Exploring IBM WebSphere ILOG JRules Integration with IBM WebSphere Process Server

A Technical Overview and Use Case

Chris Berg
Product Manager, IBM WebSphere ILOG BRMS,
Application and Integration Middleware Software,
IBM Software Group

Contents

Introduction

Business Process Management (BPM) and Business Rule Management (BRM) are two technologies that are used to improve the agility, flexibility and efficiency of operational processes. Many people have questions about the differences between the two, or use the terms interchangeably—there are, however, clear differences in terms of the functionality and value of each. BPM is focused on defining, orchestrating and monitoring long running processes that are comprised of both people- and system-based activities. BRM is focused on defining, maintaining and executing decision logic that is used at specific points within a process or as part of automated decisions within business systems. Overall, the complement, or synergy, between BPM and BRM is about reaching further—expanding the breadth of problems that can be solved with a single solution. Both technologies share goals such as improving the efficiency and visibility of business processes, but they do so at different levels in a solution.

While it is true that BPM thrives on Service Oriented Architecture (SOA) and open architectures, it cannot by itself replace many of the systems it orchestrates. At the same time, the drive for transparency and encapsulation of business logic still remains for those existing systems. BRM can open up existing systems, provide extended levels of transparency and allow both BPM and existing applications to share more than ever before. The synergy relies on the difference in orientation between BPM and BRM. First, consider the following comparison:

BPM orchestrates and improves business processes

- **Flow** orientation
- **Human** orientation
- Crosses system and organization **boundaries**
- **Process**-oriented transparency—driving awareness and improvement of business processes to an increased set of stakeholders
- **Long** and short running[1]

BRM expresses and automates business decisions

- **Data** orientation
- Encapsulates to a **single boundary** of a decision
- Promotes **reuse** for any client (BPM and otherwise)
- **Decision**-oriented transparency—increasing visibility of decisions driving critical business applications and processes
- **Straight through** processing[2]

While BPM focuses on the overall business process, BRM automates and encapsulates business decisions at every level of a solution. When they work together, BPM can resolve problems related to silos and inefficient human interactions with systems and and each other, while BRM employs data validation, transformation and selection to automate tasks or make a step in a process (or application) more productive and predictable.

[1]BPM as a technology typically supports micro-flows or short-running processes as well. However, emphasis has been placed on human-oriented processes, which tend to execute over longer periods of time.

[2]A BRMS may also be integrated in a stateful manner, which contradicts the emphasis on short-running decisions. However, the majority of clients typically integrate the technology in a stateless configuration.

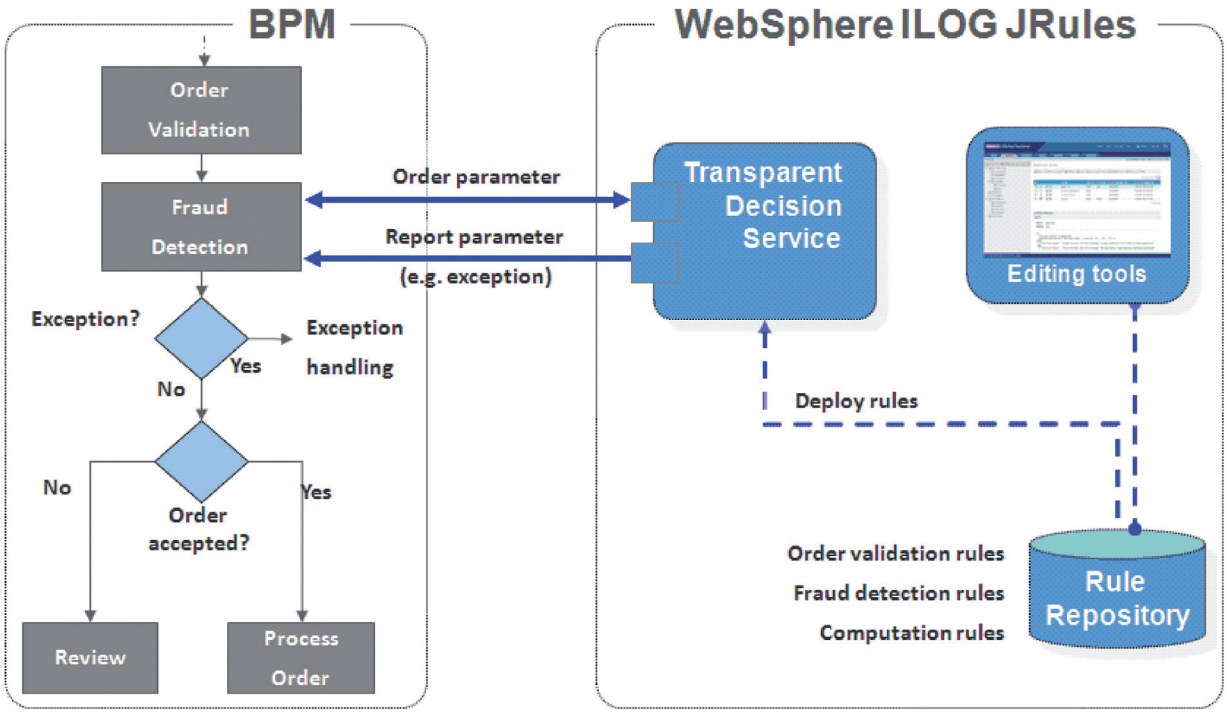

Figure 1: BPM Integration with IBM WebSphere ILOG JRules

While the orientations of the technologies may be orthogonal, they depend upon a contractual relationship that is typical in an SOA. For example, in the case of SOAP, the service signature is defined by a WSDL, an XML schema (XSD) and a policy. The XSD may in turn be shared among all upstream and downstream integration points within a business process, and is a critical artifact in the integration between BPM and a BRMS.

Managing a decision and authoring rule is primarily focused on data. At design time, a BRMS presents decision content to business users for authoring and management, since they understand the content directly and make the better owner. They also benefit from separating the BRMS life cycle from the BPM life cycle to support rapid changes to decisions that are owned by the business. Business process models tend to change less frequently than decisions, and the overall business goals may be maintained while operationally the frequency of change continues at different rates between all of the participating life cycles:

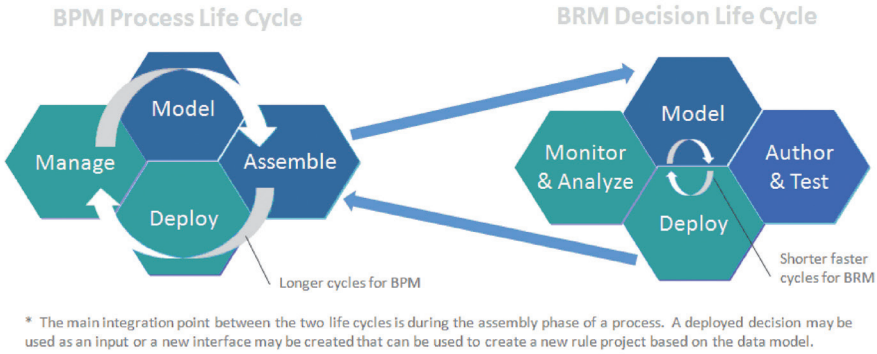

Figure 2: The BPM and BRM Life Cycles

Once a decision has been deployed, it may be reused as an SOA endpoint for any client to use. The reliance of BRMS on patterns of encapsulation and SOA enables the synergy with BPM and completes the story for better management and reuse across platforms. Both BPM and BRMS are SOA enablers and together make SOA smarter and agile.

Integration Architecture

Clients have seen the value in combining IBM® WebSphere® Process Server (WPS) and IBM WebSphere ILOG JRules for many years, and the integration of JRules has remained a consistent complement to the rich set of service components. Whether the business requirements involve human workflow or orchestrated SOA, JRules extends the reach of BPM to business users, separates the rule management life cycle by further encapsulating business logic, and makes decisions available to a wide range of consuming applications, including Java, COBOL and .NET.

JRules may be used with WPS just like other service components, such as human tasks and business state machines. Business processes may be composed of any service component and be reused as needed.

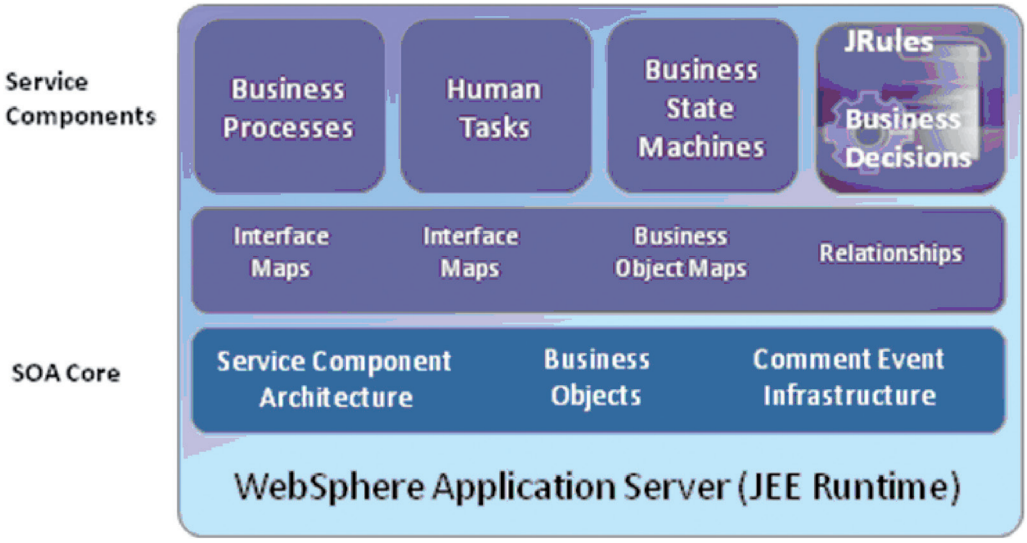

Figure 3: JRules Business Decision Integration as a Service Component

Service components relate to each other through a signature that includes interfaces, object maps and implementation. Business Process Execution Language (BPEL) is used to describe and execute the process at runtime. JRules offers two ways to integrate decisions with a BPEL model, using SOAP with Hosted Transparent Decisions Services (HTDS) and with the Decision Service Wizard, which installs as a WID plug-in.

Integrating a Business Decision as a Web Service Using HTDS and WID

Using out-of-the-box features in IBM WebSphere Integration Developer (WID), a developer may easily import a WSDL from the JRules Rule Execution Server (RES) and use it within a business process. WSDL is provided by a feature packaged with RES called Hosted Transparent Decision Services (HTDS). HTDS is an enterprise application providing the Web service endpoints within RES. It is a service wrapper for the Session Bean and facilitates the binding to SOAP.

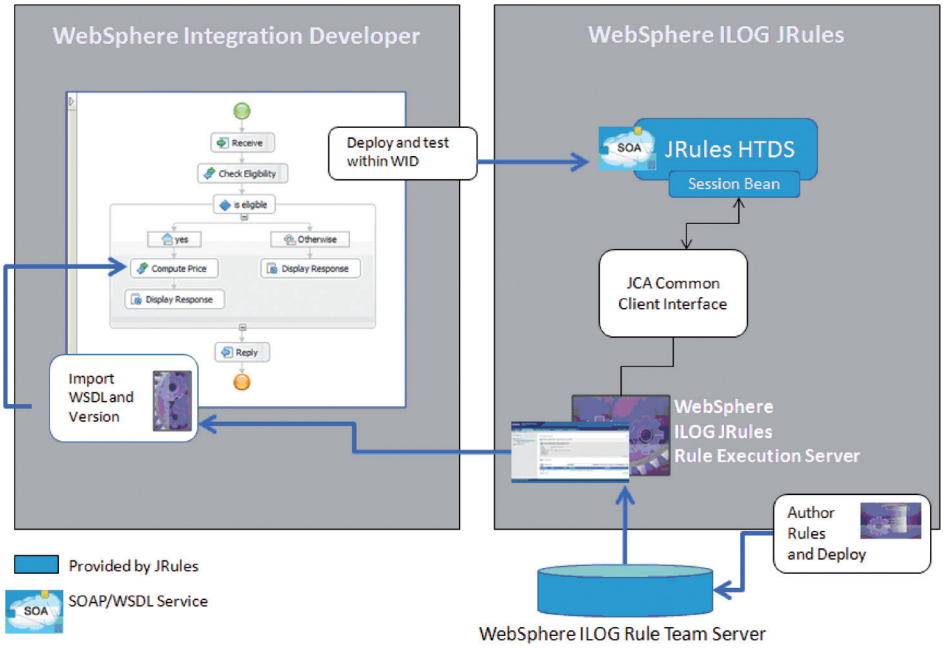

Figure 4: HTDS Integration Architecture with WID

The WSDL import process will generate many of the required WID project artifacts, including the interface and business objects (data types). JRules provides two options for WSDL:

- **Get HTDS WSDL for this "ruleset" version:** This option provides the WSDL for a specific version of a set of rules, or ruleset, for the decision. Use this version if you do not want the decision to change for existing long-running process instances.
- **Get HTDS WSDL for the latest ruleset version:** This option always provides the latest version of a decision. All existing and new process instances will take advantage of dynamic changes to the decision.

The WSDL file is portable. WID can use it from the file system or it can be pulled from UDDI after an administrator has manually deployed it.

Integrating with the Decision Service Wizard (WID Plug-in)

JRules provides a Decision Service Wizard that installs as a WID plug-in. At the time a decision is required within a process; the developer launches the wizard and completes several steps. Upon completion, the wizard generates all of the required WID project artifacts for a complete service component. The service component must be deployed before it may be unit tested or incorporated into an existing module and business process. The architecture of the SCA integration is as follows:

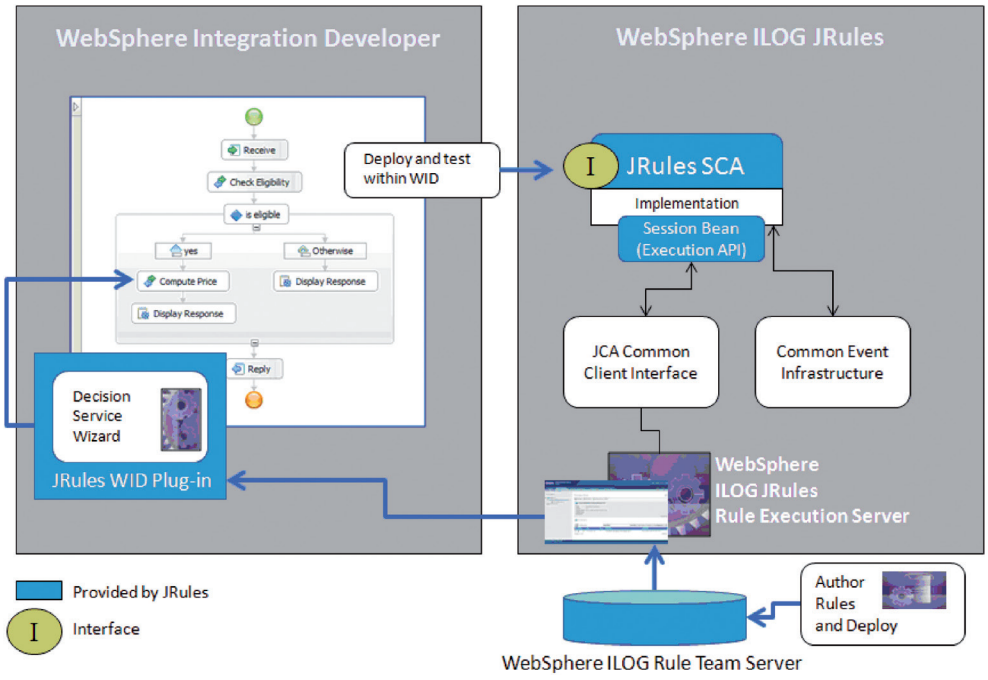

Figure 5: Decision Service Wizard Integration Architecture with WID

The implementation of the service component wraps the RES execution API and provides options for multiple rule session providers (POJO Invocation, Local EJB and Remote EJB). Choose the session provider that best matches your deployment requirements. For example, choose the POJO invocation if you want the RES to run in process with the process instance or choose remote EJB if you have an external instance of RES running in the datacenter. Local EJB is a good option if you want to avoid serialization overhead but still need transaction support. Finally, you must use one of the EJB options if the business decision participates in a transaction.[3] Below is a basic matrix showing the session providers offered by the decision service wizard:

	RES Updates	WPS Clustering	Transactions	Remote RES
POJO Invocation	×	×*		
Local EJB	×	×*	×	
Remote EJB	×	×	×	×

* Both POJO invocation and local EJB require the RES execution unit (XU)to be deployed within the same JVM.

Figure 6: Session Options for the Decision Service Wizard

Regardless of which one you choose, any changes to a ruleset that are published to RES will be reflected in the service component. A specific version can be established by updating the generated code without running the wizard a second time.

[3]JRules does not depend upon transactions for execution, and is typically considered to be an in-memory operation. However, decision support for transactions is quite useful. JRules may participate in transactions to determine if they should succeed or fail based on a decision result..

The decision service wizard offers added support for POJO object models and integration with the Common Event Infrastructure (CEI). If tracing is turned on for a ruleset, the service component can pass execution trace information on to CEI, which can be viewed later in the Common Event Browser provided by WPS. published to RES will be reflected in the service component. A specific version can be established by updating the generated code without running the wizard a second time.

The decision service wizard offers added support for POJO object models and integration with the Common Event Infrastructure (CEI). If tracing is turned on for a ruleset, the service component can pass execution trace information on to CEI, which can be viewed later in the Common Event Browser provided by WPS.

Below is a table showing the relative merits of choosing between HTDS and the decision service wizard:

	HTDS	Decision Service Wizard
XSD Model	×	×
POJO Model		×
SOAP	×	
EJB		×
Common Event Infrastructure		×
Clustering	×	×*

*Only the remote EJB session provider supports clustering while using the wizard

Figure 7: Feature Support Matrix Comparing HTDS with the Decision Service Wizard

Getting Started with Integrating JRules and WPS

The following illustration can be used to guide you through the process of using WPS and JRules together:

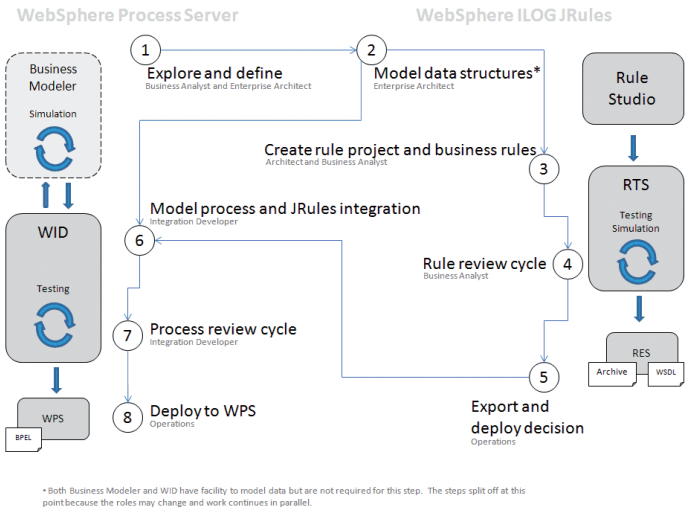

Figure 8: WPS and JRules Integration Use Case

Whether the integration architecture involves using the decision service wizard or WSDL, the steps are basically the same:

1. This is a critical step for either a BPM or BRM project. This initial step explores and identifies the candidate processes and decisions, how they related to each other, and how they impact and align the initiative with the business goals of the organization.This step informs step two and ultimately facilitates the reuse of decisions across the enterprise.
2. A shared model is created in the form of an XML schema (*.xsd). The schema provides the most widely used data types for the business domain. It is typical for an enterprise architect to establish and maintain shared models between systems, though they may collaborate with business analysts on this step. Maturing the model early in integration reduces the amount of refactoring across all of its dependencies.
3. The architect and business analyst collaborate by creating the rule project and perfecting the representation of the business object model and vocabulary together.

4. The JRules Rule Studio project is published to IBM WebSphere ILOG Rule Team Server (RTS), where the business analyst can perfect the rules through various iterations of testing and simulation using Decision Validation Services (DVS).
5. Once a decision has been matured and approved through a governance process, it can be published to RES for production use by operational staff.
6. The JRules service component is created using either HTDS (WSDL) or the Decision Service Wizard.
7. The integration developer and process owner engage in various iterations of testing and simulation with the process.
8. Operations publish the process at the appropriate time after an approval is attained.

Integrating a JRules Decision into WID Using Web Services and WSDL

Assuming that one has created a schema and followed other tutorials for creating a rule project, the process for importing WSDL from JRules into WID is straightforward and does not vary from the documentation. The first step is to download the WSDL option you want for the process. In this case, the latest version of a ruleset is chosen. Clicking on the link will prompt the user to download and save the WSDL to a hard drive.

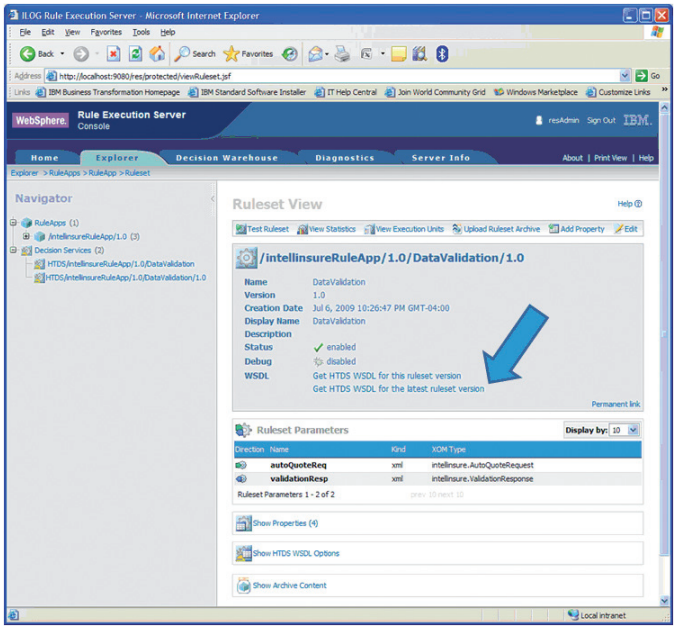

Figure 9: WSDL Options from the RES Console

Next, right click on a WID module and select import. Choose the WSDL/Interface option and point it to the WSDL recently saved on the hard drive.

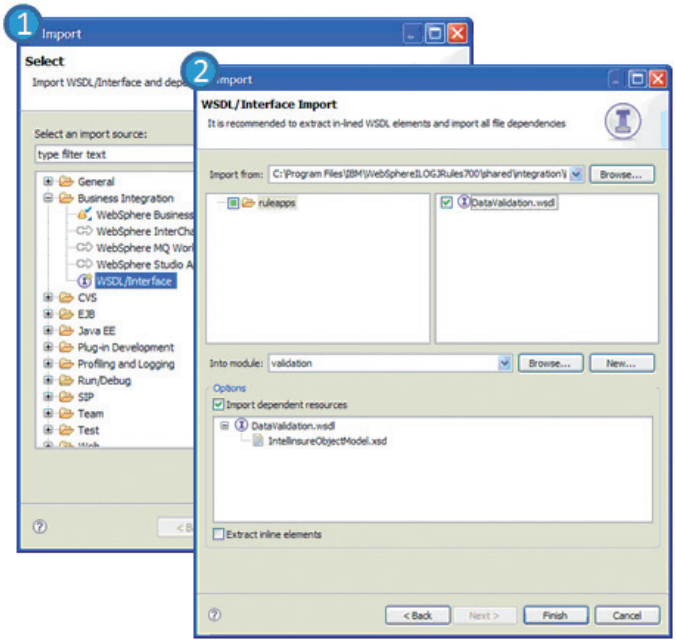

Figure 10: WSDL Import Steps

Once this is done, it takes just a few steps to establish the import of the WSDL as an assembly diagram. Here are the steps:

1. Create an import.
2. Add the existing interface from the imported WSDL.
3. Use an existing Web service port.
4. Select the port generated by the imported WSDL.
5. Select the SOAP format JAX-RPC.

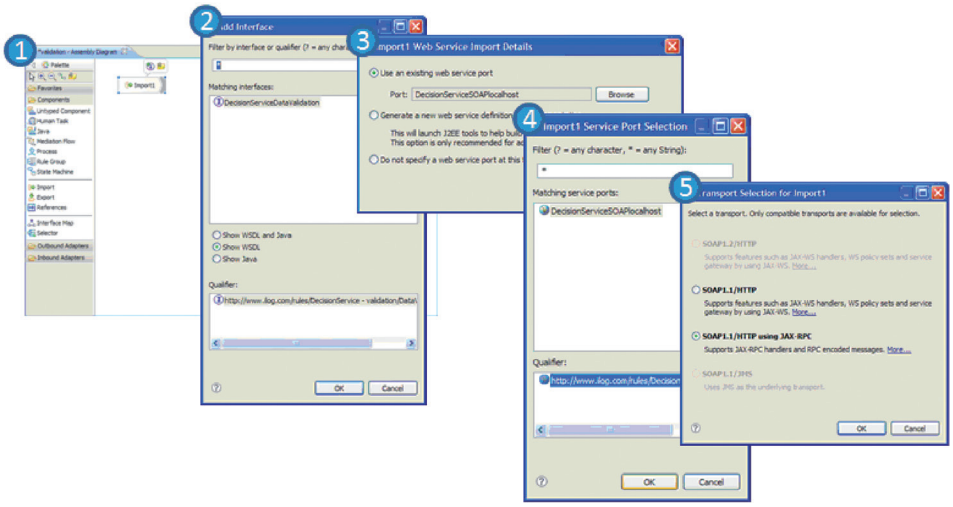

Figure 11: Steps to Import WSDL

Once you have finished these steps, the module can be tested within WID. Invoking the test will publish the module to WPS and invoke the service. Monitor the execution of HTDS by reviewing the statistics of the service itself. Navigate to the statistics view of the RES console to see any activity by the service:

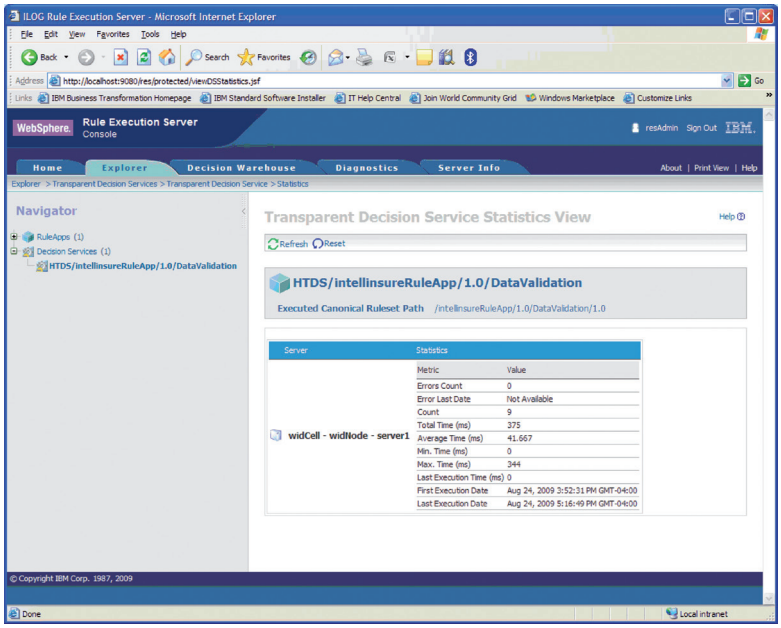

Figure 12: JRules RES Console Showing Statistics

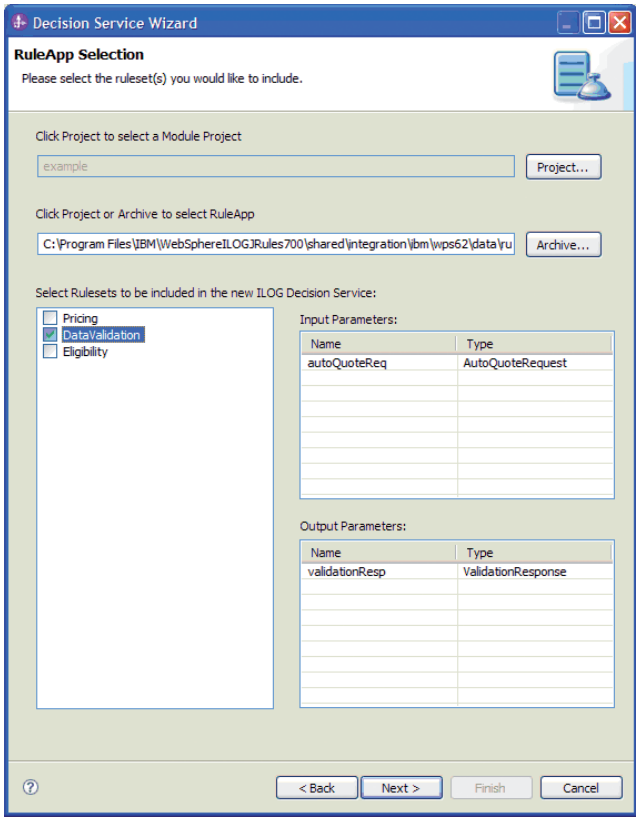

Figure 13: Steps to Launch the Decision Service Wizard

Generating a Service Component with the Decision Service Wizard

Once the WID plug-in has been installed, you may follow these steps to integrate a JRules decision within a BPEL process.[4] First, create a new WID module in the workspace. Now, move your mouse curser over "Integration Logic" and select "New" and then "Other."

[4]The WID plug-in is packaged as "SupportPac LA71: WebSphere ILOG JRules Integration for WebSphere Process Server" from the support area on IBM.com. In order to use the SupportPac LA71, one must have WID 6.2 and JRules 7.0 installed along with the WebSphere platform support JAR file.

At this point, the Decision Service Wizard should appear like the following:

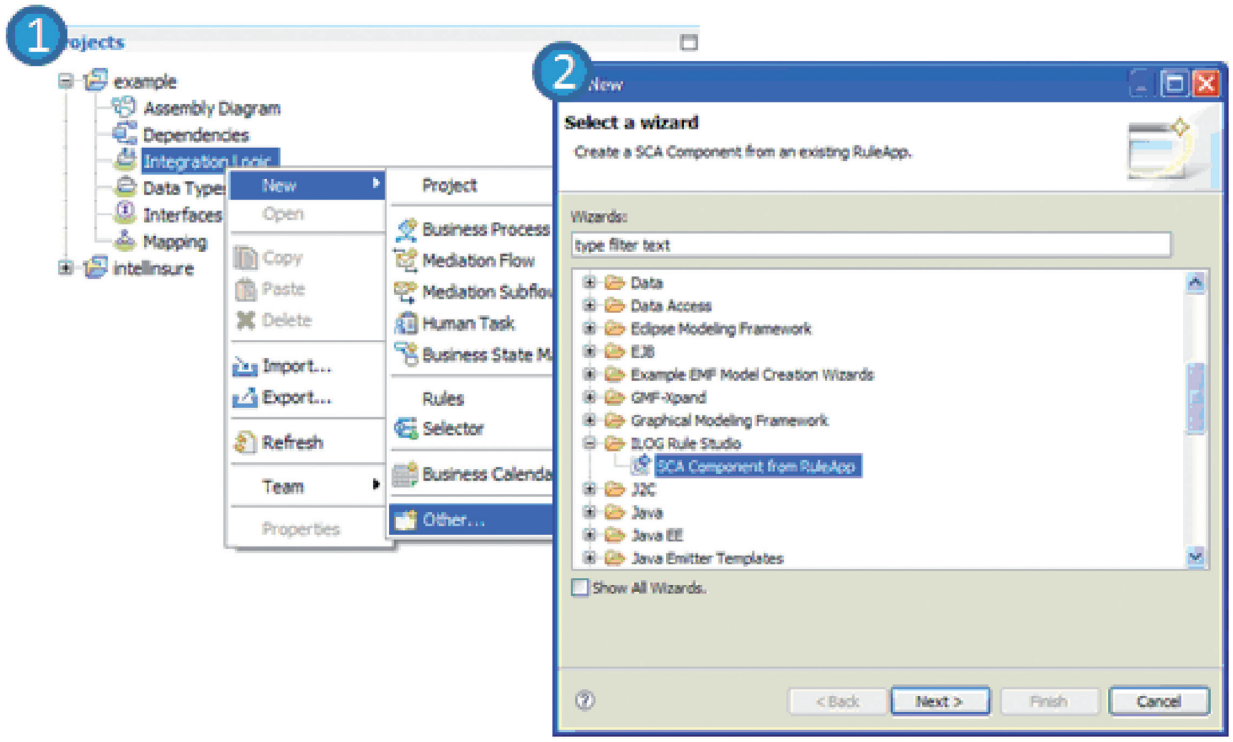

Figure 14: Selecting a RuleApp within the Decision Service Wizard

Establish the project module intended for the service component. A rule application archive must also be selected from the file system. The wizard will infer the decision signature from the archive and display the decisions available for the service. Note that a rule application archive may contain multiple decisions, but only one is chosen in this example to maintain a sense of parity with the HTDS integration example in the previous section.

The next panel in the wizard requests the object model or XML schema required by the project parameters. In this case, an XSD file selected from an external location may also reference one from the project itself.

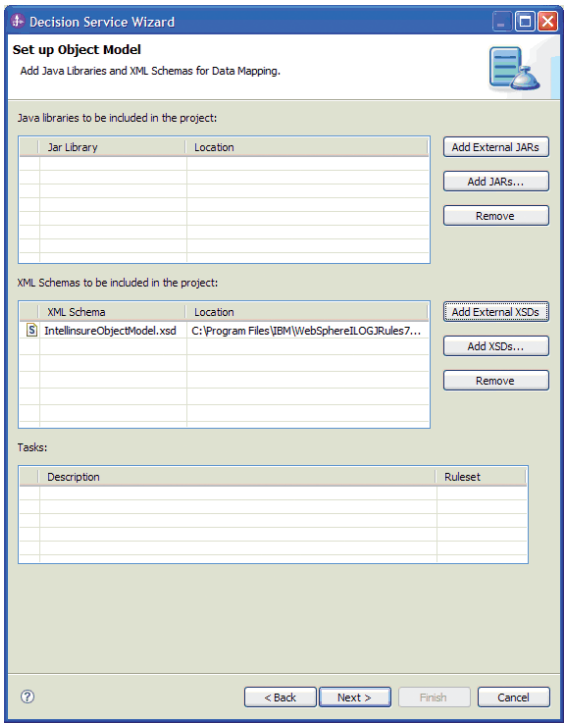

Figure 15: Add the Execution Object Model (XOM) within the Decision Service Wizard

The final step will establish content required for the implementation of the service component. This includes a namespace and object name for Java code generation. The trace option will generate code that will pass on response content to the Common Event Infrastructure (CEI) and is optional.[5] Finally, the implementation type offers options for session type based on the RES Execution API.

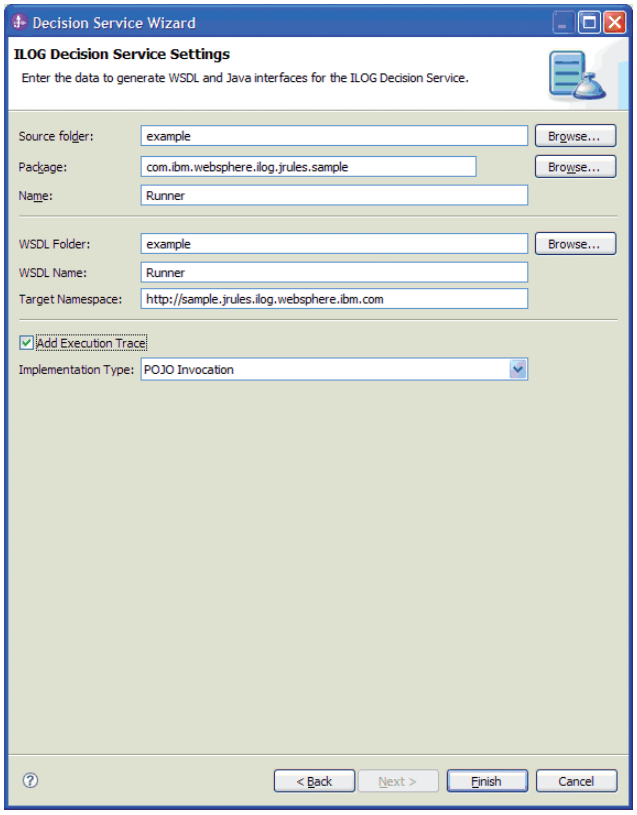

Figure 16: Establish Settings within the Decision Service Wizard

[5]When the SupportPac installs, it deploys JRules modules to the WPS instance running within WID. In doing so, it turns on the trace option for the sample project used in this example. This would otherwise be a manual step for a new project. Trace must be turned on for a ruleset deployed to RES before the CEI

Once completed, the service component may be tested.

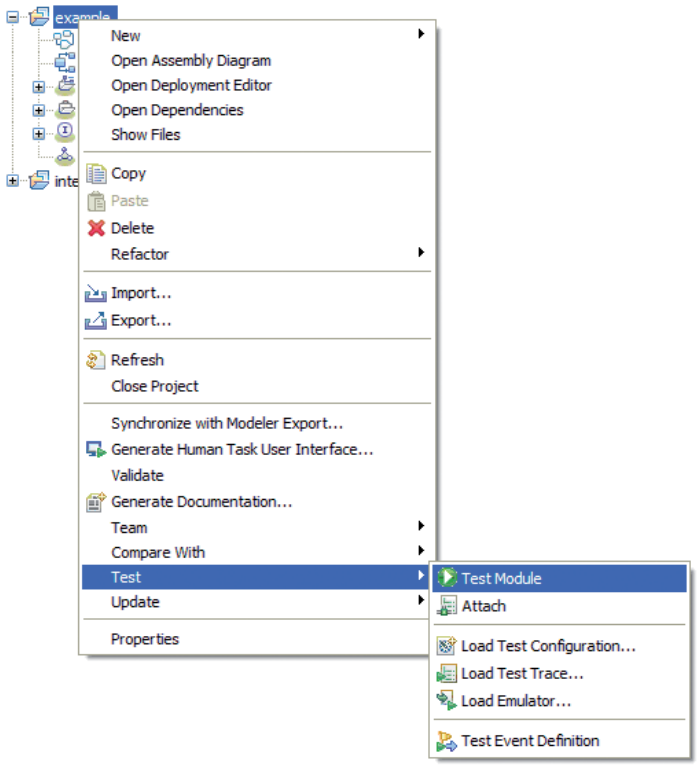

Figure 17: Testing the Service Component Generated by the Decision Service Wizard

The service component returns data from a decision. In the case below, the decision returns validation information regarding an insurance policy.

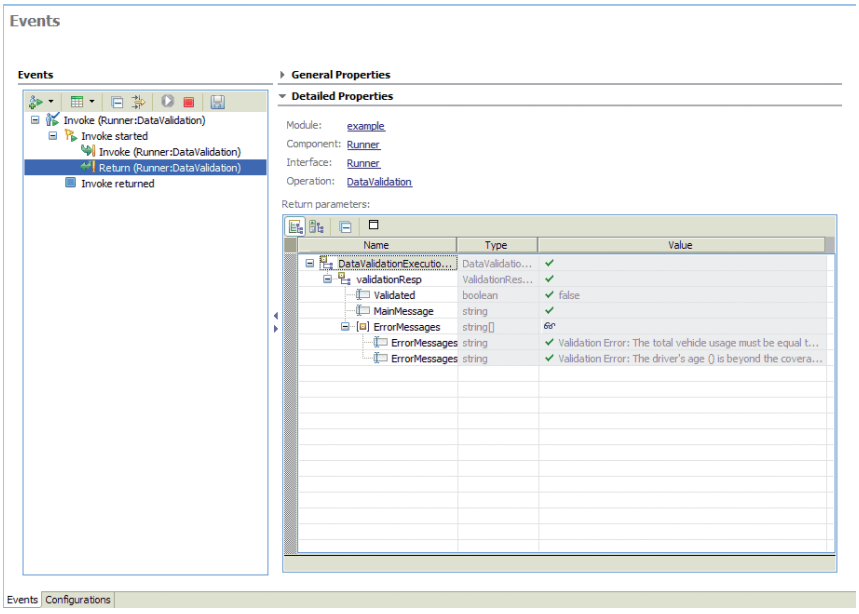

Figure 18: Creating XOM data for Testing a Service Component

Since the trace option was selected from the wizard, the Common Base Event Browser may be used to review trace information from the ruleset:

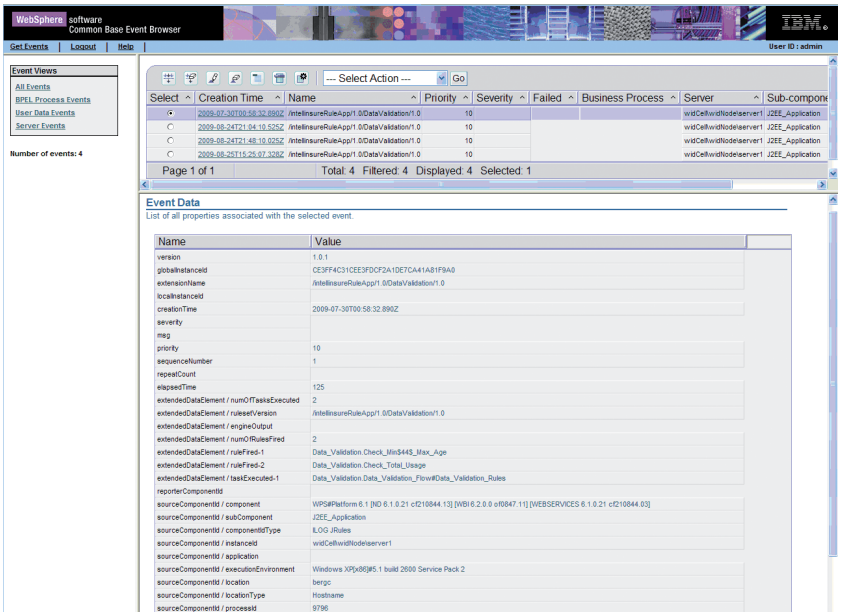

Figure 19: Common Base Event Browser View of an Executed Decision

Practice and Advanced Integration

The list of best practices continues to grow and this paper makes no attempt to substitute for a thorough statement on methodology. At the same time, there are basic steps one can take to ensure that the maintenance of the integration can stand the test of time and minimize the impact of refactoring.

Making Choices about Rule Implementation

The first steps in the integration between WPS and JRules may come down to making basic choices between the rule technology within WID and JRules. There is general agreement that the following is a solid starting point for achieving the best fit in a solution.

Use JRules when	Use WPS rules when
the rule management life cycle must be indepenent of the BPM life cycle (implies that the frequency of change is different between a decision and the process)	the rules belong to the life cycle of the process
rules are authored and maintained by business users	rules are authored and maintained by the same user that owns the process model
decisions contain many rules	decisions contain very few rules
decisions require independent testing and simulation by business users	decisions are tested and simulated at the same time as the process
decisions require inference or contain complex data structures	data used for decisions is limited to routing and basic evaluation
rule content must be shared between multiple applications and platforms (Java, COBOL, .NET)	

Figure 20: Selecting a Rules Implementation

The implication is that the choice made between JRules and WPS should hinge on the issues of governance and life cycle more so than any other.

Minimize Refactoring and Improve Testing with Service Abstraction

While a decision service interface may not change, the members of the underlying object model may undergo churn in the early phases of a project. Since the task of refactoring the object model is an ongoing concern, it makes sense to wrap the decision service as a standalone module and service component. In this way, the integration of the decision can be managed independently from the larger process model. This has the following advantages:

- As decision versions change, so may the object model. Abstracting the service into its own module allows for data mapping to take place and account for the variance.
- Specific implementation of the decision service may change without impacting the parent process. For example, one might start out with HTDS integration and then migrate to a service component generated by the Decision Service Wizard.
- Making a decision available as a reusable service within the context of WPS improves patterns of reuse within other processes that need the same service.
- If RES is not immediately available, a service stub or mock service may be created that emulates basic behaviors until the service is implemented.

The diagram below separates the process modeling step from the service integration and abstraction step.

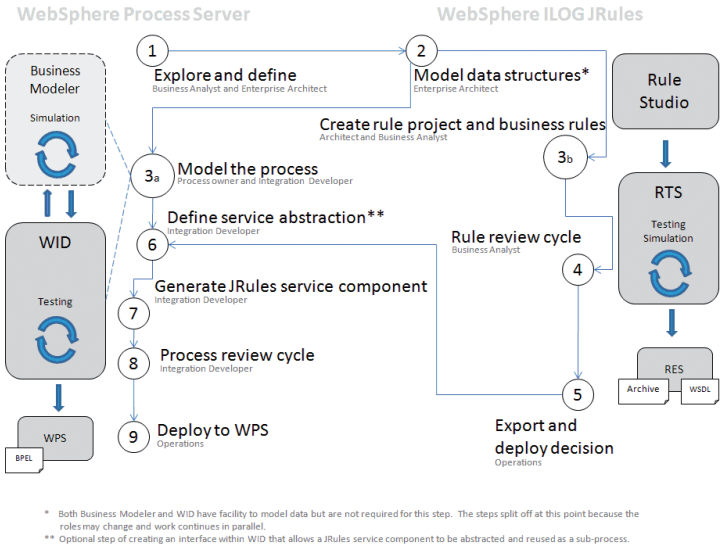

Figure 21: Additional Step for Service Abstraction

In this scenario, Steps 3a and 3b may occur at the same time. Step 6 includes both the creation of the JRules service component and the steps required to abstract it.

Testing and Simulation

Both WPS and JRules support facilities for testing and simulation. It is very likely that JRules will participate with some WPS testing and simulation scenarios. In other cases, decisions will be tested and simulated independently since they have their own life cycle. Following the practice above, abstracting the service will open the door for mock services that stand in for RES during testing and simulation.[6]

Conclusion

Productivity and predictability are welcome in the face of change drivers—the forces that stress an organization. At any given moment, the right mix of technology may be required to get more out of less, improve cycle times, and drive home margins even while the market may be in upheaval or the business turns into acquisition mode. Using WPS and JRules together extends the reach of a solution to tackle change in many of its forms.

Whether the change drivers are anticipated, the need for agility, process and decision consistency, and proper tooling for each stakeholder, will impact the overall success of a project. Although the list of using either WPS or JRules independently is quite long, the list below offers compelling reasons for using them together:

- Externalizing decision logic allows processes to be streamlined and stabilized. Business rules tend to change at a more rapid rate than the overall process model, and a BRMS reduces a significant portion of refactoring and churn that might otherwise take place within the BPEL.[7]
- WPS and JRules both provide specific environments for different users and roles. Experience shows that management of a business decision is not part of the role of a process owner. It makes sense to extend this portion of the business process to them rather than translate the content solely through a requirements process or some other form of indirection.

[6]This is applying the principal of the "mock object pattern," but applied to services. Consider reading work by Kent Beck on the topic.

[7]Early churn, it could be argued, cannot be avoided for either the WPS or JRules project. However, as a project matures, both projects will settle into their own unique rhythm. Moreover, having a mature model up front can significantly improve both projects and reduce early refactoring.

- WPS and JRules together deepen the reach of a solution for more effective management of business processes and decisions within the larger enterprise. Sharing decision content for consistency between a process model and standalone business applications is critical for maturing and improving business outcomes.
- BRM and BPM will naturally have differing life cycles, which ultimate benefits the business. For reasons stated above, embracing the different life cycles in terms of methodology and practice will promote the most gains from the BPM/BRM synergy—placing agility at all the critical and appropriate points in the solution.

Special thanks to the following people from IBM that provided helpful comments, insight and editorial commentary: Andy Ritchie, Brett Stineman, Eric Erpenbach, and YeePin Yheng.

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

Produced in the United States of America
November 2009

WSW14065-USEN-00