Table of Contents

1. Introduction ........................................................................................................................................... 2

2. Introducing Synchronized Full Life-Cycle Rule Management ................................................................. 2

3. Rule Studio: Rule Tech Developer Productivity Tool ................................................................................. 4

4. Rule Team Server: Rule Care Empowerment for Business Policy Managers ............................................. 6

5. Rule Scenario Manager: Hands-on Rule Testing and Simulation for Developers and Business Policy Managers ................................................................................................................. 7

6. Rule Execution Server: Managed, Monitored, High-Performance Rule Execution for the Enterprise ................................................................. 8

7. Conclusion ................................................................................................................................................. 9
1. **Introduction**

In a world where business agility – the ability to quickly and efficiently adapt policies to changing markets, regulatory rulings or other challenges – is a hallmark of successful information management, the enterprise business rule management system (BRMS) is an essential competitive tool for businesses and government agencies alike. Mission-critical applications constructed around an web-based BRMS place the ability to author, test and manage the rules that implement policies directly into the hands of the responsible managers. Policy changes can be driven exclusively by business imperatives and evolved 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 implementing application infrastructure such as service-oriented architectures, data management, transaction support and enterprise application integration, rather than spending most of their time and resources on repetitive activities to implement changes to business policies.

To provide these benefits, an enterprise BRMS must satisfy the divergent requirements of the many different stakeholders in an automated policy management system based on business rules. For example:

- **Architect**: The system’s architect is responsible for assuring that the application design meets long-term business needs for functionality, efficiency and performance.
- **Analyst**: The system’s analyst is responsible for modeling the business domain and business rule vocabulary.
- **Developer**: A business rule application developer creates and tests the application.
- **Policy Manager**: The policy manager is a subject expert responsible for translating business policy into detailed business rules, and testing and managing those rules.
- **System Administrator**: The system administrator manages applications in production to achieve application performance and availability goals.

ILOG JRules 6 is a new generation web-based BRMS that meets the needs of each of these stakeholders, with tools and features that match for their particular needs and work practices.

2. **Introducing Synchronized Full Life-Cycle Rule Management**

What is a business rule, and where does it reside? The answer depends on where you work within the policy automation process. Policy 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. In the top half of the figure – Application Development Cycle – application releases are driven by major requirement changes and external product release schedules. These releases are produced by architects, business analysts and developers following a traditional application development cycle of requirement specification, analysis, design, development, testing and deployment.

However, in most situations, business rules change according to different imperatives. They are driven by business policy changes that represent variants or extensions on the established functional base of an application’s current release. For example, an application may require new policies (rules) to establish specialized processing for a new customer or territory, to reflect regulatory changes or reflect a new promotional program. These types of policy changes are shown driving the business rule life cycle in the lower half of Figure 1. Changes implemented here assume a stable application logic on which the policy is elaborated, and require a smaller, more focused cycle of authoring and testing by the policy manager, and timely deployment to production. Depending on the needs of the application, a policy change implemented in this cycle can take as long as a few months or as little as a couple of hours to complete. In any case, the changes are largely asynchronous to the application development cycle.
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 application architects, analysts and developers need specialized tools that recognize and accommodate this reality.

---

**Figure 1: Application Development and Business Rule Life Cycles**

ILOG JRules 6 recognizes this by providing rich rule authoring and management environments integrated into standard software development tools for technical users in the application development cycle, such as architects, analysts and developers, and standalone rule authoring and management tools for business policy managers working in the rule management life cycle.

ILOG JRules’ technical environment is an Eclipse-based IDE for rule application development, called ILOG Rule Studio. ILOG Rule Studio for Java is a central part of ILOG’s Rule Tech, technology solutions that permit IT professionals to apply familiar skills and corporate best practices to construct state-of-the-art business rule applications. Rule Studio is explained in more detail in Section 3.

The ILOG JRules authoring and management environment for policy managers is called Rule Team Server. Rule Team Server is a thin-client Web application (for zero footprint deployment anywhere in the enterprise) with its own high-performance rule repository. Rule Team Server enables ILOG’s RuleCare technology to empower business policy managers to take charge of authoring, managing and validating dynamic business policies. Rule Team Server is explained in more detail in Section 4.

RuleSync connects Rule Studio and Rule Team Server to close the loop between the application development cycle and the business rule life cycle. RuleSync enables developers and policy managers to maintain their respective views of business rules in accordance with the needs of the application development cycle or business rule life cycle, with the ability to synchronize these views as necessary. 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. The developers can synchronize their views of the rules with the evolving rules in production as they need to, 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.
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 isolated decision implementations by policy managers. ILOG JRules 6 provides Rule Scenario Manager as part of both Rule Tech and RuleCare to provide a common environment for testing the function of rules used by an application.

Finally, in order to actually implement policy automation, rules must be executed as part of the application’s transactional environment. ILOG JRules 6 includes Rule Execution Server, a managed, monitorable execution environment for rules that can be incorporated into an application by being deployed to a J2EE application server or embedded directly in a J2SE application. Rule Execution Server incorporates ILOG’s performance-leading rule engine.

The entire suite of ILOG JRules 6 environments and modules are shown in Figure 2. These modules are explained more fully in subsequent sections. Together, they constitute a BRMS that delivers business rule management without compromise:

- Full empowerment for business teams
- Full productivity for tech teams
- Synchronized full life-cycle business rule management
- Unequalled performance in rule execution and rule management.

![Figure 2: The ILOG JRules 6 Suite of Environments and Tools](image)

### 3. Rule Studio: Rule Tech Developer Productivity Tool

A BRMS can be used to manage and execute business rules for one or more software applications. Therefore, it must provide the tools needed by business analysts, IT architects and developers to incorporate rule management and execution into these applications. ILOG JRules 6 sets a new standard for BRMS tools by addressing the real needs of the IT professional.

- **Rule Studio Works Seamlessly in Existing Development Environments**: Modern software projects are typically modeled, developed and tested within an integrated development environment (IDE), and
stored, shared, branched and versioned as files managed by a source code management (SCM) system. Rule Studio is a first-class participant in this approach. It is delivered as plug-ins that can be fully integrated with Eclipse, the leading Java IDE, but it also works as well with IBM Rational Application Developer, Rational System Architect and WebSphere Integration Developer. All rule artifacts are managed by Rule Studio as individual plain-text or XML files compatible with Eclipse’s standard file management, and can be stored and versioned in any SCM system. And because Eclipse plug-ins are available for the most widely used SCM systems, developers can do integrated source management, rule project development and Java development for most projects using the same IDE.

- **Rule Studio Supports Both Bottom-Up and Top-Down Rule Development**: In ILOG JRules 6, 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 classes or XML schemas that already define its data structures, Rule Studio can import these classes or schemas as execution models and create a corresponding BOM and verbalization in a few simple steps. This is referred to as “bottom-up” business rule development.

  Conversely, for projects that start rule development in the modeling stage, Rule Studio supports “top-down” specification of the BOM by analysts, and allows the Java classes or XML schemas representing the execution model to be specified later, when the actual software development starts.

  Rule Studio does not insist that the execution model and BOM be isomorphic images of each other. ILOG JRules’ “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.

- **Rule Studio is a “One-Stop” Rule Development Environment for Analysts and Developers**: All the artifacts and operations needed to create and test a rule application are included in Rule Studio. From within Rule Studio, an analyst or developer can:
  
  o Create a logical BOM for the application and map it to a natural language vocabulary
  o Associate the BOM to an execution model of Java classes and XML schemas
  o Create a metadata model for rules (Application-specific data fields beyond standard metadata (e.g. last modification date) can be specified, and stored and maintained with the rules.)
  o Create business rules in a natural language-like syntax
  o Create technical rules in a Java-like syntax
  o Create rules in the form of decision tables and decision trees
  o Specify packaging of rules into executable rule sets (Each rule set corresponds broadly to a single policy-driven decision, such as underwriting, credit-risk assessment or promotion management, within the application.)
  o Separate rules in a rule set into tasks, and specify a rule flow to orchestrate the execution of these tasks
  o Create default applications that invoke the rule sets for test purposes
  o Deploy rule sets to an execution server for test or production

- **Rule Studio is a “One-Stop” Rule Testing and Debugging Environment for Developers**: Testing and debugging are a key part of application development. When creating a rule application, developers must validate and potentially debug another layer of executable content.
Rule Studio provides JUnit-based integration with ILOG Rule Scenario Manager (RSM), a rule testing framework. It permits direct invocation of RSM tests from within the Rule Studio environment using the standard Java testing tool.

When a test fails, the developer needs to “dig into the application” to find out what went wrong. Rule Studio 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 ILOG JRules rule debugging information are displayed in the same Eclipse debug perspective.

- **Rule Studio and RuleSync provide a Gateway to ILOG JRules’ Policy Manager Tools:** From Rule Studio, developers can publish rule projects to Rule Team Server or update local copies from Rule Team Server. Developers can also use RuleSync to copy rule projects between multiple Rule Team Server repositories, and access prior versions of rule projects stored as baselines within Rule Team Server. Rule Team Server is explained more fully in Section 4.

Rule-based application development brings both great new capabilities and new challenges to the application development process. Rule Studio delivers rule application development without compromise: a full-featured set of tools for realizing the promise of agility through business rule applications and full integration with developers’ preferred development environment, proven methodologies and expert skills.

### 4. Rule Team Server: Rule Care Empowerment for Business Policy Managers

The rules used by a rule application are the province of the business rule life cycle and ILOG Rule Care technology. Rules in a rule application are an operational representation of business policies, which originate from policy initiatives, not with IT projects. The primary purpose for Rule Care is to enable the people most responsible for these initiatives – policy managers – to own and manage the business rule life cycle.

To this end, **ILOG JRules 6** provides a rule management application specifically for policy managers: Rule Team Server. It combines a deep, scalable, high-performance rule repository with a thin-client rule management application designed specifically for the needs of policy managers engaged in rule authoring, management and maintenance.

The Rule Team Server repository addresses the specific needs of rule-based policy management with:

- **Complete Multi-project Rule Management:** Every project stored in the Rule Team Server as a complete, self-contained entity, subject only to its explicit imports from other projects. Projects in Rule Team Server contain the project BOM, vocabulary and all the rule artifacts, and are stored without reference to a specific execution model. A working Rule Studio image of any project can be recreated from any version of the project in the repository at any time.

The Rule Team Server repository is designed to be an enterprise repository, storing in a single place multiple independent or dependent rule projects and their histories.

- **Multiuser Access:** The repository supports automatic rule-level locking, as well as user-managed persistent locks.

- **Scalability:** The Rule Team Server repository scales to dozens of users working on the same or different projects, and hundreds of thousands of individual rule artifacts.

- **Full Version and History of Rule Artifacts:** The Rule Team Server repository serves as a fully versioned content management system for BOM, vocabulary and rule artifacts. As artifacts evolve with
Policy managers author, modify, manage and deploy rules from the Rule Team Server repository using the Rule Team Server application. This is a thin-client Web-based application that gives policy managers a complete, wizard-based rule editing, management and deployment facility. Key technical features of the application include:

- **Fully J2EE Compliant**: The Rule Team Server 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.

- **Integrated Permission Management**: As a J2EE application, Rule Team Server participates in role-based J2EE authentication and authorization. Integrating permissions from an enterprise directory service is handled by standard application server-level integration.

Rule Team Server defines and enforces fine-grained permissions on rule artifacts using either a permissive or restrictive default model.

- **Complete Policy Manager Functionality**: From within Rule Team Server, 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 suits the task immediately at hand, while preserving full navigability of the overall project repository.

- **Extensibility and Branding**: ILOG JRules 6 has a rich API on the rule repository, and provides predefined extension points for modest extensions and customizations of Rule Team Server functionality. Rebranding of Rule Team Server with customer logos is supported.

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. Rule Team Server is a rich, robust platform for doing exactly that, and a key factor to ILOG JRules 6’s ability to provide business rule management without compromise.

5. **Rule Scenario Manager: Hands-on Rule Testing and Simulation for Developers and Business Policy Managers**

Because business rules are operational artifacts, testing and validation of rules are a necessary part of both the application development cycle and the business rule life cycle. ILOG JRules 6 recognizes the need for testing rule-based decisions by providing a testing framework and console called Rule Scenario Manager. Rule Scenario Manager provides rule testing functionality for both developers working in the application development context, and policy managers who author and validate rules as part of the business rule life cycle. In addition, Rule Scenario Manager 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 using customer-specified key performance indicators (KPIs).
Rule Scenario manager is a modular, extensible framework for all aspects of rule testing and simulation:

- **Rule Scenario Manager has a Flexible “Multiple Use” Architecture**: Rule Scenario Manager comprises a server component responsible for executing test suites; a Web-based console that policy managers and developers can use for creating scenarios (test cases) and tests to be applied to validate the execution of a scenario, assembling these into suites of scenarios, submitting test suites for execution, and viewing the results of test execution; and integration components that developers can use to invoke tests from within Rule Studio using the JUnit testing framework.

- **Rule Scenario Manager Supports Common Data Formats for Scenarios**: The data that defines a scenario can be provided to Rule Scenario Manager in the form of Microsoft Excel spreadsheet or XML. In addition, extension points are provided to allow developers to map scenarios to existing relational database structures for persistence.

- **Rule Scenario Manager is Inherently Extensible**: Extension points are provided throughout Rule Scenario Manager to allow customization and extension of the framework. Common extensions include: specification of custom data mappings to provide scenarios from existing data sources; creation of custom KPIs for comparing results of a simulation; and specification of custom reports filtering or detailing the results of a test or simulation.

By providing a way for both policy managers and developers to test and validate the execution of business rules, Rule Scenario Manager enhances productivity in the creation and management of business rules by policy manager, and the validation of business rule applications by developers, making it a key component in enabling business rules without compromise.

6. **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. 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. ILOG JRules 6's Rule Execution Server wraps the high-performance ILOG rule engine into a scalable, manageable and monitorable service that provides business rule execution for all server-based applications, service-oriented architectures and embedded rule applications:

- **Rule Execution Server Builds on Standard Java Architectural Patterns and Technologies for Flexible Invocation Options**: The Rule Execution Server execution unit uses Java Connector Architecture to provide pooled, managed access to rule-based decision services. Applications invoke the services through a wide choice of invocation technologies, including stateless or stateful synchronous invocation via simple Java objects (POJOs) or EJBs, or asynchronous invocation via message-driven beans (MDBs). The execution unit can be deployed in leading J2EE application servers via prepackaged resource archives or as an embedded J2SE component in non-J2EE architectures. J2EE deployments can take advantage of clustered deployments for scalability and robustness.

- **Rule Execution Server Embeds a High-Performance Engine for Both Inferential and Sequential Rule Execution**: Different kinds of rule-based decisions require different approaches to rule execution. For decisions requiring inferencing, such as underwriting, computations or clinical decision recommendations, the ILOG JRules engine offers RetePlus, a third-generation implementation of the Rete algorithm that delivers industry-leading performance on up to hundreds of objects and thousands of rules. For simpler decisions, such as validation, retail promotions and compliance monitoring, the engine offers sequential execution. For sequential applications with significant condition sharing between rules, the engine’s globally optimized FastPath technology delivers execution throughput up to 15 times that of standard sequential execution, and up to 10 times the speed of hand-coded Java.

- **Decision Services in Rule Execution Server are Manageable and Monitorable**: Whether an application requires throughput of 500 or 500 million transactions per day, system administrators and
operations managers require tools for monitoring and managing the deployment of critical decision services. A Rule Execution Server 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 Rule Execution Server console or can be scripted using ANT tasks.

- **Rule Execution Server Provides API-Based Extension Points for Customer-Specific Needs:** All Rule Execution Server management functions are available through Java Management Extensions (JMX) plug-ins, so they can be automated through standard enterprise management tools, including HP OpenView and IBM Tivoli, or customer written programs. Execution statistics by rule set and server are also accessible through both the Rule Execution Server Web console and the JMX plug-ins. By combining management and monitoring functions, system administrators can construct automated service level agreement monitoring and response triggers.

Rule Execution Server also has low-level APIs for customizing the invocation process to provide, for example, dynamic runtime selection of rule sets.

In addition Rule Execution Server, the ILOG JRules 6 engine is provided as a standalone Java component for embedded or client-side rule applications. The combined capabilities of the ILOG JRules 6 engine and Rule Execution Server make ILOG JRules 6 execution a powerful component for a BRMS without compromise.

### 7. 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 many different stakeholders, from both business and IT, to fully define, build and manage the application.
- They require secure, scalable management of the business rules that actually automate policy
- They require high-performance, scalable execution of the business rules deployed with the application.

The ILOG ILOG JRules 6 BRMS recognizes the role of each stakeholder in business rule applications, from analyst and architect to developer, policy manager and system administrator, and provides targeted tools for each. The resulting four components of ILOG JRules 6 – Rule Studio, Rule Team Server, Rule Scenario Manager and Rule Execution Server – combine to form the ideal foundation technology for building a business rule application that does not compromise the interest or needs of any stakeholder to deliver BRMS without compromise.