Proceedings of the 2009 IEEE International Conference on Systems, Man, and Cybernetics
San Antonio, TX, USA - October 2009

Inference on heterogeneous e-marketplace activities

Chin-Pang Che, Jingzhi Guo and Zhiguo Gong
Department of Computer and Information Science, University of Macau
Av.Padre Tomas, Pereira, S.J., Taipa, macau, Tel: +853-8397 4890
E-mail: {ma36517, jzguo, fstzgg}@umac.mo

Abstract - **E-marketplace activities initiated by users in general require representing the user requirements and preferences matched with a set of offerings. One issue of these activities is the heterogeneity among them, which asks for semantic consistency maintenance. This paper solves this problem by applying collaborative concept exchange technology and developing a novel RuleXPM approach. This approach transforms XPM reified documents into rule-based RuleXPM documents that are suitable for making cross-domain inference on heterogeneous e-marketplace activities, using defeasible reasoning on a newly designed RuleXPM Inference Engine made from a RuleXPM Inference Algorithm.**

I. INTRODUCTION

Recent e-business has been advocated extensively in many business fields. Many enterprises focus on their business on Internet. This makes it very important for developing the e-marketplace technology [5][17]. For the early generation of e-marketplace [16], e-marketplace is just an application for front-end web presentation (e.g. electronic product catalogues) where the sellers can post their product and buyers can search for the product they need to complete the transaction directly. This type of applications has limitations: the buyers can not search the best result since the lack of network resources and sellers lose their opportunities even if they have competitive prices because of no appropriate pricing support systems in the backend. As a consequence, the second generation of e-marketplace was formed. This generation employs the back-end technique [11] and facilitates the business interoperation between e-marketplaces using intelligent agent technology. Nevertheless, since agents are not human and cannot solve the semantic consistency problem occurred in business messaging exchange [5], the inference among e-marketplace activities between intelligent agents must first have a semantic consistency framework. In another word, a trade process between buyers and sellers can proceed correctly only on a semantically consistent basis.

According to the research of [8][9], a trade process in e-marketplace is not a single activity. It involves more complex trading activities of inquiry, offer, acceptance, etc. In general, we face two problems in dealing this complex process: (1) Semantic consistency: different firms have their own trading system and the data structures. It implies that the integration between the heterogeneous business processes is needed. (2) Inference on all activities: when the data transfer among users, e-marketplaces and firms, e-marketplaces need to handle the input and generates the output activities from its own member firms automatically. For example, when an e-marketplace acts as a tourism domain server and receives many inquiries from the travelers, it has to consider how it should disseminate those inquiries to the possible travel agents and how the travel agent should create the best offer. Obviously, all parties should have an inference engine to let the trade process run smoothly. Regarding to the first problem, this paper will utilize the framework of CONEX/CODEX/COPEX [5][7][9] to create and process business information in semantic consistency environment. By adopting this framework, the same semantic information is mapped onto the locally and contextually different information of local firms so that information among the involved parties is semantically consistent. To fulfill this goal, we adopt the specification of XML PRODUCT MAP (XPM) Schema, developed from [5][10].

Focusing on solving the second problem, this paper aims to study how to infer on heterogeneous e-marketplace activities by building or using the stored business rules, that is, how to create run-time business rules to enable heterogeneous e-marketplace activity inference. To solve this problem, this paper proposes a novel RuleXPM approach to represent and reason on heterogeneous e-marketplace activities presented in RuleXPM documents, which are transformed from XPM documents. RuleXPM approach applies defeasible logic [1][2] as reasoning method, which is based on the rules that can be defeated by other rules. In this approach, a trade process is a sequence of conditional activities of inquiry, offer, counter offer, acceptance, and contracting; and is represented by a sequence of RuleXPM documents. The processing of these documents is the responsibility of RuleXPM engine - an inference engine working on all related RuleXPM-represented e-marketplace activities.

The main advantages of RuleXPM approach are: (1) allowing one to reason with incomplete information; (2) coupled with low computational complexity; (3) able to handle priorities and preferences and permit users to define their actual needs; (4) appropriate in a highly dynamic world, such as encoding emergent business rules and policies and user inquiry.

The rest of the paper is organized as follow: Section II discusses a novel RuleXPM Approach. Section III proposes a RuleXPM Inference Algorithm that realizes the RuleXPM inference engine, which is exemplified in Section IV and implemented in Section V. In Section VI, some related works of existing inference engines are discussed. Finally, a conclusion is made and future works are given.

II. RULEXPM APPROACH

RuleXPM approach describes how an XPM-represented e-marketplace activity can be transformed into a RuleXPM-based activity to infer the subsequent RuleXPM-based activity by combining other RuleXPM rules. It is designed in a macro e-marketplace environment modeled in CPDASP (*see* [8][9]) and follows XPM specification [5]. Particularly, it is designed in a RuleXPM Transformation Framework and a RuleXPM Inference Engine, shown in Figure 1.

978-1-4244-2794-9/09/$25.00 ©2009 IEEE

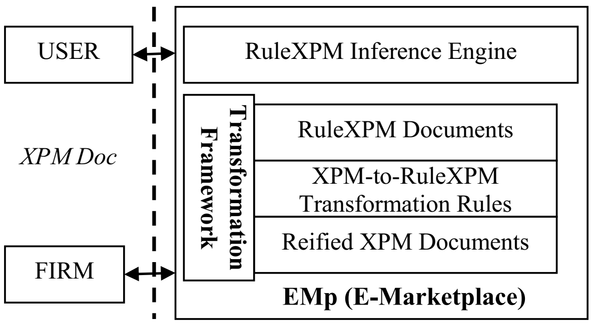

Figure 1. RuleXPM Approach

In this Figure, USER (i.e. user agent on behalf of users) sends/receives XPM reification documents to/from EMp (i.e. e-marketplace) when activity is on. EMp transforms them to RuleXPM documents and infers the next activity with the content represented in RuleXPM document. After that, EMp sends the inferred RuleXPM-based activity content to the corresponding party (EMp or USER) for further action. In this reasoning process, RuleXPM inference engine plays a very important role, which infers the RuleXPM document of one activity to that of another.

In the following, we describe RuleXPM transformation framework and then discuss the RuleXPM inference engine.

A. RuleXPM Transformation Framework

The task of RuleXPM Transformation Framework is to transform a set of reified XPM documents to a set of RuleXPM document. In this transformation process, it involves reified XPM documents.

1) Reified XPM Document Structure

In RuleXPM approach, business messages are encoded in Reified XPM documents [5] governed by XPM Reification Schema. The reified XPM document structure is simple and flexible. Its data model is extended and represented as follows to structuralize any reified concepts in any XPM documents:

Concept =: concept[*iid*, *cls*, *sel, op, gt, rt, fc*] → {*value*}

where Concept =: concept[…](concept[...], …, concept[...]), which is a recursive concept for deriving a concept hierarchy and forms a leveled connotation structure. It represents a meaningful object, which could be reified as any value = {Value}. It denotes itself with a denotation structure […] made by a set of elementary structures such as*:*

- *iid*: unique concept identifier.
- *cls*: classifier in a hierarchical placeholder.
- *sel*: selection type of defining how to process concept relations of "*choice*", "*sequence*" and "*preference*", where "*choice*" means "OR" relation where at least one of the OR-ed must be TRUE; "*sequence*" means "AND" relation where all AND-ed must be TRUE; and "*preference*" is used to identify any user's preferences.
- *op*: numeric value that represents the priority of the preference relation when sel = "*preference*" otherwise it is used to define the operator for value.
- *gt*: grammar type of the current concept.
- *rt*: referenced concept *IID* from other concept vocabulary.
- *fc*: actual human-readable concept.

The "Concept" has several variant notations for describing XPM document structure such as <phrase>, <sentence>, <paragraph>, <section>, <table>, <figure>, etc.

2) RuleXPM Document Structure

A RuleXPM document is a transformation result from a reified XPM document, and is governed by RuleXPM Schema. The purpose of a RuleXPM document transformation is to build a logically inferable document that enables reasoning from one reified XPM document to another. In RuleXPM approach, the logic applied is the defeasible logic [1][2], which handles defeasible rules and priority (i.e. preference relations defined in reified XPM documents). The document structure of RuleXPM is **ruleXPM(Rule, Pref)** of the following:

Rule =: rule[*rid, sel*](concept[*rid, rt, op*] → {*value*})

where "Rule" is a defeasible rule that constitutes of inclusion and/or exclusion rules. The *rid* is the unique rule identifier which uses the prefix "*r.*" to separate from preference. The *sel* defines the rule is essential (by using "*sequence*" to define) or elective (by using "*choice*"). The *rt* is the referenced concept *IID* from other concept vocabulary, and *op* defines the operator for value.

Pref =: preference[*rid, op*](concept[*rid, rt, op*] → {*value*})

where "Pref" denotes any priority relation with its elementary structures such that *rid* is a unique preference identifier that use the prefix "*p.*" to separate from rule, and for *op*, if *op* is a numeric value as Preference's attribute, it represents the user defined priority of preference relation. Otherwise, it is used to define the operator for value.

In Rule and Pref, the "concept" and its elementary structures *rt* and *op* are inherited from the reified XPM.

3) XPM-to-RuleXPM Transformation Rules

In RuleXPM approach, any reified XPM document is converted to a RuleXPM document in defeasible logic-like syntax based on a set of XPM-to-RuleXPM transformation rules as follows in accordance with RuleXPM Schema.

R_1: Concept[*IID*] ⇒ Rule[*IID*]
// Concept with *IID* to be a rule without value.

R_2: Concept[*IID*]→{*Value*} ⇒ Rule[*IID*]→{*Value*}
// Concept with *IID* to be a rule with value as instance.

R_3: Concept[*IID, sel*]→{*Value*} ⇒ Rule[*IID, sel*]→{*Value*}
// Concept with *sel* = "sequence" or "choice".

R_4: Concept[*IID, sel="preference", op*]→{*Value*} ⇒ Preference[*IID, op*]→{*Value*}
// Concept with *sel* = "*preference*" introduces *op* referring to priority relation between preferences.

R_5: Concept[IID_m, *sel="preference",* op_m], Concept[IID_n, *sel="preference",* op_n], $op_m < op_n$ ⇒ Preference[IID_m, op_m] > Preference[IID_n, op_n]

// Concept with less *op* value is denoted to be higher priority in superiority relation.

With these rules, reified XPM documents are converted into RuleXPM documents for e-marketplace activity inference.

B. RuleXPM Inference Engine

The aim of RuleXPM Inference Engine is to infer the result activity from the received input activity. Its inference method can be illustrated in Figure 2.

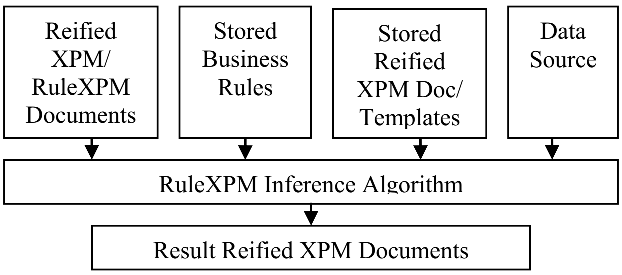

Figure 2. RuleXPM Inference Engine

In this inference engine, the message inputs and outputs for inference adopt the formats of XPM and RuleXPM, which guarantees the semantic consistency between heterogeneous e-marketplace activities. Particularly, the inference inputs could be stored as reified XPM documents or templates, reified XPM or RuleXPM documents, stored business rules or data sources; and an inference output is a reified XPM document. For each activity of an inference chain, it can be generically described as follows:

action([In]Reified XPM/RuleXPM Documents, [In]Stored Reified XPM Documents/Templates, [In]Stored Business RuleXPM, [optIn]Data Sources, [out] Result Reified XPM Document)

where an "action" is an activity in an e-marketplace business process such as inquiry, offer, counteroffer and acceptance.

III. RULEXPM INFERENCE ALGORITHM

This section describes a generic *RuleXPM inference algorithm* (RIA) used in RuleXPM inference engine. It is applicable to all roles of USER, FIRM and EMP. The goal of the algorithm is to infer a next XPM-based activity using the input of the previous XPM/RuleXPM-based activity content and any stored business rules of the corresponding roles.

A. RIA Assumptions

To maintain semantic consistency between heterogeneous e-marketplace activities, RuleXPM approach is designed based on XPM document specification. Thus, any input that is not XPM-compatible must first be converted into reified XPM or RuleXPM documents. Moreover, the stored business rule must be presented in RuleXPM format.

B. Preconditions of RIA

RIA has four preconditions, which are described as follows:

(1) $R = R_{XPM} \cup P_{XPM} = \{r_1, r_2, \ldots, r_n\} \cup \{p_1, p_2, \ldots, p_n\}$
(2) $R_b = \{r_1, r_2, \ldots, r_n\}$
(3) $TD = \{t_1, t_2, \ldots, t_n\}$
(4) $DS = \{d_1, d_2, \ldots, d_n\}$

where:

(1) R_{XPM} as a set of defeasible rules and P_{XPM} as a set of preference relational rules are converted from the reified XPM documents and/or RuleXPM documents of the information sender. For $r_i \in R_{xpm}$, r_i is an inclusion rule. For $p_i, p_j \in P_{xpm}$, p_i, p_j are preference relational rules such that $p_i > p_j$ if $i > j$.

(2) For $r_i \in R_b$, r_i is an exclusion rule of a Stored Business Rule Set R_b, which restricts the inference result. R_b is from the information receiver.

(3) For $t_i \in TD$, t_i is a Stored Reified XPM Template, which is used to create the reified XPM document applying the rules of R_b at the information receiver' side.

(4) DS is Data Sources. If DS is in relational database, d_i refers to an available record in database. Otherwise, d_i refers to a concept in a target XPM reified document.

C. Postconditions of RIA

The postcondition of RIA is a result concept set in Reified XPM Documents such that $RS = \{R_1, R_2, \ldots, R_n\}$, where $R_i \in RS$ refers to a result Reified XPM Document and each $R_i = (rs_1, rs_2, \ldots, rs_n)$.

D. RIA Procedures

To compute the postcondition from the preconditions, there are two situations. In the following, we describe the RIA computation in the following steps:

1) Step 1:Semantic Match Procedure

The first step of RIA is semantic match, which maintains the semantic consistency between the preconditions and postconditions because they occur between different semantic contexts. This is achieved through executing a Semantic Match procedure. Semantic Match Procedure checks whether the concepts of the sender's XPM Reified Documents semantically match the receiver's local concepts through a shared common action concept pool (see [8]) and the shared vocabularies such that:

SemanticMatch $\Rightarrow$ (success | fail).

2) Step 2:Activity Inference Procedure

The second step of RIA is to infer the next activity from a previous activity.

a) Case 1: $R \neq \varnothing$, $Rb \neq \varnothing$, $TD \neq \varnothing$ and $DS = \varnothing$

// find the next activity and document set

IF $\exists r \in R_b$, $a_i \in R$, $a_j \in TD$, $r =: a_i \rightarrow a_j$
THEN IF t_i associateWith a_i AND IF t_j associateWith a_j
  THEN $t_i \Rightarrow t_j$.

b) Case 2: $R \neq \varnothing$, $Rb \neq \varnothing$, $TD \neq \varnothing$ and $DS \neq \varnothing$

BEGIN {Initialize RS;}
  IF DS = relational database THEN
    Query database // use all r_i in R_{XPM} and all r_j in R_b to query database and the result records store in $RS = \{rs_1, rs_2, \ldots, rs_n\}$;
  IF DS = XPM Reified Documents THEN
    FOR i := 1 to n DO

FOR j := 1 to m DO //inclusion rules
IF all r_j in $R_{XPM} =_{match} d_i$ in DS THEN RS := RS + $\{d_i\}$;
FOR i := 1 to p DO
FOR k := 1 to n DO //exclusion rules
IF r_k in $R_b =_{match} rs_i$ in RS THEN RS := RS - $\{rs_i\}$;
IF |RS| =1 THEN
RS = $\{rs_1\}$ // final result and uses $t_j \in$ TD to create the XPM Reified Document R_1 by applying the rules of R_b.
IF |RS| = 0 THEN no possible result;
IF |RS| > 1 THEN {uses preference relation to reduce the possible results}
FOR i := 1 to n DO
FOR j := 1 to m DO
IF p_i in $P_{XPM} \neq_{match} rs_j$ in RS THEN RS := RS - $\{rs_j\}$;
IF |RS| = 1 THEN
RS = $\{rs_1\}$ // final result and terminates the cycle.
IF |RS| = 0 THEN
{// means preference relation p_j does not matched // with all possible results. Thus, RS recovers to the // previous status.}
{// Finally, use t_j in TD and RS to create the XPM Reified // Document R_i by applying the rules of R_b.}
END;

So final result is RS = $\{R_1, \ldots, R_n\}$, where $R_i = \{rs_1, \ldots, rs_n\}$.

IV. A CONCRETE EXAMPLE

To motivate the research, we provide a Kelvin's Inquiry that asks for an offer.

TABLE I. KELVIN'S INQUIRY SHEET

Inquiry Sheet	
Size	800~1000 sqf
Price	HKD 6000 ~ 12000
Bedroom	2
Floor	>10
Property Age	<=20
Furnished	True
Cat Permit	True
Preference relation	
Price – cheapest	1
Size – Largest	2
PropertyAge – newest	3

A. *Transformation of inquiry document*

Based on XPM-to-RuleXPM Transformation Specification we described in Section II, we transform Kelvin's XPM inquiry document to RuleXPM document, shown in Figure 3 which is created based on the rules derived from the Table 1, as follows.

r_1: Size between 800~1000
r_2: Price between 6000~12000
r_3: Bedroom = 2
r_4: Floor > 10
r_5: PropertyAge <= 20
r_6: Furnished is TRUE
r_7: Cat Permit is TRUE
p_1: Price-cheapest, op = 1
p_2: Size-largest, op = 2
p_3: PropertyAge-newest, op = 3
$R = R_{XPM} \cup P_{XPM}$, $R_{XPM} = \{r_1, r_2, \ldots, r_6\}$, $P_{XPM} = \{p_1, p_2, p_3\}$

When e-marketplace (EMp) receives the RuleXPM Document from User, it processes the action: ***Inquiry(R, R_b, TD, DS, RS)*** and we assume EMp uses the original XPM reified document as template TD. Since DS =∅, RS = R and using the original TD, EMp forwards the inquiry to FIRM to make offer.

```
<xpm:RuleXPM>
  <xpm:Rule xpm:rid="r.1">
     <xpm:word xpm:rid="r.1.1" xpm:rt="" xpm:fc="Size"/>
     <xpm:phrase xpm:rid="r.1.2" xpm:fc="between" xpm:sel="sequence">
        <xpm:word xpm:rid="r.1.2.1" xpm:rt="" xpm:fc="minimum"
           xpm:op="LgAndEq"> 800 </xpm:word>
        <xpm:word xpm:rid="r.1.2.2" xpm:rt="" xpm:fc="maximum"
           xpm:op="LsAndEq"> 1000 </xpm:word>
     </xpm:phrase>
  </xpm:Rule>
  <xpm:Rule xpm:rid="r.2">
     <xpm:word xpm:rid="r.2.1" xpm:rt="" xpm:fc="Price"/>
     <xpm:phrase xpm:rid="r.2.2" xpm:sel="sequence" xpm:fc="between">
        <xpm:word xpm:rid="r.2.2.1" xpm:rt="" xpm:fc="minimum"
           xpm:op="LgAndEq"> 6000 </xpm:word>
        <xpm:word xpm:rid="r.2.2.2" xpm:rt="" xpm:fc="maximum"
           xpm:op="LsAndEq"> 12000 </xpm:word>
     </xpm:phrase>
  </xpm:Rule>
  ……
  <xpm:Preference xpm:rid="p.1" xpm:op="1">
    <xpm:phrase xpm:rid="p.1.1" xpm:sel="sequence">
      <xpm:word xpm:rid="p.1.1.1" xpm:rt="" xpm:fc="cheapest"
         xpm:op="min"/>
      <xpm:word xpm:rid="p.1.1.2" xpm:rt="" xpm:fc="price"/>
    </xpm:phrase>
  </xpm:Preference>
  <xpm:Preference xpm:rid="p.2" xpm:op="2">
     <xpm:phrase xpm:rid="p.2.1" xpm:sel="sequence">
       <xpm:word xpm:rid="p.2.1.1" xpm:rt="" xpm:fc="largest"
          xpm:op="max"/>
       <xpm:word xpm:rid="p.2.1.2" xpm:rt="" xpm:fc="size"/>
     </xpm:phrase>
  </xpm:Preference>
  <xpm:Preference xpm:rid="p.3" xpm:op="3">
    <xpm:phrase xpm:rid="p.3.1" xpm:sel="sequence">
      <xpm:word xpm:rid="p.3.1.1" xpm:rt="" xpm:fc="newest"
         xpm:op="min"/>
      <xpm:word xpm:rid="p.3.1.2" xpm:rt="" xpm:fc="Property Age"/>
    </xpm:phrase>
  </xpm:Preference>
</xpm:RuleXPM>
```

Figure 3. Inquiry Sheet in RuleXPM

B. *Representation of RIA procedure on FIRM*

Based on the above, we demonstrate one firm to represent the RIA procedure. Assume that $FIRM_1$ has five available apartments shown as table II.

TABLE II. $FIRM_1$ AVAILABLE APARTMENT LIST

ID	Size	Price	Rm.	Floor	Pro. Age	Furnished	Cat Per.
1	950	8800	2	10	15	True	True
2	1071	11800	2	12	17	False	True
3	1671	19000	4	15	0	True	True
4	518	4000	1	8	3	False	False
5	928	9800	2	17	8	True	True

With the additional information, since $apartment_5$'s owner would not give any commission, regarding the company regulation, $FIRM_1$ is not willing to choose $apartment_5$ as a result so it's Stored Business Rule (R_b) should be:

r_8: ID $\neq$ 5, R_b = {r_8}

We assume $FIRM_1$'s data source is XPM Reified Documents, thus, $FIRM_1$ processes the action: ***Inquiry(R, R_b, TD, DS, RS)***.

For computing a result, RIA first processes the Semantic Match procedure and confirms that the received XPM Document concepts are semantic match with FIRM's concepts. After that, RIA processes Activity Inference Procedure such that:

For inclusion rules in R_{XPM}: apartment 1 and 5 are matched with all rules in R_{XPM}, therefore, RS = {$apartment_1$, $apartment_5$}.

For exclusion rules in R_b: $apartment_5$ is matched with R_8, so $apartment_5$ is excluded. Consequently, RS = {$apartment_1$}

We also assume that $FIRM_1$ determines the subsequent action is *offer* and uses its offer template t_j to create the reified XPM Document by applying R_b and send back to EMp.

C. Representation of RIA procedure on EMp

The received XPM reified Documents are shown as Table III. Assume that all FIRMs' offers are valid and accepted by EMp.

TABLE III. EMP RECEIVED FIRMS' OFFERS LIST

ID	Size	Price	Rm.	Floor	Pro. Age	Furnis hed	Cat Permit
1	950	8500	2	10	15	True	True
2	950	10000	2	12	17	True	True
3	800	9800	2	15	18	True	True
4	1000	8500	2	20	12	True	True
5	900	10000	2	17	14	True	True
6	950	8500	2	25	15	True	True
7	1000	8500	2	17	8	True	True

Since Data Source is the received XPM reified Documents, EMp processes the action: ***Offer(R, R_b, TD, DS, RS)*** and the steps are described as below:

(1) RIA processes the Semantic Match procedure first to ensure the semantic consistency and then it proceeds with the Activity Inference Procedure as follows:

> *For inclusion & exclusion rules:* Due to all received offers fulfill user requirements (R_{XPM}) and accepted by EMp, thus, RS = {$apartment_1$, …, $apartment_7$}.

(2) If |RS| = 7 > 1 then RIA uses preference relations p_i to reduce the possible result set RS:

- For $p_1 \in P_{XPM}$, apartment 2, 3 and 5 don't satisfy on preference relation 1 - Price is cheapest. Therefore, they are excluded from RS;
- For $p_2 \in P_{XPM}$, apartment 1 and 6 don't satisfy on preference relation 2 - Size is largest. Therefore, they are excluded from RS;
- For $p_3 \in P_{XPM}$, apartment 4 does not satisfy on preference relation 3 - Property Age is newest. Therefore, $apartment_4$ is excluded from RS.

Eventually, RS = {$apartment_7$} is the final result. Moreover, according to EMp Stored Business Rule, the subsequent action is offer and uses its offer template t_j to create the reified XPM Document and send back to USER.

V. IMPLEMENTATION

The RuleXPM Inference Engine architecture is shown as Figure 4. In this Figure, we assume the concepts exchange and mapping between EMP and FIRM accomplish in "EMP-FIRM concept exchange & mapping module" (see [8]) to simplify the architecture. Now we describe the ingredients as follows:

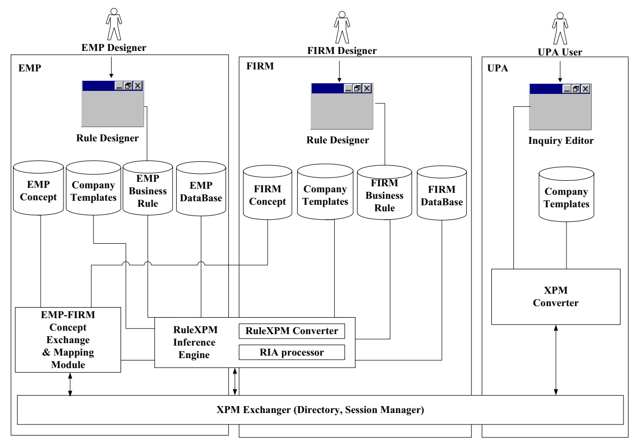

Figure 4. RuleXPM Inference engine architecture

- *USER Inquiry Editor*: responsible for creating and editing the user requirements.

- *EMP/FIRM Rule Designer*: EMP and FIRM designers can add/modify their business rules into/from their own Business Rule Storage.

- *EMP/FIRM Business Rule Storage*: to store the EMP/FIRM business rules/regulations and participates in RuleXPM Inference Engine as R_b.

- *EMP/FIRM/UPA Company Templates*: responsible for storing their own company templates and take part in RuleXPM Inference Engine as TD.

- *EMP/FIRM Database*: stores business data and involve in RuleXPM Inference Engine as DS.

- *XPM Converter*: convert the received inquiry to reified XPM Documents.

- *RuleXPM Inference Engine*: this module is utilized to generate postconditions from the received preconiditions automatically. It constitutes: (1) RuleXPM Converter: to transform XPM concepts to RuleXPM, (2) RIA processor: the core part of this module which determine the correctness of final result.

- *XPM Exchanger module*: it is utilized in exchanging business document between UPA, EMP and FIRM in XPM. It consists of search directory of users, EMP, FIRM and Session Manager for managing interactions.

The advantage of this system architecture is: (1) Flexibility: the system participants can add their dynamic business rules directly to inference engine. (2) Easiness: the system will easy to use for the system participants. (3) Automatic: once USER finish the inquiry sheet, the reified document is automatic processed by the system participants.

VI. Related Work

Inference is investigated in many aspects of SMC area [3][12][13][15], and is most important in inference engine design. In literature, some related works on inference engine are found for comparing with RuleXPM inference engine.

Wu et al. [19] describes an inference engine implemented for Oracle Semantic Data Store, which supports inference based on standard RDFS/OWL constructs and user-defined rules. The inference engine is implemented entirely as a database application on top of Oracle Database.

Jena [18] is a java framework for building semantic web applications. It provides a programmatic environment for RDF, RDFS and OWL [4], including a rule-based inference engine. Jena's inference engine can derive additional statements that the model does not express explicitly and supports OWL constructs. It also allows users to define their own rules.

Euler [14] is an inference engine supporting logic based proofs. It is a backward-chaining reasoner enhanced with Euler path detection. It has been implemented in Java, C#, Python, Javascript and Prolog and can check whether a given set of facts and rules supports a given conclusion.

In RuleXPM approach, the inference engine is designed on RuleXPM and implemented in C++. It can infer heterogeneous e-marketplace activities. It is not a database application but a collaborative e-marketplace component across domains, enabling all the involved system participants to amend and add their business rules in run-time.

VII. Conclusion and Future Work

In this paper we studied the problems on (1) how to transform business documents between heterogeneous e-marketplaces and (2) how to generate the possible result against a set of requirements and user preferences. Regarding to the first problem, we adopt XPM Schema to solve the semantic consistency problem. On the other hand, we propose the novel approach – RuleXPM Schema, which can represent Reified XPM Documents in defeasible logic-like syntax. Moreover, this approach can let user define their preference relations since RuleXPM can define defeasible rules and preference relations. It is simple and flexible for the participant's designer to modify or add their own rules. Finally, we propose a novel algorithm – RuleXPM Inference Algorithm, which infers the next activities from the received activity and generate the possible result Reified XPM Document from the received XPM Documents.

In future, we plan to (1) implement the automated negotiation system. Once user receives the offer provided by firm, user can send the counteroffer back to firm and makes negotiation request. (2) improve our approach that can demonstrate in real-world business field.

Acknowledgement

The work reported in this paper has been partially supported by University of Macau Research Committee.

References

[1] Antoniou, G. and A. Bikakis (2007) DR-Prolog: A System for Defeasible Reasoning with Rules and Ontologies on the Semantic Web. *IEEE Transactions on Knowledge Data and Engineering* 19(2) 233-245.

[2] Antoniou, G., Billington, D., Governatori, G. and M. J. Maher (2001) Representation results for defeasible logic. *ACM Transactions on Computational Logic* 2(2) pp. 255-287.

[3] Burns, J., Winstead, W. and D. Haworth (1989) Semantic Nets as Paradigms for Both Causal and Judgmental Knowledge Representation. *IEEE Transactions on Systems, Man, and Cybernetics* 19(1) pp. 58-67.

[4] Dean, M. and G.. Schreiber (Eds.). *OWL Web Ontology Language Reference* (2004). www.w3.org/TR/2004/REC-owl-ref-20040210/.

[5] Guo, J. (2007). Business-to-Business Electronic Marketplace Selection. *Enterprise Information Systems* 1(4) pp. 383-419.

[6] Guo, J. *Collaborative Concept Exchange*, VDM Publishing: Germany, 2008.

[7] Guo, J. Inter-Enterprise Business Document Exchange. In: *Proc. ICEC'06*, (ACM Press, 2006) pp. 427-437.

[8] Guo, J. (2008) Answering an Inquiry from Heterogeneous Contexts. In: *Proc. of 2008 IEEE Int'l Conf. on e-Business Engineering (ICEBE 2008)*, IEEE Computer Society (Xi'An, China, October 22-24) 113-120.

[9] Guo, J. (2008) Document-Oriented Heterogeneous Business Process Integration through Collaborative E-Marketplace. In: *Proc. of 10th Int'l Conf. on Electronic Commerce (ICEC'08)*, ACM Press (Innsbruck, Austria, August 19-22).

[10] Guo, J. (2009) Collaborative Conceptualization: Towards a Conceptual Foundation of Interoperable Electronic Product Catalogue System Design. *Enterprise Information Systems* 3(1) pp. 59-94.

[11] He, M., Jennings, N. R., and H-F. Leung (2003) On Agent-Mediated Electronic Commerce. *IEEE Transactions on Knowledge and Data Engineering* 15(4) pp. 985 – 1003.

[12] Morizet-Mahoudeaux, P. (1991) Maintaining Consistency of a Database During Monitoring of an Evolving Process by a Knowledge-Based System. *IEEE Transactions on Systems, Man, and Cybernetics* 21(1) pp. 47-60.

[13] Muro-Medrano, P., Banares, J. and J. Villarroel (1998) Knowledge Representation-Oriented Nets for Discrete Event System Applications. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans* 28(2) pp.183-198.

[14] Roo, J. D. *Euler Proof Mechanism*, http://www.agfa.com/w3c/euler/

[15] Shan, Z., Liu, T., Qu, Y. and F. Ren (2005) Modeling and Inference of Extended Interval Temporal Logic for Nondeterministic Intervals. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans* 35(5) pp. 682-696.

[16] Segev, A., Wan, D. and C. Beam. Electronic Catalogs: a Technology Overview and Survey Results. In: *Proc. CIKM'95 (ACM Press, 1995)* 11-18.

[17] Wang, S., Zheng, S., Xu, L., Li, D. and H. Meng (2008) A literature review of electronic marketplace research: Themes, theories and an integrative framework. *Information Systems Frontiers* 10(5) 555-571.

[18] Wilkinson, K., Sayers, C., Kuno, H., and D. Reynolds (2003) Efficient RDF storage and retrieval in Jena, in *Proc. VLDB Workshop on Semantic Web and Databases* (Berlin, Germany) pp. 131 - 150.

[19] Wu, Z., Eadon G., Das S., Chong E.I., Kolovski V., Annamalai M.and Srinivasan, J. (2008) Implementing an Inference Engine for RDFS/OWL Constructs and User-Defined Rules in Oracle. In *Proc. IEEE 24th Int'l Conf. on Data Engineering (ICDE.2008)* pp. 1239-1248.