



How Blaze Advisor Works

Table of Contents

Introduction to Rule Service Technology	3
Century Coverage Insurance — A Case Study	6
Step 1 - Collect and Define an Initial Set of Business Rules	8
Step 2 - Develop the Technical Architecture	9
Step 3 - Import and Refine the Enterprise Business Object Model	10
Step 4 - Construct a Visual Map of the Supported Business Process (rule flows)	11
Step 5 - Develop the Business Rules of Blaze Advisor	12
Step 6 - Develop the Rule Maintenance Templates	13
Step 7 - Generate the Rule Maintenance Applications	14
Step 8 - Test the Rule Services	15
Step 9 - Deploy the Business Rule Services	15
Rule Technology Benefits	16
About HNC	17

About This White Paper:

This white paper introduces rules management technology, defines various key terms and uses a detailed case study to show how it can be used to revolutionize your application design, development and maintenance. It describes the roles and responsibilities of those using the technology and walks through a realistic scenario step by step.

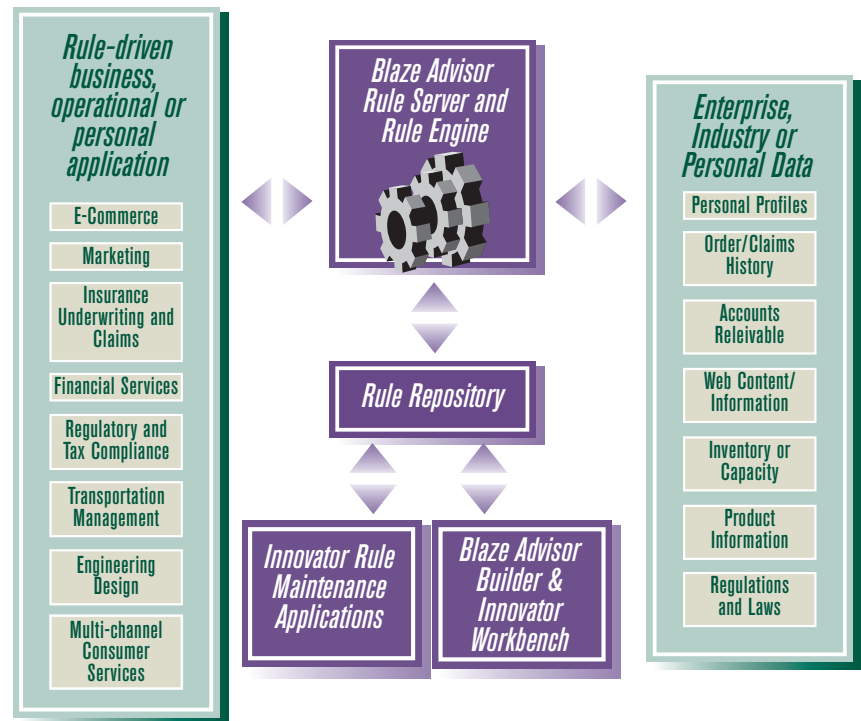
Introduction to Rule Service Technology

Blaze Advisor from HNC is a complete solution for rule service design, rule maintenance, and rule service deployment. The benefits of this approach to application development are:

1. Non-technical business managers and analysts can maintain application logic and personalization rules without being dependent on technical resources. This reduces application modification cycle time and thereby speeds organizational response to changing conditions. It also reduces overall application cost of ownership.
2. Application developers are able to separate business logic from procedural code. Application logic can be maintained without bringing the calling applications down. Business rules can be viewed and understood more easily. Separating business logic from procedural code also allows for faster development of business logic.
3. Business logic and personalization rules can be used by multiple applications and client devices, supporting enterprise consistency. This becomes more important as organizations offer the same services and access to the same information over multiple channels of communication.
4. Organizations can personalize their business processes, data display policies, interactive dialogs, content display, Web site and article links, etc., based on either general user categories or based on specific user history and interests.
5. Enterprises can deliver a new level of user-centric service by combining transaction services with information services. Enterprise data, personal alerts, and business rules can all work together in a multichannel world.

The following graphic provides a simple overview of the primary functions supported by the system:

- An e-business application (on the left) asks the Advisor Rule Server to perform some business service (for example, to develop a contingency plan, perform risk assessment of a customer, or make a promotional offer)
- If required, Blaze Advisor can reach out to enterprise data to get the information it needs to perform the service in accordance with the enterprise's business rules



Blaze Advisor from HNC is a complete rules management solution.

- and the data relevant to the case at hand
- Rule services are defined and constructed with the Blaze Advisor Builder and Innovator Workbench authoring tools. Blaze Advisor Builder is a rule service design, authoring, and testing environment, and Innovator Workbench is a rule maintenance application design tool. Innovator Runtime is software that controls safe rule editing and new rule creation in Innovator rule maintenance applications.

Interactions Between Applications, Databases, and Rules

Generally, calls from a client application initiate rule service execution. However, the initiating events might also be database update events or calls from another rule service. When called, a rule service may need to immediately perform any of the following actions in order to proceed:

1. A database look-up with a calculation and a return of a value to an object in memory.
2. A data extract composed of multiple objects (based on a single table or a database view) for use by rules at some point in the execution of the rule service.
3. The creation of a dynamic (temporary) object that can be used by the rule service for processing purposes based on a class understood by the rule service.

Definitive Rule Execution

Rule Service Components:

Ruleflows provide the sequence of execution steps, defining the function or ruleset triggered at each point in the process flow. Ruleflows can have subflows as well as unlimited branching.

Rulesets are groups of rules in which the rules have the same structure and are evaluated as a set at the same point(s) in the process flow.

Functions perform algorithmic processing (calculations).

Rulebases are large, reusable rule structures, potentially including multiple rulesets and functions.

Rule execution can be definitive or be a part of a larger process flow (next section). In a definitive situation, rules explicitly address what should happen for every possible set of

conditions in a single process step. We may have a rule service with hundreds of rules, but the subset that is relevant for a single invocation of the rule service is very small.

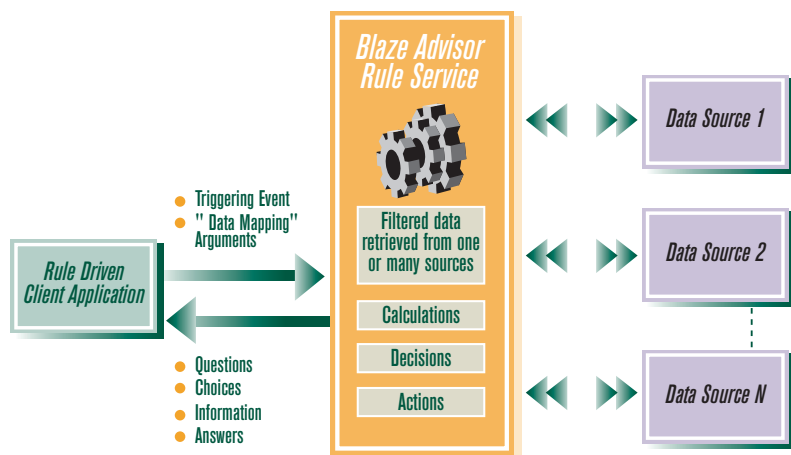
As an example, we often have a rule “problem” that looks something like this:

If Customer is a High Risk Driver* **and** has had claims less than \$250 in the last year **and** requests a quote for a boat owner’s policy,

Then present Boat Policy Question Set **and** Boat Safety Web Content **and** offer a special discount of 3% on the premium.

*High Risk Drivers are customers with more than \$5,000 in claims in the last two years.

This example is fairly simple. It does not contain any process complexity. The conditions merely seek to determine account status in light of the quote request action and then offer the appropriate follow up questions, content, and discount. Any complex calculations in the example would be handled by a



Blaze Advisor rule services can respond to triggering events from many applications, using data from many databases.

function that performs risk-adjusted quotes for insurance. This particular rule “problem” might cover four customer segments, five claim history bands and 50 insurance products resulting in 1,000 possible combinations for the three left-hand side conditional variables. The rule service may need 1,000 rules to address every possible triggering event, but each calling event is sending at most, one value for the three variables. So, in this case, only one of the 1,000 is the right one to execute.

The Rete algorithm (developed at Carnegie-Mellon) determines rule relevancy. This algorithm, which was designed for rapid pattern matching, is extremely efficient at finding the rule conditions that are relevant at a given moment. On the right-hand “action” side of the equation, the correct design depends on whether all desired actions are driven by the complete set of left-hand side conditions. In this example, the Web links might be based purely on the product interest, whereas the Web content might be personalized based on customer segment, and the discount might be a function of all three variables. As a result, it might be more efficient to manage the desired actions using separate rules:

- If Customer requests a quote for boating insurance product, then display Boat Policy Rider document
- If Customer is from segment High Risk, then display High Risk Policy holder options
- If Customer is from segment High Risk and has had claims less than \$250 in the last year and requests a quote for Boating insurance product, then offer a special discount of 3% on the premium

Larger Process Flows: Rules that Execute Other Rules

In other cases, the execution of one set of rules may necessitate the execution of another set of rules, often in a multi-step, branching ruleflow. Sometimes the transition from one rule set to another is based on a planned, automated ruleflow, and sometimes it is driven by the client’s particular response to questions posed in an earlier step. At other times, the firing of a

rule service or a ruleset is based on an event. Here is a short list of some of the triggering events that can initiate rule service execution:

1. A direct call by an external application
2. A specific answer or set of answers to a question or set of questions posed
3. A specific result determined by rules execution in a prior ruleflow step
4. When something is needed by a rule attempting to execute in a ruleflow step
5. When a database record is changed, deleted or created

The importance of having a complete system cannot be overstated. It should support structured, branching rule flow, interactive dialog question sets and event rules that can launch other rules. A rule management system that flexibly meets the needs of diverse calling applications must handle all of these requirements well.

Rule Effectivity Based on Date and Time

Another fundamental concept is rule effectivity. Some rules remain effective until the rule service is modified or deleted; others have windows of effectivity. Rules may need to be in effect until a date, after a date, between dates or between specific times. They may need to be in effect on weekends or during weekdays. Any number of scenarios can apply.

Defining explicit date and time-based effectivity in advance reduces the need to update rule services and provides a built-in audit path. If all date or time-based rules are in one compiled rule service, it is easy to know what the rule service would have done in the past. Giving application designers and business people the choice of defining their rules with or without temporal constraints is a major advantage. And once you begin making business rule effectivity based on time, you will find you need to distinguish between absolute and localized time.

Special Rule Requirements

More specialized requirements exist in certain situations such as the need for prioritization, scoring, weighting, resource consumption tracking, and optimization routines. These functions reside on top of basic business rules. Optimizing a decision for lowest cost, fastest delivery, lowest risk, or maximum use of space can involve applying a set of linear programming constraints to an optimization function. These types of rules are critical when they are needed, but are far less frequently required than declarative business rules. Blaze Advisor's rules technology supports prioritization, scoring, and weighting, but does not support the tracking of resource consumption nor certain forms of optimization. Product configuration and network management products are better solutions when these requirements are paramount.

Century Coverage Insurance: A Case Study

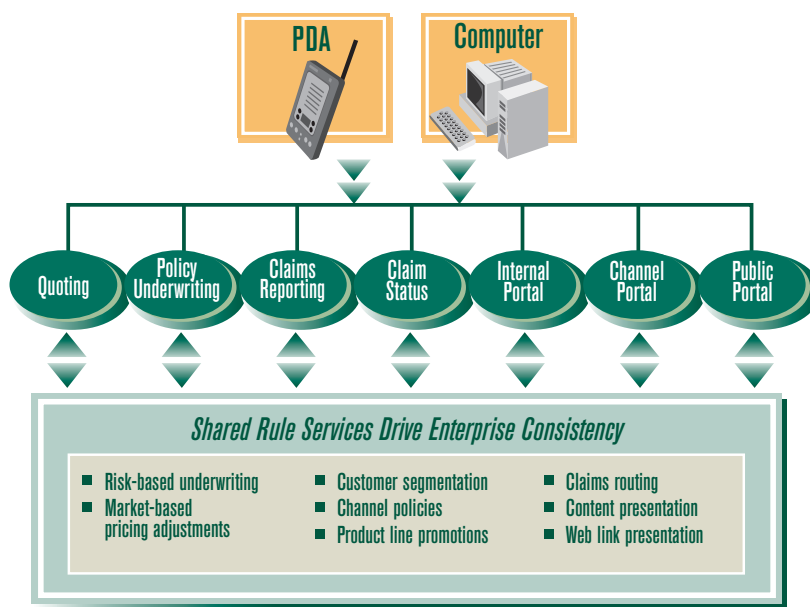
Advanced rules management systems support many usage scenarios. The two major categories of usage are:

- Assessments, decision-making and business transactions based on a combination of data evaluation, business or organizational policy, and business or industry facts. These processes can be carried out in a personalized, interactive manner or as a batch process.
- The control of access to enterprise information and/or enterprise services based on personalization rules and data evaluations for customers, prospects, partners, and employees.

The recent growth in the first category reflects the growth in custom application development for the Internet. The second category reflects the growth of personalized portal and mobile

applications. By reading this paper, we hope you will see how rules technology can support both major categories of application development.

To help make all of this come to life, we will now describe, step by step, how you would use Blaze Advisor from HNC to build and deploy rule services in support of a set of applications in both categories.



Rule services can drive consistent application behavior across a complete range of applications.

The Scenario

Century Coverage Insurance is a fictitious insurance company that sells property and casualty, auto and life insurance to individuals throughout the United States. The company has just completed a strategic planning process that resulted in many initiatives, including five major system initiatives.

The key system initiatives for the next two years are:

1. Deployment of self-service Internet applications in multiple countries for quote generation, policy underwriting and

claims processing with personalized content, Web links and customer history displays.

2. Deployment of three information portal applications for prospects/customers, agents, and internal visitors. These portals need to supply information with a high degree of personalization based on group and individual user rules. Data display, search launches, and content display need to employ a high degree of personalization.
3. Deployment of personalized mobile self-service Internet applications for simple portal, product sales, and claim reporting based on Palm O/S, Windows CE, WAP, and CHTML technologies.
4. Deployment of airport kiosks with custom front ends, but identical business rules to those used by the regular Internet applications.
5. Application Service Provider (ASP) business model support with the ability to provide:

- Full e-business selling services to large partner agencies
- Support for agencies' controlled customization of a subset of the selling and content display rules

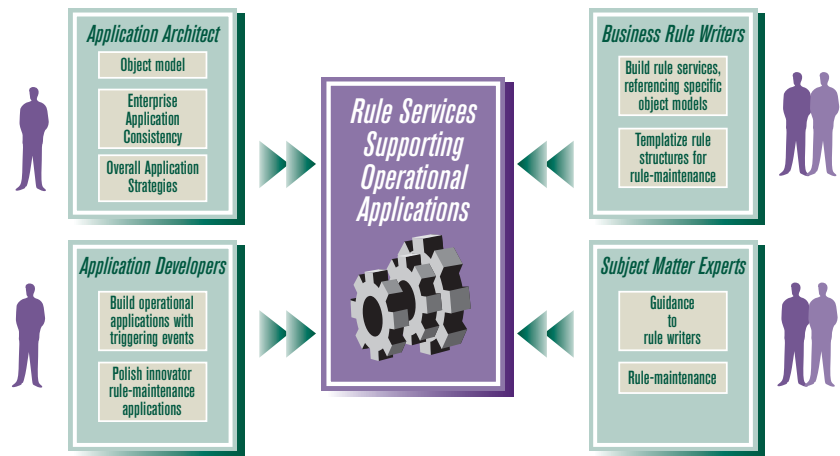
These initiatives were selected because executives felt they could collectively generate 25% annual growth in a crowded consumer insurance market-place. Century Coverage has expanded over the past 3 years through a combination of natural growth and a set of successful acquisitions. This acquisition strategy has resulted in the need to cope with multiple data models. Existing customer history is now spread throughout multiple front office and back office systems.

Historically, Century Coverage has sold through affiliated agencies, but it is now selling a growing percentage of its business directly to consumers through its Web site. Insurance is a regulated industry in most countries. The auto insurance business poses special requirements as it is regulated at the state level in

the United States, but at a national level in the rest of the world. There is also talk that the European Community (EC) will adopt central regulation of the life and property insurance businesses in the near future.

Players and their Roles

To successfully adopt rules management technology, Century Coverage Insurance requires several different jobs to be completed. Fortunately, their team already appreciates the value of using this technology. They are already committed to gaining productivity and flexibility by separately managing the business logic used in a self-service world.



Roles and responsibilities in developing rule services.

The number of people needed in each broad job category will vary over time. It also depends on the complexity and scope of rules technology adoption in the application development process. The main functional roles are:

- **Application Architects** – This group is familiar with all of Century Coverage's systems in terms of data models and technologies. They have overall responsibility for the division of labor between all software applications and for enterprise consistency and efficiency from an architectural point of view.

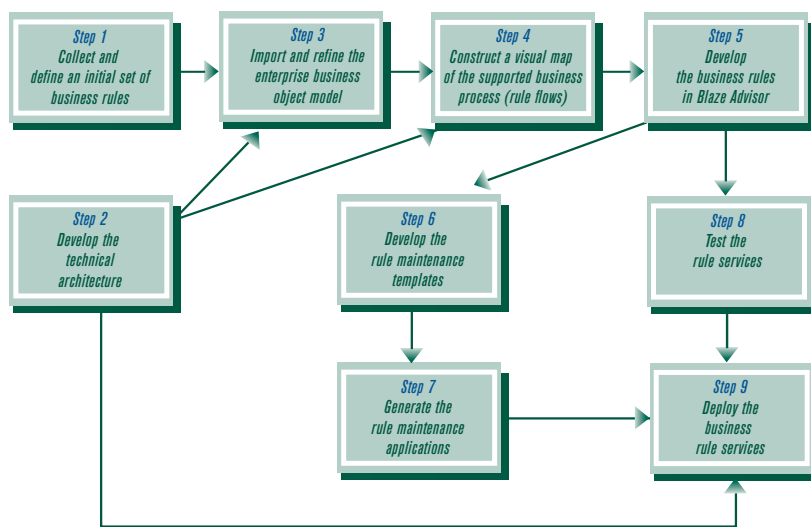
Ensuring consistency means managing the integration of standards and communication protocols between Century Coverage’s applications, data warehouses and various central systems, such as e-mail and call center IVR units. It means understanding the superset of business objects that support the complete set of business applications and the mapping between classes and databases. They must ensure enterprise consistency in terms of business transaction processing and the personalization of all customer interactions. They must define strategies for handling session “state” in multi-interaction sequenced dialogs with end users, and for handling data assembly and data retrievals.

- **Subject Matter Experts** – This group understands Century Coverage’s business and personalization objectives as well as the particular business rules associated with each line of business in each geography. These people have in-depth knowledge about Century Coverage’s products, services, competitors, regulatory constraints and obligations, channel structures, and customers.
- **Business Rule Writers** – This group takes the subject matter experts’ business knowledge and develops business process models and sets of business rules. These models and rules translate business knowledge and policies into a consistent, organized rulebase frame-work that aligns harmoniously with the self-service user interaction points and the complete set of enterprise data sources. Because Blaze Advisor business rules are in near-natural language and Advisor provides excellent visual mapping tools to model and control the sequence of logic execution, the subject matter experts generally participate in the development of the rule flows and rule-set structures. Together with the subject matter experts and the application developers, the rule writers define the content and style of the Blaze Advisor Innovator rule maintenance applications that will be used by the subject matter experts for ongoing rule maintenance.

- **Application Developers** – This group has two responsibilities. The first is end user application development, and the second is interoperability. Application development includes the design of GUIs, logic, and end-user application interfaces.

Interoperability ensures that applications work together and that the entire system can be easily adjusted when new entities, rules, or business processes must be supported. The application developers for Century Coverage will develop PC browser, mobile and kiosk applications using commonly used development tools. They will design and build required databases, GUIs, navigational frame-works, and internal application logic for the new rules-driven applications. They will decide when to put logic into procedural code and when to use rule service technology. They can insert rule services into existing enterprise applications as well as into the new applications they build from scratch. Easily defined triggering events invoke Blaze Advisor rule services that drive interactive dialogs, content display, database look-ups, assessments, decision making, and data-base updates. Blaze Advisor supports all of this by adhering to open standards and by providing tools to manage the invocation of rule services and the interoperation of Blaze Advisor from HNC with Century Coverage’s databases and applications.

Steps in Using Blaze Advisor from HNC



Step 1 – Collect and Define an Initial Set of Business Rules

The subject matter experts start the process by drawing up a first pass list of Century Coverage’s business rules by primary business process. This effort is best launched with a brainstorming session that captures a broad overview of the enterprise and its products. This ensures that all major organizational perspectives are integrated into a balanced enterprise view of the object model, the business processes, the rulesets, and functions. At this point in the planning process, the purpose is not to be exhaustive but rather to establish a framework with a specific list of the business objects required by the rules.

Here are a few examples of what the team came up with in just the first hour of its brainstorming session on selling and personalization rules:

In Century Coverage’s case, there are literally hundreds of different business rules. The fastest way to identify them all is to

group them by process step with appropriate breakdowns for product line and geography specifics.

Some rules apply to more than one line of business.

Grouping the rules by process step and, as necessary, placing them in tables when rulesets are required are the first steps in defining the rule flows and rules that will be used later in building the actual rulebase.

Step 2 – Develop the Technical Architecture

The Century Coverage Insurance application architect team then roughs out the technical architecture for its use of rule management technology. There are many choices available to today’s architects. Century Coverage opts for an open standards solution built around Java 2 Enterprise Edition (J2EE). Blaze Advisor Rule Server has built-in J2EE support, so there is no compatibility concern.

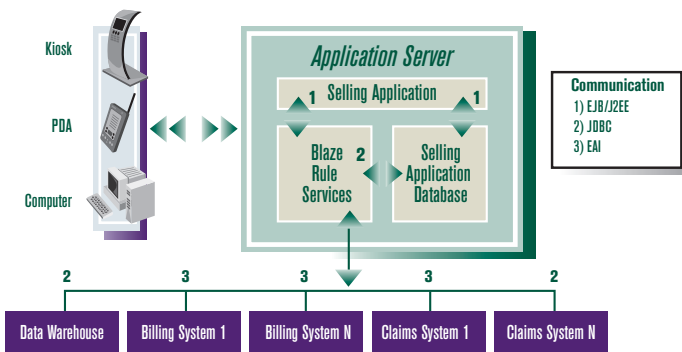
Selling Rules

- Our “Gold1” customers are those who own more than 1 policy and spend more than \$1,000 per year in total premiums. (Sum of all policies and all riders)
- Our “Gold1” prospects are those who report owing more than one policy and spending more than \$1,000 per year in total premiums
- Whenever a customer requests a quote on Fine Art insurance, promote Jewelry insurance (display content and quote \$50,000 rider)
- The discount for prepayment of total annual premium is 5% if annual premium is < \$1000
- If the site visitor lives in California, promote earthquake insurance (display content and quote \$325,000 replacement cost with \$20,000 deductible as example)
- If the site visitor lives in Texas, Oklahoma, Kansas, Alabama or Georgia, promote tornado insurance (display content and quote \$200,000 replacement cost with \$10,000 deductible as example)
- Customers who have had more than 2 property claims in the last 5 years should not receive special promotions
- Customers who are frequent late payers should not receive special property promotions
- Customers who have paid more than 60 days late in the last year more than one time are “frequent late payers”
- If the site visitor is not a customer, ask for current insurance profile (amount of policies in force with carrier and years of coverage)
- If the site visitor is a customer and doesn’t have policies in all of Century Coverage’s product lines, ask for a current insurance profile for the product lines not currently with Century Coverage

The most common invocation scenario puts a layer of customer touchpoint connectivity in front of the rule service. This way, the specific means of how Century Coverage speaks to a customer at any given point is abstracted – thus providing a consistent application of business policy across all touchpoints. Various application server vendors provide this sort of layer. Of course, there can be rules that call for personalization or another action unique to a particular touchpoint (such as mobile phones). This is important, because different touchpoints, such as desktop browsers, telephone IVR systems and mobile PDAs display information based on unique parameters. A phone, for example, might only show the top product recommendation, while the PC Web browser might show the top ten recommendations.

Similarly, a layer (sometimes known as an Enterprise Application Integration or EAI layer) is needed to facilitate data lookups and/or updates with data in other applications. Again, various vendors offer this technology. In many common cases, the Rule Engine can do the enterprise data communication itself when the data is available in open JDBC-compliant databases or via MQSeries, remote EJB, CORBA or COM. For a large established firm such as Century Coverage, there will be a mix of both approaches.

Within the application server are process objects (and objects representing data). These process objects communicate with each other to handle the various user-driven business events.



An example of part of a technical architecture.

One might manage a conversation flow through Web or WAP screens with a single JSP object. Another might coordinate the assembly of remote data that arrives asynchronously from different sources. One key process object is the rule service that performs specific business logic. In Century Coverage’s situation, these rule services underwrite policies, assess risk, determine claims eligibility, personalize information presentation, etc.

The rule services can be either stateless or stateful depending on the application design. Stateless is good when one needs a direct answer (such as, “What is the policy premium for this request-for-quotation?”). Stateful is useful when there is a series of interactions with the user and you want to remember what decisions have already been made to avoid repeating them (for example, the rules-driven conversation on what questions to ask the user in collecting information about their usage or coverage profile).

Step 3 – Import and Refine the Enterprise Business Object Model

The next step is for Century Coverage to assign the application architect and some of the subject matter experts to define the object model for the rule management universe. They start by reviewing existing enterprise database tables. Any existing classes can be imported into, Blaze Advisor to be used when writing rules. Blaze Advisor rule services can work with existing Century Coverage Insurance enterprise data. Data does not need to be re-entered and maintained separately within Blaze Advisor. Blaze Advisor uses its Business Object Model Adapters (BOMAs) to access data as required during rule service execution. Advisor provides BOMAs for Java, COM, CORBA, RDBMS, XML, and other object or data models. In our simple scenario, Century Coverage will write rules that reference a number of classes and their associated properties. A partial list of required classes includes:

- A Century Coverage customer
- An Internet visitor registration profile
- A non-Century Coverage insurance policy profile
- A Century Coverage auto policy
- U.S. state auto policy rules by U.S. state
- Country auto policy rules by country
- A Century Coverage auto claim
- A Century Coverage term life policy
- A Century Coverage term life rider
- A Century Coverage home owner's policy
- A Century Coverage personal property policy
- A Century Coverage personal property rider
- A Century Coverage personal property loss claim
- Century Coverage Web Content entities
- Web links
- A Century Coverage employee
- A Century Coverage agent
- An auto policy application
- A term life policy application
- A term life policy rider application
- A property policy application
- A property policy rider application

Blaze Advisor provides JDBC, XML, Java, and COM class import wizards with automatic data type mapping that allow the team to import one or many classes at a time. They can also use the Class Editor to add additional properties or to provide Blaze Advisor aliases when column table names are awkward or inconsistent. Since calculations and decisions made by a Blaze Advisor rule service commonly require storage in an object (such as “customer” or “visitor”) the ability to add additional class properties is crucial. Century Coverage Insurance, for instance, might add a property to the customer class for “claims history ratio” or “sum of annual premiums” based on a Blaze Advisor calculation.

Next, the team needs to think about what must be added to support the self-service Internet applications it wants to build.

Additional classes and “temporary” (or “dynamic”) objects must be defined so that the rule services can function effectively.

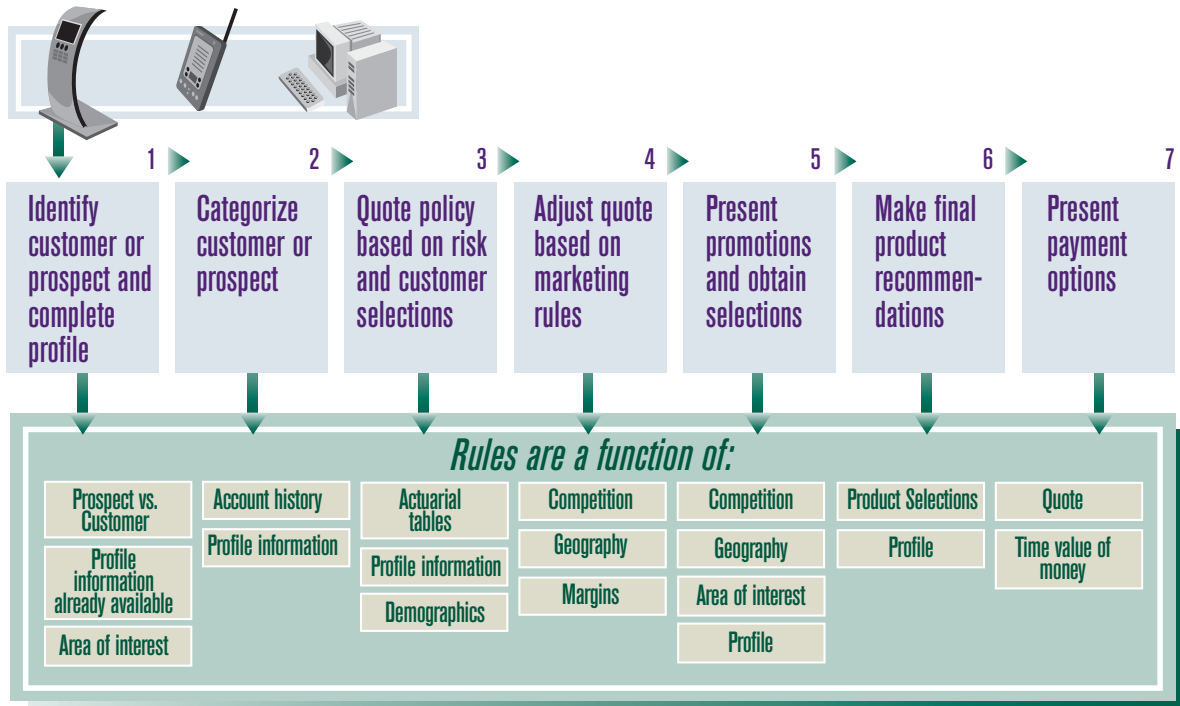
These objects are sometimes needed by the rule services to store data received and object property values calculated. As an example, a Blaze Advisor rule service deployed in support of Century Coverage’s selling processes often needs a temporary object called “policy application.” It isn’t a policy yet. It is a potential policy built up via an interactive dialog process. At the end of the dialog, and after all rules have been executed, the prospect and Century Coverage may both say “yes.” If this happens, the final step in the rule service tells Advisor to send an accepted and approved policy to the correct back office administration and billing system. This is when a permanent database record will be created. The temporary advisory “policy application” will not be stored in Advisor.

Finally, the architect and rule writers meet to review all of the rules defined in Step 1 against the object model. In particular, they identify the allowed value controls needed for each rule variable (condition or action). Some rule variables just need data type controls such as the ability to restrict data entry to integers, strings, money, or dates. But other variables require pick lists, radio button controls or checkboxes. Some of these lists are already stored in existing enterprise database tables that can be accessed as needed, while others may need to be created within Advisor. The object universe is now ready for Century Coverage to begin developing business rule services based on rule flows.

Step 4 – Construct a Visual Map of the Supported Business Processes (rule flows)

Just like diagramming a flow chart on a whiteboard, the rule writer lays out a visual map with rule flows for the various Century Coverage business processes that will be supported. Whenever an existing customer contacts Century Coverage Insurance, the first step is to complete their existing customer profile, including an update of any non-Century Coverage insurance policies. If the visitor is not a customer, then the full

Kiosk, PDA or Computer Users



A visual map of a supported business process

range of profile questions is asked while allowing for additional questions based on the visitor’s expressed interests. The second step is to categorize and score customers and prospects based on total existing business, service relationship, claims history, potential business, and simple profile. The rule writers branch quickly into the particulars of each line of insurance and each market served, but a basic set of qualification steps is used frequently in the early stages of multiple individual setting processes. The reusability inherent in rules management technology provides tremendous value here.

This visual flow map helps organize the business rules into manageable units for easy change in the future. It also helps organize the process into a sequence of steps or tasks so the rule writer can confidently write business rules that depend on intermediate results determined in a prior step.

Blaze Advisor uses a blend of declarative business rules and procedural sequence. This is an excellent way to mirror actual business thinking. There is a sequence of steps, but within each step one generally finds a set of business rules or a function. In the case of a set of business rules, execution is conditional, based on the case in hand.

Step 5 – Develop the Business Rules with Blaze Advisor

At this point, the Century Coverage rule writers possess all the necessary raw materials to develop the actual rule services: the business object definitions, allowed value controls, and process steps. Often rule development occurs on two steps, a first phase where you define at least one rule for every step in the rule flow and a second where you “templimize” the subset of the steps where the rules will be maintained using Innovator-based rule maintenance application screens.

For each step in the ruleflow, the rule writer uses the ruleset editor in Advisor Builder to enter the business rule or rules for that step. For example, the rules to classify a customer are entered declaratively in a ruleset attached to the second major ruleflow task or step we described previously. In contrast, the pricing algorithms provided by underwriting for risk-based pricing would be invoked as functions in the third rule flow step. Asking questions or launching conditionally required rule services are also options at any step in a rule flow.

Step 6 – Develop the Rule Maintenance Templates

The next step in the overall design and development process is to decide which rules should be easily changed using point-and-click Web-based rule maintenance applications. After reviewing all of the rule flow steps, it is clear that they vary widely in terms of how often the rules change and who needs to change them. Certain steps have only one or two rules that rarely change, with some requiring data lookups and assessments or calculations. Other steps use functions where algorithms might change periodically, but the rules that determine which function is called rarely change. The functions are maintained using the Builder design environment.

However, other steps have rules that change frequently. In this category, one set of rules is "owned" by the underwriting department, another by the regulatory affairs department, and others by marketing and sales. Blaze Advisor Innovator allows you to take these rules, create templates for them, and then enable non-technical people to safely edit them using intuitive rule maintenance applications. With Innovator, Century Coverage transfers responsibility for application modification, customization, and personalization from the IT department to the business analysts and managers in the various departments who are responsible for the rules.

After deciding which rules should be managed using Innovator-based rule maintenance applications, the project team must

define four things:

- 1. Century Coverage's audit requirements.** This information is collected and stored in the repository when rules are changed. The most common information is simply who changed a rule, when and why. Other information can also be collected.
- 2. Required flexibility for rule maintenance.** A rule service can be broken into one or many templates. And each template can have one or many instance files if further separation of rules is required. When rules have multiple conditional variables, it can even make sense to create a separate template for each condition so individual rules can use one or many of these conditional structures. Some rules are identical in structure but must be kept separate by state, client or branch. For these situations, you can define a single template that allows multiple instances of the rule, one for each state, client etc.
- 3. Allowed value controls that control data entry for each editable component in each rule structure template.** As discussed previously, these can be simple data entry controls based on data type, or they can be lists provided by Advisor or by existing company or even industry database tables. Often the rules reference a large number of the properties and methods of the classes in the object model. In these cases, the Business Rules Template Generation wizard can be used to automatically create code templates and allowed-value controls or "providers" automatically for the classes, methods and properties in a subset of the object model. These templates and providers will be referenced by the required rule maintenance applications.
- 4. Rule maintenance application access controls.** These define who is permitted to change which rules under which security mechanism. LDAP-based repositories work well for managing group-based access control, but JDBC repositories can work as well.

In our Century Coverage Insurance selling process example, a number of rulesets will be maintained using Innovator applications. The customer and prospect categorization rules, product recommendation rules, and promotion presentation rules are all good examples, as are the Web content and Web hyperlink dis-

play rules. Each of these rulesets will be maintained using Innovator-based rule maintenance applications. The product recommendation rules use more variables than those in any of the other rulesets. In this case the Business Rule Template Generation wizard should be run before building the rule maintenance application for this ruleset.

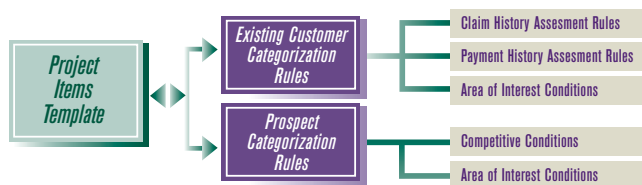
The wizard will introspect the chosen classes, with their properties, methods, and declared variables. It will then generate all the possible condition templates, action templates, value holders and providers. These will then be available for point-and-click inclusion in templates allowing a developer to quickly and precisely define how a user should be able to edit their own rules.

The wizard also automatically generates a ruleset template that integrates all these value holders and providers. When a rule maintenance application is generated for this template, the user can use a series of drop down lists to build complex rules with compound conditions and multiple actions based on the methods, properties, and declared variables of the selected subset of the object model. The templates, value-holders and providers generated by the wizard can be edited as required before generating the rule maintenance application.

Here is an example of a set of Innovator templates being used for customer and prospect categorization. Two ruleset templates each include rule condition templates that can be used or not used in creating or modifying an existing rule. Each editable rule variable (condition or action) has one or more allowed value(s) control attached to it. These allowed value controls can be set up with great flexibility. As an example, the customer and prospect categorization codes might be controlled by one or two different pull-down lists.

Step 7 – Generate the Rule Maintenance Applications

Once the rule maintenance templates with their editable components and allowed value controls have been set up, the next



Sets of templates can be used to generate suitable rule maintenance applications. Multiple instance files can be associated with some of these templates to support ASP business models where customers must have their own, customer-specific, rules.

step is to design and generate the rule maintenance pages that will be used to maintain the rules.

Blaze Advisor Innovator provides a range of options for generating rule maintenance applications. Once the templates are established, you can use the Rule Maintenance Application Generator (RMAG) to generate one or many rule maintenance applications. These are based on JSP 1.1 tags and can be polished using standard Web page development tools such as Dreamweaver from Macromedia. You can also build rule maintenance applications by hand using either the JSP 1.1 tag library or the beans provided as part of Blaze Advisor.

The generator allows you to select any "templated" rule structure in Innovator Workbench and automatically generate a complete, working rule maintenance application. This flexibility allows you to create a rule maintenance application for a subset of a rule service, a complete single rule service, or a set of related rule services. This supports your ability to create Innovator rule maintenance applications for specific functional units in an organization or for individual clients in an ASP model. The generator automatically generates the HTML pages with the JSP tags already inserted. As a result, every generated HTML field is automatically associated with its Innovator allowed value controls, and every edit, insertion, or deletion is immediately saved into the rule repository.

The generator typically generates two pages for each rule maintenance application, one for a table that displays all the existing rules in the ruleset and one for editing existing rules

and creating new ones. The generator allows you to define the application's name, location, and Innovator repository linkage as well as many stylistic and functional characteristics. In addition to defining the scope of the application, you can include banners, a table of contents, list filters, a login screen, a sign-out screen, or help. You can also design much of the basic style of the application you generate. You can define field and column labels, page text, spacing, field width, data display, font, and color. For rule variables that have multiple options and selection choices, you can select the style of data entry and display, choosing between radio buttons, check boxes, and lists.

Blaze Advisor layers the access to the template and template instance structures through a Java API at the lowest level, or at higher levels through Java Beans or JSP 1.1 tags. The Java Beans can be used by Swing, Visual Basic, Active Server Pages, or other user interfaces.

Step 8 – Test the Rule Services

Blaze Advisor provides a convenient way for the rule writer to test a rule service before hooking it up to the one or many calling applications that may need it. Within Blaze Advisor, the Century Coverage rule writer defines “named” test objects representing various test customers with various profiles using various business processes. Then, using testing features, these test customers are processed by the Blaze Advisor rule service. Blaze Advisor sends the results of the rule service execution to a console.

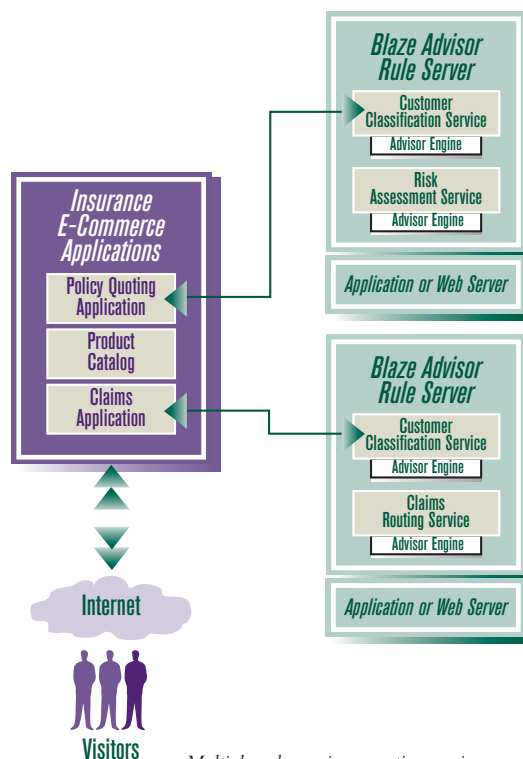
Blaze Advisor provides a wide variety of testing aids to the rule writer. These include:

- A cross-reference browser that allows you to see everything that can happen at each point in the ruleflow
- An execution browser that allows you to see exactly which rules and functions are firing in a given rule service invocation
- Breakpoints on object changes, rules, and ruleflow tasks
- Watch panels on object value changes
- Step-by-step execution control

Step 9 – Deploy the Business Rule Services

After testing a set of rule services in isolation from the rest of the Century Coverage Insurance application architecture, the set is ready for integration and deployment into Century Coverage’s personalized service applications for customers, partners, and internal visitors. Blaze Advisor Rule Server (ARS) is used to manage the loading, initializing and running of the rule services.

The Advisor Rule Server manages multiple simultaneous execution sessions for each of your rule services by assigning some number of Rule Service Agents to each service. You can specify how the server assigns agents to each service. The Advisor



Multiple rule service execution sessions can be created to handle increased load.

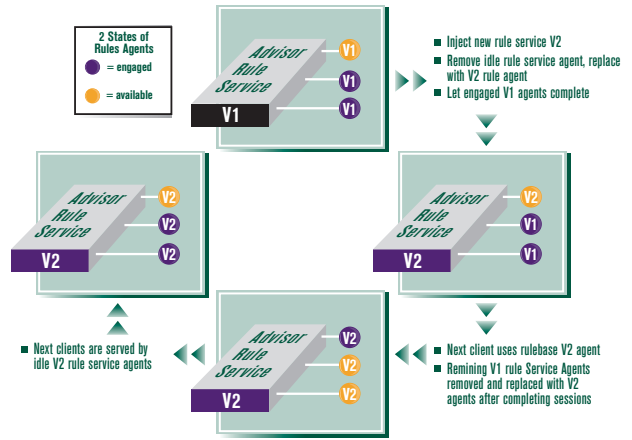
Rule Server can dynamically add or remove rule agents for a service depending on server traffic. For EJB deployments, rules can be container-managed, allowing the application server

container to manage the pooling of service agents. Rule agency policies can also be set as fixed.

Blaze Advisor comes with several Quick Deployment wizards that guide the user through configuration options and generate code configuration files, scripts, and documentation for each deployment. The wizards know what information each deployment environment needs. The EJB wizard creates a server code bean and assists in defining application server and invocation names, an application server deployment descriptor file, and an Advisor Rule Server (ARS) deployment configuration file. In contrast, the MTS/COM+ wizard generates a DLL instead of a bean and registers COM objects with the Component Services module in an IIS environment instead of providing a deployment descriptor file. The Java wizard doesn't require an application server deployment descriptor file. In all three cases, the Wizards generate an ARS deployment configuration file that handles the definition of the triggering event, object mapping, ruleflow linkage, number of rule service agents, agent management policies, and so forth.

The Quick Deployment wizards also let you specify whether the rule service is stateless, asynchronous stateful or synchronous stateful. In a stateless deployment, there is no "conversation" between the client and the rule service while the service is executing. In an asynchronous stateful deployment, the service can interact with the client by posting and handling various types of events. In a synchronous stateful deployment, the service can maintain the state of the data across multiple client interactions.

Advisor Rule Server manages rule service updates effortlessly. The ARS listens for changes and updates production rule services as required. A change in a rule service is implemented by replacing idle rule service agents with new rule service agents and allowing busy agents to complete their sessions.



Rule Services can be updated while the system is in use

Rule Technology Benefits

An independent rules technology-based architecture— one that has logic managed as an independent layer by a dedicated server within the information architecture—enables Century Coverage to deploy rule services across multiple applications and touchpoints. Domain experts within specific business units control business policies and rules without taxing IT resources and without recoding or risking conceptual miscommunication.

Century Coverage expects benefits in the following areas:

- Application development reach
- Application maintenance speed and precision
- Logic and rule reusability
- Enterprise consistency
- Personalization of the interactions with customers, prospects, agents and employees

About HNC

HNC Software Inc. (Nasdaq: HNCS) is a leading provider of Customer Insight software that enables companies to acquire, manage and retain customers. HNC's decision management and customer analytics technology analyze both structured and unstructured data so companies can derive value from all information. The company's sophisticated software is used in the telecommunications, financial services, insurance and e-commerce industries.

For more information, visit www.hnc.com

www.blazesoft.com

Headquarters 150 Almaden Blvd., San Jose, CA 95113, USA Tel: 408 817 9100 Fax: 408 535 1776
info@blazesoft.com

