Abstract

Traditional health record systems are ink and paper based. Slowly we move toward electronic health records, allowing for greater mobility, connectivity, and usability of medical records. We present a system based on Semantic Web technologies that enables physicians to visualize patient data in ways related to the type of medicine they practice. Taking advantage of open data from NIH, the system can present the physician with drugs related to their specialty, drugs for conditions the physician may treat, etc. It is designed to be generic and extensible to any number of specialities.

1 Introduction

Medical record keeping is slowly being dragged into the age of electronic computers. In some cases the dragging is done by insurance companies and hospitals (Kaiser, VA); in others (and increasingly commonly) by upcoming government regulation,[15]. Along the way medical records for patients will move from an ink, paper, filing cabinet system to an Electronic Health Record (EHR). This conversion brings new problems and challenges, but also new never-before-possible capabilities, to the fore.

Many EHR systems exist. Some are better than others. These are the realities of software systems, especially ones as large and complex as EHR management systems. The authors’ positions on technologies to implement such a system should come as no surprise to the reader. A system based on technologies that implement semantics in a standard well-understood format are not only valuable to the portability of records in an EHR but can make the data that exists more powerful by encoding what it means and any relationships it has.

1.1 Motivating Use Case

In modern medicine a patient rarely has only one physician. Each physician has the power to prescribe medications to a patient. As a result, a patient often has multiple physicians prescribing multiple medications.

Physicians cannot know the use and side-effects of each and every drug, there are simply too many. A patient may be prescribed a medication for one condition that has side-effects that affect another condition or interact with another drug they take. Finding the root cause of such occurrences can be difficult to sort out. Being able to quickly visualize the medications a patient is taking and the results of lab tests can make the task much easier.

Here we present a possible flow of events where our system would be useful.

1. A Patient with diabetes visits their General Practitioner for a routine check-up.

2. General Practitioner views record and sees that Patient has been keeping good track of their blood sugar and that it remains inside healthy levels. However, the Patient’s blood pressure is high. Since Patient already takes a drug to lower their blood pressure, General Practitioner increases the dosage of the drug in an attempt to further lower the Patient’s blood pressure.

3. Patient notices their blood sugar readings are becoming more erratic. Patient makes appointment with their Diabetes Specialist.

4. Diabetes Specialist views Patient’s medical record. It shows stable blood glucose readings for quite some time, but they have become more erratic recently. Patient claims that no behavior has changed.

5. Diabetes Specialist notices that Patient is taking a drug to lower blood pressure. Diabetes Specialist decides to visualize Patient’s blood glucose readings along with their blood pressure medication dosage.

6. The visualization clearly shows that the erratic readings begin after the increase in dosage of blood pressure medication. Though the increased dosage may not have caused the erratic blood glucose, the correlation is clear.
7. Diabetes Specialist modifies Patient’s blood pressure prescription. Diabetes Specialist chooses a drug that is known to have no such interactions with blood glucose or diabetes symptoms.

2 Methods

Here we will highlight the methods we used in developing the system. The topics will be somewhat technical, but not so much as to scare those uninitiated with the semantic technologies used.

2.1 Ontology

We first designed an ontology. This ontology maps a Physician to their specialty and uses this specialty to map the Physician to different diseases they might treat. The ontology is expressed in the Web Ontology Language (OWL). Using this ontology is a key part of our project, but also the most uncertain. The ontology was designed by people with little medical experience. In addition no real source of authority was found on medical specialties, indeed there are too many sources of authority to consider them all. Instead we decided to implement an ontology which was complete enough for our purposes and simple enough to be easily extended by some future work.

Figure 1 presents the entirety of the ontology we developed. It is clearly very simple. It presents us with the basic items with which our system deals and the relations between them. The most complex classes are MedicalPractitioner and ProblemDrugs. A MedicalPractitioner has a Specialty (indicated by the yellow arrow). A ProblemDrug encodes the relationship that a Drug can be used to treat a Problem. The class Problem does not exist in the ontology. A collection of medical problems would be beyond the scope of this present system. ProblemDrugs does contain a data property of the disorder or disease it was prescribed to treat, but this is not present in Figure 1 since it is not a class.

2.2 Data

We used a combination of data sources. For patient data, we used sample synthetic patients provided by Children’s Hospital Boston[11]. This data incorporates a variety of items and events found on a typical medical record: medications taking and taken, allergies, lab test results and problems reported by the patient. The data is provided in tab-separated flat text files. The data comes with a Python script to transform the flat text files into RDF-XML format. For the purposes of this system, though, we converted the data using csv2rdf4lod developed by Tim Lebo to convert the data into Turtle syntax for RDF. The resulting RDF was then loaded into a TDB triple store.

Some of the data from CHB was modified by the authors to facilitate greater functionality of the system. For example, included in the data is a list of medications patients are taking or have taken. We modified this data to separate out the drug name and the dosage of the drug from the provided single “Name” column. This allows the system to represent and visualize drug dosage over the time the patient is taking the drug.

Additional data was invented by the authors for the purposes of illustrating system functionality and usefulness.

2.3 Retrieval

Once the data is created and/or modified it is loaded into the TDB triple store. This triple store has a Joseki SPARQL endpoint attached. Through this endpoint SPARQL queries are made over the data in the triple store. The SPARQL queries query data not just in the triple store controlled by the authors but data provided by NIH as a part of RxNav[7].

2.4 Reasoning

Our system further enhances EHR by support automatic reasoning over EHR with semantic technologies. We use Jena to create an ontology model from reading three RDF files: the file that encodes the ontology, the file that contains our converted RDF data, and the file that encodes a synthetic drug regulation. The synthetic drug regulation impose dosage constraints on EHR, i.e. drug dosage should be within the proper range: $\text{minimum} \leq \text{value} \leq \text{maximum}$. We
can easily encode these constraints in OWL, since OWL has support for value constraints. We also have the Pellet reasoner to conduct reasoning over the whole ontology model to find out the EHR with drug dosage that is either below the minimum or above the maximum. This reasoning approach is highly flexible, because not only can we adopt new drug regulations by reading the corresponding encoded RDF files, but also switch among several drug regulations by choosing the RDF files dynamically. Having such automatic reasoning instead of manually checking the EHR could save medical practitioners’ time and efforts.

2.5 Data Presentation

The main presentation method is a simple line graph. The graph shows the dosage of drugs prescribed to a single patient over the time they were prescribed. The variance of drug dosage over time can be caused by a variety of events. The most surprising is a change made by a physician other than the one who originally prescribed the drug. In this case, without a visualization the prescribing physician would have to look through the patient’s data themselves to check for such a case. In our system, this data is presented in an easy-to-read fashion that facilitates easier inspection of patient records, even if the physician is not explicitly searching for modifications in drug dose, such a visualization may lead them in the correct direction which may have taken weeks or months to find otherwise.

2.6 Technologies Utilized

The technology stack of the system is depicted in Table 1. The front end are web pages written in HTML and Javascript. We use Google Visualization API to visualize the time series of the drug dosage and lab result. More specifically, we utilize dygraphs, which is an open source JavaScript library, because it produces interactive, zoomable charts of time series. We use SPARQL as the query language for retrieving data from the triple store or Jena model. In the middle layer, we set up a Joseki engine to provide a query point. To support reasoning over our knowledge base, we adopt the combination of Jena and Pellet. At the back end, we use csv2rdf4lod-automation for converting the raw data into RDF files. These RDF files are loaded into TDB and also Jena model. Our ontology is developed in OWL 2 using Protégé.

### Table 1: The technologies used by the system along with their purpose.

<table>
<thead>
<tr>
<th>Position</th>
<th>Technology</th>
</tr>
</thead>
<tbody>
<tr>
<td>Front End</td>
<td>HTML, Javascript, Google Visualization API, SPARQL</td>
</tr>
<tr>
<td>Middle</td>
<td>Joseki, Jena, Pellet</td>
</tr>
<tr>
<td>Back End</td>
<td>csv2rdf4lod-automation, File System, TDB</td>
</tr>
<tr>
<td>Ontology</td>
<td>Protégé, OWL 2</td>
</tr>
</tbody>
</table>

3 Discussion

3.1 Semantic Benefits

The benefits of using semantic technologies for a system like this are many. The first is portability of data. Semantic data formats are open and widely available to anyone who wants to create or read them. This allows data to move from one location to another without anything being lost. In the case of an EHR this becomes especially important. An EHR can contain many years worth of health data; problems, medications, allergies, surgeries. Losing all this data because of a change in insurance or relocation would be extremely detrimental to the patient. They would be forced to carry their physical file or to recount their history from memory. Both methods are error-prone and better if avoided.

Semantic technologies allow for encoding their namesake, semantics. No longer is data without context or standing on its own. Semantic technologies allow data to be encoded along with what the data...
3.2Claims

3.2.1 Providing Visualization Allows Medical Professionals to More Accurately Provide Medical Care

In Figure 3, the drug dosages and lab results for a patient with diabetes is visualized. We can observe that the dosage of glimepiride Oral Tablet was changed from 2mg to 4mg between July and October is correlated with the slowed increase and eventual decrease of Glucose SerPl-mCnc shortly afterward. One reasonable inference we can reach from this interaction between the changes of the drug dosage and lab results is that the physician decided to increase the dosage after the value of the lab result keeps going up for a while or hits some threshold. Moreover, we can also see that the increase of Glucose SerPl-mCnc is slowed down and controlled after the dosage change. This trend shows that the condition of the patient was improved and justifies the dosage change decision made by the physician.

3.2.2 Supporting Automatic Semantic Reasoning that Facilitates Medical Professionals to Examine and Analyze EHR

Our system also provides a quick and easy way to perform reasoning over the EHR. The medical professionals can identify EHRs that violate regulations and lab result records that are in normal ranges by button click. Such automatic EHR analysis is realized by applying semantic tools like Jena, Pellet and SPARQL to EHR. Figure 4 shows the over dosage of carvedilol Oral Tablet and one abnormal lab result of HDLc SerPl-mCnc for a patient with Chronic pulmonary heart disease.

3.2.3 Medical Role Ontology Provides an Extensible and Expandable Framework

Our system is an extensible and expandable framework that enhance EHR with semantic technologies. First, we could integrate EHRs from multiple sources. With configured data conversion, we could easily convert the EHRs into ontology-ready format. Secondly, we could incorporate new drug regulations with minimal manual efforts. We merely need to encode the regulations in a programmatic way and then import the encoding in the ontology model. We can further expand the system via linking open data like RxNav to integrate more information about drugs physicians are prescribing.

4 Related Work

4.1 Existing Technologies Tried and Dismissed

Longwell

Longwell is a web-based RDF-powered faceted browser\cite{3}, a part of the SIMILE\cite{9} project from MIT. It allows creation of simple user interfaces to browse RDF data. This plays into the goals of this project, create an interface to RDF data that allows one to discover things within the data.

A particularly interesting portion of the Longwell system is called Fresnel\cite{11}. Fresnel is an open W3C standard\cite{2} for defining RDF presentation\cite{12}. Once again this plays directly into the goal of this project, giving different doctors different views of the same set of data. Using a technology like this would greatly reduce our development time and greatly increase the functionality of the end product. Fresnel works within the Longwell system natively, so using Fresnel strongly encourages the use of Longwell.

Longwell is not available as a binary, anyone who wishes to use it must build it themselves. The distribution is easily obtained from the SIMILE site for Longwell\cite{4}. The most recent release is dated 29 October 2007. The build instructions are straightforward. The problems begin when attempting to build. Longwell depends on many libraries and calls for very specific library versions. Problematically, some of these libraries are not where the build system expects them or are not the versions the build system wants. In some cases the versions the build system wants do not exist or have never existed.

After several days of tracking down dependencies and modifying build-system XML files to use existing versions of dependencies, the authors gave up. We decided it would be better to explore other possibilities.

Paggr

Paggr is actually two products, Paggr Prospect and Paggr CMS. Paggr Prospect is a faceted browser builder for Linked Data\cite{6}. Paggr CMS is a content management system that combines Linked Data with widgets and simple agents\cite{5} \cite{13} \cite{14}. Paggr CMS won the International Semantic Web Challenge in 2008\cite{10}. Paggr is developed by semsol\cite{8}, a Semantic Web technology company in Germany.
Figure 3: System view showing a patient with Diabetes and the drugs they take (left) and lab results for blood glucose (right).

Figure 4: System showing built-in reasoning for dosage limits (left) and lab results (right).
Paggr CMS was interesting because it allows easy mashup of Linked Data and management of that same data. It seemed to be an all-in-one solution that would solve many of our problems and give us a great starting point.

Paggr Prospect was interesting for the same reasons as Longwell, giving a faceted browser over RDF data. Again, this would solve a multitude of our problems, allowing us to focus on more important issues.

Unfortunately, both Paggr products cost money; more money than we could afford. This is a shame since they looked to be genuinely quality and very interesting to use. No evaluation or academic versions were readily available.

5 Evaluation

Evaluation would be based on real-world usage and conversations with practicing physicians. We would present the application to the physician and observe their use. Once they achieve (or fail to achieve) the goal of the system (visualizing patient data and using it to make a diagnosis) we can ask them about their experience. This process would be best with real patient data, but the synthetic data would be just as illustrative of the system’s functionality and use to the physician.

The most powerful part of the evaluation is comparison with existing systems, whether paper, electronic or semantic. If our system has capabilities the others do not, ours is clearly better. If our system does the same thing as some existing system but in a better way, our system is better. A “better way” would be one that requires less time, fewer clicks, navigating fewer pages of content, etc. This would require real-world use of other systems along with ours.

6 Future Work

As it stands the system is functional, but leaves much to be desired in the aesthetics department. This is not limited to just looking nice, the overall design virtually non-existent. A better design would make the system easier to use and to understand for everyone, even though already familiar with the existing system. If the goal is to enable easier and clearer access to patient medical data then we still have work to do making the system itself easier and clearer to use.

The data used in the system was less than perfect. CHB example data was acceptable for basic data but a system like this needs finer grain data, better expressed and more available. We already modified some of the CHB data to better fit these requirements, any other data that comes through should be just as fine grained and expressed as well if not better.

The data we contrived was intended to show possible function of the system and may not be medically accurate or even medically possible. We made every effort to make it reasonable, clear and demonstrable of the system’s functionality. The real test of the system will come with real data from real people collected and viewed by real physicians.

The drug data used by the system is sorely inadequate. It only contains data about the few drugs in the CHB sample set. We can make use of data about drugs from RxNav, as of now we do not. The NIH data is trustworthy but NIH will not distribute data about drugs that may not be true (such as experimental uses of the drug). This is to be expected from a large government organization and is not a negative. Integrating other datasets about drugs would bolster the functionality of the system and overall improve the types of medical information retrievable from the system.

Our use case shows a physician with a specialty in Diabetes modifying the dosage of a blood pressure drug. Here we see a physician with that specialty modifying a drug which treats a condition not typically treated by that specialty. This type of modification would be interesting to make note of in the system, as it is not typically done, and may be dangerous. In reality, this type of interaction happens quite commonly. Identification of this type of interaction could be done in Pellet using the MedicalPractitioner class and the Specialty property.

7 Team Contributions

Peter

Peter was the driving force behind the project. His medical knowledge was the basis of our medical data synthesis and ontology construction. Peter created and gave the majority of presentations concerning the project. Attempted to find data from RxNav, even downloading quite a bit of the data from the site, trying to find a drug that may Treat a condition exhibited by one of our patients.

Ping

Ping was the main system architect. She constructed the modified ontology (from the original role versions) and the main interface, wrote the SPARQL queries,
and developed synthetic data under Peter’s supervision. Ping managed the setup and interactions of a variety of technologies to reach the final product.

**Scott**

Scott was the main author of written material, viz. this paper, and researcher of existing technologies upon which to implement the system (discussed earlier). Did initial work on ontology, attempting to express roles correctly and flexibly. Did some research into data availability and usefulness.

**All Members**

General critique of self and others’ work. Suggesting new avenues for project and types of problems that the system could solve. Possible forms solutions could take. Types of synthetic data to create and what it could illustrate.

## 8 Conclusion

Semantic technologies allow for more powerful data. Harnessing this power allows for greater impacts on any field, but medicine especially. An a person’s medical record is a person in data form; a record of every incident they have had with a medical professional. Keeping this data useful and meaningful is an important task. Keeping this data at the fingertips of medical professionals without overloading them can make this data a key component of medicine in the future.

## References


