International Journal on Advanced Science Engineering Information Technology

Vol.7 (2017) No. 6
ISSN: 2088-5334

Design and Implementation of a Software System for High Level Business Rules

Deepak Kumar Sharma[#], Naveen Prakash[*], Manish Mahajan[+] Dheerendra Singh[^]

[#]*Research Scholar, Department of Computer Science and Engineering, IKG PTU, Kapurthala, Punjab, India.*
Email: masterdeepak12@gmail.com

[*]*ICLC, 21/4, S. Bhagat Singh Marg, New Delhi 110001, India,*
Email: praknav@hotmail.com

[+] *Department of Computer Science and Engineering, CGC, Landran, Chandigarh, India*
Emai: manishmahajan4u@gmail.com

[^]*Department of Computer Science and Engineering, CCET, Chandigarh, India,*
Email: professordsingh@gmail.com

***Abstract*— The Business Rules Group has highlighted the importance of the ownership of business rules by business people. This calls for a business oriented view of business rules. Accordingly, we propose to introduce a Business Layer on top of the CIM layer of business rules that considers the essential nature of business rules, their properties and structure as well as inter-relationships between business rules. We propose a model that inhabits the business layer. This model provides (a) flat and hierarchical business rules, (b) business rules that operate on the state of an enterprise and cause state changes (c) temporal constraints and specification of long running and instantaneous business rules. Further, we develop a Business Rule Management system(BRMS) that, besides basic CRUD capability, allows construction of business rules from given ones. Our proposals are exemplified with a subset of the business rules of a Library.**

***Keywords*— business rule; business rule model; business motivation model; model driven architecture; temporal operator.**

I. INTRODUCTION

Business modeling supports the discovery of system requirements by helping the analysis team to perceive the wider business context in which the system To-Be will operate. Out of the six scenarios of business modelling identified by Kruchten [1] our interest is in the domain-modelling scenario, that is, in performing business modelling during domain analysis. Business modelling can be considered to yield high-level requirements that are elaborated to yield system requirements.

Now, according to the object-oriented approach of [2], [3] business use case diagrams, business activity diagrams, and state machines may be used in business modeling. During business modeling, the analyst may come across business rules or constraints on how functions must be performed by the system. These are treated as annotations to use cases and as 'guard conditions' in activity diagrams and state machines. Evidently, this approach emphasizes use cases, activity diagrams and state machines while treating business rules as secondary to these.

There is however another view [4], [5], [6] that gives relatively greater importance to business rules and treats these as providing an expression of system requirements. This perspective is exemplified by the Business Rules Manifesto [6], the Business Motivation Model [7] and SBVR [8]. The Business Rules Manifesto issues a call to give primacy to business rules. Articles 1.1 and 1.2 of the Manifesto say, "Rules are first-class citizens of the requirements world" and "Rules are essential for, and a discrete part of, business models and technology models" respectively. Putting these together, we conclude that the Business Rules Manifesto would be satisfied if we could develop a business oriented business rules model that treats a business rule as a first-class concept of the model. The Business Motivation Model treats business rules as

directives and leaves it to SBVR to define them. SBVR expresses rues in the extended first order logic and is position at the CIM layer of OMG MDA.

However, the question remains as to whether a CIM level proposal is the one that business people would be comfortable with. To address this, we looked at business rules as organized in the three-layered architecture of MDA [9], [7] shown in Fig.1. Business rules at the CIM layer are an abstraction of business rules, RuleML [10], R2ML [11], etc. of the PIM layer. However, the three proposals of the CIM layer of Fig. 1 do not perform an ab-initio examination of business rules and do not ask questions like

- What is the essential nature of business rules in a business?
- What is their structure?
- What are their properties?
- What are the inter-relationships between business rules?

We need a model that should answer these question and reflects business rules as they really are. **The aim of our work is to propose a model for business rules from the business perspective and develop a technique to arrive at system requirements from business rules of this model. In this paper, we address the first part, that of developing a model.** We refer to this model as Business Rules Oriented Business Model (BROBM).

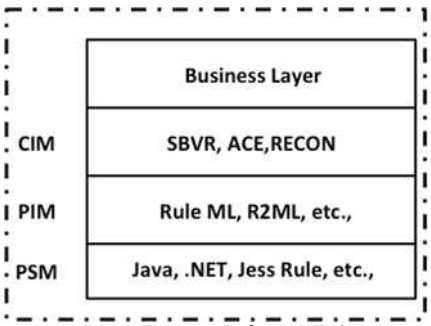

Fig. 1 Business Rules in MDA

Evidently, we need to investigate the world of business to provide us our model. It is for this reason that we propose a fourth layer, the business layer, on top of the CIM layer where we address these questions. Further, we develop a BRMS, a business rule management system, for business rules expressed in this model.

A. Desiderata of BROBM

Now, as mentioned earlier, we need to look into the world of business to obtain the concepts of our model. For this, we revisit the Business Motivation Model to extract the main concepts around business rules. This provides to us a list of desired features that business rules should have from the business perspective.

The Business Motivation Model, BMM [12] was developed from the business perspective. It "provides a scheme or structure for developing, communicating, and managing business plans in an organized manner". BMM is organized around a set of business concepts and articulates the inter-relationships between these.

BMM says that a business rule is a directive that guides/governs a course of action. A course of action may enable another course of action and a business rule governs this 'enabling' as well [13], [14]. We draw two conclusions from this:

The structure of courses of action has a strong bearing on the structure of business rules. If the former is 'flat', non-hierarchical in nature then the business rule that governs it is also flat. On the other hand, if a course of action is complex, hierarchical, then its business rule is complex and must have business rule components that govern the component courses of action of the complex course of action.

Enabling of a course of action by another is a source of relationships between business rules. That is, the business rule for enabling establishes a relationship between the business rule of the enabled course of action and the business rule of the course of action that enables it.

It follows that BROBM must have features for structuring business rules: complex rules would be defined from simpler component rules until atomic rules are reached. Additionally, we need to represent the relationship, enables, between business rules.

We can infer the third requirement of the business layer from the BMM view that an End can be expressed in terms of states. Thus, BMM says that a Vision is about the 'future state' of a business. Similarly, an intended result (goal or objective) is a state that is to be brought about or sustained. Changes in state are the result of courses of action. Thus, state changes are governed/guided by business rules and it is therefore possible to relate business rules to state changes. Evidently, we need to model states, state changes and their relationship with business rules.

The fourth and last requirement of the business layer is that business layers have temporal properties. We obtain this from the notion of the objective of BMM. An objective is SMART, Specific, Measurable, Achievable, Relevant and Time bound. If the intended result is time bound, then the course of action that produces it must be time bound and therefore, the business rule that governs the course of action must govern this punctuality. Evidently, we need to specify the temporal properties of business rules.

II. MATERIAL AND METHOD

In this section, we present our rule model that meets these four identified requirements then we have presented temporal property of business rule and their extension in first order based rule representation.

The business rule model presented in Fig. 2 captures the four requirements of BROBM discussed earlier. Accordingly, this section is organized in four sub-sections, (i) Business Rule Structure, (ii) Relationships across Rules, (iii) the notion of state and its relationship to a business rule and (iv) temporal properties of business rules.

A. Business Rule Structure

The basic structure of a business rule is in two parts a) what is to be governed and b) how it is governed. Consequently, we define a business rule (see Fig. 2) as an aggregate of antecedent and consequent, in which the 'how' aspect is represented in the antecedent and the 'what' aspect in the consequent of the rule. The figure shows that a consequent is a business act. A business act is an active component that is executable. As the figure shows, there are

two kinds of business acts, atomic and complex. An atomic business act cannot be decomposed further. The figure shows that a complex business act is composed of one or more business acts. For example, Late_return (book) is a business act built over the business acts *Return book* and Levy fine respectively.

An antecedent may be a situation that is a state of the enterprise, a business act or any combination of these formed using the Boolean operators AND, OR, NOT. When the situation is a state, we will say that the antecedent is a condition upon the satisfaction of which the consequent is enacted. When it is a business act then **its enactment enables** another business function. An example of a condition is the rule in which the consequent Issue book is governed by the antecedent "the requestor is a registered borrower" giving rise to the rule <the requestor is a registered borrower, Issue book>. An example of enablement is <Register requestor, Deregister requestor>, that is the enactment of Register enables the enactment of Deregister.

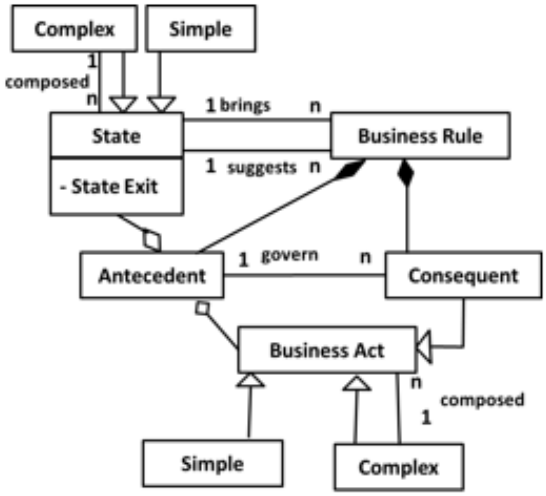

Fig.2. The Business Rule Model

B. The Notion of State

Now, consider the notion of a state. A state can be simple, that cannot be decomposed into simpler states, and complex, one that is composed of other simpler states. In other words, a state is complex if it uses conjunction or disjunction; it is simple otherwise.

When enacted, a business rule changes the state of the enterprise as shown by the relationship, changes, in the figure. The cardinality of this relationship says that the state of the enterprise can be changed by more than one business rule but a business rule changes only one (simple or complex) state. For example, the state, book availability, can be changed by two business rules for issuing a book (availability → issued) and returning a book (issued→available) respectively.

When the enterprise is in a state, s, then it can be forced out of s by **enacting** one or more business rules. This 'forcing out' may be

- Constrained: Here the forcing out is subject to an exit condition that specifies a limit of occupancy of the state. This is captured in the attribute, State Exit of State of Fig. 2. For example, let it be that a book can be issued for up to 10 days only. That is, when the issue rule changes the state of book from available to issued, then the State Exit should be 10 days. This says that the book must be forced move out of state, not available, within 10 days. We capture constrained State Exit using UNTIL [8] for example UNTIL 10 days. The formal representation of UNTIL is presented in section 3.
- Unconstrained: This is the case when State Exit is unspecified. It is possible for the state never to be 'forced out'. For example, given that a book is in the state, available, it is possible that the book may never be issued and remains available.

The business rules that can take the enterprise out of a state is modeled by the relationship, suggests, shown in Fig. 2. It is so named because s suggests the several business rules that can apply to it. Again, there is a 1: N relationship between state and business rule. Suggests is the inverse of the relationship, brings, which says that a business rules brings the business into a state.

We illustrate the foregoing by considering a small example from a library system as follows. The set of business rules, B is B = {Issue, Normal return, Late return, Overdue rule}

Issue is for issuing books; normal return is for those cases in which the book is returned before the due date has expired; the Overdue rule deals with books not returned by the due date; and Late return is for processing books returned after the due date.

The set of state S for books is

S = {available, issued, overdue}

The relationship suggests is represented in Table 1.

TABLE I
THE SUGGESTS RELATIONSHIP

State	Suggested Rule
Available	Issue book
Issued	Normal return, overdue rule
Overdue	Late return rule

That is, if a book is available then the only applicable rules is Issue; if 'issued' then the applicable rules are normal return rule and overdue rule; if it is in overdue state then the applicable rule is the late return rule. One interpretation of Table 1 is as a state transition diagram as given in Fig. 3.

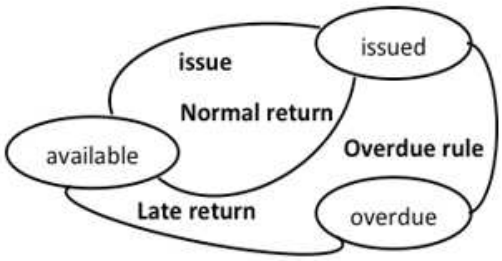

Fig. 3. State Diagram of Book of Library

We can use relationship, *suggests*, in a number of ways as follows:

1) Discovering new rules and states: Analysis of the states of the relationship can reveal new rules. For example, in Table 1, when the book is available then it can be sent for binding. Thus, we get a book binding rule and the state "being bound". Similarly, if a book is issued, then it can also be reserved by some borrower. This gives us the rule reserve book and the state, reserved.

2) Inverse Rules: With each new state-business rule relationship that is discovered, we ask the question, can the state be reversed? This gives us new rules, for example Reserve yields Free that undoes the reservation.

3) Consistency checking: We formulate two consistency checks as follows:

- *Rule consistency:* Given the set of business rules B, the set of suggested rules must be equal to B. Our example shows consistency because the set of suggested rules from Table 1 is {Issue book, normal return, overdue rule, late return} is equal to the set of business rules that we postulated in section 2.2.
- *State Consistency:* The set of states participating in state changes must be equal to the set of states participating in the relationship, suggests. Table 1 shows the latter set of states to be {available, issued, overdue} and this set is equal to the set of states in section 2.2 that participate in state changes. Thus, our example is state consistent.

Notice that a consistent definition does not mean that the enterprise is completely defined. In our example, the three business rules define a consistent enterprise but rules for indenting, stock taking etc. are missing.

C. Typology of Business Rules

Fig. 2 did not, for graphical reasons, present the different types of business rules. This typology is shown in Fig. 4. As seen there are three types of rules, atomic, complex, and abstract [14]. Complex and abstract business rules are constructed from simpler ones until atomic business rules are reached. Complex business rules are themselves of three kinds, aggregates, transitive rules, and bunches. We consider each of these below.

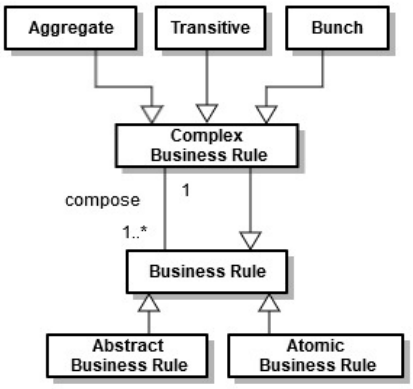

Fig. 4. The Business Rule Typology

1) Atomic Business Rules: An atomic business rule is one whose consequent is an atomic business act and whose antecedent is simple. Examples of atomic business rule are as follows:

< Borrower. Type= 'Student', Issue Book>

< Register Borrower, Issue Book>

In the first rule, the consequent, Issue Book, is an atomic business act and its antecedent is a simple state. In the second rule, the antecedent is simple; it is an atomic business act. Further, its consequent, Issue Book, is also simple.

Complex Business Rules: A complex business rule is a meaningful collection of simpler business rules. A complex rule implies that the antecedent is complex or the consequent is complex. There are three kinds (see Fig. 4) of complex business rules, namely 1) Bunch, 2) Aggregate, and 3) Transitive. The first two of these use a complex antecedent to construct business rules having the same common consequent. This consequent may be complex or atomic. The third kind, transitive, must have a complex business act as its consequent. Its antecedent may or may not be complex. We consider each of these in turn.

- *Bunch:* A bunch is a named collection of business rules having an antecedent on the same common state variable and a common consequent. For example, consider the collection of atomic business rules as follows:

a) <Borrower.Type = 'Student', Register Borrower>
b) <Borrower.Type = 'Teacher', Register Borrower>
c) <Borrower.Type = 'Administrator', Register Borrower>

The Bunch formed is as follows:

Bunch

< Borrower.Type = 'Student' OR Borrower.Type = 'Faculty' OR Borrower.Type = 'Administrator', Register Borrower>

Of

<Borrower.Type = 'Student', Register Borrower>

<Borrower.Type = 'Teacher', Register Borrower>

<Borrower.Type = 'Administrator', Register Borrower>

- *Aggregate:* An Aggregate is a named collection of business rules having an antecedent on different state variables but with a common consequent. Notice the difference with bunch where business rules having antecedents on a common state variable. As an example of aggregate, consider the two of atomic business rule as follows:

a) <Borrower.Type= 'Student', Issue Material>
b) <Borrower.NoIssued<=10, Issue Material>

These rules may be aggregated as

Aggregate

< Borrower.Type= 'Student' AND Borrower.No Issued <=10, Issue Material >

Of

<Borrower.Type= 'Student', Issue Material>

<Borrower.No Issued<=10, Issue Material >

- *Transitive:* It is possible to construct rules using the notion of transitivity. There are two ways in which transitivity arises, directly or indirectly. Direct

transitivity occurs between business acts. Let A1, A2, and A3 be business acts then the following holds by transitivity:

<A1, A2> <A2, A3> implies <A1, A3>

The rule *<A1, A3>* is a complex business rule built over two simpler ones. As an example, from our library, consider the rules,

<Borrower.RegistrationRequest, Register Borrower >
<Register Borrower, Provide Services>

We obtain the transitive business rule as follows.

<Borrower.RegistrationRequest, Provide Services>

transitivity on

<Borrower.RegistrationRequest, Register Borrower >

<Register Borrower, Provide Services>

Indirect transitivity occurs when the consequent of one rule affects the antecedent of another rule. Let us be given the rule

a) < A1, A2>
Let A2 change the value of S, that we express as, *Affects (A2, S)*

Let there be another rule as follows:

b) <S, A3>
Then we get, by indirect transitivity the rule *<A1, A3>*. Taking an example from our library, let us be given

Rule (a) calls for damaged material to be withdrawn. This withdrawal changes the values of quantity on hand, Q_O_H (rule b). Rule (c) reorders material if it falls below the threshold level. By indirect transitivity we get

Transitive

a) <Material.Status= 'Damaged', Reorder Material>

On

b) <Material.Status= 'Damage', Withdraw Material>

Affects (Withdraw Material, Material.Q_O_H)

c) <Material.Q_O_H <= threshold, Reorder Material>

2) Abstract: An abstract business rule is a generalization of other business rules. This generalization can occur when the antecedent and/or consequent of business rules enter into generalization/specialization relationship. Let us given two business rules

a) <Student Borrower.Status='valid' AND Student Borrower.No issued < 4, Issue Book>
b) <Faculty Borrower.Status='valid' AND Faculty Borrower.No issued < 10, Issue Book>

The antecedent of rule contains state variables Student Borrower and Faculty Borrower. These can be generalized into the state variable Borrower. As result, we obtain the abstract rule

Abstraction

<Borrower.Status='valid' AND Borrower.No issued < maximum, Issue Book>

generalization of

<Student Borrower.Status='valid' AND Student Borrower.No issued < 4, Issue Book>

<Faculty Borrower.Status='valid' AND Faculty Borrower.No issued < 10, Issue Book>

Similarly, there can be abstraction based on the consequent. As an example, consider the abstract rule constructed above.

a) <Borrower.Status='valid' AND Borrower.Type= 'Faculty', Issue Text Book>
b) <Borrower.Status='valid' AND Borrower.Type= 'Faculty', Issue Ref. Book>

The consequents Issue Text Book and Issue Reference Book respectively may be generalized as a business act Issue Book giving rise to the abstract business rule

Abstraction

<Borrower.Status='valid' AND Borrower.Type = 'Faculty', Issue Book>

generalization of

<Borrower.Status='valid' AND Borrower.Type ='Faculty', Issue Text Book>

<Borrower.Status='valid' AND Borrower.Type ='Faculty', Issue Ref. Book>

D. Temporal Properties of Business Rules

As mentioned in the Introduction, a business oriented business rule may be time-bound. This implies that it needs to be understood whether the rule must finish within a period of time or it is instantaneous. In the former case, we need to represent the fact that the business rule is long running. Given that a business rule consists of an antecedent and consequent part, there are two factors that we need to examine, namely, (a) the temporal properties of antecedent and consequent respectively, and (b) time interval between antecedent and consequent. We consider each of these in turn.

First notice that time can be viewed as point time or as time interval. Let both the antecedent *A* and the consequent *C* be time points. Since it is not possible for the consequent to precede the antecedent, we get two cases (i) the consequent succeeds the antecedent (row 1 of Table 2) and (ii) both occur at the same time (row 2 of Table 2).

TABLE II
TEMPORAL NATURE OF BUSINESS RULE

Antecedent (A)	**Consequent (C)**	**Condition**	**Execution**
Time point	Time point	Ct > At	Long running
Time point	Time point	Ct = At	Instantaneous
Time point	Time interval		Long duration
Time interval	Time point		Long duration
Time interval	Time interval		Long duration

In the former case, the business rule is long running whereas in the latter case it is instantaneous. On the other hand, it is possible that either *A* or *C* or both are spread over

a time interval. Evidently, in these cases (see rows 3, 4, and 5 of Table 2) the business rule is long running.

Now consider the factor (b), that is, there is a time interval between *A* and *C*. Even if it were the case that both *A* and *C* are time points, the mere presence of this time interval says that the business rule is long running.

Overall then, if we are to handle both our situations, we need to do an analysis based on the start and end times of antecedents and consequents. For example, if the start time of a consequent is greater than the end time of its antecedent then there is a time interval between the two that results in a long running business rules.

Using the notion of 'start time' and 'end time' of time intervals, Allen [15] proposed seven temporal relations, namely, *BEFORE*, *MEETS*, *EQUALS*, *STARTS*, *DURING*, *OVERLAPS* and *FINISHES* between intervals. These, along with their conditions are shown in Table 3 where *A* and *C* are two-time intervals. Note that the subscript, *st*, refers to start time and the subscript, *et*, refers to the end time.

TABLE III
TEMPORAL RELATIONS WITH CONDITIONS

Temporal Relation	Condition
A before C	Aet < Cst
A meets C	Aet = Cst
A equal C	Ast=Cst and Aet=Cet
A starts C	Ast=Cst and Aet<Cet
A during C	Ast>Cst and Aet<Cet
A overlaps C	Ast<Cst and Aet<Cet and Aet>Cst
A finishes C	Ast>Cst and Aet=Cet

In the rest of this section, we consider these relations from the point of view of business rules. As we will see, we need to introduce a new relation to handle instantaneous business rules and use only two out of these seven for long running business rules.

1) Instantaneous Business Rule: First, notice that due to the assumption that A and C are time intervals, the relations of Table 3 cannot be used to express time point. This is because for a time point $A_{st} = C_{st} = A_{et} = C_{et}$ and these relations are not defined to handle this condition.

Therefore, we need to explicitly introduce a relation, *INSTANT*, for specifying an instantaneous business rule. We define *INSTANT* as having two arguments, *INSTANT (A, C)* which says that both *A* and *C* occur at the same moment, or that, $A_{st} = C_{st} = A_{et} = C_{et}$.

2) Long Running Business Rule: Now, let us consider long running business rules. There are two basic constraints that such business rules should comply with. The first is that the start time of a consequent cannot be earlier than the end time of its antecedent. This is because the truth value of the antecedent is known only when it completes.

Constraint 1: $C_{st} >= A_{et}$

Following from this constraint is the second constraint that the start time of the consequent cannot be earlier than the start time of its antecedent. We need this to reason about the conditions of Table 2.

Constraint 2: $A_{st} <= C_{st}$

Now, let us determine which of temporal relations of Table 2 comply with these conditions. Let us look at the third column of Table 4 that shows the relevance of temporal relations to long running business rules.

The first row of the table says that the end time of the antecedent is less than the start time of the consequent. This satisfies constraints 1 and 2 above and results in a long running business rule as shown in the third column of the table. Similarly, the second row of the table says that the end time of the antecedent is the same as the start time of the consequent. This satisfies constraints 1 and 2. As before, this results in a long running business rule. The third row of the table satisfies the second constraint. It violates the first constraint: since A_{et} is greater than A_{st} which is equal to C_{st}, A_{et} is greater than C_{st}. Thus, the relation *EQUALS* of the third row is not relevant to us.

TABLE IV
LONG RUNNING RULES

Temporal Relation	Condition	Business Rule
A before C	Aet < Cst	Long running
A meets C	Aet = Cst	Long running
A equal C	Ast=Cst and Aet=Cet	Violates condition 1
A starts C	Ast=Cst and Aet<Cet	Equivalent to MEETS
A during C	Ast>Cst and Aet<Cet	Violates condition 2
A overlaps C	Ast<Cst and Aet<Cet and Aet>Cst	Violates condition 1
A finishes C	Ast>Cst and Aet=Cet	Violates condition 1

The fourth row meets the second constraint but not the first. The same argument as for *EQUALS* applies. Thus, *STARTS* is not relevant to business rules. The fifth row violates condition (2); the sixth row violates condition (1) and the seventh row violates condition (2).

Therefore, *DURING*, *OVERLAPS*, and *FINISHES* are also not relevant to our analysis.

It can thus be seen that only two, *BEFORE* and *MEETS* yield a long running business rule. To illustrate, let us apply these to a business rule of a library:

If a borrower pays the library fee then the borrower is provided library service.

The antecedent of this rule is Payment and the consequent is service provision. We consider a few cases of the temporal inter-relationships between these as shown in Table 5.

TABLE V
TEMPORAL RELATIONSHIPS BETWEEN THE ANTECEDENT AND CONSEQUENT

Antecedent	Consequent	Nature of Business Rule
Payment is instantaneous at time, t	Service provision is instantaneous at t	Instantaneous, INSTANT(A, C)
Payment is instantaneous at time, t	Service provision is instantaneous but after a delay at t' > t	Long running, BEFORE(A, C)
Payment is over an interval	Service provision is instantaneous but starts at end time of payment	Long running, MEETS(A, C)
Payment is over an interval	Service provision is over an interval but with a delay after end time of payment	Long running, BEFORE (A, C)

3) Representing UNTIL: The last temporal issue is that of representing the UNTIL condition applicable the notion of

a state. For us, UNTIL can be represented using the corresponding notion of UNTIL found in temporal logic [16]. This logic introduces operators like NEXT, UNTIL, RELEASE, FINALLY etc. The difference between non-temporal and temporal logic is that the latter allows the truth value of its predicate to change whereas the former treats the truth value as never changing. Therefore, temporal logic is capable of dealing with dynamic situations. Temporal logic defines UNTIL as a binary operator, for example, (x UNTIL y). This says that x holds during the entire period when y does not hold. The moment y holds, x ceases to be true. Examples of this situation are: talk UNTIL lecture end, alive UNTIL dead.

Notice that we use UNTIL as a way to express precisely such a situation. Thus, we say, in our example of a library, "issued UNTIL end semester"; "issued UNTIL returned"; "issued UNTIL 10 days". In information systems/software engineering, a number of proposals exist that include a clock in the system [17], [18]. This enables the modeler to treat time as a state of the clock and we obtain the notion of a temporal state. Thus, given x UNTIL y, x is a state whereas y can be a temporal or a non-temporal state.

4) Packaging Rules

While constructing business rules, we need to group business rules to make them intellectually manageable. Thus, the set of business rules related to a business area may be packaged together in one package. We refer to such a package as a Rule package. For example, the collection of business rule comprising Procurement may be packaged together in the rule package Procurement Material. We allow the possibility of rule packages being contained in other rule packages. This constructs a hierarchy to rule packages. Thus, two rule packages Procure Material and Maintain Material may comprise rule package Manage Material.

In Fig. 5, Run Library is a rule package that consists of three sub rule packages, Manage User, Inventory Control and Staff Management. Manage User rule itself has two sub rule packages Issue and Return. Issue and Return contain the collection of business rules for issuing and returning material from the library.

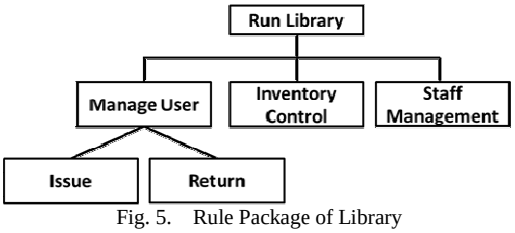

Fig. 5. Rule Package of Library

E. Representing Business Rules

In this section, we use the extended first order logic [19] for our business rules. In this presentation, we have not considered extensions like inclusion of modals that have already been proposed.

Instead, we specifically consider extensions proposed by us as follows:

- A new binary operator that captures the enablement of one business act by another. This operator called enables, is =>. It has two operands, both of which are functions, for example, F1 => F2, which says that F1 enables F2. Enables is similar to implication can also be represented as (NOT F1 OR F2). However, we propose to use the => operator because of its clarity in representing enablement.
- Five temporal predicates,
  - MEETS,
  - BEFORE and its inverse AFTER,
  - INSTANT, and
  - UNTIL.

  As already explained all of these 2-place predicates.

The logic is defined as follows:

- A constant is an individual object in world
- A variable denotes an object in the world.
- A term is a constant or a variable.
- There are n-argument functions of the form G(x1, x2, …. xn) where xi is a term. Functions reflect the mapping of an individual object to another object.
- There are n-place predicates of the form P F(y1, y2, …. yn) where yi is a term. A predicate reflects the mapping of an individual to a truth value.
- An atom is a term, function, or predicate
- There are standard 2-place predicates EQ, NE, LT, GT, GTE, and LTE corresponding to the six relational operators.
- There are standard 2-place predicates namely BEFORE, AFTER, MEET, INSTANT and UNTIL.
- If P and Q are atoms, then ~P, P V Q, P ^ Q, and P => Q are atoms.
- Every atom is a formula. If F is a formula, then Q(F) is a formula where Q can be the universal quantifier or the existential quantifier.
- Brackets may be put as required.
- Nothing else is a formula.

EXAMPLE

- Books can be issued to Faculty members and teaching assistants for a maximum of one semester.

((Type(borrower,'Faculty') OR Type (borrower, 'TeachingAssistants')) AND (Name(material, 'Book')) => (issue(material, borrower) UNTIL (1semester); MEET(A,C)

- Faculty members and teaching assistants can be issued a maximum of 10 books, Journals, magazines, or Project Reports.

((Type (borrower,'Faculty') OR Type (borrower, 'TeachingAssistants')) AND (Name (material, 'Journals') OR Name (material, 'Journals') OR Name(material, 'Magazines') OR Name(material, 'Project report')) AND LTE (No.issue(borrower),10)) => (issue(material, borrower); INSTANT(A,C)

- All non-teaching staff members can be issued 4 books for one semester.

(Type (borrower, 'Non-teaching staff')) AND (Name(material, 'Book') AND LTE (No.issue(borrower),4)) => (issue(material, borrower) UNTIL (1 semester)); MEET(A,C)

III. Results and Discussion

A. Business Rules Management System

We developed a Business Rules Management System, BRMS, for our rules. The architecture of the BRMS is shown in Fig. 6. The tool has three interfaces, namely, Rule Editor, Vocabulary Editor and Package Editor. The repository is organized in two parts, Vocabulary Repository and Rule repository.

1) Rule Editor

The Rule Editor allows us to create, edit and delete business rules that are available in the rule repository. The basis of the Rule Editor is an atomic rule from which it constructs other business rules, bunch, aggregate etc. For this construction purpose, the Rule Editor uses axioms specific to the rule being constructed.

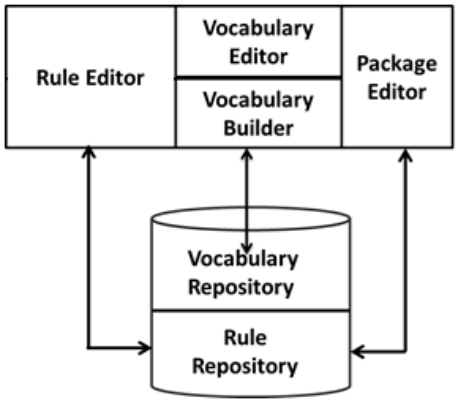

Fig. 6. BRMS architecture

That is, for a bunch, it uses the bunch axiom, for an aggregate it uses the aggregate axiom, and so on.

Fig. 7. Atomic Business Rule

The screen of atomic rule is shown in Fig. 7. The top of the screen allows selection of an existing project or the creation of a new project. Business rules of the project may then be entered. The name of the business rule is entered in the top of the screen. The body of business rule is in Antecedent and Consequent sections of the screen. In the former, a business rule is entered as either a Condition - Business Act type or Business Act – Business Act type rule. As shown in the figure, a business object is represented by Entity and state of business object is represented by Attribute. Similarly, Business act is represented as a Business Function that may take business objects as a parameter. The Consequent section of the screen consists of the Business Function associated with the antecedent along with its optional parameters.

The bottom of the screen deals with state and temporal information the section entitled State Brings, captures the change of state as a result of business rule execution (discussed in 2.2). This is done in terms of the entities and attributes that determine business objects. This section also specifies the temporal constraint that applies. As discussed in section 3 this means that the UNTIL has to be specified.

Lastly, on the left bottom of the screen, the temporal relationship between antecedent and consequent is captured. This is done by selecting one temporal operator from BEFORE, MEETS and INSTANT defined in section 3. The screen automatically fills in the temporal nature of rule either Long Running or Instantaneous.

2) Package Editor

Notice that a package of rules, since it is a collection of logically related business rules, does not have any temporal property associated with it nor does it have a State or a State Exit. A package is just a convenient grouping/classification device for easy partitioning of related business rules.

The architecture of Fig. 6 shows a package editor. This editor is responsible for creation and deletion packages as well as modification of package content by inserting rule in a package, deleting a rule from a package. Package construction does not allow the modification of rules. Rule modification is allowed in the rule editor as discussed above. Any modification of a rule has implication on rule construction.

The package editor operates in two modes: in the first mode, business rules from the rule repository are displayed and the user makes an appropriate selection of these to form the package. In the second mode, the user can search the rule repository to retrieve rules heaving specified values in the antecedent or consequent. Selection from the retrieved rules can then be made to form the package.

3) Vocabulary Editor and Builder (VEB)

This part of the architecture is used for the construction of the vocabulary repository (shown in Fig. 6.). The vocabulary consists of business objects used in the business rules along with the respective states and business acts of business objects.

VEB operates in two modes. In the first mode, the business vocabulary can be managed through the vocabulary editor for direct input by add, update and delete capacity as shown in Fig 8. Once selection of the project is down (see top of screen) the feature to create new entity or to add attributes and associate business acts in existing entity is available. For adding attributes, a choice is provided to add new attributes or specify default values of the attribute. All existing business acts and attributes as well as the added ones are shown in their respective data grids of Fig 8.

In the second mode, VEB picks vocabulary from the rule editor. When a new business rule is written in the rule editor, the vocabulary that is not available in the vocabulary

repository then is picked up by VEB and stored in this repository. As a result, all new vocabulary is made available in the vocabulary for future rule writing. As business expend and its business entities increases, the business vocabulary is also expended.

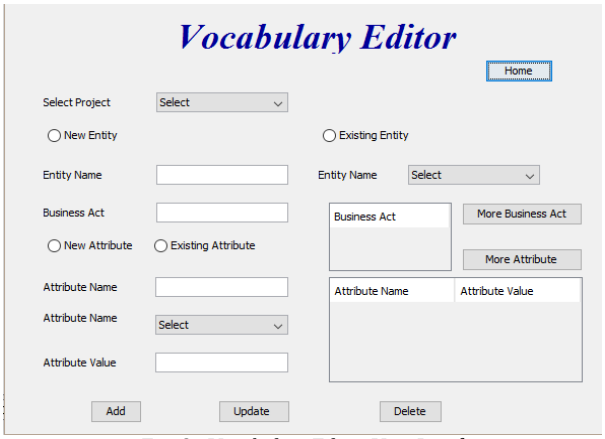

Fig. 8. Vocabulary Editor User Interface

B. Case Study

The foregoing business rule model was applied to the business rules as given to us by the Library Committee of the library of our Institute. We were given a total of 34 business rules covering issue of material (books, journals etc.) by different types of borrowers; return and reservation of library material; stock verification, handling of misplaced material; and requisition to procure new material.

In order to provide a flavor, we consider 4 of these rules related to the task of issuing library material to borrowers. We now illustrate the rules by first given the English statement that was provided to us.

Thereafter, we convert the rules into our form. We include the state changes as well as the temporal conditions of each rule.

Rule 1: Books can be issued to Faculty members and teaching assistants for a maximum of one semester.

{
<((Borrower.BType= 'Faculty Member' OR Borrower.BType= 'Teaching Assistant') AND Material.Name='Book') , Issue Material >
State brings: *Borrower. No issued +1*
Material.Status='issued' UNTIL 1 semester on Material.Name.
Temporal Relation: *MEET*
}

Notice that the consequent is a business act called Issue Material. The state change includes raising the number of books issued by one and changing the state of book from available to not available. The temporal condition imposed is that the material can be kept for one semester. This condition is imposed on *Material.Name* and rule itself is long running as issue is valid for 1 semester.

Rule 2: Print Journals/Magazines/Project report will be issued for 9 days to faculty members and teaching assistants.

{
<(Material.Name = 'Print Journal' OR Material.Name ='Magazine' OR Material.Name = 'Project Report') AND (Borrower.Btype = 'Faculty' OR Borrower.BType= 'Teaching assistant'), Issue Material>
State brings:
Borrower. No issued +1
Material. Status=' issued' UNTIL 9 days on Material.Name.
Temporal Relation: *MEET*
}

Rule 3: Students can be issued books from library for a maximum period of 14 days during a semester.

{
<(Borrower.Btype='Student' AND Material.Name ='Book'), Issue Material>
State brings:
Borrower. No issued +1
Material. Status= 'issued' UNTIL 14 days on Material.Name
Temporal Relation: *MEET*
}

Rule 4: No material in the reference section of the Library will be issued.

{
<(Material.type != 'Reference') , Issue Material>
State change:
Material. Status= 'issued'
Temporal Relation: *MEET*
}

C. Discussion and Related Work

At the outset, we would like to state that we are not aware of any proposal that calls for the introduction of a business layer on top of the CIM layer for business rules. However, such proposals exist [20] in the area of business process modeling,

We have identified four essential requirements of business layer:

- Structuring of Business rules
- Course of action enablement course of action
- Notion of state and state changes
- Temporal Property

The three proposals populating the CIM layer in Fig. 1, do not provide for (I) and (II) above. In SBVR[8], facts are passive elements which led [21] to introduce Activity Fact Type to model business processes. However, factors to integrated modeling of business rule with business process is presented in his research work[22]. Since there is no notion of an activity in SBVR, it is not possible to deal with courses of actions. Similarly, RECON [23],[24] does not deal with courses of action. Finally, ACE [25] in its *'IF <condition> THEN <consequent>'* form allows the condition and consequent to be simple or composite sentences. A simple sentence describes a situation that can be an event or a state. Notice that the BMM notion of a course of action is different from that of an ACE event. Whereas the former tells us what is to be done in a business, the latter is a happening, a single occurrence that is instantaneous. Thus, we see that the CIM layer is unable to address the concerns of business modeling.

A comparison of our proposals with those of SBVR, ACE and RECON is presented in Table 6. The second column of Table 5 contains the feature of our business rule model whereas the other columns of Table 5 indicate whether or not these are present in SBVR, ACE and RECON.

SBVR does not explicitly postulate the notion of a state though it can be done through objectification in its logical formulation. ACE explicitly deals with states but RECON is silent on them. The second row of the table says that temporal conditions are available in all proposals. The third row of the table though there is no notion of business act as operation/function in ACE, it explicitly deals with the notion of an event.

SBVR again, does not explicitly allow for a business act or an event. RECON associates operations with its verbs. The fourth and fifth rows of the table show that whereas all proposals have atomic business rules, no proposal other than ours allows for business rules to be constructed from others.

TABLE VI
FEATURE ANALYSIS OF BUSINESS RULE MODEL

Business Rule Model	**SBVR**	**ACE**	**RECON**
State	No	Yes	No
Temporal Condition	Yes	YES	Yes
Business Act	No	Event	No
Atomic rule	Yes	Yes	Yes
Complex rule	No	No	No

Our business rule model provides for states, temporal conditions on states, and business acts that cause state changes. Business rules govern/direct the manner in which state changes occur. We view business rules as simple and complex. For the latter, we define business rules closure based on the bunch, aggregate, transitive and abstract axioms.

IV. CONCLUSION

Our business oriented view of business rules is positioned in the Model Driven Architecture of OMG as a Business Layer sitting on top of the CIM layer. The business layer details the essential nature of business rules, their structure, their properties, and structural and temporal inter-relationships between business rules. The model inhabiting this new layer provides (a) flat and hierarchical business rules, (b) business rules that operate on the state of an enterprise and cause state changes, (c) temporal properties of business rules for instantaneous and long running business rules. The BRMS contains facilities of creating, updating and deleting rules along with temporal constraints and state information. Logically related rules may be group in to different packages. The next part of our work is to build computer based application systems from given business orient business rules. This requires the conversion of business oriented business rules to a suitable implementation. We are currently developing a strategy and mechanism for this conversion.

REFERENCES

[1] Kruchten P.,(2003). *The Rational Unified Process: An Introduction*, Addison-Wesley Professional, Boston, MA, USA.

[2] OMG.(2015). UML 2.5 [Online]. Available: http://www.omg.org/spec/UML/2.5/

[3] Wazlawick R.,(2014). Object-Oriented Analysis and Design for Information Systems: Modeling with UML, OCL, and IFML (1st ed.). Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

[4] Gladys S. Lam W, (2006). "Business Rules vs. Business Requirements, Business Rules Journal", Vol. 7, No. 5.

[5] Kardasis P., Loucopoulos P.,(2004). "Expressing and organizing business rules", Information and Software Technology 46 701–718.

[6] Ross R. G. (ed.), Business Rules Group, Business Rules Manifesto The Principles of Rule Independence, Version 2.0, www.BusinessRulesGroup.org.

[7] Mohammed A. E.,Sahibuddin S., Ibrahim R,. (2015). "Model driven architecture a review of current literature", Journal of Theoretical and Applied Information Technology, Little Lion Scientific, Vol.59,pp. 122-127.

[8] OMG.(2015). Semantics of Business Vocabulary and Business Rules (SBVR), v1.3 [Online]. Available: http://www.omg.org/spec/SBVR/1.3

[9] Gaweł B, Skalna I, (2014). "Model driven architecture and classification of Business rules modelling languages", Advance in Business ICT, Springer, pp 123-131.

[10] Boley H., (2016). "The RuleML Knowledge-Interoperation Hub" RuleML 2016. Lecture Notes in Computer Science, Springer, vol 9718.

[11] Wagner G, Giurca A, Lukichev S, (2006). R2ML - The REWERSE I1 Rule Markup Language. [Online]. Available: http://www.w3.org /2005/rules/wg/wiki/R2ML.html

[12] OMG.(2012). Business Motivation Model, v1.2 [Online]. Available: http://www.omg.org/spec/BMM/

[13] Prakash N., Deepak S., Deepika, Singh D, (2013). "A Framework for Business Rules", 5th International Workshop on Requirements, Intentions and Goals in Conceptual Modeling -36th International Conference on Conceptual Modeling, HongKong.

[14] Prakash N, Sharma D. K., Singh D, (2014). "Business rule for Business Governance", 16th International conference on Enterprise Information system, Proceeding in LNBIP Springer, Portugal.

[15] Allen J. F., (1983). "Maintaining knowledge about temporal intervals, Commun". ACM, vol. 26, no. 11, pp. 832–843.

[16] Goranko, Valentin and Galton A., (2015) Temporal Logic The Stanford Encyclopedia of Philosophy. [Online]. Available: http://plato.stanford.edu /archives/sum2015/entries/logic-temporal

[17] Hasegawa T.,Fukazawa Y., (2009). "Model Checking by Generating Observers from an Interface Specification Between Components, Information Systems: Modeling", Development, and Integration series, Lecture Notes in Business Information Processing, Vol. 20, pp 526-538.

[18] Geiger M.,Harrer S., Lenhard J.,Wirtz G.,(2016), "On the Evolution of BPMN 2.0 Support and Implementation",IEEE Symposium on Service-Oriented System Engineering (SOSE), Oxford, pp. 101-110.

[19] Ross R. G. (ed.), Business Rules Group, Business Rules Manifesto The Principles of Rule Independence, Version 2.0, www.BusinessRulesGroup.org

[20] Sheridan J, Fouad A and Phalp K, (2008). "Extending the Model Driven Architecture with a pre-CIM level", 1st International Workshop on Business Support for MDA ,Switzerland.

[21] Steen B.,Pires L.F and Iacob M.E.(2010). "Automatic Generation of Optimal Business Processes from Business Rules," 14th IEEE International Enterprise Distributed Object Computing Conference Workshops, Vitoria, pp. 117-126.

[22] Wang W., Indulska M., Sadiq S. (2016). "To Integrate or Not to Integrate – The Business Rules Question", Advanced Information Systems Engineering. CAiSE 2016. Lecture Notes in Computer Science, Springer, vol 9694.

[23] Barkmeyer Ed, Mattas A., (2012). A Restricted English for Constructing Ontologies (RECON).[Online]. Available: http://dx.doi.org /10.6028/NIST.IR.7868

[24] Barkmeyer Ed, Neuhaus F, (2013). "RECON - A Controlled English for Business Rules". RuleML2013@Human Language Technology, 7th International Rule Challenge, Seattle.

[25] Norbert E, Fuchs, Schwertel U, Schwitter R., (2003). Attempto Controlled English (ACE). [Online]. Available: http://web.science .mq.edu.au/~rolfs/papers/ifi-99.03.pdf.