Ontology Based Rule Acquisition from similar Web with Knuth Morris Pratt Algorithm

M. Maria Alex
Assistant Prof in CSE-Dept
Jayamatha Engineering College,
Aralvaimozhi, India,

M.Amal Rajan
Assistant Prof in CSE-Dept
Jayamatha Engineering College,
Aralvaimozhi, India,

Abstract—**Rule acquisition is also an important issue, and the Web that implies inferential rules can be a major source of rule acquisition. It is not easier to acquire rules from a site by using similar rules of other sites in the same domain. Here it proposed an automatic rule acquisition procedure using an ontology, named RuleToOnto, that includes information about the rule components and their structures. The rule acquisition procedure consists of the privacy preserving rule component identification step and the rule composition step. Then A* algorithm is developed for the rule composition. By using Knuth-morris-pratt Algorithm (Kmp), it finds the string is present in pattern and to select exact parts that contain rules from Web pages. The Result demonstrating that our ontology-based rule acquisition approach works in a real-world application.**

Keywords—**Rule acquisition, rule ontology, best-first search, KMP algorithm.**

I. INTRODUCTION

The Semantic Web, which is the key component of Web 2.0 and Web 3.0, is an evolving development of the World Wide Web in which the semantics of information and services on the Web are being defined. This is enabling the Web to understand and satisfy the requests of people and machines to use the Web content [1]. Knowledge is an essential part of most Semantic Web applications and ontology, which is a formal explicit description of concepts or classes in a domain of discourse [2], is the most important part of the knowledge. However, ontology is not sufficient to represent inferential knowledge. This is because ontology-based reasoning has limitations compared with rule-based reasoning, even though ontology-based reasoning with description logic is a popular issue of the Semantic Web. That is, inferential rules are also the essential part of knowledge of the Semantic Web. SWRL [5] is a proposal for a rule representation standard based on ontology. Many attempts have been made at knowledge acquisition in order to obtain enough knowledge for Semantic Web applications. Ontology learning, which refers to extracting conceptual knowledge from several sources and building an ontology from scratch, enriching, or adapting an existing ontology, is one of the attempts at knowledge acquisition. Most ontology learning approaches acquire knowledge from the Web, because it offers a large amount of valuable information for every possible domain. Rule acquisition is as essential as ontology acquisition, even though rule acquisition is still a bottleneck in the deployment of rule-based systems.

Let us suppose that we have to acquire rules from several sites of the same domain. The sites have similar Web pages explaining similar rules from each other. A comparison shopping portal can be an example. The comparison of simple data such as book prices does not need rules, but delivery cost calculation with various options and applying free shipping rules and return policies needs rules [6]. Therefore, the portal should acquire rules about delivery options, shipping rules, and return policies from shopping.

Ontology Learning

The algorithm builds the taxonomy with linguistic analysis and identifies relevant candidates of classes and instances based on statistical analysis. Also, relations can be identified with statistical measures of “connectedness” between identified concepts [3]. The ontologies are composed from automatically obtained taxonomies. OntoLT [3] follows a similar procedure and uses mapping rules between linguistic structure and ontological knowledge. The way OntoLT differs from previous works is that linguistic knowledge remains associated with the constructed ontology. Some approaches used somewhat different learning methods for identifying instances and relations. The rules are generated by expert verification and accumulated by repeating the Ontology Learning.

Rule Acquisition

The Semantic Web becomes more popular, combining rule and ontology reasoning is becoming an important research issue, in addition to ontology inference based on OWL [4]. However, reasoning over OWL ontology and rules is undecidable. Therefore, there is an approach that rewrites a certain form of rules into OWL axioms in order to reduce the complexity. It could be a type of rule acquisition that acquires axioms for ontology reasoning from existing rules. However, the acquisition results are limited to axioms and not general purpose rules, and the target is existing complete rules and not text or Web documents.

In this paper, we want to propose a full automatic procedure of rule acquisition that includes rule composition from identified rule components. We converted the frame-based ontology OntoRule of the XRML approach [8] into the OWL [9] version RuleToOnto, and condensed it by extracting the essential parts of the old rule ontology for the rule composition process. That is, RuleToOnto is an improved version of OntoRule in a different format. We excluded issues on ontology-based rule acquisition that have already been proposed in the previous XRML approach [8], and only proposed new issues on rule composition.

II. THE MODEL

A. Rule Acquisition Procedure

The rule acquisition process following steps,

In step 1, RuleToOnto is generated from rules which are acquired in another site.

In step 2, variables and values are automatically identified from the Web page using RuleToOnto and the first rule draft is generated.

In step 3, rules are automatically composed by combining the identified variables and values. We developed A* algorithm [34] for this purpose. However, the generated rules may be incomplete. Therefore, the knowledge engineer needs to refine the second rule draft to make it complete in step 4.

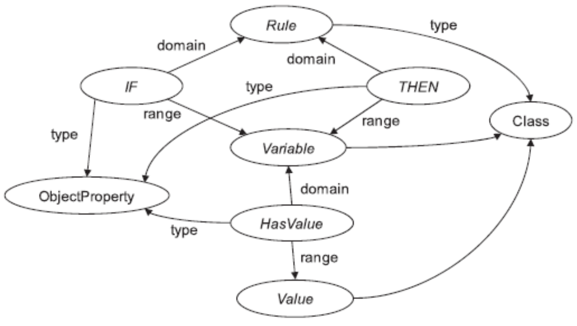


Fig 1: RuleToOnto schema.

B. Rule Ontology Generation

RuleToOnto is domain specific knowledge that provides information about rule components and structures. It is possible to directly use the rules of the previous system instead of the proposed ontology. However, it requires a large space and adon rules, while RuleToOnto is a generalized compact set of information for rule acquisition. Thus, we use RuleToOnto instead of the rules themselves. While the rule component identification step needs variables, values, and the relationship between them, the rule composition step requires generalized rule structures. Therefore, RuleToOnto represents the IF and THEN parts of each rule by connecting rules with variables with the IF and THEN relations, in addition to basic information about variables, values, and connections between variables and values. The RuleToOnto schema has three object properties HasValue, IF and THEN, and three classes, Variable, Value, and Rule, as shown in Fig. 1, which is an RDF graph generated from the OWL ontology by the RDF validator (http://www.w3.org/RDF/Validator/). In order to utilize ontology inference, we added some axioms. First, something is a Variable precisely if all the values of the has Value property are instances of the Value class, as shown in the following axiom represented in the Functional-Style syntax [7].

EquivalentClasses(
:Variable
ObjectAllValuesFrom(:hasValue :Value))

Moreover, Rule can only have instances of Variable for its values of if and then properties as follows:

EquivalentClasses(
:Rule
ObjectIntersectionOf(
ObjectAllValuesFrom(:if :Variable)
ObjectAllValuesFrom(:then :Variable)))

We also added property cardinality restrictions. An instance of Variable should have at least one Value instance for it has Value property, and Rule should have at least on e Variable for each of the if and then properties as follows:

SubClasses(
:Variable
ObjectMinCardinality(1 :hasValue :Value))
SubClasses(
:Rule
ObjectIntersectionOf(
ObjectMinCardinality(1 :if :Variable)
ObjectMinCardinality(1 :then :Variable)))

We excluded connectives such as AND and OR from

RuleToOnto, because it is hard to represent the complex nested structure of connectives in a simple frame representation, and generalization has no effect if we represent all connectives in the ontology [8]. For example, let us assume that we have three rules from different sources as follows:

Rule1:

IF Book Price >= 25 AND Shipping_Method = "Standard Shipping"
THEN Free Charge = TRUE
Rule2:
IF Book Price >= 50 AND (Shipping_Method ="Economy" OR Shipping_Method = "USPS International Surface")
THEN Free Charge = TRUE
Rule3:
IF Book Price >= 50 OR Shipping_Method = "Free Super Saver Shipping"
THEN Free Charge = TRUE

If we make a rule ontology including connectives, the three rules cannot be the same type, because they have different connective structures. However, if we exclude connectives, we can make only one rule type generalized from the three rules. We suppose that this assumption is general, because the same type of rules can have different connective structures in any domain. The ontology is represented in OWL [9] format and we constructed it with Protege [3].

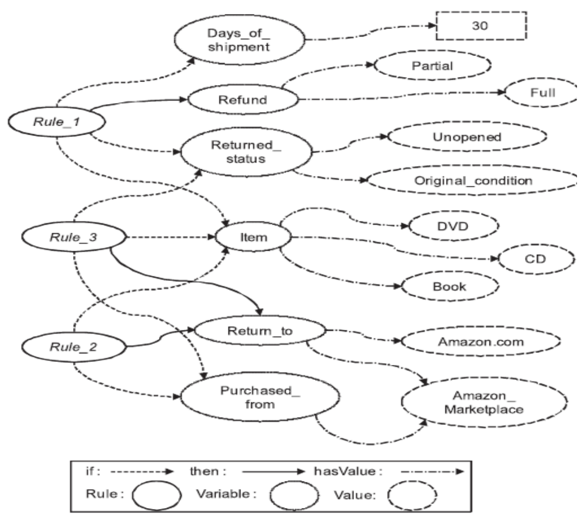


Fig 2. An example of RuleToOnto instance.

Fig. 2 shows an example of RuleToOnto that is built from the rules of Amazon. For example, Rule_1 has days_of_ shipment, returned_status, and item in the IF part and refund in the THEN part.

C. Rule Component Identification

The goal of rule component identification is to elicit variables and values by comparing parsed words of the given text with the variables and values of RuleToOnto. The issues and logical procedures involved have already been discussed in the previous work [8]. We want to focus on the practical implementation and the new issue, semantic similarity. For example, we will be able to detect negations in the text and add them to matching variables if we use grammatical analysis. Moreover, if we use dependency relationships, we can find

the relationships between words and apply them to acquiring variable and value relationships. There is further potential for improvements in HTML tag analysis. For example, we used <P> tags to divide sentences in a document, and used each sentence for an identification and composition range in order to decrease complexity. We expect that there is further potential for improvements by using HTML tag analysis.

Return Policy

- ***We'll refund your purchase if you:***
- ***Return books (items)in their original condition(returned status)***
- ***Return within 30 days of the shipment date listed on your packing slip***
- ***Used books(items)purchased from our PC & Video Games store must be returned to GameShop. These items must be unopened(returned status).***

Fig 3. Target text of the web page.

In order to facilitate better understanding, we make an example scenario. Fig. 3 shows the target text in the Web page of our example. We will identify rule components from the text and combine rules from the components in order to show how to apply our approach. By using the ontology shown in Fig. 2 and the procedure shown in Fig. 6, we can extract the following variable instances from the text:

refund, item, returned status, days of the shipment, item, purchased from, returned to, item, purchased from, returned to, items, returned status.

The variables, refund, days of the shipment, purchased from, returned to, items are directly identified with the ontology, while returned status and three items are identified from the relation between variables and values. For example, we can notice that returned status is omitted, because we found original condition, which is linked to returned status as a value in RuleToOnto. The rule identification step of a previous study [8] identifies the omitted variables by using the relations between variables and values in RuleToOnto. For the sake of convenience, we will use abbreviations for the variable instances as follows:

RF= refund; IT = item;RS = returned status;

DS = days of the shipment; PF = purchased from;

RT = returned to

The identified variable instances can be noted as a set as follows:

VI = {RF1; IT1;RS1;DS1; IT2; PF1;RT1; IT3; PF2;RT2;IT4;RS2}:

We denoted the variable instances with variable abbreviation and number, because one variable has several instances in the text. For example, IT3 is the third instance of IT in VI.

D. Rule Composition through Best-First Search

The objective of rule composition is to combine identified variable instances into rules. There are several possible variable instances for one variable on a Web page. The first step of rule composition is the preparation step, where we find appropriate rules from RuleToOnto. This is done by comparing the identified variable instances with the variables of the rules in RuleToOnto.

The input of this step is the variable instances of rule draft 1 that is an output of the rule component identification step and RuleToOnto. The next step is rule ordering, which generates RuleOrder from the variable instances and the candidate rules. The third step is variable ordering, which generates TotalOrder with the RuleOrder and VariableOrder that is calculated in this step. The last step is best-first search that makes rule draft 2.

E. Preparations for Rule Composition

The input of the Best-First Search (BFS) algorithm is a set of identified variable instances, V I = {V I1; V I2; . . . ; V Ii; . . . ; V In} that is given from the rule component identification stage [32]. The output is a set of rule instances, RI = {RI1;RI2; . . .;RIp; . . .;RIq}, where RIp is a set of variable instances assigned to the rule. The first job of preparation is extracting rule candidates from RuleToOnto. Every variable of each rule candidate should be matched to the variable instances of VI. At this time, the rule candidates are just rule templates with variables to which the variable instances are not yet assigned.

Therefore, we use variables to denote the rule candidates. Rule candidates are denoted as a set RC = {R1;R2; . . .;Rj; . . .;Rl}, where a rule candidate Rj is {Vj1; Vj2; . . . ; Vjk; . . . ; Vjm} and every Vjk is matched to one or more variable instances of VI. Shortly, rule composition generates rule instances by assigning variable instances to rule candidates. By comparing VI the ontology shown in Fig. 2 in our example, we can make the following rule candidates:

R1 = {DS;RS; IT;RF};R2 = {PF; IT;RT};

R3 = {RS; PF; IT;RT};RC= {R1;R2;R3}:

From the above example, we can calculate the following values that will be useful in the next section, where

Count(Vjk) is the number of instances of Vjk in VI that are not assigned to any rule instance yet:

Count(RF) = 1;Count(IT) = 4;Count(RS) = 2;

Count(DS) = 1;Count(PF) = 2;Count(RT) = 2:

In the beginning, the values of Count() are the number of all instances of each variable. However, these values change as we continue the rule composition step and assign variable instances to the rule candidates.

F. Rule Ordering and Variable Ordering

The objective of rule ordering is to decrease the complexity of making rules with identified variable instances . Therefore, we calculate the number of possible combinations of assigning variables for each rule. Subsequently, we choose the rule with the smallest number of combinations and start to assign variable instances to the rule. A smaller number of combinations mean that the number of options we have to consider initially is also small and we can simplify the problem. At the same time, we start from the variable that has the smallest number of instances in the procedure of assigning variable instances to one rule, because it can also decrease the complexity. We call this variable ordering within each rule. We imported the concept of rule and variable ordering from the Constrained Heuristic Search.

Assigning Variable and Value Pairs to IF or THEN

Once the rules are determined, the next step is to complete the rules by assigning variable and value pairs to IF or THEN. For example, the identified rule instance R1 = {DS1;RF1;RS1; IT1}can be converted to the following variable-value pairs by matching variables and values with identified values and the ontology shown in Fig. 2

{(days_of_shipment, 30), (refund, “”), (returned_status, “original condition”), (item, “books”)}.

Assigning the pairs to IF or THEN is very simple. If the variable belongs to an IF part in the rule instance of RuleToOnto, we assign the pair to the IF part of the rule. Otherwise, if it belongs to a THEN part, we assign it to the THEN part. The following shows the generated rule from the above set of pairs by applying Rule 1 of the ontology shown in Fig. 2. Only the variable refund belongs to the THEN part of the rule.

IF days of shipment = 30
AND returned_status = “original condition”
AND item = “books”
THEN
refund = “”

G. Rule Refinement

The rules automatically generated in Section D (*Rule Composition through Best-First Search)* are not complete in most cases, as we can see from the final rule in Section D *(Assigning Variable and Value Pairs to IF or THEN)*. Therefore, we need to refine them. At this step, the knowledge engineer checks the rules and modifies/adds connectives and values. The following rule is an example of the refined rule. The knowledge engineer changed the operator of days_of_shipment from “=” to “<=” and added the value full by referencing the ontology shown in Fig. 2 and the target Web page.

IF days of shipment <= 30
AND returned_status ¼ “original condition”
AND item = “books”
THEN
refund = “full”

H. KNUTH-MORRIS-PRATT ALGORITHM:

It is to select exact parts that contain rules from Web pages. By using knuth-morris-pratt algorithm (kmp) it find the string is present in pattern or not.KMP is just an array of "pointers" (which represents the "internal rules") and a separate "external" pointer to some index of that array (which represents the "current state").

The Knuth–Morris–Pratt string searching algorithm (or KMP algorithm) searches for occurrences of a "word" W within a main "text string" S by employing the observation that when a mismatch occurs, the word itself embodies sufficient information to determine where the next match could begin, thus bypassing re-examination of previously matched characters.

The KMP algorithm does not have the horrendous worst-case performance of the straightforward algorithm. KMP spends a little time precomputing a table (on the order of the size of W[], O(n)), and then it uses that table to do an efficient search of the string in O(k)**.**

pseudocode for the search algorithm

```
algorithm kmp_search:
 input:
    an array of characters, S (the text to be searched)
    an array of characters, W (the word sought)
 output:
    an integer (the zero-based position in S at which W is
                found)

 define variables:
    an integer, m ← 0 (the beginning of the current match in
                S)
    an integer, i ← 0 (the position of the current character in
                W)
    an array of integers, T (the table, computed elsewhere)

 while m + i < length(S) do
    if W[i] = S[m + i] then
       if i = length(W) - 1 then
          return m
       let i ← i + 1
    else
        if $T[i] > -1$ then
            let $m \leftarrow m + i - T[i],\ i \leftarrow T[i]$
        else
            let $i \leftarrow 0,\ m \leftarrow m + 1$

(if we reach here, we have searched all of S unsuccessfully)
return the length of S
```

III. RESULT FOR THE SYSTEM

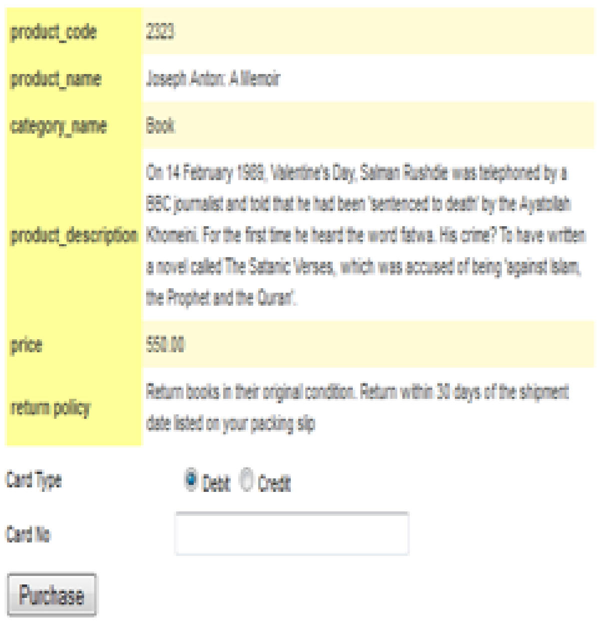
In this module it is to develop the semantic structure for web application.In this structure add some product category with some policies and add the item under its category. It define the rules for individual policy.

Parsing & Streaming Html:

The goal of rule component identification is to extract variables and values by comparing parsed words of the given text with the variables and values of RuleToOnto.

Semantic Matching:

To find the relationships between words and apply them to acquiring variable and value relationships.



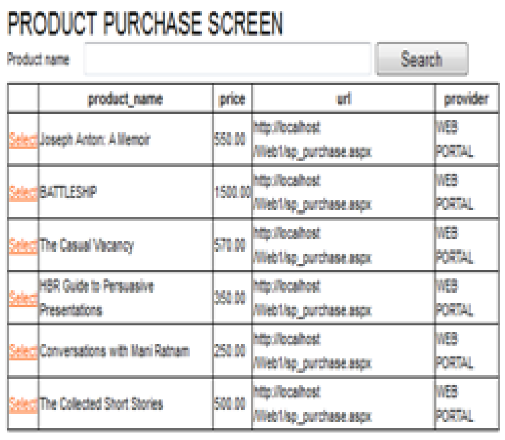
product_code	2323
product_name	Joseph Anton: A Memoir
category_name	Book
product_description	On 14 February 1989, Valentine's Day, Salman Rushdie was telephoned by a BBC journalist and told that he had been 'sentenced to death' by the Ayatollah Khomeini. For the first time he heard the word fatwa. His crime? To have written a novel called The Satanic Verses, which was accused of being 'against Islam, the Prophet and the Quran'.
price	550.00
return policy	Return books in their original condition. Return within 30 days of the shipment date listed on your packing slip

Card Type [✓ Debit] Credit

Card No

Purchase

Fig 4. Web Portal.



PRODUCT PURCHASE SCREEN

Product name Search

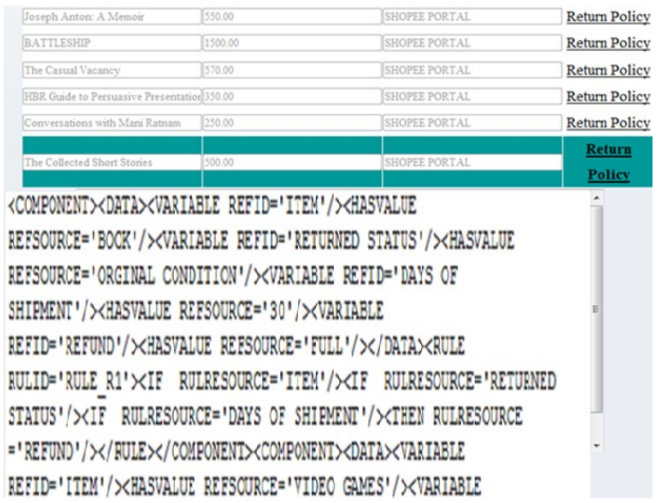
	product_name	price	url	provider
Select	Joseph Anton: A Memoir	550.00	http://localhost /Web1/sp_purchase.aspx	WEB PORTAL
Select	BATTLESHIP	1500.00	http://localhost /Web1/sp_purchase.aspx	WEB PORTAL
Select	The Casual Vacancy	570.00	http://localhost /Web1/sp_purchase.aspx	WEB PORTAL
Select	HBR Guide to Persuasive Presentations	350.00	http://localhost /Web1/sp_purchase.aspx	WEB PORTAL
Select	Conversations with Mani Ratnam	250.00	http://localhost /Web1/sp_purchase.aspx	WEB PORTAL
Select	The Collected Short Stories	500.00	http://localhost /Web1/sp_purchase.aspx	WEB PORTAL

Fig: 5 To Purchase The Product

RuleToOnto, which represents information about the rule components and their structures.

The rule acquisition procedure consists of the rule component identification step and the rule composition step. A* algorithm is used for the rule composition

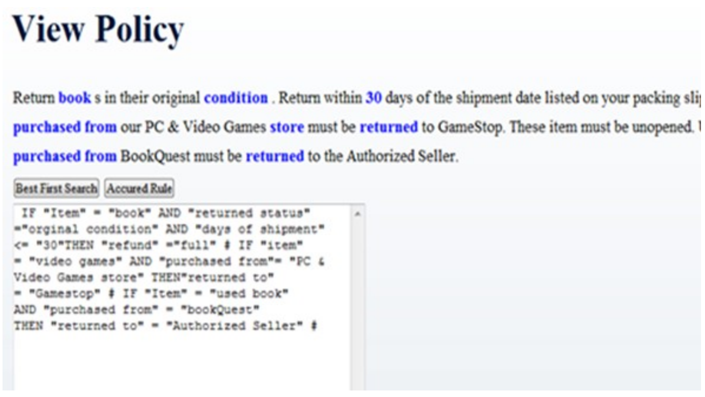
Knuth–Morris–Pratt string searching algorithm (or KMP algorithm) searches for occurrences of a "word" within a main "text string" by employing the observation that when a mismatch occurs, the word itself embodies sufficient information to determine where the next match could begin, thus bypassing re-examination of previously matched characters. The figures 4, 5 and, 6 show the result of the system.



Joseph Anton: A Memoir	550.00	SHOPEE PORTAL	Return Policy
BATTLESHIP	1500.00	SHOPEE PORTAL	Return Policy
The Casual Vacancy	570.00	SHOPEE PORTAL	Return Policy
HBR Guide to Persuasive Presentatior	350.00	SHOPEE PORTAL	Return Policy
Conversations with Mani Ratnam	250.00	SHOPEE PORTAL	Return Policy
The Collected Short Stories	500.00	SHOPEE PORTAL	Return Policy

```
<COMPONENT><DATA><VARIABLE REFID='ITEM'/><HASVALUE
REFSOURCE='BOOK'/><VARIABLE REFID='RETURNED STATUS'/><HASVALUE
REFSOURCE='ORGINAL CONDITION'/><VARIABLE REFID='DAYS OF
SHIPMENT'/><HASVALUE REFSOURCE='30'/><VARIABLE
REFID='REFUND'/><HASVALUE REFSOURCE='FULL'/></DATA><RULE
RULID='RULE_R1'><IF  RULRESOURCE='ITEM'/><IF  RULRESOURCE='RETURNED
STATUS'/><IF  RULRESOURCE='DAYS OF SHIPMENT'/><THEN RULRESOURCE
='REFUND'/></RULE></COMPONENT><COMPONENT><DATA><VARIABLE
REFID='ITEM'/><HASVALUE REFSOURCE='VIDEO GAMES'/><VARIABLE
```

Fig 6: Return Policy



View Policy

Return **book** s in their original **condition** . Return within **30** days of the shipment date listed on your packing sli
purchased from our PC & Video Games **store** must be **returned** to GameStop. These item must be unopened.
purchased from BookQuest must be **returned** to the Authorized Seller.

```
 IF "Item" = "book" AND "returned status"
="orginal condition" AND "days of shipment"
<= "30"THEN "refund" ="full" # IF "item"
= "video games" AND "purchased from"= "PC &
Video Games store" THEN"returned to"
= "Gamestop" # IF "Item" = "used book"
AND "purchased from" = "bookQuest"
THEN "returned to" = "Authorized Seller" #
```

Fig 7: View Policy

CONCLUSIONS

We developed the semantic structure for web application. In this structure add some product category with some policies and add the item under its category. It defines the rules for individual policy. This is enabling the Web to understand and satisfy the requests of people and machines to use the Web content. We can generate the ontology based on previous acquired rule. In first get the previous acquired rule and generate the rule ontology. RuleToOnto is domain specific knowledge that provides information about rule components and structures. It is possible to directly use the rules of the previous system instead of the proposed ontology. The goal of rule component identification is to extract variables and values by comparing parsed words of the given text with the variables and values of RuleToOnto. In this project the contribution is Knuth morris pratt algorithm. The Result show that the performance of our approach to reduces burden of knowledge engineer

REFERENCES

[1] T. Berners-Lee, J. Hendler, and O. Lassila, “The Semantic Web,” Scientific Am. Magazine, 2001.

[2] T. Gruber, “A Translation Approach to Portable Ontology Specifications,” Knowledge Acquisition, vol. 5, no. 2, pp. 199-220,1993.

[3] P. Buitelaar, D. Olejnik, and M. Sintek, “A Prote´ge´ Plug-in for Ontology Extraction from Text Based on Linguistic Analysis,” Proc. First European Semantic Web Symp. (ESWS), 2004.

[4] C. Golbreich, “Combining Rule and Ontology Reasoners for the SemanticWeb,” Proc. RuleML, pp. 6-22, 2004.

[5] I. Horrocks, P.F. Patel-Schneider, H. Boley, S. Tabet, B. Grosof, and M. Dean, “SWRL: A Semantic Web Rule Language Combining OWL and RuleML,” W3C Member Submission, http://www.w3.org/Submission/2004/SUBM-SWRL-20040521/, 2004.

[6] S. Park, J. Kang, and W. Kim, “Rule Acquisition Using Ontology Based on Graph Search,” J. Korean Intelligent Information System, vol. 12, no. pp. 95-110, 2006.

[7] “OWL 2 Web Ontology Language: Structural Specification And Functional-Style Syntax,” B. Motik, P.F. Patel-Schneider and B. Parsia, eds. W3C Recommendation, http://www.w3.org/TR/ 2009/REC-owl2-syntax-20091027/, Oct. 2009.

[8] S. Park and J.K. Lee, “Rule Identification Using Ontology While Acquiring Rules from Web Pages,” Int’l J. Human-Computer Studies, vol. 65, no. 7, pp. 644-658, 2007.

[9] M.K. Smith, C. Welty, and D. McGuinness, “OWL Web Ontology Language Guide,” http://www.w3c.org/TR/owl-guide/, 2004.