

Business Rule Standards -- Interoperability and Portability

April 2005

Mark H. Linehan
Senior Technical Staff Member
IBM Software Group Emerging Technology
mlinehan@us.ibm.com

Donald F. Ferguson
IBM Fellow
Software Group Chief Architect and Chairman, SWG Architecture Board
dff@us.ibm.com

A Business Scenario

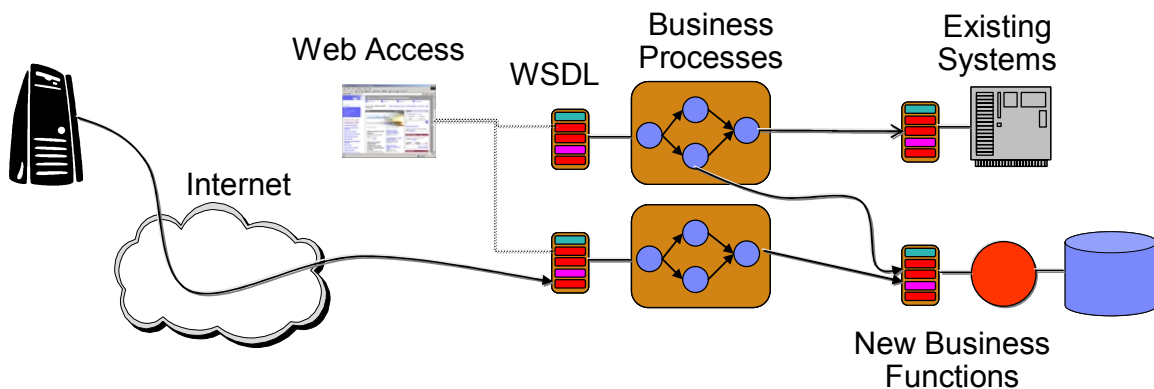


Figure 1: Example insurance scenario

Consider a commercial Web site for an insurance company (Figure 1). The web site surfaces Web pages that allow customers, agents, employees and partners to interact with applications. The site also exposes a Web service interface using WSDL [\[WSDL\]](#) to document interfaces and a WS-Interoperability [\[WSI\]](#) binding for access from partner systems. This scenario introduces several important use cases for rule technology. These include enabling interoperability, model

driven architecture and development, implementing services using rules, customization of process and services, portability, and event analysis. We explore these concepts below.

Interoperability

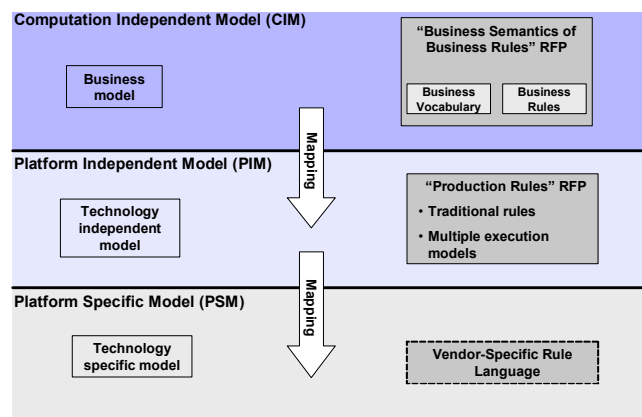
WS-Interoperability protocols provide support for runtime interoperability. There is also a need for interoperability between “tools,” based on a common way to describe service interfaces, the data that is being interchanged and related metadata. WSDL provides basic support for defining the interfaces to services. WS-Policy [\[WSPO\]](#) provides support for documenting Web service protocol extensions for various qualities of service, for example security [\[WSSE\]](#) or reliable messaging [\[WSRM\]](#). WS-BPEL [\[BPEL\]](#) defines abstract processes that describe valid sequences of invoking service operation. Ontology languages [\[OWL\]](#) allow annotations of Web services and XML Schema Definitions of messages to provide semantic information. The eXtensible Access Control Markup Language [\[XACML\]](#) specifies access control rules.

These technologies alone are not complete for defining interfaces to services. There is a requirement to annotate an interface definition with pre-conditions, post-conditions, and invariants for the service. These constrain the service input, output, and transformation behavior. For example, in the insurance scenario outlined above, an access control rule might limit agent services to registered agents. We believe that an XML language for annotating service interfaces and XML messages with constraints and rules is complementary to WS-BPEL, WSDL, etc. and will provide valuable functions.

As the previous discussion starts to show, we are beginning to see the unification of concepts from the worlds of Service Oriented architecture (W3C, OASIS are driving key standards), Model Driven Architecture (OMG is driving these standards) and the Semantic Web (W3C is driving this work).

Model Driven Development

The Unified Modeling Language [\[UML\]](#) & Meta Object Facility [\[MOF\]](#) provides a foundation for model driven architecture, metadata management and development based on open standards. UML supports modeling interfaces, structure (classes, associations, inheritance) and behavior (interactions, activities, state machines). UML also defines the Object Constraint Language (OCL). OCL supports modeling pre-conditions, post-conditions, constraints, invariants, etc. OCL can provide the basis for rule based extensions to service interface definition. MOF supports the definition of specific metadata models and supports programmatic and XML based interchange of model information – including rule specifications.



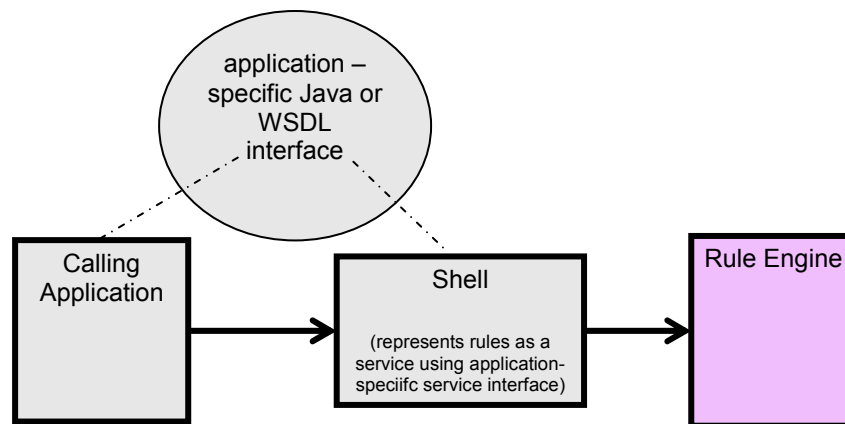
OMG is expanding the Model Driven Architecture (MDA) efforts to include formal support for Business Process and Business Rule modeling. W3C should work with the OMG effort to define an XML based, Platform Independent Model (PIM) for rules. This will complement the future Computation Independent Model (Business level models) that captures the semantics of business vocabularies and rules. The three levels of abstraction will provide end-to-end support for MDA.

1. The OMG CIM extends modeling to support rules and vocabularies oriented to business users. This is the business rule model layer.
2. An OMG and W3C XML and Web Services based PIM model for rules and constraints will complement WSDL and ontology languages for documenting interfaces, and WS-BPEL for defining service implementations. This layer also provides a new way to implement services, as described below. This is the logical rule model layer.
3. The XML PIM standard will provide a bridge into vendor specific rule languages and engines. This is the physical rule layer,

Note that these layers correspond to the 'Conceptual/Business', 'Logical', and 'Physical' model layers commonly used in the field of data modeling and data base design.

Rule sets often operate solely on the inputs and outputs, which is a good fit for the service abstraction. If rule sets require additional information, the rules should use services for accessing it. This requires WSDL/WS-Interoperability integration with the PIM and CIM models, and implementations in rule engines.

Implementing Services

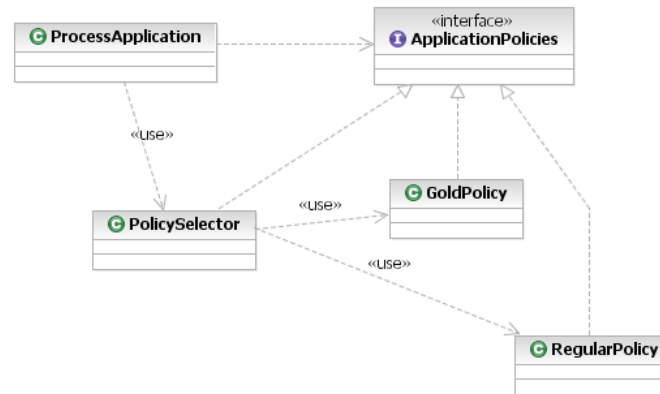


Historically, rule engines have been isolated runtimes or have had special interfaces for interaction with more traditional programming languages and models. IBM's programming model "wraps" rule engines and rule sets with a service abstraction. Rules become a way to implement a service. The service encapsulates all linkage/calling conventions and programming logic specific to the rule engine interfaces. In our example, inference rules could provide a mechanism for computing risk and assigning rates to policy applications. The user portal, workflow processes, etc. would interact with the service through the WSDL interface using standard protocols.

Customizing Services

There is a special use case of rules in complex business applications – *customizing services*. This use of rules becomes important when we discuss languages and models for implementing

services using rules. The figure below provides an overview of the customization model. A service, in our example a WS-BPEL process for processing an application for insurance (*ProcessApplication*), documents points of variability or policy enforcement points by declaring a dependency on a service interface *ApplicationPolicies*. This is a use of the *Strategy Pattern* [GOF4]. To tailor or customize a service, programmers simply modify or provide a new implementation of the customizing service – *GoldPolicy* and *RegularPolicy* in our example.



There will typically be multiple implementations of the customizing service. A distinguished service, a *selector*, mediates the call and routes the request to the correct implementation. In summary, the base service declares and uses an interface. There may be multiple implementations. The base service actually calls a selector, which routes the request to the correct implementation.

Rule Models

The preceding discussion demonstrates that there are many use cases for rule languages in Service-Oriented Architecture (SOA) solutions. Complex inference rule sets can implement services that classify policies and compute risks. There are also many simpler use cases. For example, an automobile policy base rate may depend on the tuple of {make, year, cost, state}. This is a natural problem for *Decision Tables*. There are also common use cases for *Decision Trees* and sequential execution of sets of *if ... then ...* rules. Routing rules in selectors often fit the pattern of *if ... then ...*, with the *if* clause selecting elements of the messages and the *then* clause selecting the instance.

Customization and selection provide an opening for enabling non-traditional programmers (“business professionals”) to customize and tailor business solutions. These professionals routinely use spreadsheets, which is a natural metaphor for *Decision Tables*. Business schools often offer classes that teach *Decision Trees* and *if ... then ...* constructs (structured language). We use the term *simple business rules* to describe rule concepts like *Decision Trees*, *Decision Tables* and *if ... then ...* sets. Our solution to rules and SOA must surface models that are intuitive to business professionals and for which it is possible to build easy to use tools.

It is possible to implement simple business rules in a more complex and powerful engine, but this should be a decision at the Platform Specific Model. The CIM and PIM layers should provide first class support for simple business rules, enabling intuitive tools for business professionals and mapping to multiple implementation technologies.

The CIM, PIM and PSM layer should also support more powerful rule models.

Event Management

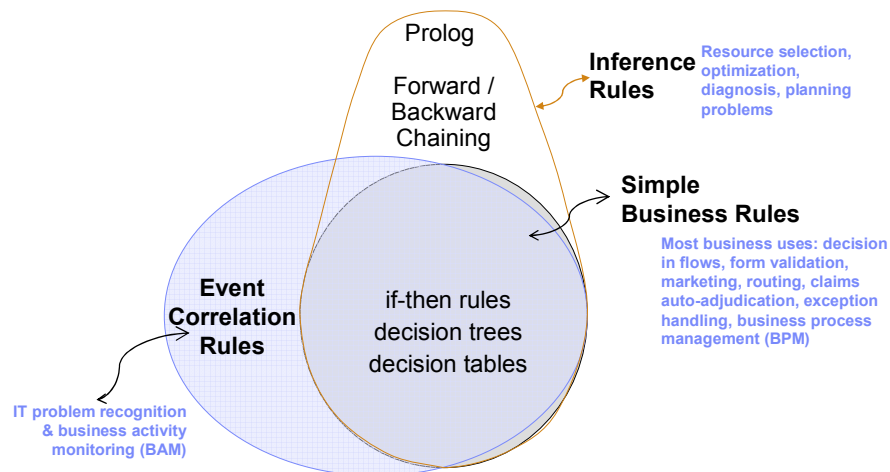
Event correlation rules recognize relationships across multiple events arriving in an event stream within a time window. For example,

- When an application server failure occurs within 30 seconds of a database failure, assert that the root problem is the database failure.
- Email the managing director if the sum of the insurance policies for new policies in a 24 hour period exceeds \$1000K.

Event correlation rules are another important use case that requires special consideration. These rules can fit into the SOA model. Services emit events to a topic at an event broker using WS-Notification [WSNO]. Other services subscribe to topics using filters. The listening service's implementation uses event correlation rules to process the event stream, maintain internal state during the processing of streams of events, assert new events or invoke other services.

Summary

One of the largest challenges in the computer industry today is complexity – in programming models, system designs, user interfaces, and standards. Complexity overwhelms our customers, significantly limiting their ability to extract value from our technology. Complexity also challenges the pace and value of industry standardization efforts.



We have proposed three principles that we believe bring simpler approaches to business rules:

- Applying SOA concepts to integration of rules with applications.
- Ensuring that standards for business rules integrate with model driven architectures and complement the capabilities of UML and related OMG Business Process and Rule standards and Web services standards.
- Focusing on a family of standards for rules that ensures a natural fit between the use cases and the technology. Different vendors will choose different engine technology (code generation, rule engines, etc.) to implement the Platform Specific Model. The Platform Independent Model needs to be a simple, natural fit for the important use cases.

This “simple things should be simple” approach reduces complexity, better matches common skill sets and enables intuitive, simple tools. The figure above provides an overview of various rule models.

References

- [BPEL] OASIS Web Services Business Process Execution Language (WSBPEL) Technical Committee. http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wsbpel.
- [GOF4] Eric Gamma et al. *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley. 1994.
- [OWL] Deborah L. McGuinness and Frank van Harmelen. "OWL Web Ontology Language." February, 2004. <http://www.w3.org/TR/owl-features/>.
- [ODM] Object Management Group : Ontology Definition Metamodel , <http://www.omg.org/cgi-bin/doc?ad/05-01-01>
- [UML] Object Management Group, "Unified Modeling Language". www.omg.org/uml
- [MDA] Object Management Group ."Model Driven Architecture". www.omg.org/mda
- [MOF] Object Management Group "Meta Object Facility"
http://www.omg.org/technology/documents/modeling_spec_catalog.htm#MOF
- [WSDL] David Booth, Canyang Kevin Liu. "Web Services Description Language (WSDL) Version 2.0 Part 0: Primer.» December, 2004. <http://www.w3.org/TR/2004/WD-wsdl20-primer-20041221/>
- [WSI] Web Service Interoperability Organization, <http://www.ws-i.org/>.
- [WSNO] OASIS Web Services Notification (WSN) Technical Committee. http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wsn
- [WSPO] Lee et al. "W3C Workshop on Constraints and Capabilities for Web Services." September, 2004. <http://www.w3.org/2004/08/ws-cc/wsp-20040903>.
- [WSRM] R. Bilorusets et al. "Web Services Reliable Messaging Protocol (WS-ReliableMessaging)." February, 2005. <http://www-128.ibm.com/developerworks/library/specification/ws-rm/>.
- [WSSE] OASIS Web Services Security (WSS) Technical Committee. http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wss.

[XACML]

OASIS eXtensible Access Control Markup Language, http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=xacml.