KARMA: Managing Business Rules from Specification to Implementation

Jacqueline Sobieski
Fannie Mae
3900 Wisconsin Avenue
Washington, DC 20016
(202) 752-4994
Fax: (202) 752-4205
sxujas@fnma.com

Srinivas Krovvidy
Brightware, Inc.
2200 Columbia Pike, #919
Arlington, VA 22204
(415) 899-9070 (x-508)
Fax: (202) 752-4205
krovvidy@brightware.com

Colleen McClintock and Margaret Thorpe
Tangram, Inc.
1155 Connecticut Avenue, #500
Washington, DC 20036
(202) 467-8539
colleen@tangram-inc.com and margaret@tangram-inc.com

Abstract

Fannie Mae is a congressionally chartered, shareholder-owned company and the nation's largest source of conventional home mortgage funds. Fannie Mae purchases and securitizes loans and is considered the leader in the secondary mortgage market. Because of its strong leadership role, Fannie Mae's policies for loan eligibility set the standard in the mortgage industry and applying these policies consistently and effectively is critical to Fannie Mae's mission and profitability.

Fannie Mae's policies for selling and servicing mortgage loans span the business functions of the secondary mortgage market and therefore are contained in many different software applications. Managing policy across multiple business applications became increasingly complex.

To meet these demands, Fannie Mae developed KARMA (Knowledge Acquisition and Rule Management Assistant) and the Business Rule Server to allow policy changes to be implemented quickly throughout its software application environment and to provide business users with direct ownership and management of Fannie Mae's policies in a way that seamlessly integrates policy into the software applications. KARMA is designed to support the management of these policies independent of the applications in which they are embedded. KARMA generates executable business rules which become part of the Business Rule Server. As a result, policy is managed centrally and no longer embedded in multiple applications. KARMA and the Business Rule Server have been running in production supporting the Cash Delivery application since July, 1995.

Background

Fannie Mae is a congressionally chartered, shareholder-owned company that was created in 1938 to provide liquidity to the U.S. housing market. It is the largest supplier of home mortgage funds, the nation's largest corporation in terms of assets, and the second largest borrower in the capital markets, next to the U.S. Treasury. Fannie Mae's corporate mission is to provide financial products and services that increase the availability of affordable housing for low-, moderate-, and middle-income Americans. It accomplishes this mission by channeling funds between primary market lenders that originate mortgages (commercial banks, savings institutions and mortgage companies) and capital market investors that purchase securities backed by those mortgages, thus helping create the secondary mortgage market.

In order to maintain the credit quality of their portfolio and the broad acceptance of their Mortgage-Backed Securities (MBS) by capital market investors, Fannie Mae must ensure that the loans that they purchase are of the highest quality. Fannie Mae accomplishes this through the establishment of underwriting guidelines and eligibility criteria, which must be adhered to by those lenders wishing to sell loans to Fannie Mae. Fannie Mae's business policies and procedural requirements are published in the Fannie Mae Selling and Servicing Guides, which are distributed to Fannie Mae customers in both electronic and paper form. For the purposes of this paper, business policy is defined as "business principles and guidelines, considered to be expedient, prudent or advantageous that are designed to influence and determine the decisions and actions of the business". The following is an example of a business policy from the Fannie Mae Selling Guide (FannieMae 1993):

> We will now accept second homes as the security for Two-Step adjustable-rate mortgages and for fixed-rate balloon mortgages -- as long as such mortgages are not subject to an interest rate buydown plan. The maximum allowable loan-to-value ratio for these mortgages will be 80% for purchase money transactions and 70% for limited cash-out transactions.

The Representation Problem

Within Fannie Mae, the English language is the primary means of specifying and communicating these business policies. Natural language, with its heavy dependence on domain knowledge, its ambiguity and its imprecision, works well for verbal and written communications between people with similar levels of knowledge (people within the same company, the same industry, etc.). However, when these policies become very complex, or when they are being communicated and interpreted with the goal of encoding them in computer systems, they must be translated into some sort of formal specification which can be checked for completeness and logical consistency prior to implementation. The existence of this type of specification increases the speed, efficiency and accuracy with which systems can be built and maintained. Without a formal specification, it becomes difficult to ensure that the business requirements are being accurately implemented in the software. In addition, as was the case at Fannie Mae, systems maintenance becomes an unwieldy process, unable to keep up with the rapidly changing business environment.

Like many corporations, Fannie Mae's computer systems execute on a variety of different hardware platforms and operating systems. They have been developed in a number of different programming languages and access different DBMSs. They have been developed using different methodologies, and the standards and procedures by which they are maintained often differ according to environment. Many of the legacy systems running at Fannie Mae today evolved over time based on changing business demands. These systems were not necessarily designed from the ground up to do the kind of processing that they are doing today. In many cases, individual systems have sprung up to accommodate narrow slices of business functions, rather than broader, integrated applications designed to handle an entire business function. As a result, the same business policies are often implemented in multiple applications, and coded in different programming languages, with no traceability back to a common set of requirements. This creates a real maintenance problem, as it is difficult to keep the various pieces of code current and synchronized with the business requirements, and almost impossible to keep them synchronized with each other.

Project Objectives

The mission of the Business Rule Services project was to develop the tools and techniques necessary to address the problems inherent in the representation of business policy at Fannie Mae. Specifically, we set out to accomplish the following:

I. Define a simple, English-like specification language for specifying business policies.
II. Create a single, shared repository where specified business policies can be stored, updated and accessed.
III. Give business users the ability to control and manage the specification and implementation of business policies.
IV. Eliminate the need for human interpretation and translation of specified business policies into executable code for implementation in computer systems.
V. Reuse Fannie Mae business policies reusable across multiple applications, and ensure that they are implemented consistently.

These objectives were accomplished through the development of the following set of related application components:

I. **Business Rule specification language** - a grammar-based representation for the specification of business policies
II. **Knowledge Acquisition & Rule Management Assistant (KARMA)** - a policy management application consisting of the following:
   - a GUI through which policy specifications can be defined and queried,

- a set of databases containing the policy specifications, their underlying models and all related metadata,
- a code generation component, which generates executable code directly from the policy specifications.

III. **Business Rule Server** - a knowledge base which provides software applications, acting as clients, with executable policy knowledge. KARMA generates the ART-IM rules which are executed in the Business Rule Server.

IV. **Data Translator** - a tool that enables sharing of data across software applications through the mapping of application data models to the data model upon which the business rules are based, the business object model.

Project Significance

Several AI applications have been successfully deployed in the mortgage industry over the past few years. The CLUES system (Talebzadeh et al. 1994) focused on automating the underwriting process. In CLUES, business policy is embedded within the knowledge base rules. GECCO (Bynum et al. 1995), is an automated compliance checker which checks loans against investor guidelines. This compliance checking is done at different stages in the mortgage loan processing pipeline. GECCO enabled different applications to use the same business policy by embedding the GECCO knowledge base in different applications. Our project differs from the above efforts in the following ways:

- **High-Level Knowledge Representation**

  Business policy is modeled as business rules in an English-like specification language that can be understood by business users. KARMA's business rule language is general enough to represent business rules in any policy related domain.

- **Knowledge Acquisition Tool**

  A knowledge acquisition tool, KARMA, was developed to define and manage these business rules, giving business users direct access to business policy implemented in the computer systems.

- **Automatic Code Generation**

  Executable business rules are automatically generated by KARMA from the business rule specification language.

- **Knowledge Server**

  Business Rules execute in a Business Rule Server which different applications, acting as clients to the Business Rule Server, can access.

- **Data Mapping**

  A data translation capability was developed from which data model translation code is generated from a high-level specification language to enable client applications to access the Business Rule Server regardless of their differing data models.

Overview of Application Components

These components, as illustrated in Figure 1, work together to provide an effective means of specifying and implementing Fannie Mae business policies. Business policies are conceived of and validated by the responsible business persons. They are then specified in the form of textual business rules using the KARMA Rule Editor. KARMA stores the logical representation of the business rule in the Business Rule database. Both the textual and the logical representation are based on an underlying data model which must first be defined through the KARMA Data Dictionary Editor, and stored in the Data Dictionary database. KARMA uses object-oriented and database technology to facilitate defining and managing business rules and uses AI technology to perform consistency checking on business rules and to generate executable business rules which become part of the Business Rule Server. The Business Rule Server is an ART-IM knowledge base which processes requests from client applications to execute the business rules. Since the client applications requiring access to the Business Rule Server may have different data models, the Data Translator translates the client application data into the data model used by the Business Rule Server. The following sections describe each application component in more detail.

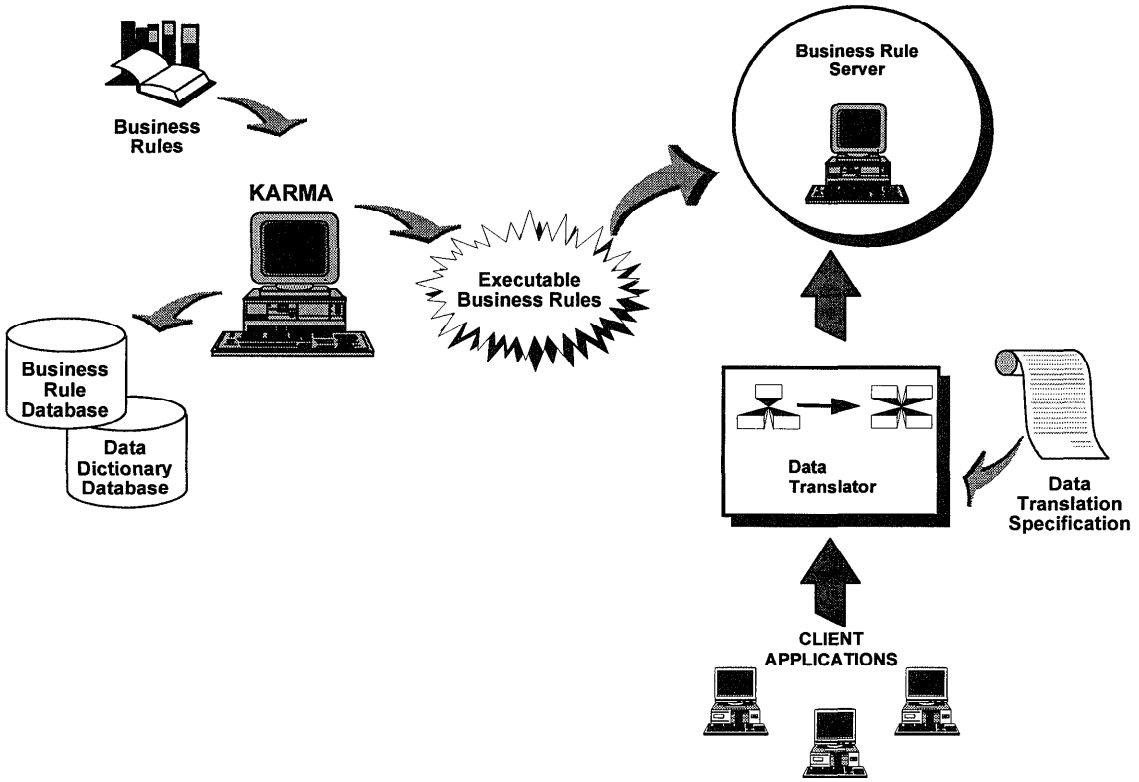

Figure 1. Application Component Overview

Business Rule Specification Language

The term "business rule" has become very popular over the last several years, particularly in the database and application development tool sectors of the software industry. It means different things to different people, but can be broadly defined as "an explicit statement stipulating a condition that must exist in a business information environment for information extracted from that environment to be consistent with business policy" (Appleton 1988). Business rules are usually described as discrete and atomic, implying that they represent the smallest units of business policy - that they cannot be broken down any further without losing their meaning. In the absence of a more precise industry definition, the term has been used to refer to everything from entity-relationship and attribute-domain constraints (which are traditional components of data models), to inference rules. There are a few researchers proposing business rule formalisms and categorization schemes, but it appears that more work will need to be done in this area before a common classification scheme will be complete and theoretically sound enough to gain general acceptance.

The term "business rule", as it is used within the context of this paper, actually refers to a very specific type of *fact constraint* - a *declarative* sentence that places restrictions on the relationships between people, places and things. These business rules do not include the simple data integrity constraints that are represented in traditional data models, instead they consist of the more complex and dynamic conditional business restrictions that are typically coded in computer programs. We created a specification language with a restricted vocabulary and relatively simple structures to precisely describe these business rules. In this artificial language, business rules consist of left-hand side and right-hand side clauses. Business rules may have one or more clauses ANDed together on the left-hand side but may only have a single clause on the right-hand side. This right-hand side clause restricts the value of a single attribute when the left-hand side conditions are satisfied. Therefore, business rules are represented as:

```
IF <clause>
AND <clause> .....
THEN <clause>
```

Where these clauses are of the following forms:

```
<Attribute> <Operator> <Attribute>
<Attribute> <Operator> <Value>
<Attribute> <Operator> <Attribute List>
<Attribute> <Operator> <Value List>
<Object> <Operator>
```

For example, in the business rule:

```
IF    Lien Type is Second Mortgage
THEN  Occupancy Status must be Principal
      Residence
```

The clause "Lien Type is Second Mortgage" is an `<Attribute> <Operator> <Value>` clause (Lien Type, is, Second Mortgage). The clause "Occupancy Status must be Principal Residence" is also an `<Attribute> <Operator> <Value>` clause.

The structure of the business rule lends itself naturally to knowledge representation as a production rule in a data-driven rule-based system and in fact, that is how the executable version of the business rules are represented. The Business Rule Server section describes this in more detail.

KARMA

KARMA is a policy management application developed to support the collection, analysis and implementation of business rules at Fannie Mae. KARMA has three main components: the Data Dictionary Editor, Rule Editor, and the Rule Browser. The objects and attributes available to create business rules are defined using the KARMA Data Dictionary Editor. Using the KARMA Rule Editor, business policy is formally specified in business rules using the English-like syntax described above. Rules defined in KARMA can be queried and browsed through the KARMA Rule Browser.

KARMA Data Dictionary Editor. The business object model is defined using the KARMA Data Dictionary Editor, shown in Figure 2. The user first defines an object, giving the object a name, and providing the business definition for the object. Any number of attributes can be defined for an object. For each attribute the user selects a data type, and provides the name and definition for the attribute. For enumerated data types, the user must define the set of enumerated values. The business object model defined using the Data Dictionary Editor is stored in a relational database, the Data Dictionary database. The objects, attributes and values stored in the Data Dictionary database are then available to be used in defining rules in the KARMA Rule Editor.

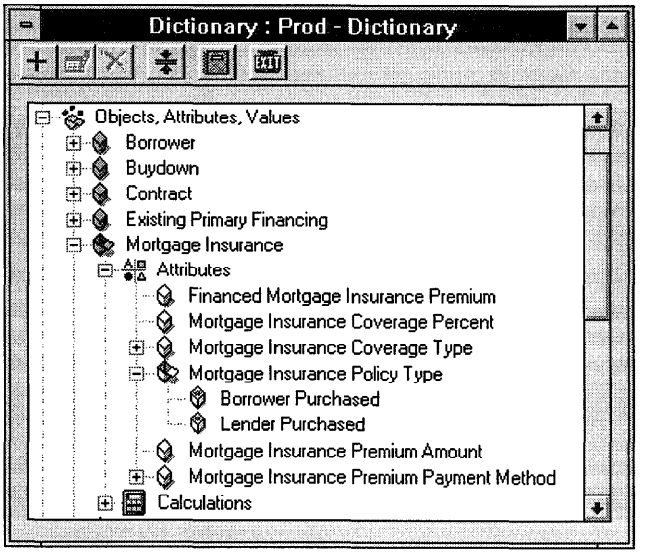

Figure 2. KARMA Data Dictionary Editor

KARMA Rule Editor. The KARMA Rule Editor, shown in Figure 3, allows users to define new rules and modify existing rules or rule properties. Once defined, rules are stored in the local MS-ACCESS Business Rule database or in the shared SYBASE Business Rule database. This allows users to keep a local copy of the Business Rules and Data Dictionary databases to work with during knowledge acquisition until they are ready to update the master Business Rules and Data Dictionary databases.

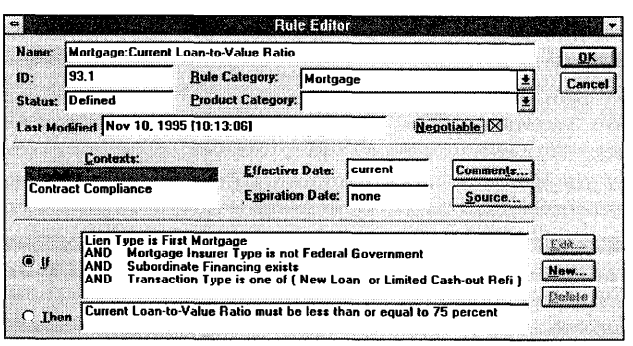

Figure 3. KARMA Rule Editor

The rule clauses are defined or modified through the KARMA Clause Editor, shown in Figure 4. The Clause Editor steps the user through the process of defining a rule clause by displaying only valid selections in the hierarchical list box control. A clause is defined to have the following structure `<Operand> <Operator> <Operand List>`. Valid selections are determined by the data type of the first operand selected. For each data type, a list of valid operators is defined in the Data Dictionary database. The valid operators for a data type are then made available in the KARMA Clause Editor when the first operand of a

clause is selected. Once an operator is selected, the second operand or operand list of the clause is restricted by this operator. Multiple clauses can be defined for any given business rule.

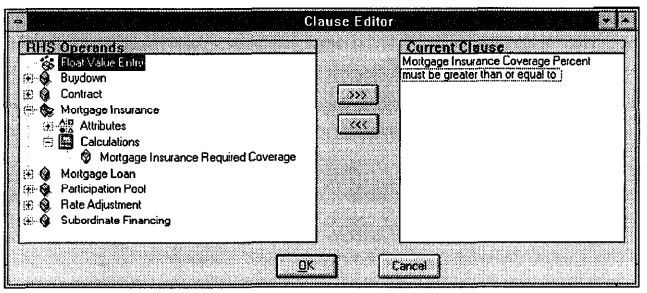

Figure 4. KARMA Clause Editor

KARMA Rule Browser. The KARMA Rule Browser, shown in Figure 5, displays all the rules defined in KARMA. Users can scroll through the rules defined in the Business Rule database. The rule text for the selected rule is displayed in the lower section of the Rule Browser. From the Rule Browser users can invoke the Rule Query capability which allows the user to specify criteria for which matching rules will be displayed in the Rule Browser. Rules displayed in the Rule Browser can also be sorted and printed.

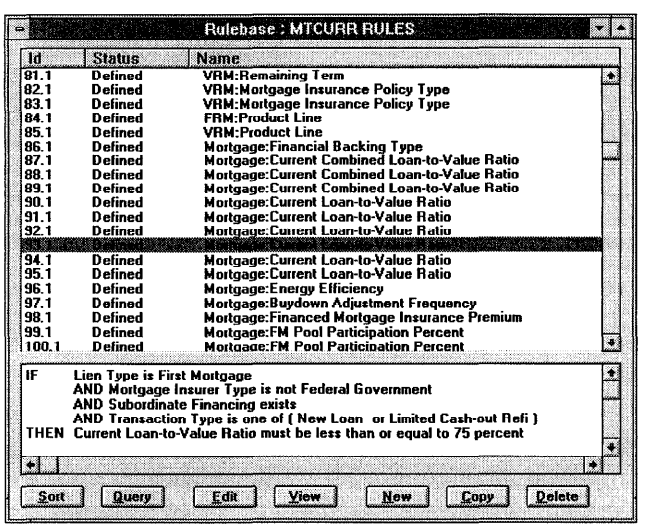

Figure 5. KARMA Rule Browser

Consistency Checking in KARMA. In order to fully support the business users in defining high-quality business rules, KARMA must ensure not only that the rule has a valid syntax but that it is consistent with other rules in the Business Rule database. These requirements are met through the GUI, which restricts users to creating rules using valid syntax, and the consistency checking component, which keeps users from defining rules which are inconsistent with rules that have already been defined.

KARMA's consistency checking implementation assumes that no nested conditions exist in the rules and that the consequents have only one literal and the antecedents have multiple literals with an "AND" connector. A unification based algorithm is used to perform consistency checking. The consistency checking capability identifies the following relationships among business rules: inferred rules, redundant rules, conflicting rules, and subsumed rules (Polat & Guvenir 1993).

For the verification of the rules, the comparison of the clauses is the primary operation. Let `C->lhs` be the left-hand side operand in the clause `C`, `C->op` be the operator in the clause `C` and `C-rhs` be the list of right-hand side operands in the clause `C`. Comparing two clauses C_1 and C_2 yields the following results with the respective substitution lists:

1. $C_1 \equiv C_2$ if { C_1->lhs = C_2->lhs; C_1->op = C_2->op; C_1->rhs = C_2->rhs } SUB_LIST = {} or
   if { C_1->lhs = C_2->lhs; C_1->op = C_2->op; C_1->rhs ≠ C_2->rhs } SUB_LIST = UNIFY(C_1->rhs = C_2- >rhs)

2. C1 ≡ ~C2 if { C1->lhs = C2->lhs; C1->op = ~(C2->op); C1->rhs = C2->rhs } SUB_LIST = {} or
   if { C1->lhs = C2->lhs; C1->op = ~(C2->op); C1->rhs ≠ C2->rhs } SUB_LIST = UNIFY(C1->rhs = C2- >rhs)

3. C1 ⊂ C2 if { C1->lhs = C2->lhs, C1->op ⊆ C2->op; C1->rhs ⊆ C2->rhs } SUB_LIST = UNIFY{C1->rhs ⊆ C2->rhs } or
   if { C1->lhs = C2->lhs, C1->op = C2->op; C1->rhs ⊆ C2->rhs } SUB_LIST = UNIFY{C1->rhs ⊆ C2->rhs } or
   if { C1->lhs = C2->lhs, C1->op ⊆ C2->op; C1->rhs = C2->rhs } SUB_LIST = {}

4. C1 ≠ C2

In each case, if SUB_LIST ≠ {}, then that list must consist of a consistent set of substitutions for each variable. Based on these relationships between clauses, any two rules R_i and R_j are compared as follows:

1. **IF** the right-hand side clause of Ri ≡ right-hand side clause of Rj with a consistent substitution list for unifying all the clauses on their left-hand sides **THEN** Ri and Rj are redundant.

2. **IF** the right-hand side clause of R_i ≡ ~right-hand side clause of R_j with a consistent substitution list for

unifying all the clauses on their left-hand sides **THEN** R_i and R_j are conflicting.

3. **IF** the right-hand side clause of $R_i \equiv$ right-hand side clause of R_j with a consistent substitution list for subsuming R_i's left-hand side clauses with those of R_j **THEN** R_i subsumes R_j.

4. **IF** the right-hand side clause of $R_j \equiv$ right-hand side clause of R_i with a consistent substitution list for subsuming R_j's left-hand side clauses with those of R_i **THEN** R_j subsumes R_i.

The left-hand side of a rule R_i subsumes that of R_j in the following cases:

- All the clauses in the left side of R_i have equivalent clauses in R_j and R_i has at least one more clause than R_j on its left side.
- At least one clause from the left-hand side of R_i subsumes those of R_j and the rest of the clauses from the left-hand side of R_i have equivalent clauses in the left-hand side of R_j.

Consistency checking in KARMA is implemented based on the above ideas. The results from the consistency checking have proven to be very valuable in the knowledge base verification. The output from KARMA consistency checking is shown in Figure 6.

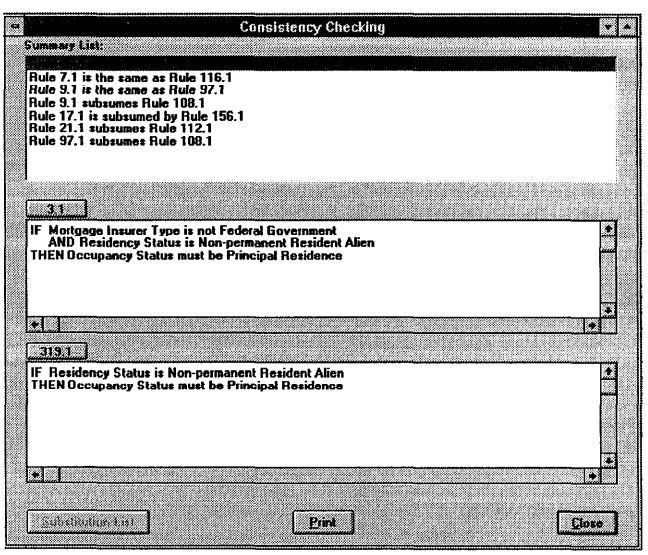

Figure 6. KARMA Consistency Checking

At the top of the Consistency Checking window, the summary results are displayed. The two rules for the selected result are displayed below the summary list.

Code Generation in KARMA. KARMA generates executable ART-IM rules for the Business Rule Server from the business rule representation stored in the Business Rule database. All code which is dependent upon the business rules is generated by KARMA. This means that there is no manual maintenance necessary for any application using the Business Rule Server when business policy changes.

In terms of rule generation, the business rule representation stored in the Business Rule database is the source and the knowledge base rule representation is the target; therefore, rules are generated in the form expected by the knowledge base from the database representation. In the knowledge base, rules are not represented as clauses. Rather, rules are represented as patterns on the left-hand side of the rule and actions on the right-hand side of the rule. In the case of executable business rules for the Business Rule Server, the right-hand side action is to create a violation message representing a policy violation.

The left-hand side patterns are simply conditions which are evaluated against the loan data. If all the conditions of the left-hand side are met, the rule fires and the violation message is created. For the rule:

```
IF    Lien Type is Second Mortgage
THEN  Occupancy Status must be Principal
      Residence
```

a violation of policy occurs when the Lien Type is Second Mortgage, and Occupancy Status is not Principal Residence. Notice that the THEN (right-hand side) clause of the business rule specification representation must be negated in the knowledge base representation when testing for a violation.

The ART-IM representation of the above rule is:

```
(defrule charter:occupancy-status-7-1
 (declare (salience 100 ))
 (schema ?mortgage-loan
    (instance-of mortgage-loan )
    (lien-type  ?lien-type &: (= ?Lien-Type
secondary-mortgage))
    )
 (schema ?property
    (instance-of property )
    (occupancy-status  ?occupancy-status
&:(NOT (= ?Occupancy-Status
              principal-residence)))
         )
  =>
 (generate-violation-message occupancy-
status-7-1)
 )
```

Rules are generated through the use of an intermediate representation (IR). A variant of BNF formalism is used to specify this IR. The variant is achieved by imposing certain restrictions to reduce the complexity of BNF. The use of an intermediate representation provides the flexibility to generate rules in any target language (not just ART-IM rules). Since ART-IM rules are generated through the IR, the knowledge base can be dealt with on a syntactic level allowing the business rules to be abstractly

developed based on their knowledge composition rather that their detailed textual structure.

The IR is composed of three types of constructs:

- **Lexical Nodes**

  Lexical Nodes are atomic (they cannot be decomposed). They describe the syntactic elements on a character-by-character basis.

- **Repetition Nodes**

  Repetition Nodes are list nodes which specify one or more occurrences of a node of any type.

- **Construction Nodes**

  Construction Nodes are nodes which are composed of a fixed number of other nodes which may be lexical nodes, repetition nodes, or construction nodes.

The IR is defined using a grammar containing these three types of nodes, as shown in Figure 7. The IR contains the components needed to build the target business rules along with their associated unparsing schemes. Generation of the rules from the intermediate representation to the knowledge base representation is achieved by unparsing the intermediate representation. For example, at the highest level a business rule is specified in the IR as:

```
<eligibility_rule>: "(" <ruleheader> "\n"
<left_side> "\n=>\n" <right_side> ")"
```

So, an eligibility rule is composed of a `<ruleheader>`, a `<left_side>`, and a `<right_side>`. This is a construction node composed of three constructs. The character strings defined within quotes are terminal symbols, the other constructs are non-terminal symbols. During code generation, the strings are emitted before visiting each non-terminal node.

These constructs are defined at the next lowest level as:

```
<ruleheader> : "(defrule" <name>
"/n(salience " <salience>
")/n"
<left_side>: <pattern_list>

<right_side>: "(generate-violation-
message" ")"
```

The `<ruleheader>` is a construction node with two components, `<name>` and `<salience>`. The `<right_side>` is a lexical node. The `<left_side>` is a repetition node composed of several patterns.

The IR is implemented as a set of C++ classes representing the IR nodes. All IR classes are subclasses of the lexical, repetition, and construction node classes. During rule generation, the database representation is used to construct the IR classes. Once the IR classes are constructed, rules are generated by unparsing the IR nodes (Krovvidy & Wee 1988).

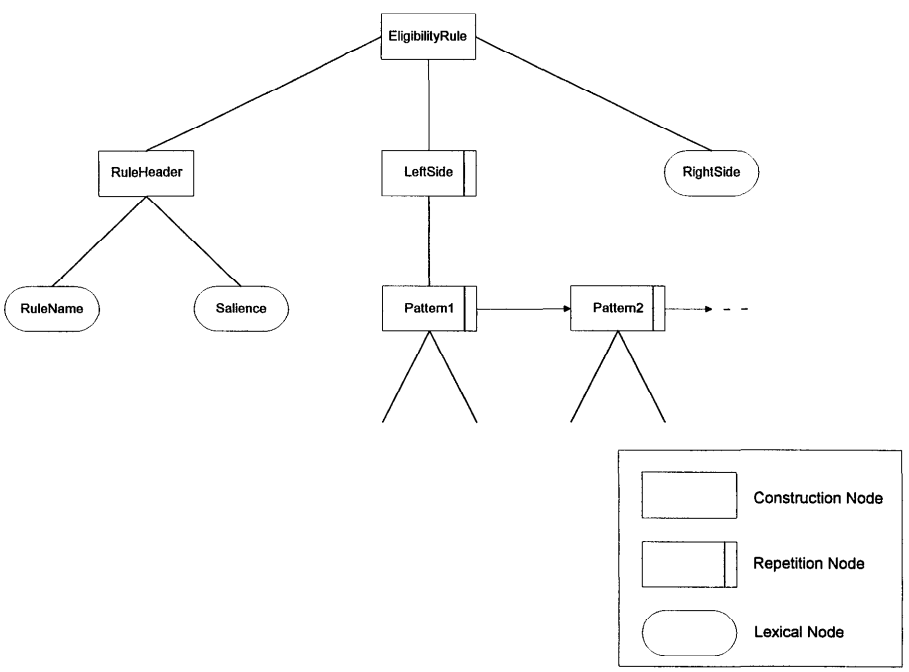

Figure 7. Intermediate Representation for Business Rules

Business Rule Server

The Business Rule Server is a client/server application capable of servicing multiple client applications simultaneously. The Business Rule Server makes Fannie Mae business policy available to applications as a service. Applications no longer need to contain these policies but instead can request access to them as a service from the Business Rule Server. Any application requiring the use of Fannie Mae business policy for compliance checking can send the loan data to the server with a request for loan validation. The server checks the loan for compliance with Fannie Mae policy and returns policy violations to the application. The complete compliance checking takes less than 0.5 seconds including network time.

The Business Rule Server consists of control structures written in ART-IM and C, an RPC based API developed using ONC RPC, executable business rules generated by KARMA, and data translation code generated by the Data Translator (explained in the next section). In order to request the loan validation service from the server, the client application first obtains the loan data to be passed to the server. Since the Business Rule Server only contains Fannie Mae business policy rules, data integrity checking and other system related editing must be performed by the client application before making a request to the Business Rule Server.

The API to the Business Rule Server is provided as a C library to the client application. Loan data is loaded into the data structures using the provided accessor functions and the server is invoked with an API call. Once the server receives the request and data, it translates the client application's source data into the data model defined in the KARMA Data Dictionary database (upon which the business rules are based). This translation is accomplished using the Data Translator. Translated data is mapped into the knowledge base along with the validate loan request. Inside the knowledge base all business rules are evaluated. Violated business rules produce violation messages which are returned to the client in a data structure. The client application can then retrieve violation messages using the provided accessor functions and process those violations.

Data Translator

The Data Translator was developed to enable applications with dissimilar data models to share data with the Business Rule Server, without modifying their data models. The Data Translator allows the Business Rule Server to be independent of data sources. There are many valid reasons why an application may not be able to modify their data model to match the data model upon which a particular set of business rules is based. Using the Data Translator, any application regardless of how its data is represented can request the services of the Business Rule Server. To facilitate a uniform access to the rule server, a translation layer was introduced between the client and the server and a data translation language was developed. This language is used by the client application team to specify the mapping of the data from their data model to the business rules data model. From these specifications, the Data Translator generates the required translation code. The data translation specification language currently supports the following features: object- based data models, conditional statements, assignment statements, multiple instances of objects, date arithmetic and local variables for complex mappings.

Data Translation Language (DTL) is defined using a context-free grammar. A complete set of DTL specifications consist of a source data model, a target data model and one or more translation rules to map the data from the source data model to the target data model. A sample data translation rule is specified as:

```
RefinanceRule
  {IF
    (ANY S-SpecialFeature:SpecialFeatureCode ==
    "refinance")
  THEN T-Mortgage_Loan:Refinance_Option =
    "Yes"};
```

The name of this rule is "RefinanceRule". The prefix S- corresponds to the objects from the source model and T- corresponds to the objects from the target model. This rule specifies that if there is an instance of the object SpecialFeature whose attribute SpecialFeatureCode has a value of "refinance", then the attribute "Refinance_Option" of the target object MortgageLoan is assigned a value of "Yes". These translation rules can include very complex logic. The code generation module of the Data Translator uses the same techniques as the generation of the executable business rules.

Application Development, Deployment and Maintenance

Development

Initially, the Business Rule Server and Rule Editor components of KARMA were prototyped in KBMS, an expert system shell by Trinzic Corporation (which has since been acquired by Platinum Technology). The English-like rule language of KBMS was well suited for business policy rules. At the end of 1993, an in-depth analysis of expert system shells on the market was performed. ART-IM by Inference Corporation (now Brightware) was selected for its powerful pattern matching capabilities and its embeddability. The Business Rule Server was prototyped as an ART-IM knowledge base imbedded in a C application during the first quarter of 1994. Following the prototype, Fannie Mae's Cash

Delivery application was selected as the first client application of the Business Rule Server because it was undergoing a major re-write.

The Cash Delivery application receives loans which Fannie Mae will purchase and hold in their portfolio. During the process of receiving the loans, known as the delivery process, loans are checked for compliance with Fannie Mae policy and contractual obligations. The business rules related to this process are applied by the Business Rule Server. The Cash Delivery application acts as a client application to the Business Rule Server, sending loans to the Business Rule Server for validation.

Knowledge Acquisition for the Cash Delivery application, during which all of the Fannie Mae business rules pertaining to cash delivery were acquired, took approximately six months with two full-time knowledge engineers and significant support of the business experts. During this Knowledge Acquisition phase, development on the Business Rule Server commenced. The Business Rule Server was developed using ART-IM and C under Solaris. The RPC capability was developed using ONC RPC. Development time for the Business Rule Server was approximately 15 months with two developers.

In July of 1994, a three month prototype of KARMA was completed using ART*Enterprise (A*E). Although A*E was valuable for rapid prototyping, the resource requirements and performance problems encountered with the early version used for the prototype prohibited using it for developing KARMA. Furthermore, the A*E rule language was inappropriate for the application, since KARMA is an event driven rather than a data driven system. Since KARMA is a procedural object-oriented application, Microsoft C++ under Windows NT was selected as the development environment for the production version. The Data Translator was also developed in Visual C++ under Windows NT using MKS Lex and YACC. KARMA and the Data Translator were developed by one full-time developer and one part-time developer over an 18 month period.

Deployment

Prior to production implementation, the Business Rule Server was tested by a dedicated testing team. The team consisted of three business users and one technical representative from the Business Rules team. Approximately 2600 test cases were created to test the 400+ business rules in the Business Rule Server. These test cases were carefully hand-crafted to test each business rule and the interdependencies among the business rules. Preparing the test cases took three business analysts two months to complete. Following the preparation of the test cases, three testing cycles were performed in which all cases were executed and all resulting problems were fixed. Testing was conducted over a three month period. During testing, the business analysts began developing a process for managing business rules through KARMA. KARMA introduced a powerful new capability that required new procedures to create a streamlined policy management process that could allow changes that previously had taken months to be implemented in days.

Following this testing, the Business Rule Server was moved to a production environment along with the Cash Delivery application to run in parallel with the old cash delivery application. This parallel production run lasted for approximately four months during which extensive analysis was performed to determine the impact of the Business Rule Server on cash loan purchasing. For example, would the Business Rule Server apply policy more strictly than the old cash delivery system? If so, were these good risk decisions or was the policy implemented too restrictively? Business users were presented an abundance of information about the loans that Fannie Mae was purchasing and specific reasons for those it chose to reject. KARMA and the Business Rule Server have had an important impact by providing business users with timely information combined with the ability to quickly react and adjust constraints to optimize business decision making.

The Cash Delivery application and Business Rule Server have been running in production since July of 1995. KARMA is being used to maintain all business policy related to cash purchasing.

Maintenance

KARMA was designed to enable quick decision-making related to policy. One of its key benefits is the ease with which the business rules can be maintained in the production Business Rule Server. All domain specific code for the Business Rule Server and the client API library is generated by KARMA and the Data Translator. In legacy systems, the turnaround time for implementing new policy frequently takes several months because so many different systems are impacted by a single business change. KARMA has accelerated the process to a maximum of several days for the Cash Delivery application. Business rules can actually be modified and re-generated in minutes; however, the production migration process can take several days. During the parallel production run, eight different rule changes were required and all were implemented in production in under three days. This benefit will be realized over and over again as new client applications use the Business Rule Server. Policy changes will be made in one place and become available to all impacted client applications simultaneously.

KARMA, the Data Translator, and the control structures of the Business Rule Server are maintained by the development team. Currently we are enhancing KARMA by adding more rule management capabilities and extending the rule language to provide additional language

features. We are also preparing to support more client applications. As we acquire rules for these new domains, we are finding that many rules already existing in the Business Rule Server will be reused by these client applications.

Application Use and Payoff

The Business Rule Server is currently being used by a single Fannie Mae application, the Cash Delivery application. Use of the Business Rule Server has had an immediate and significant impact by improving the quality of information available to resolve policy issues for loans submitted to Fannie Mae for its Cash Portfolio business. The Business Rule Server has been processing an average of 1000 mortgage loans per day since it was implemented in production. Although the majority of the loans do not generate messages indicating policy violations, a significant portion of the loans do. These loans require special handling to review and resolve these policy violations in order to determine if Fannie Mae will purchase these loans. This review process is tedious and labor-intensive. The quality of the information supplied by the Business Rule Server has significantly aided this review process and is already resulting in reduced operational costs. In the future, use of the information provided by the Business Rule Server is expected to result in additional revenue for Fannie Mae.

In supporting Fannie Mae's Cash Delivery application, the Business Rule Server and KARMA have already provided Fannie Mae with important benefits, but the real payoff will result when other applications begin using these tools. Several strategic applications at Fannie Mae that require the use of policy information are currently preparing to use KARMA and the Business Rule Server. Without KARMA and the Business Rule Server, each application would have to develop and code to do their own policy checking. This code would be embedded in each application and therefore inaccessible to other applications. Policies in all these applications would need to be updated and maintained redundantly. The result would be high maintenance costs, the potential for inconsistent implementations of the same policies, and slower response to changes in the mortgage industry.

The Business Rule Server and KARMA not only eliminate these redundancies, but also provide new development projects with reduced development costs. New projects will require 70-80% less funding to develop the policy component of their application. Most of the costs associated with using KARMA and the Business Rule Server will be dedicated to knowledge acquisition to acquire new rules for the application (if they are not already available in the Business Rule Server). With KARMA and the Business Rule Server, business users can devote their resources to crafting the business policies rather than planning around lengthy implementations.

Lastly, KARMA is providing a powerful long-term benefit by making policy information in the form of business rules clear and unambiguous, easily modifiable and, most importantly, accessible to the business users. Business users can see exactly what business rules are currently implemented by querying KARMA from their desktops. They can also perform "what if" analysis to determine the impact of proposed policy changes as well as trend analysis to review the performance and impact of policies on loans that Fannie Mae has already purchased. The benefits to business users will accelerate as more business rules are acquired and defined in KARMA and new opportunities for using this knowledge emerge.

In summary, Fannie Mae is already finding its competitive position enhanced by using the Business Rule Server and KARMA. Fannie Mae can now respond quickly and efficiently to the changing economic conditions that are so prevalent in the mortgage industry today. Their policies can be easily modified and implemented to keep pace with new product developments and to proactively seek additional investment opportunities.

Acknowledgments

In addition to the authors, many individuals contributed to the successful development and deployment of KARMA and the Business Rule Server. The authors want to extend a special gratitude to Andrew Weiss for his continued support and guidance since the inception of this project. They also want to thank Peter Kopperman for directing this project and Bill Tucker, Brian Pannell and Paula Marlowe for their efforts in testing KARMA and the Business Rule Server. Finally, the authors want to express their thanks to Cathy Doman, Raza Hashim, Greg Close and Pete Silvestre for their ideas and development efforts and Carol Borchardt for providing the domain knowledge.

References

Appleton, D.S. 1988. Second Generation Languages. *Database Programming & Design* February 1988:48-54.

Bynum, S.; Noble, R.; Todd, C.; and Bloom, B. 1995. The GE Compliance Checker: A Generic Tool for Assessing Mortgage Loan Resale Requirements. In Proceedings of the Seventh Innovative Applications of Artificial Intelligence Conference, 29-40. Menlo Park, Calif: American Association for Artificial Intelligence.

Fannie Mae Selling Guide. 1993. Fannie Mae, Washington, DC.

Krovvidy, S., and Wee, W.G. 1988. Retargetable rule generation for expert systems. In Proceedings of the third international symposium on methodologies for intelligent systems, colloquia program 37-46.

Polat, F., and Guvenir, H.A.1993. UVT: A Unification-Based Tool for Knowledge Base Verification. *IEEE Expert* June 1993:69-75.

Talebzadeh, H.; Mandutianu, S.; Winner, C.F.; and Crane, L. 1994. Countrywide Loan Underwriting Expert System. In Proceedings of the Sixth Innovative Applications of Artificial Intelligence Conference, 141-152. Menlo Park, Calif: American Association for Artificial Intelligence.