Fast facts

ILOG JRules puts the rules that define the business back into the hands of the business. With traditional application development, business rules have been implemented deep within application code or even the supporting database. JRules externalises all these rules and makes them visible and available to the business. This ensures that the rules and policies defined by the management are the same as those implemented in the computer systems. It also enables the business to change the rules quickly and accurately to reflect changes in policy or to react to external events.

The J in JRules stands for Java and this means that the rules can be seamlessly integrated into the application suites that process the business transactions. ILOG also sells ILOG Rules, which is a C++ rule engine, and ILOG Rules for .NET.

Key findings

In the opinion of Bloor Research, the following represent the key facts of which the prospective user should be aware:

- JRules supports the externalisation of business rules from the rest of the application. This separation is a very powerful paradigm to bridge the business/IT divide.

- The development environment is intuitive to use and allows the business analysts and users to develop and review the rules directly, therefore helping to bridge the business/IT communication gap.

- The rules can be defined using a point and click interface, a decision table spreadsheet or a decision tree graph. This choice allows the user to use the mode with which they are most comfortable.

- JRules has extensive support for the full life-cycle: Java development, business definition, distributed deployment, monitoring, auditing and governance.

- The rules execution environment can be embedded in an application so that users can make changes to the rules in real time. For example, the rules relating to a call centre could be altered hour by hour to reflect changes to special promotions.

- JRules has been embedded into a number of application suites as well as custom applications developed by IT staff.

- JRules supports business rules and works with the systems of other vendors to provide complete solutions; for example, ILOG has a close relationship with Software AG, Filenet, EMC (Documentum), who all supply Business Process Management environment.
JRules supports J2EE in servers from IBM, BEA, JBoss, ObjectWeb, Oracle and Apache.

To exploit the separation of business rules will require some changes in the application development cycle; however, the benefits of these changes should be significant.

The bottom line

JRules provides a very powerful tool supporting the externalisation of business rules that would normally be embedded in business processes, messaging routing or transaction processing.

The externalisation of the business rules not only helps to bridge the business/IT divide but also improves the business governance by making the rules visible and clear to senior management and external auditors.

If an application can be analysed and designed so that the rules can be externalised then JRules provides excellent support.
Vendor information

## Background information

ILOG was founded in 1987 to develop visualization and optimisation software as a component that could be incorporated into third party products. Today these still form an important part of the ILOG product set.

In the early 90s ILOG extended into the rules engine arena with ILOG Rules. ILOG Rules is a C++ based product and is still actively marketed. Based on the experience, ILOG developed ILOG JRules, in 1997, which provides similar functions to Rules but built for Java. In 2004 ILOG announced a product for the .NET and Microsoft Office environment.

Web address: www.ilog.com

## Product availability

ILOG JRules 6.0 shipped in March 2006.

## Financial results

ILOG is an international company quoted both on NASDAQ and Euronext under the symbol ILOG and 006673 respectively. The company has corporate headquarters in Gentilly, France and Mountain View, California. In fiscal year 2005, it had revenues of 125m USD. Revenue has increased continuously over the last eight years and the company has shown profits over the last six years.
What are Business Rules?

The term 'business rule' is in common use by IT vendors but is not used in exactly the same way by all vendors. ILOG defines a business rule as any statement of the form "IF criteria THEN action" that is of interest to the business. The business will be interested if the rules represent policy defined by the business managers or external regulators. Such policies could apply to compliance, eligibility, pricing, sales initiatives, risk etc. A simple example would be "IF value of shopping cart greater than 200 USD THEN give 10% discount". The business managers will be interested in this as it represents policy on pricing and sales initiative. They will also wish to modify the values 200 and 10% easily to reflect changing market conditions. They may also wish to modify the criteria by adding other tests, such as number of items in the basket; or to modify the action by, for example, replacing the discount by loyalty points.

These rules may need to be changed regularly and/or quickly by the business. They also need to be explicit and accessible to the business.

Technically these rules may be part of validation, message routing, message transformation, database update or application logic, but should not be buried in the code. The rules should be externalized irrespective of where they are implemented.

By externalizing the rules and making their definition and modification available to the business the elapsed time, from the requirement for a rule change to its deployment, can be reduced. Figure 1 shows where savings should occur:

- In the time business analysts used to develop a request for a rule change, they can now change the rule.
- There is no need for IT to do any coding as the business analyst has done it.
- The business will take less time validating the changes as they made them directly.
- As the rules are externalised only the changes to the rules need to be deployed. This should be faster than the deployment of modified components.
Product description

Introduction

ILOG believes that business rules should be considered in a similar way to data. Databases are external to the applications and business rules should be the same. For this to be possible there has to be a separate Business Rules Management System (BRMS) just in the same way as there is a DBMS. ILOG JRules is a BRMS.

Architecture

Figure 2 illustrates the set of components that make up the ILOG JRules 6.0 BRMS.

When the rules are in production, they are processed by the rules engine. The engine is a set of Java classes that runs in a J2EE or J2SE environment. The engine can run by itself, or it can run within the Rules Execution Server which also provides performance management, remote control and monitoring.

ILOG provides separate development environments for the Java Developers (RuleTech) and the Business Analysts (RuleCare). These development environments and the operations environment are coordinated by Full-Circle BRM.

- **RuleTech** supports technical teams by providing the tools for applying best practices and processes to rule applications. The tools run in the Eclipse Integrated Development Environment (IDE) that the developer already uses for other development tasks.

- **RuleCare** provides business team members with their own web interface and rule-editing environment that is separate but synchronized with IT.

- **FullCircle BRM** refers to the tools and processes that enable full life-cycle enterprise business rule management. It enables a business rule management cycle that is independent of application development cycles while enabling synchronization between the business user and development environments.

The three functional areas described above are implemented in ILOG JRules 6.0 in four major components:

- Rule Team Server—the browser-based business-user environment.

- Rule Studio—the Eclipse based Java Integrated Development Environment.
• Rule Scenario Manager—for synchronising between the business and IT environments.

• Rule Execution Server—the runtime engine.

The rules themselves and the related business object models are kept in a repository.

### Business user environment

Business analysts and business managers need to be able to define, review and modify business rules without the assistance of a professional IT developer. Rule Team Server is an easy to use browser-based environment specifically designed for the business user.

It allows these teams to learn business rule management quickly using guided editors, templates and online help. Also RuleCare provides:

• Testing and simulation tools for the business team to assess the business effectiveness of a policy change prior to deployment.

• Tools to create and store reusable business scenarios and apply them against the business rules, allowing them to perform ‘what-if’ scenario testing.

The rules can be defined and presented in a variety of formats, and the user can choose the most suitable format for a requirement. For example, in a mail order system relationships may exist between criteria such as age, revenue and category, and special offers and promotions. Businesses typically document these as spreadsheets with a line for each combination of criteria. JRules supports this format directly and under the covers will convert it into technical rules.

In Figure 3 the question of what risk grade to give a loan is a question for the business itself to answer not IT. The spreadsheet and the underlying rule are both available to check and modify.

The spreadsheet view is a very productive way of documenting the rules, as it is concise and understandable by the business. The decision table support increases productivity further by:

• Enabling cutting and pasting of sections of the decision tables from existing Excel spreadsheets.

• Supporting asymmetric tables (for example the example in Figure 3 could include a single line to say if corporate score is below 300 then the grade is unacceptable).

Figure 3: Business users can set criteria using business logic in a familiar spreadsheet-like interface
• Checking completeness and consistency on the fly (so the missing row for corporate scores below 300 would be automatically flagged).

## Java development environment

A developer of an operational Java solution will want to be able to incorporate business rules into an application. JRules Rule Studio runs as an Eclipse plug-in so enabling the developer to create standard Java and business rules within a single Integrated Development Environment and stored in a single Source Code Control repository.

The rules are defined using a rules language. The language is implemented in several flavours.

A technical flavour (ILOG Rule Language) that is very similar to Java and can reference Java objects directly, as follows:

```
When {
    ?s: ShoppingCart(value > 100);
    ?c: Customer (category == Category.Gold);
} then {
    ?s.setDiscount(15);
}
```

This is fine for Java developers but is not easily understood by business people. Therefore a higher level of abstraction would improve the IT/business communication and this is provided by a Business Rules Language (BRL). JRules comes with a default BRL call the Business Action Language (BAL) and in this the above rule would read.

```
If

the shopping cart value is greater than $100
and the customer category is Gold

Then

Apply a 15% discount
```

Figure 4 shows the Eclipse interface for building such rules. Using this standard interface reduces the learning curve for the developer and also allows the developer to switch between different tasks quickly and seamlessly. The developer can create some Java, then a related set of rules, then test them together, and finally promote them to production, all from one interface.
The sets of rules can be called from a Java application or can be implemented as strongly typed SOA services on the leading web and application servers.

Rules can be applied against XML schemas, Java classes, databases and Web Services. The translation of the business term to the underlying Java or XML code is defined in a Business Object Model (BOM) stored in the repository.

The BOM Editor has introduced the concept of vocabulary—the representation of the business object model in business terms. Vocabulary is generated automatically through a new process called verbalization. The vocabulary can then be manually edited to further improve the natural-language feel of rules written with the vocabulary. This screenshot from Rule Studio (Figure 5) shows the editor for the Borrower class, the default verbalization, and the generated vocabulary specific to the members of the Borrower class (bottom panel).

In the examples the RHS is made up of simple methods (just changing a value) but it is possible to develop rules where the RHS uses methods that are arbitrarily complex. The end result of a method could include the updating of databases, inserting messages onto queues, running algorithms or writing to a browser.

A basic rule is of the form 'if … then …'. However a particular business process is likely to have multiple rules related to it.

In some cases these are independent of each other and the order they are processed is unimportant but in many cases the order is important with the output of one rule having an effect on another. ILOG supports these requirements by allowing rules to flow using a flowchart GUI. This function required the basics rules engine to be extended to support algorithms that understand tasks and sequences.

**Partners**

Besides being used to develop rules for bespoke applications, JRules has also been embedded into application suites to enable easy configuration of the business rules related to the suite. A good example is the InterConnect Suite from Equifax.

BRMS is a complimentary technology to Business Process Management and ILOG has partnered with IBM, EMC (Documentum), Software AG, BEA, FileNet and others to ensure a close integration between their products.
Synchronise, deploy, manage

When business rules are included in an operational system it is essential that they are governed and controlled as rigorously as the standard application code. Deployment of new rules and changes to existing production rules must be done in a regulated manner so that the state of the production system is known at all times. This will ensure that changes are not being made for malevolent reasons; it will also ensure that if a transaction is later queried by a user or the auditors that the decision processes can be replayed to see if the results are correct or not.

To ensure proper governance several facilities have been incorporated into JRules.

A rule in production may have been developed cooperatively by Java developers and business users. JRules provides a function to synchronise both environments so that they reflect the same underlying rules.

Multiple releases of a rule can be saved in the repository at one time. This may include:

- Several historical versions that may be required by the auditors.
- The current production version.
- A new version that has a planned deployment date.
- Several future minor and major releases that are in development.

Rules in production can be monitored and logged. The monitoring can be incorporated into an external operations management suite so that out of line performance can be analysed across the Java and the business rules to quickly identify and resolve the problem.

Logging of the production system will provide the input to subsequent analysis of the rules processing. This analysis can suggest changes to the rules to better meet the objectives of the business. The logs can also be used to drive what-if analysis and the testing of new versions of the rules.

Run time environment

The engine is a forward-chaining logic engine. For those of you not familiar with this concept the next box gives a brief description.

**Forward-Chaining Logic Engine**

The processing of a forward-chaining logic engine is defined by a series of 'IF condition THEN action' rules (note there is no ELSE clause).

When the engine consumes a message it looks at its rules base to see if any of the conditions evaluate to true. If so the action in the rule is processed; this action may change some information in working storage.
When the action is complete the engine checks to see if the condition of any other rule evaluates to true; if so, the process is repeated. Rules that previously evaluated false may now be true because of changes in the working storage.

This continues until no rules that have not been triggered evaluate to true. Then one or more messages may be emitted.

Such engines will have mechanisms to decide which order rules should be processed if more than one evaluates as true and also to ensure that a rule does not trigger twice.

As can be imagined, defining processes using a rules base is distinctly different from writing procedural code. However it is a very powerful construction and particularly suitable for business rules as it tends to be the way business people define business rules.

The engine is implemented as Java classes and methods.

When an application needs to use the rules engine:

- It instantiates a rules context. The context contains working memory, a rule set and an agenda.
- The application asserts the object that the rules engine is to work on.
- A set of rules is loaded from the repository or from a file.
- Once the application has asserted the objects, the application will execute the fireAllRules method.
- The rule engine determines which rules are eligible for execution and places an instance of those rules in the agenda.
- Each rule in the agenda is executed and then removed from the agenda.
- It is possible that a rule will change an object and this change may mean that a new rule becomes eligible for execution and this will be put into the agenda.
- This process continues until there are no more eligible rules in the agenda.
- The rule engine will then pass control back to the application.

JRules directly supports Java Message Services (JMS). The rule engine can be instantiated as a Message Driven Bean (MDB). This means that when a message is received by JMS the rule engine method is automatically fired and the message processed by the rule engine. The rule engine can then, if required, send messages via JMS back to the originator or on to other queues.

JRules also supports ‘temporal logic’, so that a rule can be defined to wait for a period of time, or wait until an event occurs before proceeding.
Support for executing rules in an SOA environment is also included, allowing rules and groups of rules to be deployed as decision services. This includes support for runtime security, logging and auditing that help to address SOA governance challenges.

JRules 6.0 incorporates significant performance and scalability improvements. This is essential as JRules is being incorporated in high volume transactional systems.

**Future Directions**

A further release in 2006 will enable deployment to .NET. Future releases will merge the functions of JRules and Rules .NET.
Summary

ILOG JRules provides a full function business rule management system (BRMS) that enables all business rules to be externalised. The externalisation enables the business to develop the rules directly thereby ensuring that the rules accurately reflect the business policies. Externalising the rules also ensures that external auditors and regulators can view the rules to verify that the enterprise is enforcing their requirements.

The rules engine is a set of Java classes and therefore can become an integral and seamless component of a complete application.

No other product takes the separation of business rules from application logic to the level of JRules. The recently released JRules 6.0 has greatly improved functionality, chiefly in its support of the business user, and also the performance and scalability of the run time engine.

The effective use of JRules does require some rethinking of the development process, but this should not be hard and the benefits obtained will definitely repay the effort.
Bloor Research has spent the last decade developing what is recognised as Europe’s leading independent IT research organisation. With its core research activities underpinning a range of services, from research and consulting to events and publishing, Bloor Research is committed to turning knowledge into client value across all of its products and engagements. Our objectives are:

- Save clients’ time by providing comparison and analysis that is clear and succinct.
- Update clients’ expertise, enabling them to have a clear understanding of IT issues and facts and validate existing technology strategies.
- Bring an independent perspective, minimising the inherent risks of product selection and decision-making.
- Communicate our visionary perspective of the future of IT.

Founded in 1989, Bloor Research is one of the world’s leading IT research, analysis and consultancy organisations—distributing research and analysis to IT user and vendor organisations throughout the world via online subscriptions, tailored research services and consultancy projects.
Copyright & Disclaimer

This document is subject to copyright. No part of this publication may be reproduced by any method whatsoever without the prior consent of Bloor Research.

Due to the nature of this material, numerous hardware and software products have been mentioned by name. In the majority, if not all, of the cases, these product names are claimed as trademarks by the companies that manufacture the products. It is not Bloor Research’s intent to claim these names or trademarks as our own.

Whilst every care has been taken in the preparation of this document to ensure that the information is correct, the publishers cannot accept responsibility for any errors or omissions.