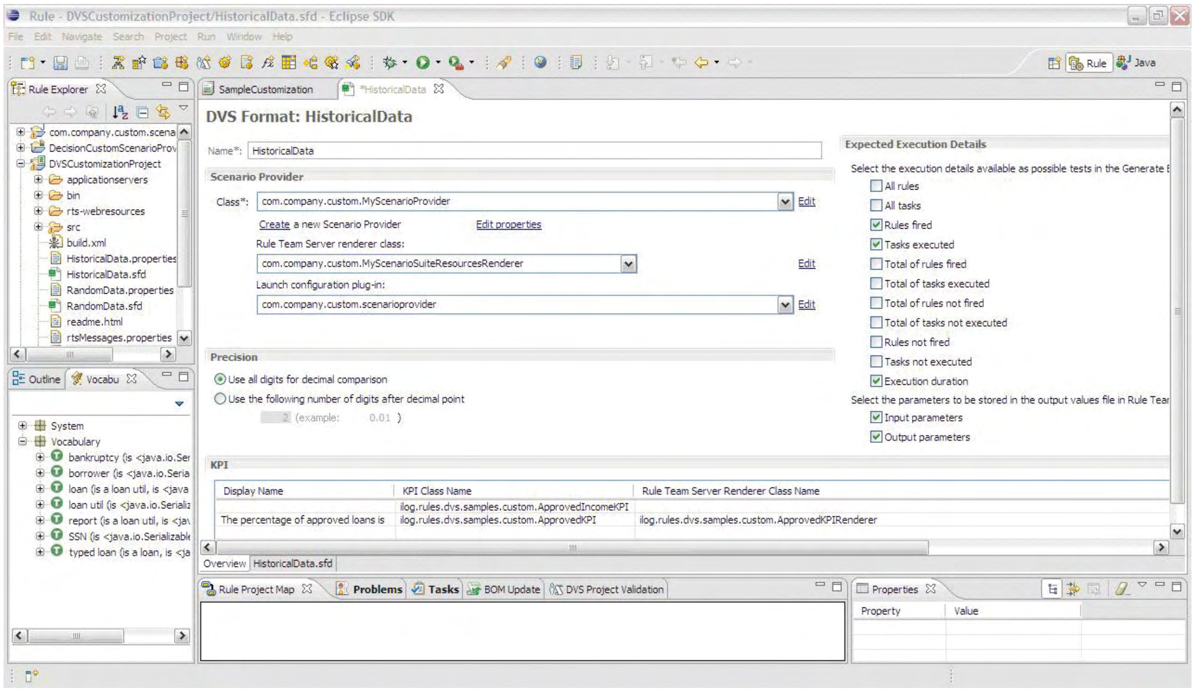
- Unit and regression testing to ensure rules execute as expected
- Functional testing to execute rule sets with data and capture the results
- Simulation to measure or verify the business results of rule sets against either historical or test data
- Rule execution auditing to review decision outputs

Because business rules are operational artifacts, testing and validation of rules are important for both the application development cycle and the business rule life cycle. DVS provides rule testing functionality for both developers working in the application development context, and policy managers authoring and validating rules as part of the business rule life cycle.

In addition, DVS provides facilities for simulations in which the results from a modified "candidate" rule set executed against a suite of test cases is compared to a baseline rule set running against the same data; simulations enable LOB policy managers to understand how rule changes affect business systems in terms of the business objectives of the organization by allowing for specification of key performance indicators (KPIs). As an example, a DVS simulation could use the previous 12 months of data to determine if a rule change increases or decreases the number of automated approvals for a government social service, along with a comparison of the results to the target accept/reject ratio.

DVS is a modular, extensible framework for all aspects of creating rule test and simulation scenarios:

- It provides a set of configuration and test execution components integrated within the Rule Studio for Java IDE. Developers create scenarios to be applied to validate the execution of rule sets, assembling these into suites of scenarios, as well as configuring the Scenario Service Provider (SSP), a server component responsible for executing test suites. Developers can also use DVS to invoke tests from within Rule Studio using the JUnit testing framework.
- The data that defines a scenario can be provided to test suites in the form of a Microsoft Excel spreadsheet, allowing developers to create test suite templates that business users can populate with input and output data specific to their rules and business functions. In addition, extension points are provided to allow developers to map scenarios to existing relational database structures if needed.
- Other DVS extensions include creation of KPIs, and specification of custom reports for filtering or detailing the results of a test or simulation.



DVS Format: HistoricalData

Name*: HistoricalData

Scenario Provider

Class*: com.company.custom.MyScenarioProvider Edit

Create a new Scenario Provider Edit properties

Rule Team Server renderer class:
com.company.custom.MyScenarioSuiteResourcesRenderer Edit

Launch configuration plug-in:
com.company.custom.scenarioprovider Edit

Precision

● Use all digits for decimal comparison
○ Use the following number of digits after decimal point
2 (example: 0.01)

Expected Execution Details

Select the execution details available as possible tests in the Generate E

□ All rules
□ All tasks
☑ Rules fired
☑ Tasks executed
□ Total of rules fired
□ Total of tasks executed
□ Total of rules not fired
□ Total of tasks not executed
□ Rules not fired
□ Tasks not executed
☑ Execution duration

Select the parameters to be stored in the output values file in Rule Tear

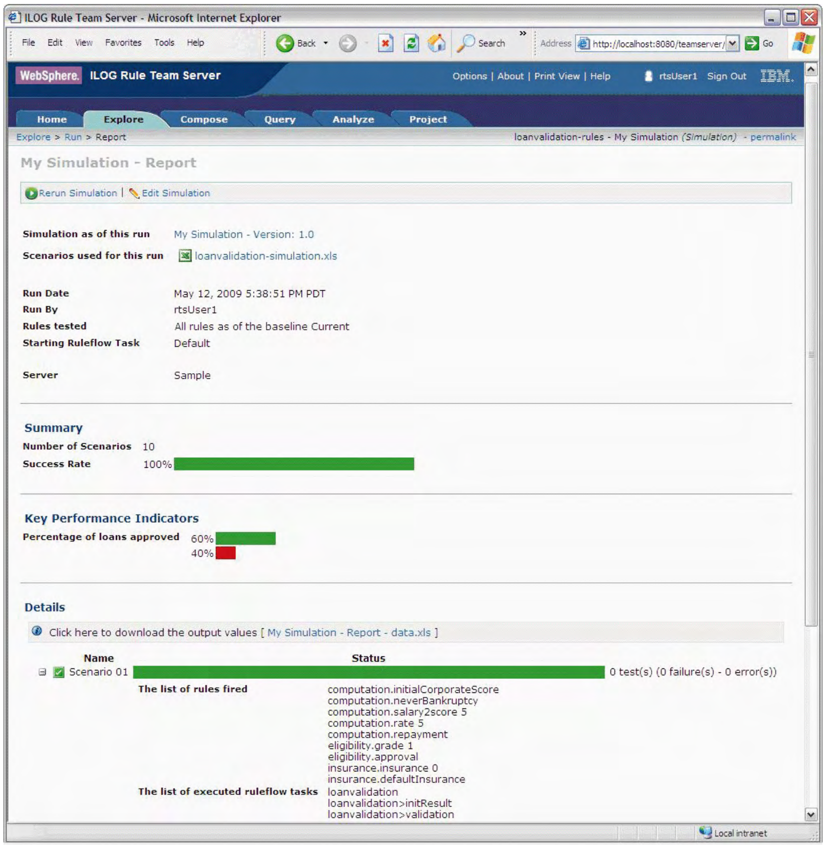
☑ Input parameters
☑ Output parameters

KPI

Display Name	KPI Class Name	Rule Team Server Renderer Class Name
	ilog.rules.dvs.samples.custom.ApprovedIncomeKPI	
The percentage of approved loans is	ilog.rules.dvs.samples.custom.ApprovedKPI	ilog.rules.dvs.samples.custom.ApprovedKPIRenderer

Figure 7: Decision Validation Services (Rule Studio Test Scenario Configuration)

For business users, DVS test and simulation scenarios can be run from within Rule Team Server, with result outputs provided through the RTS interface and the option of sending the results to Micosoft Excel. IT has the ability to incorporate more extensive tests as part of the rule life cycle, but the first level of testing and validation can become the responsibility of the LOB functions, empowering them to be more involved in their business systems and helping to decrease the overall workload on IT.


ILOG Rule Team Server - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back
Search
Address http://localhost:8080/teamserver/
Go
WebSphere. ILOG Rule Team Server
Options | About | Print View | Help
rtsUser1 Sign Out IBM.
Home
Explore
Compose
Query
Analyze
Project
Explore > Run > Report
loanvalidation-rules - My Simulation (Simulation) - permalink
Rerun Simulation | Edit Simulation
Local intranet

My Simulation - Report

Simulation as of this run	My Simulation - Version: 1.0
Scenarios used for this run	loanvalidation-simulation.xls
Run Date	May 12, 2009 5:38:51 PM PDT
Run By	rtsUser1
Rules tested	All rules as of the baseline Current
Starting Ruleflow Task	Default
Server	Sample

Summary

Number of Scenarios	10
Success Rate	100% 🟩

Key Performance Indicators

Percentage of loans approved	60% 🟩 40% 🟥

Details

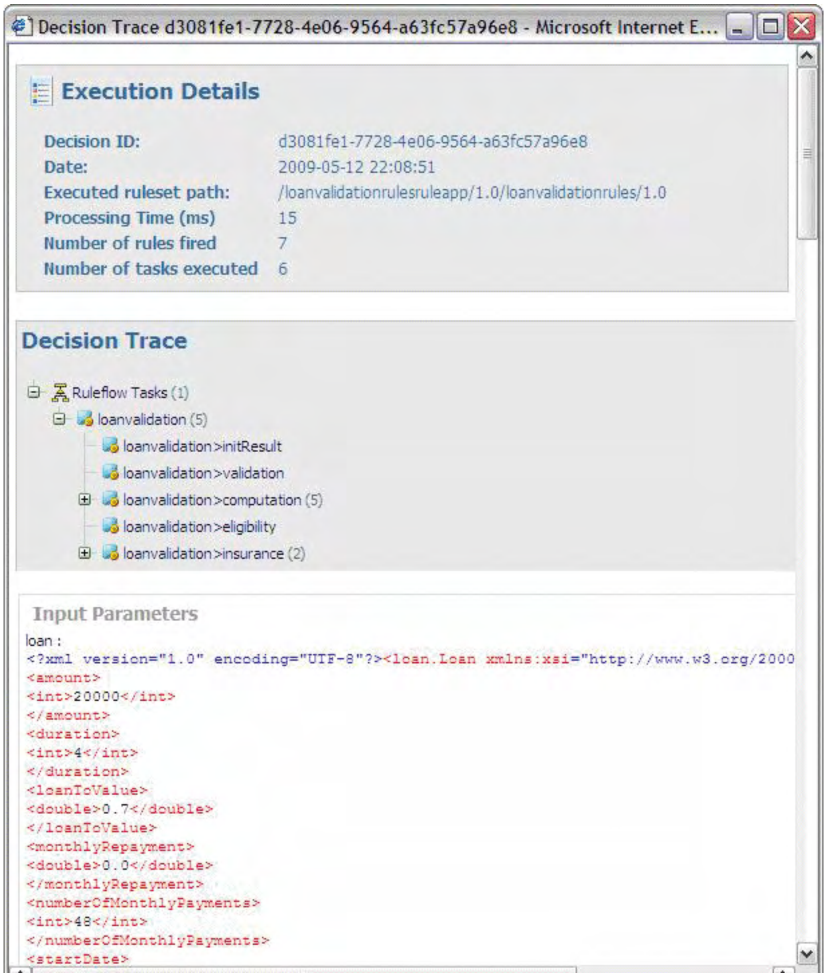
Click here to download the output values [My Simulation - Report - data.xls]

Name	Status	
⊟ ☑ Scenario 01	🟩	0 test(s) (0 failure(s) - 0 error(s))

The list of rules fired	computation.initialCorporateScore computation.neverBankruptcy computation.salary2score 5 computation.rate 5 computation.repayment eligibility.grade 1 eligibility.approval insurance.insurance 0 insurance.defaultInsurance
The list of executed ruleflow tasks	loanvalidation loanvalidation>initResult loanvalidation>validation

Figure 8: Decision Validation Services (Rule Team Server Simulation Results)

Another capability of DVS is the Decision Warehouse, a rule execution audit and reporting feature that is available through the JRules Rule Execution Server administration console. The Decision Warehouse allows rule execution data to be queried against a number of different parameters, and for the results to be drilled down to the individual rules fired. The results also provide Rule Team Server links to repository, detailing the exact version of each rule fired at the time of execution. By providing this information, it is possible to gain valuable insight into how operational decisions occur in the production environment and to selectively perform audits as required by internal or external compliance mandates.



Decision Trace d3081fe1-7728-4e06-9564-a63fc57a96e8 - Microsoft Internet E...

Execution Details

Decision ID:	d3081fe1-7728-4e06-9564-a63fc57a96e8
Date:	2009-05-12 22:08:51
Executed ruleset path:	/loanvalidationrulesruleapp/1.0/loanvalidationrules/1.0
Processing Time (ms)	15
Number of rules fired	7
Number of tasks executed	6

Decision Trace

```
⊟ Ruleflow Tasks (1)
    ⊟ loanvalidation (5)
          loanvalidation>initResult
          loanvalidation>validation
        ⊞ loanvalidation>computation (5)
          loanvalidation>eligibility
        ⊞ loanvalidation>insurance (2)
```

Input Parameters

loan :

```
<?xml version="1.0" encoding="UTF-8"?><loan.Loan xmlns:xsi="http://www.w3.org/2000
<amount>
<int>20000</int>
</amount>
<duration>
<int>4</int>
</duration>
<loanToValue>
<double>0.7</double>
</loanToValue>
<monthlyRepayment>
<double>0.0</double>
</monthlyRepayment>
<numberOfMonthlyPayments>
<int>48</int>
</numberOfMonthlyPayments>
<startDate>
```

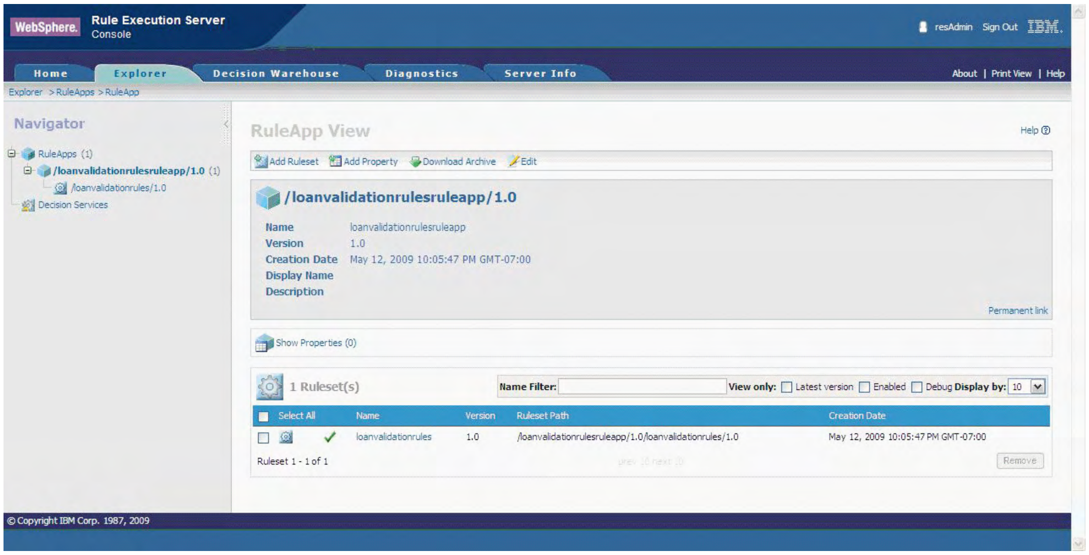
Figure 9: Decision Validation Services (Decision Warehouse)

By providing a way for both policy managers and developers to test and validate the execution of business rules, DVS enhances productivity in the creation, management and governance of business rules. DVS enables IT to define appropriate tests and simulations, and then provide the functionality to non-technical users so that they can verify the correctness of any rule changes, as well as understand the overall business impact.

Rule Execution Server: Managed, Monitored, High-Performance Rule Execution for the Enterprise

Architects and developers of business rule applications must out of necessity be concerned about the execution of business rules. Business policies are automated by integrating all the execution elements into a rule application ~~in~~ order to actually implement policy automation, rules must be executed as part of the application's transactional environment. The WebSphere ILOG BRMS offerings include Rule Execution Server (RES), a managed, monitorable execution environment for rules that can be incorporated into an application by being deployed to a J2EE or .NET application server, or embedded directly in an application. Both the JRules and Rules for .NET versions of RES incorporate platform-specific versions of IBM ILOG's high-performance rule engine.

Once an application is deployed, system administrators require insight into and control over the execution services to make sure service level agreements are met and new rule sets are deployed correctly and in a timely manner. The RES console for JRules and Rules for .NET provides a central, Web-based environment for business rule execution management of server-based rule applications, SOA-based decision services and embedded rule applications.



/loanvalidationrulesruleapp/1.0

Name loanvalidationrulesruleapp
Version 1.0
Creation Date May 12, 2009 10:05:47 PM GMT-07:00
Display Name
Description

1 Ruleset(s)

Name Filter: View only: □ Latest version □ Enabled □ Debug Display by: 10

□ Select All	Name	Version	Ruleset Path	Creation Date
□ ✓	loanvalidationrules	1.0	/loanvalidationrulesruleapp/1.0/loanvalidationrules/1.0	May 12, 2009 10:05:47 PM GMT-07:00

Ruleset 1 - 1 of 1

Figure 10: Rule Execution Server (Administration Console)

The JRules and Rules for .NET versions of RES are essential components of a comprehensive BRMS:

- RES provides pooled, managed access to rule-based decision services. Rule applications can take advantage of clustered deployments for scalability and robustness.
- Different kinds of rule-based decisions require different approaches to rule execution. For decisions requiring inference-based rule execution, such as clinical decision recommendations or data validity computations, the JRules and Rules for .NET rule engines offer RetePlus, a third-generation implementation of the Rete algorithm, delivering industry-leading performance on up to hundreds of objects and thousands of rules. For simpler decisions, such as eligibility determinations, retail promotions and compliance monitoring, the engine offers sequential execution. For sequential applications with significant condition sharing between rules, the IBM ILOG optimized FastPath technology delivers execution throughput up to 15 times that of standard sequential execution.
- Both JRules and Rules for .NET allow rule sets to be implemented as Transparent Decision Services, fully defined Web services (WSDL) that are generated through the RES console without requiring any coding. Transparent Decision Services allow rule sets managed by either BRMS to be deployed into cross-platform service-oriented architectures.
- A RES installation can manage multiple versions of each of many decision services (rule sets). Management options to deploy, activate and deactivate individual rule sets or versions are available through the RES console or can be scripted using ANT or MSBuild tasks.
- All RES management functions are available through Java Management Extensions (JMX) or Windows Management Instrumentation (WMI), so they can be automated through standard enterprise management tools or custom programs. Execution statistics by rule set and server are also accessible through the RES Web console and can be extended to external tools. By combining management and monitoring functions, system administrators can construct automated service level agreement monitoring and response triggers.

The JRules BRMS provides an additional option for embedding rules directly into COBOL applications running in mainframe environments. While the JRules RES can run directly on z/OS and z/Linux production environments, IBM WebSphere Rules for COBOL provides a COBOL copybook code generation capability for those applications that need to run the rules natively. This provides flexible options for utilizing a BRMS with mainframe-based business systems.

Whether an application requires throughput of 500 or millions of transactions per day, system administrators and operations managers require tools for monitoring and managing the deployment of critical decision services. The WebSphere ILOG BRMS provides comprehensive support to make managed rules available and integrated with the variety of systems and architectures that need to interact with production rule applications.

Conclusion

There are as many reasons to build a business rule software application as there are unique decisions to be automated in business and government. But all such applications share crucial common features:

- They rely on a number of stakeholders, from both IT and LOB, to fully define, build and maintain and deploy rule-based applications.
- They require a comprehensive set of management capabilities designed to meet the specific needs of each stakeholder.
- They require high-performance, scalable, managed execution of the business rules deployed into production.

The implementation of a BRMS brings organizations wide-ranging benefits in terms of the accessibility and oversight of automated decision logic, along with the ability to accelerate the implementation of decision changes. But there are differences between business rule offerings in the market and the functionality they deliver to ensure easy, safe and reliable rule management. The WebSphere ILOG BRMS offerings deliver the most comprehensive set of rule management capabilities. Designed to fit a wide range of enterprise architectures, they empower both LOB and IT teams, and facilitate communication and collaboration between all rule management stakeholders.

IBM Corporation
Software Group
Route 100
Somers, NY 10589
U.S.A.

Produced in the United States of America
November 2009
All Rights Reserved

IBM assumes no responsibility regarding the accuracy of the information provided herein and use of such information is at the recipient’s own risk. Information herein may be changed or updated without notice. IBM may also make improvements and/or changes in the products and/or the programs described herein at any time without notice.

References in this publication to IBM products and services do not imply that IBM intends to make them available in all countries in which IBM operates.