Improving Patient Care in Transport Medicine through an Ontological Approach

Philip DePalo, Kyungeun Park, Yeong-Tae Song

Dept. of Computer & Information Sciences
Towson University
Towson, USA

+1 410 371 8049 pdepal1@students.towson.edu
+1 410 704 4662 kpark@towson.edu
+1 410 704 3557 ysong@towson.edu

ABSTRACT

Electronic Health Records enhance patient care across medicine enabling providers to view historical information from the patients chart, viewing results from participating agencies, sharing notes with providers outside their institution and utilizing integrated support systems which protect against medication errors and redundancies. The field of transport medicine revolves around emergent patient transports by ambulance, helicopter and plane. These are resource poor environments and, coincidentally, occur during the most critical times of the patient's condition. The need to support transport clinicians with the most valid pertinent information about each patient is the main focus to our research. We proposed an ontological approach around transport medicine protocols which associated multiple diseases and their associated symptoms. We have developed semantic queries using the patient's current symptoms as input and the query result is analyzed by an algorithm that we created to derive probable diseases. The algorithm uses types of associated symptoms based on the ontology to quantify a confidence level for each possible disease. If the disease ruled in, we presented this information to the clinician as part of a decision support system. We used this output to query the patient's existing EHR for relevant medical history regarding the current disease process. We provide both the probable diagnoses along with the patient's relevant history in a single XML resource document.

Categories and Subject Descriptors

• **Information systems~Expert systems** • **Applied computing~Health care information systems** • Applied computing~Health informatics • *Software and its engineering~Search-based software engineering* • Networks~Data center networks

General Terms

Algorithms, Standardization, Theory, Measurement

Keywords

Diagnosis support system, electronic health record, ontology, SPARQL query, transport medicine

IMCOM (ICUIMC)'14, January 9–11, 2014, Siem Reap, Cambodia.

1 INTRODUCTION

The ability to share medical data with a variety of institutions has come to the forefront of healthcare in the US and other countries. Currently documents are being created for a variety of patient encounters and reports. These documents are stored in large central databases so that others may access them for their own use as shown through the steps of Figure 1. When a clinician accesses this registry of information they are able to search for all documents available for any given patient. This is enabled by Health Level 7(HL7) Clinical Document Architecture (CDA) [1] standard which is an XML based version of the medical record produced by HL7 and ANSI-accredited standards developing organization for healthcare. Multiple types of CDA documents exist, where there is often a lot of information, some duplicated, among all these documents. In transport medicine, clinical providers need quick access to relevant information about the patient they are treating due to the time contingent environment in which they work. In such an environment, providers may not have the time or resources to sift through a large number of electronic CDA documents in order to diagnose the patient or determine their relevant past medical history as it relates to the current problem. CDA documents might contain a variety of similar terms relating to the same diagnosis repetitively, which may further delay critical treatment of the patient

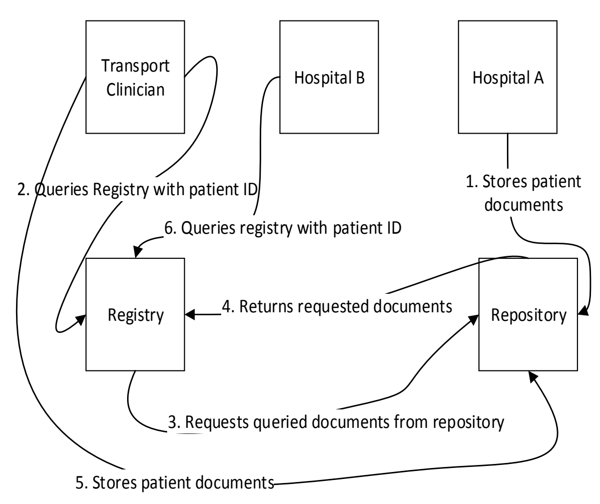

Figure 1. Transport Medicine Overview

population [2]. To date there has been no proposal for a standardized search through this type of data, especially a search that is meaningful and effective for the clinician [3]. Even though current search technology utilizes relational databases, their ability to represent relationships dynamically among each different element located in different locations is limited. To overcome such limitations, an ontological approach is investigated and considered. An ontological search is an attempt to go beyond simple keyword search. Ontologies support URI objects that can be uniquely identified across areas of all internet domains [4]. Agarwal [5] states that an ontology is, therefore, the manifestation of a shared understanding of a domain that is agreed between a number of agents. Thus such agreement facilitates accurate and effective communications of meaning, which in turn leads to other benefits such as inter-operability, reuse and sharing. The ability to share data is a meaningful use objective [6]. Ontologies have a lot more flexibility in handling their data than relational databases, but deductions cannot be made with such functionality [7]. Our focus is on the electronic health record (EHR) and its ability to consume, reuse and share available data within the EHR in order to achieve interoperability in transport medicine. This may prevent repeated tests, medication errors and potential redundancy.

The rest of this paper is structured as follows. Identifying the available resources that create the health IT domain and how they play a role in semantic queries of the EHR to support transport medicine, in section 2. In section 3 we discuss semantic interoperability and the steps we took to begin implementation of our Transport Record Summary (TRS) Consolidator. In section 4, we examine actual queries of our medical ontology that we created in OWL web ontology language and show how to create a unique CDA document for transport medicine that is shared with the Health Information Exchange (HIE). Finally, a summary of our work and conclusions are presented in section 5.

2 HEALTHCARE EXCHANGE DOMAIN

In order to achieve interoperability among healthcare institutions and various government agencies, public health and private sector participants have organized standards that define a network of systems that will support the exchange of electronic health data. HealtheWay, formerly the national health information network, is an organized body of smaller HIEs. An HIE is a network of healthcare organizations such as hospitals, rehabilitation clinics, doctors' offices and laboratories that have agreed to participate in an exchange of records to support healthcare. These records are individual documents based on individual encounters with each of these organizations, the whole of which comprises a patient's EHR. Therefore an EHR is a conceptual single record about a patient's entire medical history, which consists of multiple documents from a variety of encounters. If each provider were to create their own proprietary document, the communication among providers would be problematic. HL7 has enforced a standardized set of rules to govern these documents and as such, the Clinical Document Architecture (CDA) [1] is the governing standard for electronic medical documents. The agreements or business relationships among organizations is based upon the ebXML standard for e-commerce.

2.1 *ebXML*

ebXML is comprised of multiple components: Business Process Models, Core Components, Messaging Services, Registry and Repository and Trading Partner Information. Business process models define the basic rules of specific transactions among trading partners. These rules are based upon simple business transactions such as a customer inquiring about an item, the seller providing the information and the price and finally the buyer agreeing to purchase the item. The resulting business process specification schema is part of the environment created for business collaboration [8]. Core Components are business objects. Objects define a real-world concept such as a customer. Objects are part of the building blocks of ebXML. Registry and repository concepts in ebXML are similar to a database. The registry is a listing of information that is stored in the repository. The registry allows for fast indexing without the need to access the actual data elements in order to direct an inquiry. Trading partner information is an electronic contract that defines interaction protocols. It does not contain any business information, but rather the technical system specifications.

2.2 *The Registry*

The registry in ebXML is a database of XML artifacts, schemas, data elements and metadata which are details about those artifacts. There are many benefits to using a registry. It does promote services discovery and maintenance of registered objects. It allows for fast indexing and queries for specific artifacts as well as enabling security and efficient control versions of artifacts. It provides availability and reuse of various artifacts. It allows for multiple users to improve current artifacts and

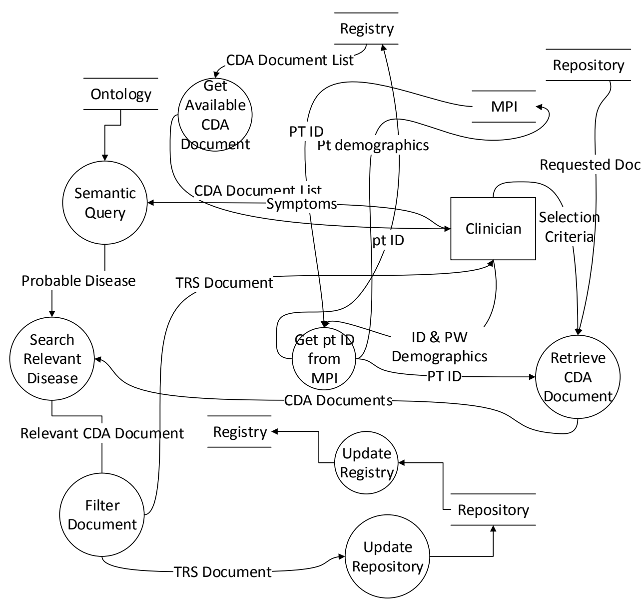

Figure 2. Transport Messages

```
PREFIX patient: <
http://www.lifelinetransports.org/Transports/patient#>
PREFIX symptom:
<http://www.lifelinetransports.org/Symptoms>
SELECT ?Disease
  WHERE {
            ?who patient:fullName ?givenName .
            ?who patient:symtpom ?Symptoms.

 http://lifelinetransports.org/Symptoms
              }
```

Figure 3. SPARQL Query

submit new ones to be further enhanced.

The technology can be equally applicable to a registry in healthcare information technology. The standard is called Cross Enterprise Document Sharing (XDS). The goal is for a healthcare institution or entity to be able to provide and register a document set. The documents are provided to the repository and then the repository is asked to register them with the registry. At the same time, the documents may be provided directly to a document recipient. This request must contain the metadata, such as document type, provider, document ID, that describes the documents, at least one object per document, a link to the new documents and references to existing documents. If the request does not complete, the repository will send an error.

2.3 *The Repository*

Though discussed secondly, the repository is actually the gateway to storing documents and initiating the storage of information about those documents, the metadata. A request to store a document is sent to the repository and the repository then will update the registry with information. More transactions requests will hit the registry, but it will be the repository that enables the registry to have a reason to exist. There is no need to store information about documents that do not exist. Once the information is stored, then the registry can provide multiple users with much information about any patient that is queried. The goal of the repository is to act as the heavy hitter storing the large files for use in patient records as seen in Figure 2.

3 ONTOLOGY

Ontology is a specification of a conceptualization, that is, it's a description of its context and the relationships that can exist between different objects within its structure [9]. Semantic interoperability focuses on the relevance of the transmitted information to both organizations. If an organization sends data that is relevant to them, but that has no meaning for the recipient, then it is not interoperable [10].

```
http://www.lifelinetransports.org/Transports/patient#info,
http://www.lifelinetransports.org/Transports/patient#fullName,
"John Doe"

http://www.lifelinetransports.org/Transports/patient#info,
http://www.lifelinetransports.org/Transports/patient#symptoms
, "Chest Pain, Bradycardia, Chest Discomfort "

http://www.lifelinetransports.org/Transports/patient#info,
http://www.lifelinetransports.org/Transports/patient#transport
Method, "Ambulance"

http://www.lifelinetransports.org/Transports/patient#info,
http://www.w3.org/1999/02/22-rdf-syntax-ns#type,
http://www.lifelinetransports.org/Transports/patient#person
```

Figure 4. RDF Triples

Semantic interoperability will provide unique queries among multiple diverse data sources. The results of these queries will provide useful information that may not have otherwise been located due to the time constraints of patient transport. Very little time is spent with the patient during an emergency transport. During this time focus is on diagnosis and emergency intervention. Using unique query results that are based on the patient's own available medical history is an intelligent way to make the most of the limited time available. This information can prove the difference between similar medical issues that require significant different treatments. Currently a patient transported by 911 receives immediate care for their chief complaint, but previous medical problems are unavailable and not considered. A semantic query for this information could provide essential medical records of previously documented problems, allergies and medications that could prove lifesaving to an unresponsive individual without a spokesperson. Given the resource poor, time constrained environment surrounding 911 emergencies, queries that utilize logic to quickly and usefully produce results are paramount. Existing communications technology used in transport medicine does not provide the ability to exchange health information and can hinder essential care.

3.1 *Semantic Interoperability*

In order to achieve semantic interoperability of electronic medical records in transport medicine, we researched the basis of the resource description framework (RDF). RDF is the basis for the semantic web ontology. RDF decomposes any knowledge into smaller pieces called triples [11]. To make the relationship between elements as meaningful as possible, we determine the level of detail for each triple. The more detailed this information becomes, the more useful the ontology. For instance *heart problems* are not detailed enough. This term can be broken further down into myocardial infarction, congestive heart failure, cardiomyopathy, etc. Figure 2 details the process flow of information through a semantic query of CDA documents. The clinicians must first provide demographic information about the patient so that their ID can be discovered from the Master Patient Index (MPI). The MPI stores multiple IDs for patients and is able to cross reference institution specific IDs for each patient based on demographic information. With this ID, we are able to query a list of available CDA documents and eventually choose the CDA documents we want to search. The patient's symptoms are then processed against our medical ontology described in a web ontology language (OWL) file. This ontology associates symptoms with possible diagnoses. Then based on the type of symptoms, it implies a confidence level to the most likely differential diagnosis for the current patient. These differential diagnoses are returned and a search is made on the list of CDA documents for those specific sections that relate to past medical history. A single XML file called the Transport Record Summary (TRS) is created along with a list of the probable diseases in situations where there is no relevant past medical history. This CDA consolidation using the TRS constructor and decisions support system accelerates the time to treatment for the clinicians [12]. The TRS constructor decomposes CDA documents by the XML parser and these decomposed documents have specific sections extracted using the semantic analyzer technology explained later in Figure 7.

3.2 *RDF Triples*

Our resource is here: http://www.lifelinetransports.org/Transports/patient#info. This is shown in Figure 4. The details of our message state there is a person identified by the above resource whose name is John Doe whose location is Johns Hopkins whose transport method is an ambulance. We break down this information into the RDF subject, predicate and object. Our subject is our resource URI; our objects are John Doe, Symptoms and ambulance. The predicates that describe those objects are: whose name is, whose symptoms are and whose transport method is. The subject's type is described by the World Wide Web Consortium (W3C) as http://www.w3.org/1999/02/22-rdf-syntax-ns#type, which is a person. We then form the RDF Triples. The RDF triple use a URI to describe the unique objects and then shows a relationship between things as well as the two ends of the link. Since HL7 messages ultimately are described by XML and specifically ebXML constructs, we convert the above RDF triples into RDF/XML in order to be exchanged, stored, etc. Our conversion of RDF to RDF/XML is shown in Figure 5.

3.3 *Querying RDF graphs*

With our data now defined using RDF, we use SPARQL Protocol and RDF Query Language (SPARQL), a recursive acronym. SPARQL allows us to query and retrieve data from RDF graph formats as previously described. SPARQL can query disparate data sources, which in healthcare is an excellent benefit. The ability to query records from multiple healthcare sources is invaluable to obtaining the most relevant information about a patient. In our example, we can query for the specific name of a patient across multiple transport requests based on the location of the request as seen in Figure 3. Another example would be to query a specific set or all sets of medical records across all healthcare institutions based on a single patient identifier. The next logical step in our example would be to query information about the specific patient identified from the first query using only the healthcare institution requesting the transport and the timestamp on the patient's current admission to the requesting institution. This would provide the transport unit with only the specific medical records for the patient currently being transported. The advantage of this specific unique query is that only relevant information about the patient's problems and their transport needs is received.

Our research produces context aware results. A query for a patient suffering from, e.g., an acute urinary tract infection will produce medical record results that include previous treatment for bladder cancer. A SPARQL query using RDF is able to produce these types of linked results.

When information is located these files are brought together based on a unique ID and can be stripped of all redundancies by comparing results to those results already harvested through the TRS constructor. This stripped set of documents are then brought together as a single XML file with multiple URIs to each bit of data contained in the multiple documents. This provides a single time use document for providing relevant information during patient transport. Using data from this document, plus new data generated during the transport another patient document is created based on the TRS Profiles from the IHE. This document details the specific transport and can be used in similar fashion by the receiving medical institution as the above process describes.

3.4 *XML Parsing*

Using LINQ (Language-Integrated Query) for XML we can parse the XML file using the XElement method shown in Figure 6.

We load the retrieved CDA XML documents into the XElement method using the path variable. We then determine any namespaces associated with the XML document, so that we can strip them from the appropriate section headings while searching terms in the standardized set. Next, since our semantic search returns relevant key words, we have to place these multiple words into a string array. We then search through the section headings of the CDA document for those words using the "*foreach*" loop.

```
<?xml version="1.0" encoding="UTF-8"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:patient="http://www.lifelinetransports.org/Transports/patient#" xmlns:hospital="http://www.lifelinetransports.org/Symptoms/">
        <rdf:Description rdf:about="http://www.lifelinetransports.org/Transports/patient#info">
                <patient:fullName>John Doe</patient:fullName>
                <patient:Symptoms> Chest Pain, Bradycardia, Chest Discomfort </patient:Symptoms>
                <patient:transportMethod>Ambulance</patient:transportMethod>
        </rdf:Description>
        <rdf:Description rdf:about="http://www.lifelinetransports.org/Symptoms/">
                <patient:Symptoms> Chest Pain, Bradycardia, Chest Discomfort </patient:Symptoms>
        </rdf:Description>
</rdf:RDF>
```

Figure 5. RDF/XML

```
XElement xml = XElement.Load(path);
XNamespace ns =
xml.GetDefaultNamespace();
foreach (string str in lstOfStrings)
{
   IEnumerable<XElement> symptoms =
   from item in xml.Descendants(ns +
   "section")
   where item.Element(ns +
   "title").Value.Contains
   (DropDownList1.Text)
   where item.Element(ns +
   "text").Value.Contains(str)
   select item;
   foreach (var complaints in
   symptoms)
      {
      _writer = new

TextBoxStreamWriter(txtConsole);
      Console.SetOut(_writer);
      Console.WriteLine(complaints);
      }
   Console.ReadLine();
}
```

Figure 6. XML Parsing

This loop queries each section for the relevant keyword information. Once located, the block is extracted to the console writer which in turn begins to build a new XML tree with that block. It is possible that the block will contain more than one word, though it doesn't match. In this case, it truncates everything before and after that word and only imports the single word plus its encapsulated XML heading. Searching multiple documents will yield multiple blocks with the same section title, if this is the case, it will then search the newly generated XML document for the same section title and if there is a match it will add the word to that section, therefore removing redundancies.

When searching for relevant information, relevancy is based on a few factors. Relevancy can be based on the most recent visit (DATE) or by type of document (TYPE). A person who has been seen recently multiple times for the same problem at the local hospital and is now requesting a 911 ambulance might require a search by DATE to see what has been going on recently. An interfacility transport for a patient who needs a higher level of Oncology care might require a search by TYPE to see their entire Oncology history. In addition there is no hard fast rule and a search of both kinds might be required.

Our method searches for the parts of these documents that are relevant to the transport and does not require the entire document. This information is specified by the specific URIs shown in Figure 4. These records can be updated with each visit [13]. One example would be when a patient is scheduled for transfer to a major medical institution for treatment after reaching some Hospital C. Utilizing a semantic search, we would be able to query all the information that is relevant to this patient based on the current chief complaint. The relationship can be defined through ICD-10-CM codes or SNOMED as two examples. ICD-10-CM codes use a hierarchical structure to define conditions and traverse this structure so as to provide links to additional diagnoses for each patient. A person who has an ICD-10-CM code of M81.0: "Age –related osteoporosis without current pathological fractures" under the Diseases of the Musculoskeletal System and Connective Tissue search item may link to another category, M19.9: "Secondary Osteoarthritis, Unspecified Site". These code based systems create a symbiotic environment for semantic queries. The TRS profiles provide similar coding systems for patients and are a source for additional linked information on patients who are transported. Semantic queries can also be used in the public health domain to find out about certain common epidemics, either locally or globally based on queries across the healthcare domain. These queries will link common medical conditions and histories leading up to the current epidemic and will find common links.

Our queries are serialized to make a single patient specific XML file that can be updated and stored in the transport RDF store. This single XML file is retrievable by the receiving institution for similar actions.

4 DIAGNOSIS SUPPORT SYSTEM

An ontological search can help users who are not familiar with the current domain find relevant information. Given a search for "Deafness", the results themselves may provide more detailed scientific terms to narrow the search and make it more specific to the disease process, perhaps a search on "Cochlear Implants" [4]. Our search may return diseases based on symptoms, but we can then use the related symptoms with each returned disease to see if those symptoms as a whole represent a high confidence level towards our main disease process based on the current symptoms. This can rule out semantically related, but medically insignificant differential diagnosis from the current issue and provide decision support for appropriate treatment.

4.1 *911 Scenario*

Patient Smith experiences severe chest pain, a sure sign he

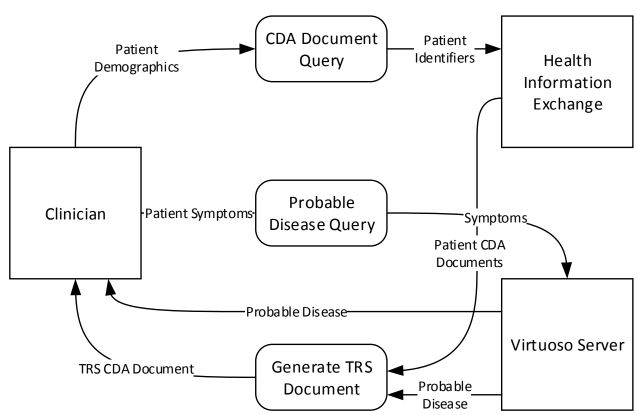

Figure 7. Module Diagram

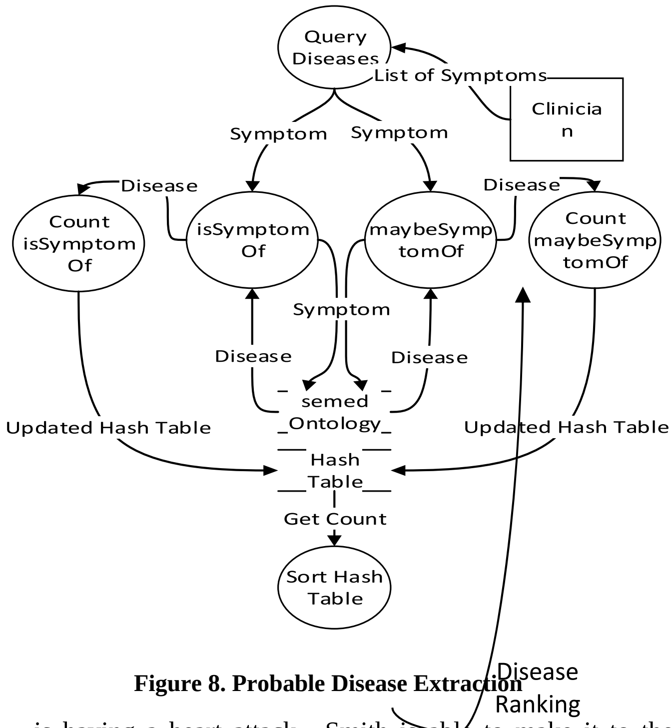

Figure 8. Probable Disease Extraction

is having a heart attack. Smith is able to make it to the phone to dial 911 and report his emergency, but soon after collapses to the ground unconscious. Patient Smith has never contacted 911 before and the fire department is not familiar with him or his medical history. Upon arrival paramedics are able to find identifying demographics and log into the local health information exchange where these demographics are matched up with the patient ID using the master patient index, a list of previously used IDs associated with his demographics as shown in Figure 7 under the *CDA Document Query* module. Now able to access his previous health record's CDA documents, the paramedics input his symptoms and search possible diseases both in his past medical history and as a new diagnosis as seen in the Figure 7 *Probable Disease Query* module. This completed record is created by the Figure 7 *Generate TRS Document* module and becomes a new CDA document based on our Transport Record Summary (TRS) profile [12] and will continue to guide treatment for this unconscious patient who otherwise would have presented as a mystery to the paramedics.

A similar study on ontological queries in Electronic Health Records in the Massachusetts General Hospital was performed in 2011 [13]. This study sought to search EHR using ontological methods for relevant past medical history in order to avoid excessive radiology studies. The results showed that quick results provided physicians with salient data that could aid in their decision about radiology imaging as part of their work up [14]. The study indicates similar results to our search across EHRs in the emergency medical services field, in that it is limited by the available information at the time of the query and must ultimately be validated by a clinician. It was used as a tool in the decisions support process, but it was limited to providing a read only resource. Our method takes the results and creates a new CDA document based on our previous TRS profile and incorporates suggested differential diagnoses to the list to guide the provider through their treatment. Currently the only way to view CDA documents is to select individual documents, sorted by type, and view them one at a time. Our approach streamlines this process for the transport environment where time is a very limited resource; however, our methodology can be applied to other areas of healthcare where viewing a single medical document can prove advantageous. This could be in the emergency department setting or as an anesthesia report during preoperative surgery procedures.

4.2 *Patient Symptom SPARQL Query*

Our disease and symptom ontology was created in RDF by Using Protégé [15]. Protégé is a software product that allows one to build an ontology using classes and keywords and then define the relationships between those keywords and the classes through a variety of operands that describe how they are joined or disjoined. We used Openlink's Virtuoso [16] to query the RDF file that we have created. Virtuoso is a server that hosts an ontology file and supports SPARQL queries. When defining a set of queries, a prefix

```
public class Probable_disease {
     string disease_name;
     int shouldHaveSymptom_count;
     int mayHaveSymptom_count;
     Probable_disease( string t, int a,
     int b) {
            disease_name = t;
            shouldHaveSymptom_count = a;
            mayHaveSymptom_count = b;
     }
     string disease_list (){
      return ( disease_name + ":" +
      shouldHaveSymptom_count + "major
      matches" + "and" +
      mayHaveSymptom_count + "minor
      matches") }}

  Foreach (string disease_name in
   hashtable Temp_PDS) {
   If (Search_hash_table(disease_name)){
     shouldHaveSymptom_count =
     shouldHaveSymptom_count + 1;
    }
 // if the disease doesn't exist in   table
   Else {
     add_hash_table(disease_name);
     shouldHaveSympton_count =
      shouldHaveSymptom_count + 1;
     }
     }
```

Figure 9. Pseudo Code Disease Algorithm

for the semantic query needs to be set. This prefix defines the ontology file we will be using for the specific query. We used the prefix “semed” and declared it in the following example.

```
PREFIX semed:
http://www.semanticweb.org/semedicine
/ontologies/2013/5/semedicine-
ontology-7#
```

Once set, we then used a select statement to find out which disease a certain symptom is a subset of. In this example we search using the symptom Dyspnea, or shortness of breath:

```
SELECT ?d
WHERE { semed:symDyspnea
semed:isSymptomOf  ?d . }
```

This returns a list of results:

```
(?d =
<http://www.semanticweb.org/semedicin
e/ontologies/2013/5/semedicine-
ontology-7#CarbonMonoxidePoisoning> )
(?d =
<http://www.semanticweb.org/semedicin
e/ontologies/2013/5/semedicine-
ontology-7#NearDrowning> )
(?d =
<http://www.semanticweb.org/semedicin
e/ontologies/2013/5/semedicine-
ontology-7#Overpressurization> )
```

The returned results describe diseases based on the provided symptom. The *?d* describes the matching disease. The URI refers to the prefix *semed* (already defined) and the information following the # sign lists the actual disease. So the above results refer to three diseases: Carbon Monoxide Poisoning, Near Drowning and Over-pressurization. We can also search the opposite information based on the results. We can search which symptoms may be symptoms of the previous result set. In this example we use the first returned result *CarbonMonoxidePoisoning*:

```
SELECT ?s
WHERE { ?s semed:maybeSymptomOf
semed:CarbonMonoxidePoisoning . }
```

This returns a list of results:

```
( ?s =
<http://www.semanticweb.org/semedicin
e/ontologies/2013/5/semedicine-
ontology-7#symVomitting> )
( ?s =
<http://www.semanticweb.org/semedicin
e/ontologies/2013/5/semedicine-
ontology-7#symUnconscious> )
( ?s =
<http://www.semanticweb.org/semedicin
e/ontologies/2013/5/semedicine-
ontology-7#symSeizure> )

( ?s =
<http://www.semanticweb.org/semedicin
e/ontologies/2013/5/semedicine-
ontology-7#symBurns> )
```

The returned result set here is similar in format to the previous result set, except the “*?s*” refers to the symptoms

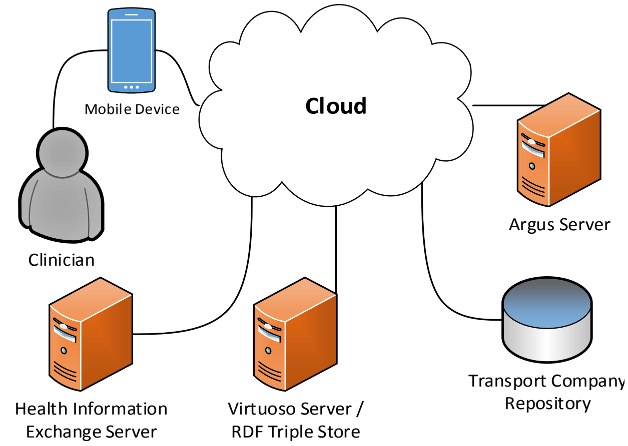

Figure 10. Deployment Diagram

and the actual symptom appears after the # symbol. The *sym* before the symptom identifies it as a symptom and so it is stripped from the result set to give us the following above returned symptoms: Vomiting, Unconscious, Seizure and Burns. We continue these searches through the remaining original result set including *NearDrowning* and *Overpressurization*.

4.3 *Probable Disease Algorithm*

With a valid ontology in place, we can then include its function within the TRS consolidator. Figure 8 shows the software architecture diagram for probable disease extraction. Using a list of symptoms provided by the clinician, the system queries the ontology for probable diseases. Lists of diseases that must have the symptom are listed with an attribute “isSymptomOf” and those that may or may not have the symptom are listed as “maybeSymptomOf”.

Lines 1-11 show the Probable disease class and the output table in Figure 9.

In lines 12-14 of Figure 9, for each symptom with a corresponding disease that is returned, the counter in the “hash table” of diseases is incremented by one for the specific disease. Thus searching for probable diseases based on the confidence level counter associated with each symptom and corresponding disease. Using a hash table allows for quick search each time it is to be incremented, vs. an array which requires the entire array to be search each time as opposed to jumping to the specific associated disease in the hash table. Using iterative steps we can count the number of should have symptoms and may have symptoms and add +1 to the counter variable seen in line 14 of Figure 9.

In line 15, if the disease does not exist in the count, add the name to the table of probable diseases seen in line 16 and set the count to +1 in line 17.

The deployment diagram in Figure 10 shows the Argus Server which provides a cloud based application to run the probable disease algorithm from any mobile device. A clinician can connect to the Argus server which in turn facilitates communication between the HIE, our developed ontology in the form of a RDF triple store on the virtuoso

Myocardial Infarction: 2 major matches and 2 minor matches
CarbonMonoxidePoisoning: 0 major matches and 2 minor matches
Overpressurization: 0 major matches and 2 minor matches

Figure 11. Confidence Results before Filter Criteria

server and the local repository of the transport company where the TRS document is stored for future HIE access.

Finally we can clinically rule in or rule out a probable disease by sorting the hash table with those diseases that have "isSymptomOf" count of at least 1 and "maybeSymptomOf" count of > 2 that was created in line 11. The limits of 1 and 2 respectively above are based on the correlation of minor symptoms to major symptoms of a specific disease. That is to say if a person experiences a symptom listed as a major symptom, the probability that the person is suffering from that specific disease is high. In addition if a person exhibits symptoms that are considered minor symptoms of the same disease, this adds to the probability of being diagnosed with that specific disease. Since minor symptoms can also be symptoms of other diseases, thus we must include a minimum limit value to the number of minor symptoms that must be included to consider the probable disease valid. An example of such symptoms associated with a disease would be a person experiencing chest pain, ST elevation in their EKG, nausea and dizziness. Chest pain and ST elevation are major symptoms associated with a Myocardial Infarction (Heart Attack). In addition to these major symptoms, nausea and dizziness can also be present; however nausea and dizziness can also be present when a person suffers from carbon monoxide poisoning, as well as other diseases. However, since chest pain and ST elevation are NOT major symptoms of carbon monoxide poisoning, it will be ruled out. We will then count the number of major and minor symptoms and discover that for the heart attack there are 2 major and 2 minor symptoms present. We would initially still list carbon monoxide poisoning in the hash table and it would have 0 major symptoms and 2 minor symptoms. Thus when we filter against our rule in criteria, carbon monoxide poisoning would be ruled out and only a heart attack would remain. Validation of the above ranking criteria will ultimately be performed by a licensed clinician or consensus of such. Those that will be discarded will have values less than the rule specifies. The specific rule is based on a confidence level of all inclusive symptoms available and a set of results is seen in Figure 11.

5 CONCLUSION

We proposed an ontological approach to treating patients in an emergency situation. This helps emergency medical services providers assess a patient according to their current symptoms. The proposed approach searches all available electronic health records in the health information exchange using the symptoms as a seed for an ontological query. The developed medical ontology associates diseases with symptoms and assigns a confidence level to each disease based on the number of present associated symptoms. Based on the chief complaints from a patient, the proposed algorithm queries the ontology and returns a list of probable symptoms. Using this list we associated each disease with each returned symptom and if a symptom returned multiple times associated with the same disease we include this disease in our list of probable diseases. In addition, the list of symptoms and related diseases are used to search the EHRs and return a single document with all relevant past medical history based on the current chief complaint and set of symptoms.

6 REFERENCES

1. **7, Health Level.** Clinical Document Architecture. *Health Level 7.* [Online] [Cited: May 1, 2013.] http://www.hl7.org.

2. *Supporting Ontology-based Keyword Search over Medical Databases.* **Kementsietsidis, Anastasios, Lim, Lipyeow and Wang, Min.** s.l. : AMIA, 2008. AMIA 2008 Symposium Proceedings. pp. 409-413.

3. *XOntoRank: Ontology-Aware Search of Electronic Medical Records.* **Farfan, Fernando, et al.** New York : s.n.

4. *An Ontology-Based Intelligent Agent System for Semantic Search in Medicine.* **Yang, Jung-Jin.** Berlin : Springer, 2003, Vol. 2891.

5. *International Journal of Geographical Information Science.* **Agarwal, P.** 5, Leeds : Taylor & Francis Group, 2005, Vol. 19, pp. 501-536. ISSN 1365-8816.

6. *The "Meaningful Use" Regulation for Electronic Health Records.* **Blumenthal, David and Tavenner, Marilyn.** 6, Massachusetts : The New England Journal of Medicine, 2010, Vol. 363.

7. *Ontolologies versus relational databases: are they so different? A comparison.* **Martinez-Cruz, Carmen, Blanco, Ignacio J and Vila, Amparo.** Granada : Springer, 2012, Vol. 38. DOI 10.1007/s10462-011-9251-9.

8. **Chiu, Eric.** *ebXML Simplified: A Guide to the New Standard for Global E-Commerce.* New York : John Wiley & Sons Inc., 2002. ISBN 0-471-20475-7.

9. **Gruber, T. R.** *A Translation Approach to Portable Ontology Specifications.* s.l. : Academic Press, 1993.

10. *Healthcare Interoperability through Enterprise Architecture.* **DePalo, Philip and Song, Yeong-Tae.** Jeju Island : ICUIMC, 2012.

11. **World Wide Web Consortium.** RDF - Semantic Web Standards. *World Wide Web Consortium.* [Online] MIT, ERCIM, Keio. [Cited: July 20, 2012.] http://www.w3.org/RDF/.

12. *Healthcare Interoperability: CDA Documents Consolidation Using Transport Record Summary (TRS) Construction.* **DePalo, Philip, Park, Kyung Eun and Song, Yeong-Tae.** Vegas : Springer, 2013. Human Computer Interaction, LNCS 8005. Vol. II, pp. 56-65.

13. *Implementing Interoperability using an IHE Profile for Interfacility Transport.* **DePalo, Philip and Song, Yeong-Tae.** Jeju Island : ACIS, 2011.

14. *Validation of an Ontology Based Search Engine for the Electronic Medical Record: Application in the Emergency Department Setting.* **Krishnaraj, Arun, et al.** Orlando : American Medical Informatics Association, 2012.

15. **Stanford Center for Biomedical Informatics Research.** *Protege.* Connecticut : s.n.

16. **OpenLink Software.** *Virtuoso Universal Server.* 2013.

17. **World Wide Web Consortium.** Turtle. *World Wide Web Consortium.* [Online] MIT, ERCIM, Keio. [Cited: July 20, 2012.] http://www.w3.org/TR/2011/WD-turtle-20110809/.