A web application using RDF/RDFS for metadata navigation

Xi S. Guo, Mark Chaudhary, Christopher Dozier
Yogi Arumainayagam, Venkatesan Subramanian

Research & Development
Thomson Legal & Regulatory
610 Opperman Drive
Eagan, MN 55123, USA
xi.guo@thomson.com

Abstract

This paper describes using RDF/RDFS/XML to create and navigate a metadata model of relationships among entities in text. The metadata we create is roughly an order of magnitude smaller than the content being modeled, it provides the end-user with context sensitive information about the hyperlinked entities in focus. These entities at the core of the model are originally found and resolved using a combination of information extraction and record linkage techniques. The RDF/RDFS metadata model is then used to "look ahead" and navigate to related information. An RDF aware front-end web application streamlines the presentation of information to the end user.

1 Introduction

As an information provider, Thomson West stores vast quantities of documents that are served up in response to user queries. Determining the relationships between entities of interest in these documents can be a complex and time consuming part of end-user research. Nor is this sort of information always explicitly presented in the documents retrieved by searches. Automating the process of discovery is complicated by the need to uniquely identify and resolve ambiguities and co-references between entities.

Our system relies on various NLP techniques and name/entity taggers to identify attorney and judge names in news articles on $Westlaw^{TM}$. These names are then tagged with unique reference identifiers that link them to their records in our legal directory. The relationships between these individuals and other entities like their firm (or court name for judges), and title of the document in which they are found are stored as RDF metadata.

A simple representation of relationships among these entities is shown in Figure 1. Documents make references to attorneys. Using NLP techniques, each occurrence is resolved to a unique reference identification. The metadata then allows us to expose meaningful relationships among entities in text. Storing this information as metadata in the UI allows us to look ahead. Hovering over a name, the end user is able to see which firms they are affiliated with. The user is also able to look ahead to see all the other documents that the person occurs in. In addition, we also know which firm each attorney works for and this relationship allows us to see all the other attorneys who work for the same firm. This information is not present in any of the documents retrieved but is inferred from our RDF/RDFS (Lassila, 2000), (Klyne and Carroll, 2004), (W3C, 1999), (W3C, 2004) metadata model. The RDF/RDFS metadata model helps to dynamically resolve relationship among entities during the time of front end rendering. This system could be extended to incorporate additional relationships between other kinds of data.

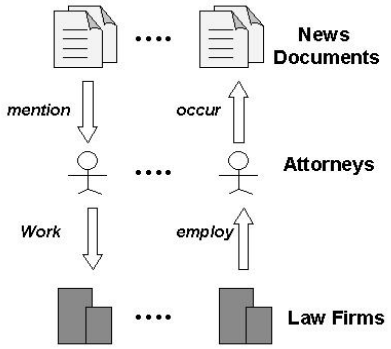

Figure 1: Relationships between entities

2 Architecture

Content in our architecture consists of plain text news documents and RDF metadata. Both are stored in an XML content repository. In addition we also store Thomson West's legal database of attorney profiles in the same repository as well. With the content stored, we use a name/entity tagger in combination with methods described in (Dozier and Haschart, 2000) to link occurrences of attorney

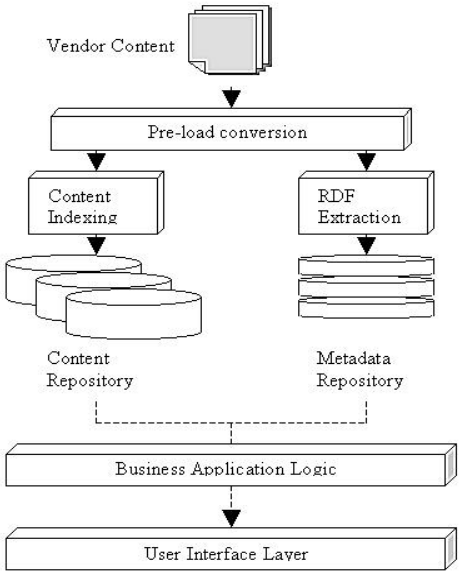

Figure 2: High Level Architecture

names within the plain text news documents to their database profile record.

There are several reasons that motivate us to build this web application using RDF/RDFS. Firstly, our existing data model put metadata and content in the same data repository, the relationships or links are embedded inside content. This makes it very difficult to build new business products since developers have to write programs to look at content first, extract information out of it and then put this extracted information somewhere to enable front-end rendering. The disadvantage of this approach is being able to dynamically maintain the integrity of both data repository and relationship repository in a rapidly changing environment. Both of these repositories need to be updated whenever any relationships get updated. The use of RDF/RDFS separates relationships from content so manipulation of metadata is easier and less expensive.

RDF/RDFS's ability to provide a data infrastructure for entities, relationships extracted from NLP applications is the second reason for choosing it as our data model. In our domain, we have different kinds of entities embedded in news articles, law reviews, legal cases etc. These entities include attorney name, judge name, and law firm names. We are interested in not only identifying them in content but also finding their relationships and linking them together. RDF/RDFS allows us to accomplish this.

Architecture for this application uses MVC (Model View Controller) design pattern for separating graphical interface of one application from its backend artifacts such as code and data. This classic architectural design pattern provided the flexibility to maintain multiple views of backend data.

2.1 RDF/RDFS/XML Data Model

Using the MVC design pattern, our data model represents data used by the application and the rules for accessing this data. A RDF/RDFS/XML model is created to represent the data and a set of APIs is provided for data accessing purpose.

Our prototype contains 911274 legal professionals' profiles from West's Legal Directory and 2000 news documents. The news documents are preprocessed using our name entity tagger. The tagging process is able to generate a list of people templates that are then fed into an entity reference resolution program. This allows us to resolve each extracted name template to its specific record from West's Legal Directory.

Our data model environment contains separate metadata and content repositories, the XML content repository and the RDF metadata repository. We convert the news articles to XML and load them to XML content repository. Our search API features of this repository allow us to perform full text searching inside content. Each news article takes the form of one XML document identified by a unique reference number. Names found inside these documents by the name tagger are identified with xml elements. Besides 2000 news articles, WLD legal professionals' profiles are also loaded to this content repository with each profile also associated with a unique identifying number.

Our RDF metadata repository employs on RDF/RDFS model. A simple RDF schema formally specifies groups of related resources and the relationships between these resources. Figure 3 demonstrates three major RDF resources; Document, People and Organization. The Attorney and Judge resources are subclasses of the People resource. Each instance of these resources has a URI associated with it. Resource related properties are also defined in this schema. The ranges of some properties of resources are themselves resources from other domains. For example, resource Document has a property PeopleInDocument. This property has its domain in Document but its range is in the People domain. The schema allows us to specify the data model so our metadata navigation application could follow relationship links specified in it. More details about this schema can be found in Appendix A.

Based on this schema, the RDF metadata repository is built to represent the relationships among

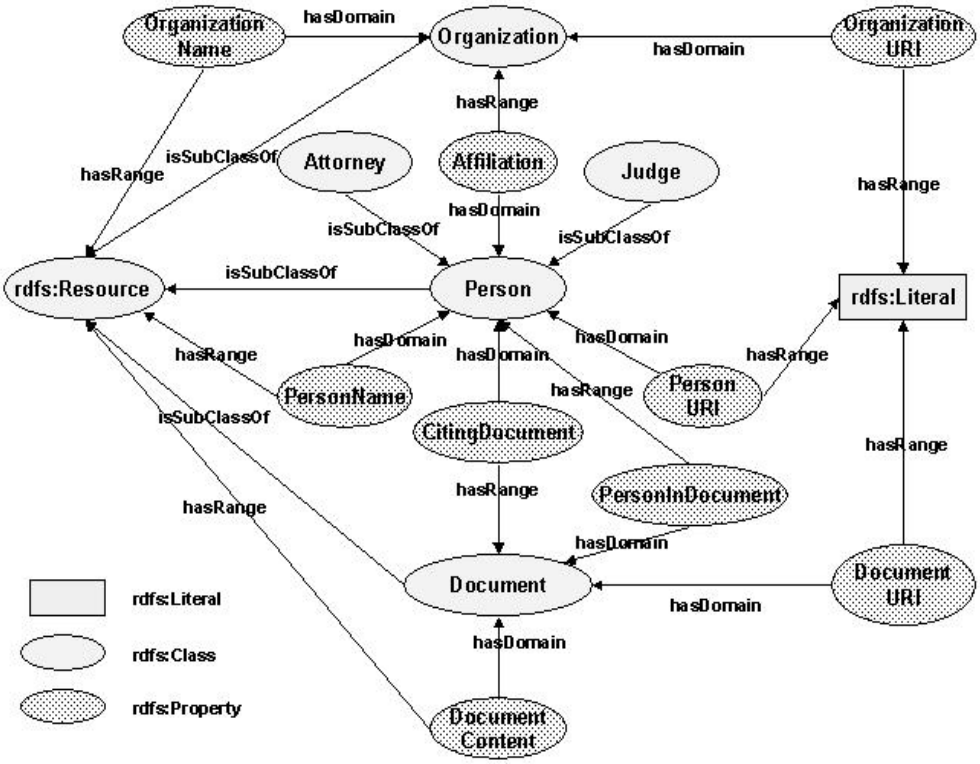

Figure 3: RDF schema of the application

news articles, attorneys, judges, courts and law firms. The metadata building process involves several steps that are entity and relation extraction from the tagged XML content repository, RDF metadata generation, and RDF metadata loading. The end result is an RDF metadata repository with full text search capability. Figure 4 shows samples of a portion of the metadata model depicting the occurrence of two attorneys in a Wall Street Journal document.

During the time the metadata repository was built, our schema was only used for data validation purpose. Currently we are exploring one approach that leverages the expressive power of logic programming tool such as Prolog to navigate the RDF schema graph; this schema navigation should be able to enable automatic metadata collection about particular concepts and then build corresponded RDF metadata based upon.

Note that in this application, URIs (unique reference identification) are used extensively. Each document in both content and metadata repositories has a unique number associated with it. This unique number works as a unique resource link and is utilized by the RDF documents in the metadata repository. With this unique number, the RDF document can then be linked to any xml or rdf document, and even to elements inside these documents using XPATH.

NEWS DOCUMENT RDF

```
<?xml version='1.0' encoding='ISO-8859-1'?>
<rdf:RDF xmlns:PeopleCite='ns://www.westgroup.com/PeopleCite/WSJ#'
        xmlns:rdf='http://www.w3.org/1999/02/22-rdf-syntax-ns#'>

<PeopleCite:WSJ_DOC rdf:about='WSJ210572229'>
        <PeopleCite:DocTitle rdf:resource='//WSJ210572229/DOC/TITLE'/>
        <PeopleCite:Person rdf:resource='0293087701'/>
        <PeopleCite:Person rdf:resource='0170855601'/>
</PeopleCite:WSJ_DOC>

<PeopleCite:Person rdf:about='0293087701'>
        <PeopleCite:PersonResource rdf:resource='WLD0293087701'/>
        <PeopleCite:DocContent
rdf:resource='//WSJ210572229/DOC/TI/NAME2'/>
</PeopleCite:Person>
<PeopleCite:Person rdf:about='0170855601'>
        <PeopleCite:PersonResource rdf:resource='WLD0293086676'/>
        <PeopleCite:DocContent
rdf:resource='//WSJ210572229/DOC/TI/NAME1'/>
</PeopleCite:Person>
</rdf:RDF>
```

ATTORNEY RDF

```
<?xml version='1.0' encoding='ISO-8859-1'?>
<rdf:RDF xmlns:rdf='http://www.w3.org/1999/02/22-rdf-syntax-ns#'
xmlns:PeopleCite='ns://www.westgroup.com/PeopleCite/WLD#'>
<PeopleCite:WLD_DOC rdf:about='WLD0293087701'>
  <PeopleCite:Organization
rdf:resource='i000000000000787f000000eb7bb3715f'/>
  <PeopleCite:QuotingNewsResource rdf:resource=' 'WSJ210572229''/>
  <PeopleCite:QuotingNewsResource rdf:resource=''WSJ210578888''/>
</PeopleCite:WLD_DOC>
</rdf:RDF>
```

Figure 4: Sample RDF metadata

In the sample of the RDF data presented in Table 1, the WSJ document with URI ”WSJ210572229” entitled ”Market on a High Wire” contains references to two attorneys; Froehlich and Madden.

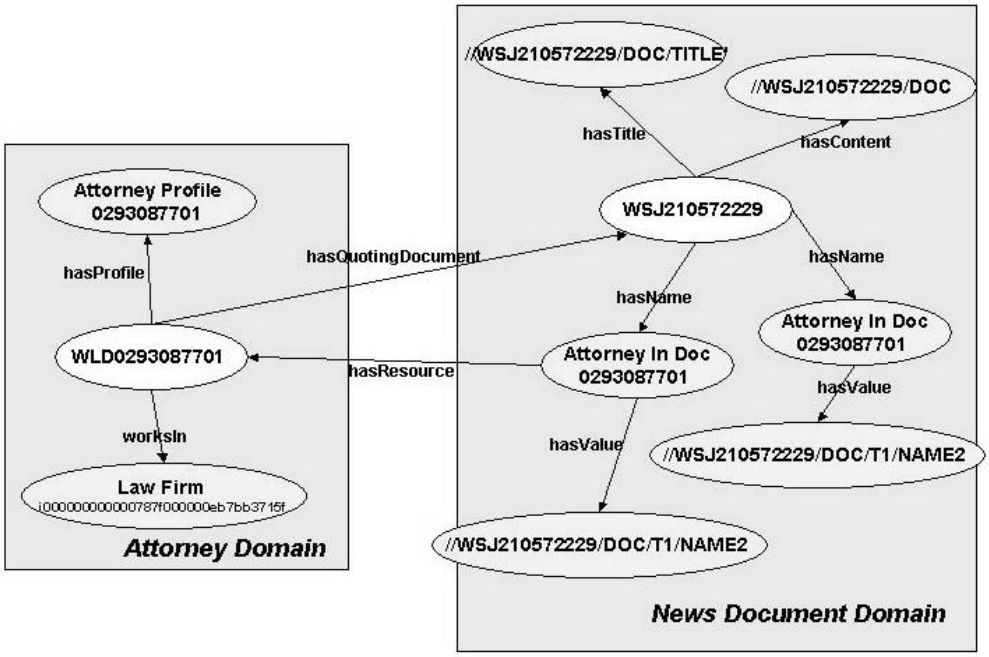

Figure 5: Small RDF Graph of one metadata sample

Froehlich has URI "WLD0293087701" and Madden has URI "WLD0293086676". The metadata also contains the XPATH of the attorney names inside this WSJ document as well as the XPATH to other properties of the document such as news title and news content.

Figure 5 shows a small RDF graph generated from samples in Table 1. In this graph, "WSJ210572229" and "WLD0293087701" are two major resources from two different domains. The RDF properties of both resources point to each other through predicates. These pointing edges represent relationships among multiple entities and they form the infrastructure for our navigational map that will eventually be presented to end-user.

Besides metadata and content storage, the data model in MVC also provides a set of APIs for accessing both metadata and content. In XML content repository, APIs exist for single XML document retrieval by URI and full text search by user queries. In the RDF metadata repository, APIs exist for single RDF document retrieval by URI, RDF resource link retrieval using ARP, an RDF parser from HP and RDF metadata full text search.

2.2 Application Controller

The Controller in our MVC patterned application contains our metadata navigation logic. The purpose of this layer is to capture all requests from the front view and to interact with the data model to provide the data wanted by the end user.

The general scenario of our application starts out with a user typing in queries. These queries are then passed to the XML content repository which returns matched search results with navigation metadata embedded inside. All of this metadata is generated through the controller layer that interacts with both RDF and XML repository. The results then are presented to the user who can click on entities of interest (which are RDF resources) and thus navigate through our metadata repository.

2.3 Front View

All information rendering happens in the front view layer. This layer interacts with end users and specifies how final data can be represented. Since back-end data is either RDF or XML, we use XSLT to convert this to HTML/JSP pages that work in the front end browser.

Appendix B shows a snapshot of our application depicting a single Wall Street Journal article containing attorney names. The end user can roll over this name link and using the pop-up menu, navigate to other corresponding entities such as other news documents that mention the same name, or law firm this attorney is working in. This metadata-based navigation is described in detail in next section.

3 Metadata based Navigation

By tagging entity information and resolving cross document co-references for attorneys and judges, we were able to identify all the documents a particular attorney or judge appeared in. The RDF metadata model goes a step further weaving together the relationships between attorneys, judges, firms, courts and the documents that reference them.

With the metadata model it now becomes easier for the user to see all related information from any particular node. The combination of information extracted from documents with information from authority files, gives us a dynamic view of relationships in the content that can answer questions such as "What other attorneys were mentioned in the same article?" and "Who else works at the same firm as this attorney?" These relationships facilitate navigation between related entities. Figure 6 shows how the metadata model allows the user to navigate from one related node to the next. Not only are we able to tell the firm an attorney belongs to even if that wasn't specifically mentioned in the text of the document, but we can also use the metadata model to shift our focus onto the firm node and immediately see a list of other attorneys related to that firm. Switching to any one of those nodes (attorneys) immediately shows us articles related to the next attorney. In a similar fashion we can move from judges to courts and articles and back.

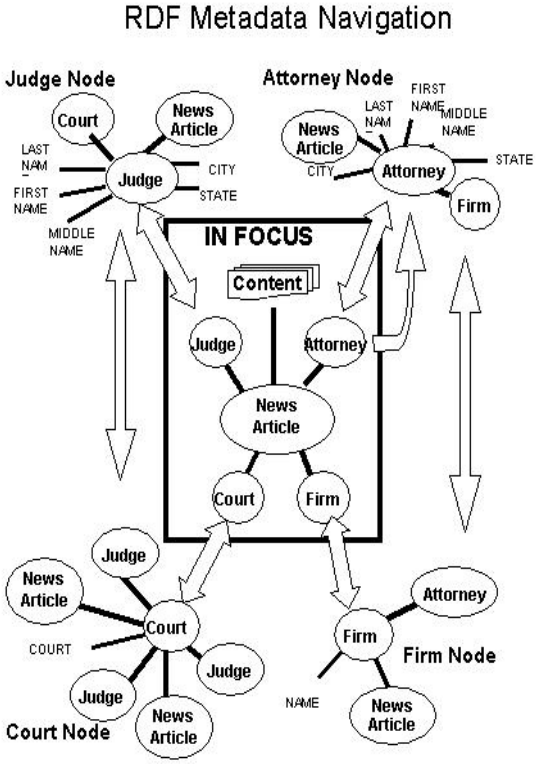

Figure 6: Navigation between related metadata

4 Conclusion

This application utilizes RDF/RDFS to build a data model that allows for easy maintenance of reference links embedded in content. This data model also facilitates development of metadata navigation. By just looking through metadata repository, the application can decide the best way to utilize rich information buried inside content repository.

We feel that this application can be extended to provide inferencing capability. The hard wiring of the logic inside the metadata repository does not currently provide any formalism to infer hidden relationships from the facts. Implementing this inferencing mechanism would bring us closer to our semantic web goal.

References

Christopher Dozier and Robert Haschart. 2000. Automatic extraction and linking of person names in legal text. *Proceedings of RIAO-2000: Recherche d'Informations Assiste par Ordinateur.*

Graham Klyne and Jeremy J. Carroll. 2004. Resource description framework (rdf): Concepts and abstract syntax. *http://www.w3.org/TR/2004/REC-rdf-concepts-20040210/.*

Ora Lassila. 2000. The resource description framework. *IEEE Intelligent Systems*, 15(6):67–69.

W3C. 1999. Resource description framework (rdf) model and syntax. *http://www.w3.org/TR/1999/REC-rdf-syntax-19990222/.*

W3C. 2004. Rdf vocabulary description language 1.0: Rdf schema. *http://www.w3.org/TR/2004/REC-rdf-schema-20040210/.*

Appendix

A A RDF Schema for data model of our application

```
<?xml version='1.0' encoding='ISO-8859-1'?>
<!DOCTYPE rdf:RDF [ <!ENTITY rdf 'http://www.w3.org/1999/02/22-rdf-syntax-ns#'>
<!ENTITY PeopleCite 'http://www.thomson.com/PeopleCite#'>
<!ENTITY rdfs 'http://www.w3.org/TR/1999/PR-rdf-schema-19990303#'>]>
<rdf:RDF xmlns:rdf="&rdf;" xmlns:PeopleCite="&PeopleCite;" xmlns:rdfs="&rdfs;">
<rdfs:Class rdf:about="&PeopleCite;Document">
<rdfs:subClassOf rdf:resource="&rdfs;Resource"/>
</rdfs:Class>
<rdf:Property rdf:about="&PeopleCite;ContentOfDocument">
<rdfs:domain rdf:resource="&PeopleCite;Document"/>
<rdfs:range rdf:resource="&rdfs;Resource"/>
</rdf:Property>
<rdf:Property rdf:about="&PeopleCite;DocumentURI">
<rdfs:comment> The Unique Identification Number of each document </rdfs:comment>
<rdfs:domain rdf:resource="&PeopleCite;Document"/>
<rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdfs:Class rdf:about="&PeopleCite;WSJ">
<rdfs:comment xml:space='preserve'>
<![CDATA[<<Wall Street Journal>>News Data Repository]]>
</rdfs:comment>
<rdfs:subClassOf rdf:resource="&PeopleCite;Document"/>
</rdfs:Class>
<rdf:Property rdf:about="&PeopleCite;PersonInDocument">
<rdfs:domain rdf:resource="&PeopleCite;Document"/>
<rdfs:range rdf:resource="&PeopleCite;Person"/>
</rdf:Property>
<rdfs:Class rdf:about="&PeopleCite;Person">
<rdfs:subClassOf rdf:resource="&rdfs;Resource"/>
</rdfs:Class>
<rdfs:Class rdf:about="&PeopleCite;Attorney">
<rdfs:subClassOf rdf:resource="&PeopleCite;Person"/>
</rdfs:Class>
<rdfs:Class rdf:about="&PeopleCite;Judge">
<rdfs:subClassOf rdf:resource="&PeopleCite;Person"/>
</rdfs:Class>
<rdf:Property rdf:about="&PeopleCite;PersonURI">
<rdfs:domain rdf:resource="&PeopleCite;Person"/>
<rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
<rdf:Property rdf:about="&PeopleCite;LastNameOfPerson">
<rdfs:domain rdf:resource="&PeopleCite;Person"/>
<rdfs:range rdf:resource="&rdfs;Resource"/>
</rdf:Property>
<rdf:Property rdf:about="&PeopleCite;FirstNameOfPerson">
<rdfs:domain rdf:resource="&PeopleCite;Person"/>
<rdfs:range rdf:resource="&rdfs;Resource"/>
</rdf:Property>
<rdf:Property rdf:about="&PeopleCite;MiddleNameOfPerson">
<rdfs:domain rdf:resource="&PeopleCite;Person"/>
```

<rdfs:range rdf:resource=”&rdfs;Resource”/>
</rdf:Property>
<rdf:Property rdf:about=”&PeopleCite;AffilicationOfPerson”>
<rdfs:domain rdf:resource=”&PeopleCite;Person”/>
<rdfs:range rdf:resource=”&PeopleCite;Organization”/>
</rdf:Property>
<rdf:Property rdf:about=”&PeopleCite;AddressOfPerson”>
<rdfs:domain rdf:resource=”&PeopleCite;Person”/>
<rdfs:range rdf:resource=”&rdfs;Resource”/>
</rdf:Property>
<rdf:Property rdf:about=”&PeopleCite;CitingDocumentOfPerson”>
<rdfs:domain rdf:resource=”&PeopleCite;Person”/>
<rdfs:range rdf:resource=”&PeopleCite;Document”/>
</rdf:Property>
<rdfs:Class rdf:about=”&PeopleCite;Organization”>
<rdfs:subClassOf rdf:resource=”&rdfs;Resource”/>
</rdfs:Class>
<rdf:Property rdf:about=”&PeopleCite;OrganizationURI”>
<rdfs:domain rdf:resource=”&PeopleCite;Organization”/>
<rdfs:range rdf:resource=”&rdfs;Literal”/>
</rdf:Property>
<rdf:Property rdf:about=”&PeopleCite;NameOfOrganization”>
<rdfs:domain rdf:resource=”&PeopleCite;Organization”/>
<rdfs:range rdf:resource=”&rdfs;Resource”/>
</rdf:Property>
<rdf:Property rdf:about=”&PeopleCite;AddressOfOrganization”>
<rdfs:domain rdf:resource=”&PeopleCite;Organization”/>
<rdfs:range rdf:resource=”&rdfs;Resource”/>
</rdf:Property>
</rdf:RDF>

B One snapshot of our metadata web application