Semantic Web Rules for Business Information

Anna Maclachlan
Idilia, Inc.
Montreal, Quebec, Canada
email: anna.maclachlan at idilia.com

Harold Boley
Institute for Information Technology
Centre for e-Business
National Research Council of Canada
Fredericton, New Brunswick, Canada
email: Harold.Boley at nrc-cnrc.gc.ca

ABSTRACT
A description of the New Brunswick Business Knowledge Base (NBBizKB) is provided and is made available online in RuleML. NBBizKB realizes a two-step design. First, business facts are extracted, once from static CSV tables and, repeatedly from dynamic semi-structured HTML pages. Second, Semantic Web rules are developed to derive information implicit in the fact base. Fact extraction comprises an XML DTD design, CSV-to-XML conversion, HTML mining, and XSLT translations. Rule derivation employs the Java-based RuleML implementation of OO jDREW to perform data validation, classification mapping, and information integration. Quantitative rule derivation results and findings about the original business data are reported. This rule-based reasoning over extracted facts about New Brunswick business comprises both a case study in business information mining and a use case for Semantic Web rules.

KEY WORDS
Web knowledge bases. Data mining. Rule-based reasoning. Taxonomy alignment. Semantic Web rules. RuleML.

1 Introduction

Many areas of e-Business such as product catalog search can take advantage of the Semantic Web, which has gained strong momentum since its start as a W3C Activity [http://www.w3.org/2001/sw]. The goal of the Semantic New Brunswick initiative of our Semantic Web Lab is making Semantic Web and AI techniques available to business analysts, venture capitalists, and entrepreneurs in a particular region, namely the province of New Brunswick (NB) in Canada. The aim of our project within this initiative is to build tools for regional business analysis and for improving semantically-based business search via taxonomies and rules. It resulted in the NB Business Knowledge Base (NBBizKB) described in this paper and related tools (appendix A).

The utility of semantic tools is apparent in the fact that there are existing resources available on the Web containing business information that we can leverage. One can manually compare NB enterprises of a specific size or in a specific industry sector or in a specific geographic area of the province by consulting the "Biznet Directory of Manufacturers and Selected Services to Industry" (henceforth Biznet), available online. One can also find the contact details for almost any business in the province in the form of the Yahoo! Canada Business Finder (henceforth Yahoo!). NBBizKB exploits the semantics implicit in these sources and goes beyond the intent of the original sources, thereby making already useful resources accesible in novel ways.

NBBizKB is implemented in two steps. The first step is the extraction of a fact base in Object-Oriented RuleML [3] format from the two distinct Web sources. The extraction process as described in section 2 requires transformation from the native format of the sources using XSLT, DTD design and HTML mining. The second step is the realization of rules, again in RuleML format, that operate over the two kinds of facts. As described in section 3, this was achieved by analyzing the fact base and devising rules useful in the context of our collections and relevant to our goal of providing semantic business analysis tools for the region.

NBBizKB has already served two intertwined purposes, helping in the development of several generations of our software:

(1) **Case study in business information mining**, presenting our end-to-end methodology for generating the NBBizKB facts. Since databases like the NB Biznet Directory exist for many regions and the Yahoo! Business Finder has a wide coverage, our case study can be transferred to other places. With deductive techniques from (2), several regional knowledge bases can then be integrated in order to proceed to a global scale.

(2) **Use case for Semantic Web rules** in an e-Business environment, demonstrating the use of RuleML's RDF-inspired, XML-based facts and rules, contributing to the preparation of a W3C Workshop [http://www.w3.org/2004/12/rules-ws/cfp]. Since NBBizKB enhances the many large, nested facts from (1) by complex rules, it has been used to benchmark indexing techniques, translators, etc. (in implementations such as F-logic [12], TRIPLE [10], and jDREW [2], [11]). With inductive techniques, additional rules can be generated from the facts.

These purposes are reflected by the structure of the main sections of this paper: Section 2 describes (2.1) the transition from Biznet's

static CSV tables to flat and nested XML forms, (2.2) the mining of Yahoo!'s dynamic HTML content into a different XML form, (2.3) the classification systems of Biznet and Yahoo!, and (2.4) the transformation of their specialized XML forms into generic RuleML database facts. Section 3 specifies (3.1) rules for validating these data, (3.2) rules that map between the Biznet/Yahoo! classification systems, and (3.3) rules for integrating the Biznet and Yahoo! facts under a common view.

This work builds on related work at IIT e-Business, including the NBBizMapper prototype [http://www.humanweb.ca/metaxtract/nbmap.html], which is a visualization of enterprise-sector distribution in New Brunswick, and the music-recommendation case study RACOFI [1], which led to inDiscover [http://indiscover.net]. There are related use cases for OWL [6] and SWRL [5].

The paper explains our principles employing only slightly idealized examples using substantial sets of real-world data. NBBizKB has thus provided a stress test of RuleML implementations and at the same time allows us to exemplify some as yet undescribed RuleML rule types. The resources used in this study are enumerated in appendix A and are available online for follow-up work.

2 RuleML Facts from Static and Dynamic Web Sources

First, we will describe the extraction of RuleML facts from two Web sources, **Biznet** and **Yahoo!**, that contain structured data on NB enterprises. These sources differ significantly and some of the differences can be briefly outlined as follows. Biznet has a simple table format while the Yahoo! data is from dynamically generated HTML. Biznet is considered static for our purposes and the Yahoo! collection is a dynamic source. Biznet contains more extensive data on few enterprises while Yahoo! provides limited data on more enterprises.

2.1 Enriching CSV Tables for an XML DTD

The Biznet collection was taken to be the primary Web data source for our project. The original Biznet data is available at [http://biznet.gnb.ca/ManuDirectory] and was compiled by the government of the province of New Brunswick in 2002. This Directory of Manufacturers and Selected Services to Industry is not expected to be updated on a frequent basis since it is labor intensive to do so. As such, it is the static Web source of our project. It consists of a comma separated value (CSV) table of 2108 NB enterprises, with a special emphasis on manufacturing companies. For each enterprise, 28 cells are supplied falling under these column headings from the first row of the table (note that these make up a single CSV line):

```
ENT_NUM, OPER_NAME, YEAR_ESTA, LANG_SPOK,
LANG_WRIT, EXPORTER, PHONE, FAX, E-MAIL,
WEB_SITE, EMPL_YEAR, CURRENT_EMPLOYMENT_MAX,
NAICS_SEQUENCE, CNTCT_TITL, CNTCT_FIRS,
CNTCT_LAST, CNTCT_WORK_ENG, CNTCT_WORK_FR,
MAIL_ENG, MAIL_FRN, MAIL_PLACE, MAIL_PROV,
MAIL_POST, PLNT_ENG, PLNT_FRN, PLNT_PLACE,
PLNT_CNTY, CURRENT_EMPLOYMENT_MIN
```

Here is an example of one row of 28 cells for a typical enterprise with a null value for the E-MAIL cell (again forming a single CSV line):

```
91,Nackawic Mechanical Ltd.,1973,ENGL,
ENGL,Y,575-2218,575-2766,,
http://www.sdi-aviation.com,
2002,50,333920,Mr.,David,Young,President,
Président,55 Pinder Rd.,"55, ch. Pinder",
Nackawic, NB,E6G 1W4, 56 Pinder Rd.,
"56, ch. Pinder", Nackawic, York, 30
```

The contents of the CSV table were imported (using the XML tool XMLSpy) into an XML format with XML tags corresponding to each table heading (e.g. <ENT_NUM>).

This flat Biznet XML collection was transformed via XSLT into a semantically-motivated, nested XML format whose document type definition (DTD) was designed with knowledge representation considerations in mind (see appendix A for links to the DTD and the XSLT used). Under this enrichment, tag names were rendered more comprehensible (e.g. YEAR_ESTA was replaced with <ESTABLISHED>) and related items were grouped together under a common tag (e.g. PLNT_CNTY was replaced with <COUNTY> under <PLANT_ADDRESS> under <CONTACT>). Structuring the XML in this manner allowed us to analyze the data more effectively. For example, it suggested potential inconsistencies in the data which motivate the validation rules of section 3.1. Having worked with the data in this form, more efficient structuring of the XML for the purpose of applying rules suggests itself, as we will point out in section 4. The following illustrates the complete XML encoding for the above Biznet enterprise from the Biznet collection according to the enriched DTD.

```
<BIZNET_ENT>
    <NUMBER>91</NUMBER>
    <NAME>Nackawic Mechanical Ltd.</NAME>
    <ESTABLISHED>1973</ESTABLISHED>
    <EXPORT>Y</EXPORT>
    <NAICS_INDUSTRY_SECTOR>
        333920
    </NAICS_INDUSTRY_SECTOR>
    <CONTACT>
        <PHONE>575-2218</PHONE>
        <FAX>575-2766</FAX>
        <WEB_SITE>
            http://www.sdi-aviation.com
        </WEB_SITE>
        <CONTACT_PERSON>
            <PERSON_TITLE>Mr.</PERSON_TITLE>
            <FIRSTNAME>David</FIRSTNAME>
            <LASTNAME>Young</LASTNAME>
            <POSITION_ENG>
                President
            </POSITION_ENG>
            <POSITION_FR>
                Président
            </POSITION_FR>
        </CONTACT_PERSON>
        <MAILING_ADDRESS>
            <ADDRESS_LINE1_ENG>
                55 Pinder Rd.
            </ADDRESS_LINE1_ENG>
            <ADDRESS_LINE1_FR>
                55, ch. Pinder
            </ADDRESS_LINE1_FR>
            <CITY>Nackawic</CITY>
            <PROVINCE>NB</PROVINCE>
            <POSTAL_CODE>
                E6G 1W4
            </POSTAL_CODE>
        </MAILING_ADDRESS>
        <PLANT_ADDRESS>
            <ADDRESS_LINE1_ENG>
                56 Pinder Rd.
            </ADDRESS_LINE1_ENG>
            <ADDRESS_LINE1_FR>
                56, ch. Pinder
            </ADDRESS_LINE1_FR>
            <CITY>Nackawic</CITY>
            <COUNTY>York</COUNTY>
        </PLANT_ADDRESS>
    </CONTACT>
    <LANGUAGE_SPOK>ENGL</LANGUAGE_SPOK>
    <LANGUAGE_WRIT>ENGL</LANGUAGE_WRIT>
    <CURRENT_EMPLOYMENT>
        <MAX>50</MAX>
        <MIN>30</MIN>
    </CURRENT_EMPLOYMENT>
</BIZNET_ENT>
```

2.2 Harvesting Dynamically Generated HTML for Business Data

We also consulted the Yahoo! Canada Business Finder, available at [http://ca.yp.yahoo.com], as the secondary, dynamic Web source for our project. This source, unrelated to the static Biznet source, provides a comprehensive listing of contact and category information for a wide range of businesses. It covers all of Canada but crucially allows search to be limited by province. In order to extract the information contained in this source, we implemented a method to mine specific HTML content in Perl ([4], [9]). The Yahoo! data is accessible only through dynamically generated HTML pages minimally requiring selection of a province and a business category by the user. The miner automatically selects the province of New Brunswick and iterates through the Yahoo! business categories to obtain all the relevant data for NB enterprises. The mining process (performed on December 2, 2003) detected 10,756 enterprise entries, some of which are near duplicates. Typical extractable data from the Yahoo! site for the NB enterprise also used in section 2.1 is illustrated here in its rendered version:

Nackawic Mechanical Ltd
56 Pinder Road, Nackawic, NB E6G1W3 Map
(506) 575-2218
(506) 575-2766 (fax)

and in its slightly simplified HTML source version to illustrate the lack of structure:

```
<b>Nackawic Mechanical Ltd</b><br>56 Pinder
Road, Nackawic, NB E6G1W3
<a href=http://...>Map</a><br>
(506) 575-2218 <br> (506) 575-2766 (fax)
```

The Yahoo! category under which each enterprise fell was also saved out during the HTML mining. For example, the category for the above enterprise was "Mechanical Contractors". Some enterprise listings also contained a URL but most, like the one illustrated, did not. The data extracted from each listing was structured in XML as illustrated below. An empty `<WEB_SITE/>` tag appears when an enterprise URL was not provided.

```
<YAHOO_ENT>
    <OPER_NAME>Nackawic Mechanical Ltd
    </OPER_NAME>
    <PHONE>(506) 575-2218</PHONE>
    <FAX>(506) 575-2766</FAX>
    <MAIL_ENG>56 Pinder Road</MAIL_ENG>
    <MAIL_PLACE>Nackawic</MAIL_PLACE>
    <MAIL_PROV>NB</MAIL_PROV>
    <MAIL_POST>E6G 1W3</MAIL_POST>
    <CATEGORY>Mechanical Contractors
    </CATEGORY>
    <WEB_SITE/>
</YAHOO_ENT>
```

The data was mined several times over the course of our project and proved to change significantly over time. This dynamic source can be used to track changes in business data and hence can also be used to maintain the original data collection. Thus, Yahoo! serves as a secondary dynamic Web data source in support of the primary static source described in section 2.1. This maintenance task is one motivation for the integration rules of section 3.3.

2.3 Two Business Taxonomies

Both of the collections classify their enterprises according to a hierarchy of enterprise types which we can consider to be a taxonomy. In the case of Biznet, the North American Industry Classification System (NAICS) is used to specify what sector each enterprise belongs to. In our `BIZNET_ENT` example, the numeric NAICS code 333920 refers to the sector "Material Handling Equipment Manufacturing". Yahoo! categories, in contrast, are mainly used for navigation purposes, and are similar to yellow pages headings. The Yahoo! category in the `YAHOO_ENT` presented in section 2.2 is "Mechanical Contractors", for example. The enterprise facts in our extracted fact bases contain the classifications encoded in the slots `NAICS_INDUSTRY_SECTOR` and `CATEGORY`, but all taxonomic subsumption relationships are left implicit. We can nevertheless investigate the correspondence between the two business taxonomies in our two fact bases. This motivates our mapping rules in section 3.2.

2.4 Making the XML Formats into Generic RuleML via XSLT

Once we had special-purpose `<BIZNET_ENT>` and `<YAHOO_ENT>` XML elements, our task was to convert these into generic RuleML facts, which can be directly accessed by the RuleML rules of section 3. This conversion was accomplished using XSLT. A fragment of the chosen RuleML encoding of the `<YAHOO_ENT>` example in section 2.2 is given here, showing the structure of an Object-Oriented RuleML (OO RuleML) fact.

```
<Atom>
  <Rel>YAHOO_ENT</Rel>
  <slot>
    <Ind>OPER_NAME</Ind>
    <Ind>Nackawic Mechanical Ltd</Ind>
  </slot>
  <slot>
    <Ind>PHONE</Ind>
    <Ind>(506) 575-2218</Ind>
  </slot>  ...
  <slot>
    <Ind>CATEGORY</Ind>
    <Ind>Mechanical Contractors</Ind>
  </slot>
  <slot><Ind>WEB_SITE</Ind><Ind/></slot>
</Atom>
```

The `<Atom>` uses the `YAHOO_ENT` relation and several generic `<slot>` roles. These slots contain two child elements. The first is the former XML tag name surrounded by `<Ind>` tags. The second is the former XML tag content also marked up by `<Ind>` tags. When the content is missing, no null-value placeholder is needed since the entire slot can be omitted. However, if a slot name such as `WEB_SITE` should be remembered, the empty individual `<Ind/>` can be used as a null value for possible later filling.

In the rest of this paper we will use the more compact POSL (POsitional-SLotted) notation [http://www.ruleml.org/submission/ruleml-shortation.html] for OO RuleML as the presentation syntax. In POSL, the above fact looks much like an F-logic [12] assertion, with a null value for the `WEB_SITE` slot.

```
YAHOO_ENT(
    OPER_NAME->"Nackawic Mechanical Ltd";
    PHONE->"(506) 575-2218";
    ...
    CATEGORY->"Mechanical Contractors";
    WEB_SITE->).
```

Note that in this object-centered modeling of n-ary relations, the semicolon infix indicates an unordered set of n slots, each consisting of an arrow-infixed 'slot->filler' pair. This modeling is preferable to a purely positional representation in Prolog, POSL, or RuleML, because the roles of the n argument positions need not be memorized and new slots can be flexibly added.

3 RuleML Rules Operating on the Fact Base

The NBBizKB rules described here are based on the business data formalized in section 2. We have implemented derivation rules to perform data validation, business classification mapping, and information integration. We also present a rule for duplicate removal that we did not implement and explain why. Rules are developed in POSL, a human-oriented syntax endowed with transformers to and from the machine-oriented XML syntax of RuleML. In each case we present our rules in this format with an informal rule description in English, which can be read as comments, preceding the POSL formalization. Section 2.4 provides an example of the actual RuleML syntax. Building on jDREW [11], rule inference employs the Java-based RuleML implementation of OO jDREW [http://www.jdrew.org/oojdrew], in a combination of bottom-up (BU) and top-down (TD) derivations. The following subsections present a selection of NBBizKB rules from each of the types and their derivation results.

3.1 Validation Rules

Validation rules in NBBizKB are conceived as derivation rules that can prove the distinguished relation `Alert`. They can thus be viewed as simulating reaction rules that

alert KB managers about possible KB problems such as inconsistencies or implausible argument combinations (positional or slotted). Moreover, alerts can point us to any slots and values involved in suspicious facts, and include any key-like fact-identifying arguments.

The following rules are examples validating slot combinations of the BIZNET_ENT facts.

1. *Issue an* Alert *if a* BIZNET_ENT *with some* NUMBER *N has a* LANGUAGE_SPOK *of ENGL but a* LANGUAGE_WRIT *value of FREN.*

```
Alert(NUMBER->?N;
      LANGUAGE_SPOK->ENGL;
      LANGUAGE_WRIT->FREN) :-
  BIZNET_ENT(NUMBER->?N;
             LANGUAGE_SPOK->ENGL;
             LANGUAGE_WRIT->FREN ! ?).
```

2. *Issue an* Alert *if a* BIZNET_ENT *with some* NUMBER *N has a* LANGUAGE_SPOK *S not equal to* BILI *(bilingual) but a* LANGUAGE_WRIT *value of* BILI.

```
Alert(NUMBER->?N;
      LANGUAGE_SPOK->?S;
      LANGUAGE_WRIT->BILI) :-
  BIZNET_ENT(NUMBER->?N;
             LANGUAGE_SPOK->?S;
             LANGUAGE_WRIT->BILI ! ?),
  naf(eq(?S,BILI)).
```

3. *Issue an* Alert *if a* BIZNET_ENT *with some* NUMBER *N has a* CURRENT_EMPLOYMENT *with a* MAX *of EMax and a* MIN *of EMin but EMin is greater than EMax.*

```
Alert(NUMBER->?N;
      CURRENT_EMPLOYMENT->
      [MAX->?EMax; MIN->?EMin]) :-
  BIZNET_ENT(NUMBER->?N;
      CURRENT_EMPLOYMENT->
      [MAX->?EMax; MIN->?EMin] ! ?),
  >(?EMin,?EMax).
```

Note that in POSL notation, each named variable is prefixed by a question mark, and an arbitrary-length rest slot is made explicit via a variable that can be anonymous (written as a question mark) following an exclamation mark. Prolog's 'neck' symbol ":-" here separates a slotted conclusion from comma-separated conjoined premises in backward rules. The first premise always accesses BIZNET_ENT slots to be validated; the second premises in rules 2. and 3. call positional built-ins for disequality checking and arithmetic comparison.

Applied to the NBBizKB facts, rule 1 and a converse of it resulted in one alert each for the 2,108 enterprises, while rules 2 and 3 did not discover any inconsistencies. Further rules, similar to rule 3, found 1,022 enterprises with *EMax* = *EMin* and a surprising 208 with *EMax* = *EMin* = 1.

3.2 Mapping Rules

As seen in section 2.3, NBBizKB's BIZNET_ENT facts have a numeric slot for the NAICS_INDUSTRY_SECTOR while YAHOO_ENT facts have a symbolic CATEGORY slot. Our mapping method realizes instance-based taxonomy alignment, where the instances are RuleML facts. In brief, this proceeds conceptually as follows. First, we remove duplicates from each fact base. Then we find the set of enterprises that are in both fact bases by some identity criterion. For each overlapping pair of enterprise facts we derive a candidate sector-category mapping. A mapping is considered valid if at least n enterprise pairs support it. Valid mappings are used for further rules in section 3.3.

Before applying the business classification mapping rule, it is useful to remove duplicate entries from each fact base, where duplicate is defined specifically in support of the mapping operation. The relevant definition is formalized in the following recursive rule generating new YahooUniqEnt facts free of duplicates bottom-up.

The YahooUniqEnt *relation with* PHONE, CATEGORY, *and Rest slots holds if* YAHOO_ENT *holds for the same slots and* YahooUniqEnt *is not (already) known for the same* PHONE *and* CATEGORY *slots and arbitrary other slots.*

```
YahooUniqEnt(PHONE->?P;
             CATEGORY->?C | ?Rest) :-
  YAHOO_ENT(PHONE->?P;
            CATEGORY->?C | ?Rest),
  naf(YahooUniqEnt(PHONE->?P;
                   CATEGORY->?C | ?)).
```

Our currently available bottom-up implementation OO jDREW BU does not support naf (negation-as-failure) since this is the first application that requires it (the naf(eq(...)) of rule 2 in section 3.1 is implemented by special treatment of disequality). NBBizKB provides some motivation for such an extension to jDREW since the rule just described, and several others suggested by the application presented in this paper, do require it. A further point worth noting about this rule is that it is not stratified since there is a recursion through the naf. The XSB-based implementation of TRIPLE [10] benefits from SLG-WAM's tabling, which can handle such non-stratified programs [8]. It would thus be interesting to combine RuleML's slots with XSB's tabling in OO jDREW. In a preprocessing step, we were able to simulate the effect of the above rule and removed 18 duplicate YAHOO_ENT facts, reducing our mapping rule Yahoo! input from 10,756 to 10,738 facts.

We can now define the sector-category mapping rules. The rules first ascertain the identity of an enterprise across BIZNET_ENT and YAHOO_ENT facts and then look up its respective NAICS_INDUSTRY_SECTOR and CATEGORY slots. Because a primary telephone number was almost always provided in both fact bases and required minimal normalization, our current single-slot identity check is based on BIZNET_ENT's CONTACT/PHONE slot and YAHOO_ENT's PHONE slot. This could be extended to a multiple-slot check over other contact details or the enterprise name, but these slot types may require more extensive normalization than phone numbers or may have no

fillers for a large proportion of enterprises. Character-wise identity of telephone numbers is realized for mapping via a single `equalphone` fact. A refined version would add missing area codes and handle variations in the use of, e.g., spaces, hyphens, and parentheses. The overlap between the 10,738 Yahoo! facts and the 2,108 Biznet facts according to `equalphone` identity was 706 enterprises.

Because we will need to remember the enterprise identity on which a sector-category pair is based, the following rule defines a ternary `IdSectorCategory` relation on top of `equalphone`.

In `IdSectorCategory`*, P, S and C constitute a Phone-keyed Sector-Category pair if there is some* `BIZNET_ENT` *with* `CONTACT/PHONE` *P having* `NAICS_INDUSTRY_SECTOR` *S and some* `YAHOO_ENT` *with* `PHONE` *Q having* `CATEGORY` *C and* `equalphone` *holds for P and Q. In* `equalphone`*, Phone is equal to itself.*

```
IdSectorCategory(?P, ?S, ?C) :-
  BIZNET_ENT(CONTACT[PHONE->?P ! ?];
     NAICS_INDUSTRY_SECTOR->?S ! ?),
  YAHOO_ENT(PHONE->?Q; CATEGORY->?C ! ?),
  equalphone(?P, ?Q).
equalphone(?Phone, ?Phone).
```

Our mapping analysis quickly showed that the sector and category form a many-to-many relationship. For example, sector 336612 (Boat Building) maps to categories "Boat Builders" and "Canoes Kayaks", while category "Draperies & Curtains Retail & Custom Made" maps to sectors 314120 (Curtain and Linen Mills) and 337920 (Blind and Shade Manufacturing).

In order to correct possible sector and/or category misclassifications in the facts, a sector-category pair found via any identity criterion is only accepted if it also occurs in a separate enterprise, where non-identity is based here again on telephone numbers. Generalizing such 2-enterprise-supported pairs, an n-enterprise-supported pair could also be introduced. We define our binary `SectorCategory` relation on top of `IdSectorCategory`.

Sector S and Category C form a 2-enterprise-supported `SectorCategory` *pair if* `IdSectorCategory` *holds for a first* `PHONE` *P1 with S and C, and for a second* `PHONE` *P2 with S and C, where P1, P2 fail equalphone.*

```
SectorCategory(?S, ?C) :-
  IdSectorCategory(?P1, ?S, ?C),
  IdSectorCategory(?P2, ?S, ?C),
  naf(equalphone(?P1,?P2)).
```

From the 706 overlapping facts, the above mapping rules discovered 84 2-enterprise-supported sector-category pairs in our collection. Note that the prior removal of duplicates (as per `YahooUniqEnt`) eliminated 17 redundant 2-enterprise-supported sector-category pairs. In addition, 27 of the 84 pairs are eliminated if, for every category, only the maximally-supported sector is chosen leaving 57 reliable sector-category pairs.

Several useful NBBizKB rules have been defined on top of the `SectorCategory` relation, one of which will be presented in the next subsection. Another example which we will not present is `uniqueCategory`, which tests whether a `YAHOO_ENT` category from a global category list is unique for a `BIZNET_ENT` sector. Another is `CSCPath`, which recursively chains arbitrary-length Category-Sector-Category paths connecting categories via intermediate sectors. These rules help analyze the semantic coherence of the `YAHOO_ENT`/`BIZNET_ENT` alignment.

3.3 Integration Rules

NBBizKB defines an `NBEnterprise` relation through view-like rules integrating the `BIZNET_ENT` and `YAHOO_ENT` facts. This virtual relation can be enriched by further public enterprise information available from additional sources. Of special interest here, the sector-category mappings of section 3.2 can be used to derive sectors from categories for information integration.

The initial structure and slots of the `NBEnterprise` relation are modeled on the `BIZNET_ENT` relation given in section 2.1; in particular, its `NAICS_INDUSTRY_SECTOR` slot is reused. Rules for viewing `NBEnterprise` as `BIZNET_ENT` thus entail no information loss. The rule shown here parallels the `BIZNET_ENT` facts and thereby incorporates all 2,108 facts.

`NBEnterprise` *assumes all slots unchanged from* `BIZNET_ENT` *facts.*

```
NBEnterprise(! ?All) :- BIZNET_ENT(! ?All).
```

In contrast, rules for viewing the `NBEnterprise` relation as the `YAHOO_ENT` relation do entail information loss, so that missing slots need to be filled in from the other available sources or by the KB manager. The basic rule unifies all the slots of `YAHOO_ENT` with variables and employs the sector-category mapping right-to-left in its premises; its conclusion is an `NBEnterprise` relation containing renamed versions of those slots, in the appropriate form and filled with those variables, and inserts the derived `NAICS_INDUSTRY_SECTOR`.

`NBEnterprise` *assumes all slots from* `YAHOO_ENT` *facts, using a* `SectorCategory` *mapping, where the* `CONTACT` *slots are renamed and restructured as defined in* `BIZNET_ENT`.

```
NBEnterprise(
  NAME->?Ne;
  NAICS_INDUSTRY_SECTOR->?S;
  CONTACT->
    [PHONE->?P;
     FAX->?Fax;
     WEB_SITE->?URL;
     MAILING_ADDRESS->
       [ADDRESS_LINE1_ENG->?MLe;
        CITY->?MCity;
        PROVINCE->?MProv;
        POSTAL_CODE->?MCode]])
:-
  YAHOO_ENT(
    OPER_NAME->?Ne;
    PHONE->?P;
    FAX->?Fax;
    MAIL_ENG->?MLe;
    MAIL_PLACE->?MCity;
    MAIL_PROV->?MProv;
    MAIL_POST->?MCode;
    CATEGORY->?C;
    WEB_SITE->?URL),
  SectorCategory(?S, ?C).
```

Note that the `SectorCategory` relation called here avoids circularity by only using the stored `YAHOO_ENT` and `BIZNET_ENT` facts, not any to-be-derived facts.

Under the two `NBEnterprise` rules as stated above, 706 duplicate facts would be introduced from the `BIZNET_ENT`/`YAHOO_ENT` overlap computed in section 3.2. We could avoid these duplicates by augmenting the second `NBEnterprise` rule with a `naf` premise to ensure that an enterprise is not also described by a `BIZNET_ENT` fact. Unlike with the duplicate elimination by the `YahooUniqEnt` rule discussed in section 3.2, this application of `naf` need not involve recursion: the `BIZNET_ENT` fact base can be accessed directly, since all `BIZNET_ENT` facts will necessarily also be in the integrated `NBEnterprise` fact base by the first rule.

The second `NBEnterprise` rule creates 2,700 `NBEnterprise` facts from the 10,738 `YAHOO_ENT` facts if categories are mapped to multiple sectors, or 1,470 if categories are mapped to only the maximally-supported sector. Thus we have demonstrated that our secondary dynamic source plus rules could help to maintain our primary static source.

Other rules provide further `NBEnterprise` slots such as the `COUNTY`. In our Java environment we used the RuleML engine jDREW in bottom-up mode for deriving `NBEnterprise` facts, and in top-down mode for proving queries over those precomputed facts.

4 Conclusions

Applying Metaxtract techniques to business information mining, the New Brunswick Business Knowledge Base has been successful as an e-Business case study: It helped us to discover (ir)regularities in two regional data collections and to construct an integration view as well as broadening and deepening our analysis of the current business situation in NB; its scope has already been broadened to Atlantic Canada in an NRC/UNB collaboration [http://teclantic.ca]. NBBizKB has also been successful as a use case for RuleML: The rule development and testing helped us to find problems in the initial OO jDREW BU implementation and gave rise to suggestions regarding `naf` and built-ins in OO jDREW and RuleML.

Our quantitative results can be summarized as follows. We harvested data from the Web to build two large fact bases consisting of 10,756 and 2,108 RuleML facts respectively. We analyzed these fact bases and their interrelation through the application of RuleML rules. Several findings regarding the first collection were produced through validation rules, including two inconsistencies. A simple identity criterion yielded an overlap of 706 facts. From this overlap, rules derived 57 validated taxonomy mappings. These mappings were in turn used to augment 2,108 primary facts with 1,470 secondary partially-specified facts.

This investigation suggests many directions for further work. In particular, we are working on a refined schema for NBBizKB. For the purpose of the kinds of RuleML rules described here, it would be more efficient to have identifying XML elements or OO RuleML slots such as phone number at the highest level since this would simplify indexing. Such changes can be achieved via an XSLT translator or in conjunction with redefining our DTD in the form of an XML Schema.

We would have based the identity criterion of section 3.2 on the `WEB_SITE` slot if a URL had been provided for more enterprises (currently URLs are provided for fewer than half of the Biznet enterprises). As portal URIs are becoming the standard way of identifying enterprises, they will replace phone numbers for that purpose. This would also permit us to regard NBBizKB facts as RDF-like metadata about the enterprises and move the URIs into fact labels obtaining URI-grounded facts [3]. OO jDREW BU should then attempt to merge derived facts grounded by the same URI.

The hierarchies of business categories in section 2.3 could be merged on the basis of the taxonomy alignment in section 3.2, and the result populated with the facts integrated in section 3.3. The merged taxonomy could be represented explicitly in TX RuleML [http://www.ruleml.org/txruleml] or RDFS as an independently usable standard business taxonomy for the region and beyond. It could then be accessed from rule variables as OO RuleML types (or sorts) to reduce the search space [3]. The indexing system of OO jDREW has already been extended for such types, but the NBBizKB real-world data would certainly lead to further improvements.

Following the current automated fact extraction in NBBizKB, another important avenue of further research concerns the stepwise automation of rule formation. Rules implicit in databases or HTML documents must be parsed

and represented in a format that can be translated by XSLT to RuleML. For this purpose we will be able to directly build on work done in the XRML [7] project.

A NBBizKB Resources

1. The Metaxtract Project at the Institute for Information Technology
[http://iit-iti.nrc-cnrc.gc.ca/projects-projets/metaxtract_e.html]
2. NBBizKB harvesting resources
[http://www.humanweb.ca/metaxtract/nbbizkb]
    a. The DTD for the Biznet XML collection
    b. The XSLT for creating the Biznet XML collection
    c. The XSLT for creating the RuleML fact bases
3. RuleML [http://www.ruleml.org]
4. OO RuleML [http://www.ruleml.org/indoo]
5. NBBizKB rules and selected facts
[http://www.ruleml.org/usecases/nbbizkb]
6. OO jDREW [http://www.jdrew.org/oojdrew]

B Acknowledgements

We wish to thank University of New Brunswick students Mike Cote and Marcel Ball, who contributed to NBBizKB in crucial ways, both in the design and implementation of various layers of the system. Thanks also to our colleagues in the Human Web and Internet Logic groups of NRC for helpful discussions. Anna Maclachlan conducted her NBBizKB research while affiliated with NRC and acknowledges their support of this work as part of the Metaxtract project within the Semantic Web Lab.

References

[1] Michelle Anderson, Marcel Ball, Harold Boley, Stephen Greene, Nancy Howse, Daniel Lemire, and Sean McGrath. RACOFI: A Rule-Applying Collaborative Filtering System. In *Proc. Collaboration Agents: Autonomous Agents for Collaborative Environments*. Halifax, Canada, October 2003.

[2] Marcel Ball, Harold Boley, David Hirtle, Jing Mei, and Bruce Spencer. Implementing RuleML Using Schemas, Translators, and Bidirectional Interpreters. W3C Workshop on Rule Languages for Interoperability Position Paper.

[3] Harold Boley. Object-Oriented RuleML: User-Level Roles, URI-Grounded Clauses, and Order-Sorted Terms. In *Proc. Rules and Rule Markup Languages for the Semantic Web (RuleML-2003)*. LNCS 2876, Springer-Verlag, October 2003.

[4] Soumen Chakrabarti. *Mining the Web: Discovering Knowledge from Hypertext Data*. Morgan-Kauffman, 2002.

[5] Stefan Decker, Mike Dean, and Deborah McGuinness. Requirements and Use Cases for a Semantic Web Rule Language. Technical report, RuleML/DAML, March 2003.

[6] Jeff Heflin. OWL Web Ontology Language Use Cases and Requirements. W3C Recommendation, W3C, February 2004.

[7] Jae Kyu Lee and Mye M. Sohn. The eXtensible Rule Markup Language. *Commun. ACM*, 46(5):59–64, 2003.

[8] Konstantinos Sagonas and Terrance Swift. An abstract machine for tabled execution of fixed-order stratified logic programs. *ACM Transactions on Programming Languages and Systems*, 20(3):586–634, May 1998.

[9] Randal Schwartz. Screen scraping for fun and profit. *Linux Magazine*, April 2003.

[10] Michael Sintek and Stefan Decker. TRIPLE – A Query, Inference, and Transformation Language for the Semantic Web. In *1st International Semantic Web Conference (ISWC2002)*. Sardinia, Italy, June 2002.

[11] Bruce Spencer. The Design of j-Drew: A Deductive Reasoning Engine for the Web. In *Joint CoLogNet Workshop on Component-based Software Development and Implementation Technology for Computational Logic Systems of LOPSTR '02*. Technical University of Madrid, Spain, September 2002.

[12] Guizhen Yang and Michael Kifer. Reasoning about Anonymous Resources and Meta Statements on the Semantic Web. In Stefano Spaccapietra, Salvatore T. March, and Karl Aberer, editors, *J. Data Semantics I*, volume 2800 of *Lecture Notes in Computer Science*, pages 69–97. Springer, 2003.