Leveraging Semantic Web to Retrieve Customized Medical Information

Sifat Islam
Computer & Electrical Engineering and Computer Science
Florida Atlantic University
Boca Raton, FL

Glenn Freytag
Shilpico, Inc.
Deerfield Beach, FL

Ravi Shankar
Center for Systems Integration
Florida Atlantic University
Boca Raton, FL

I. INTRODUCTION

We are developing a web based application for use by diabetes patients to manage their chronic condition. The application will allow patients to keep abreast of the latest developments as pertinent to their specific condition (type, risk factors, medication, history, preferences, etc), interact with others in a community of users, and have more effective dialog with their health care providers [1]. This paper will document our recent developmental efforts to incorporate semantic web concepts and health IT (information technology) infrastructure.

Recent US Health expenditures reached $2.6 trillion per year, ten times the amount spent 30 years ago. Longer life spans and chronic illnesses have contributed to this. Health care costs for chronic disease treatment account for over 75% of national health expenditures [2]. The Office of the National Coordinator for Health Information Technology (ONC) has identified 'patient focused health care' as one of its two major goals [3]. Cost containment based on prevention and patient-centric health care management are actively being addressed today. We focus here specifically on chronic diabetes management as the vehicle for our web-based system development project. For a good general discussion and other Health IT efforts see [2-4].

Web-based tools, especially the interactive ones, have the potential to improve health outcomes and complement health care delivery. However, current implementations suffer from limited effectiveness and usability errors, and consequently have high user attrition rate [5]. Our App uses open source tools for software development, semantic web and Health IT [6-8], so other researchers and entrepreneurs can build on our work and develop their own creative extensions. We hope this will help software developers work closely with all the stakeholders for a given application and find the most optimal implementation, in terms of usability, content and interactivity.

II. APPROACH

We wish to empower patients by providing them with a tool to customize their profiles and retrieve pertinent information. To retrieve relevant medical information from medical databases and the web, we have leveraged Semantic Web technology. By capturing the patient's medical history and current health conditions in a profile, our application can search for information specific to the patient's need. In this paper, we propose a framework that can be used to build complex systems to retrieve personalized medical information. The framework utilizes ontologies, which are graphs of the relationships between terms that are relevant to a given topic. Instead of storing patient profiles in a database, our tool will allow patients to supplement/modify an ontology based on their information. The benefit of utilizing an ontology is that it is easy to customize. We will use the APIs (application programming interfaces) of OWL (Web Ontology Language) to access the ontology. A semantic reasoner is used to check the validity and consistency of the ontology after each modification. To inform patients regarding medical domain knowledge, our system contains a predefined domain ontology, which is shown in Figure 1. We want to provide patients with a tool that can retrieve information not only specific to their search, but can recommend similar categories to investigate. We have utilized the Semantic Web Rule Language (SWRL) to create rules that will suggest medical terminologies that patients may find useful to explore. SWRL provides our application with the intelligence required to guide patients in staying informed with the latest health information. We also plan to dynamically add rules based on their community's interests. Our system will place the patient's profile in the context of the medical domain and provide a method to consider alternative options.

III. CURRENT IMPLEMENTATION

Our current implementation allows the patient to select search terms from the ontology and the user profile. A Web Crawler and a set of Web Service Engines then use the terms to search exhaustively through online databases for any articles that the patient may find useful. The engines generate search results, and the application sorts them by relevance to the patient's needs, taking the user profile into account. Thus, our application can obtain valuable results that might otherwise be missed, discard results which are not relevant, and rank the results in order of interest to the patient. The application also integrates the information available in many diverse database systems in a way that is transparent to the patient. The application shown in Figure 2 was developed using version 3.7.0 of the free, open-source Eclipse SDK [6] and the WindowBuilder Pro user interface toolkit [9]. WindowBuilder Pro generates Java code that contains Swing user interface components [10]. This code was then merged with lower-level

code that manages the ontology and user profile, performs the web searches and organizes the search results.

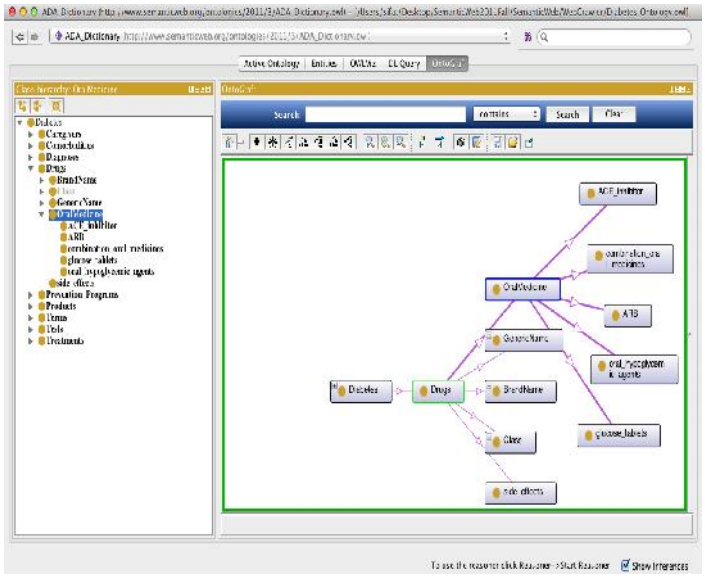

Figure 1. Diabetes Domain Ontology.

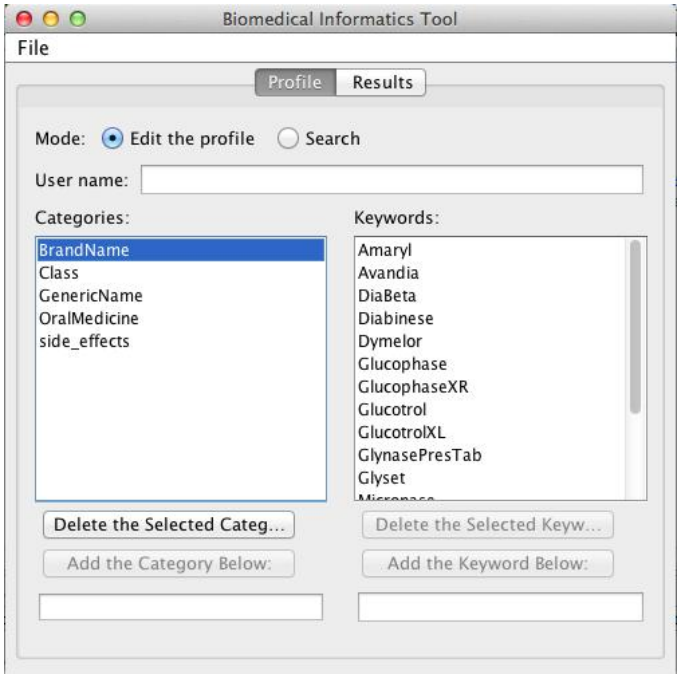

Figure 2. Our Semantic Web Application.

In Figure 3 below, we show our code for the OntologyHandler() methods, where we used OWL API [11] to interact with our Diabetes_Ontology.owl file. To write this method, we used code from OWL API's code examples [12]. In particular, we used OWL API's Simple Hierarchy code example to display information about the classes in the ontology by determining the class hierarchy and traversing the structure acquired [12]. Note that we have selected the HermiT OWL Reasoner [13] as our choice of reasoner.

```
public static void OntologyHandler() throws
MalformedURLException, FileNotFoundException,
OWLException, InstantiationException,
IllegalAccessException, ClassNotFoundException{

final String Ontology = "Diabetes_Ontology.owl";

try {     // Create our manager
          OWLOntologyManager manager =
          OWLManager.createOWLOntologyManager();

          File file = new File(Ontology);

          OWLOntology localOntology =
          manager.loadOntologyFromOntologyDocument(file);

          IRI classIRI = null;

          String reasonerFactoryClassName =
          "org.semanticweb.HermiT.Reasoner$ReasonerFactor
          y";

          Reasoner hermit =new Reasoner(localOntology);
          //   System.out.println(hermit.isConsistent());
          OWLReasonerFactory reasonerFactory = new
          Reasoner.ReasonerFactory();

          // Create a new SimpleHierarchy object with the
          given reasoner.
          SimpleHierarchyExample simpleHierarchy =
          new SimpleHierarchyExample(manager,
          (OWLReasonerFactory)
          Class.forName(reasonerFactoryClassName).newInsta
          nce();
          // Get Thing
          if (classIRI==null) { classIRI =
          IRI.create("http://www.semanticweb.org/ontologies/
          2011/3/ADA_Dictionary.owl#OWLClassImpl_01326
          002017572546000");}

          OWLClass clazz =
          manager.getOWLDataFactory().getOWLClass(classI
          RI);
          // Print the hierarchy below thing
          simpleHierarchy.printHierarchy(localOntology, clazz
);
}catch (OWLOntologyCreationException e)
{System.out.println("Could not create ontology.");
} catch (OWLOntologyStorageException e)
{System.out.println("Could not save ontology.");
}
}
```

Figure 3. The Main and OntologyHandler code.

In their paper, The HermiT OWL Reasoner, Ian Horrocks et al mention that this is the only reasoner that they are aware of which fully complies with the OWL 2 standard, and that properly reasons about properties in addition to classes. It is founded on an innovative hypertableau calculus that fixes performance issues caused by non-determinism and model size, which is the key cause of difficulty in OWL reasoners. HermiT also utilizes few new optimizations, containing an optimized ontology classification technique. Ian Horrocks et al found that HermiT does well in contrast to available tableau reasoners and is usually significantly faster when classifying intricate ontologies [14].

One cannot describe all relations using the OWL 2 language. For example, OWL 2 cannot describe the relation *child of married parents*. The reason for this limitation is that this language is unable to describe relations between people with whom a person has relations. This limitation of OWL can be overcome by incorporating SWRL rules to ontology. In Protégé's OWL editor, one can add SWRL rules and furthermore, the HermiT reasoner works with SWRL rules. Utilizing SWRL, one can describe the *child of married parents* relation. But using random SWRL rules will cause uncertainty; therefore DL-safe rules are coded in reasoners. DL-safe rules are used just for named individuals rather than individuals that are unnamed but nevertheless recognized to exist [15].

IV. Discussion

We are developing a framework which incorporates Semantic Web technology, web services, and web crawler for a user-friendly, interactive, and up-to-date Health IT application. A recent systematic review of web-accessible tools for management of diabetes [5] found that few web-based tools met all their health outcome criteria for clinical effectiveness, clinical usefulness, sustainability, and usability. They also found that greater interactivity resulted in better health outcomes. However, greater interactivity may require higher levels of health literacy, navigation skills, and computer experience. The authors conclude that strategies are needed to minimize website attrition (that occurs with static web sites and even interactive sites that are not easy to navigate). They wish to facilitate this by enabling patients and clinicians to make informed decisions about website choice with a standardized set of website quality indicators. They also indicate that with careful user testing, highly interactive applications can be designed to be user friendly. We propose that an open source application development environment, such as ours, allows such user testing, incremental improvement, and customization in terms of the stakeholder comfort level, content needs, and objectives for their usage.

In terms of the technological aspects, we have created so far a diabetes ontology and utilized OWL API to load the ontology in our application. We will then utilize a semantic reasoner via OWL API to check for validity and consistency of patient ontologies as they modify their ontology with our application. In addition, we will utilize SWRL to create rules to suggest related search terms. We expect to integrate SWRL rules and demonstrate at the conference a functioning system that addresses these issues.

At present, our app shown in Figure 3 provides each user the latest research articles; this will be extended to include articles from reliable clinical sources, and a rank listing of the latest articles read by their disease-centric community. Later, we plan to incorporate more avenues to facilitate direct interaction among patients, and formation of physician-centered groups of patients, to support each other and learn from each other.

The Indivo open source framework [8] will be used to store and maintain patient databases, and our extensions that embed their preferences. In development for about five years so far, Indivo [8] is now in its version 2.0.0 and is a free/open-source Personally Controlled Health Record server developed by the Children's Hospital in Boston, MA [17]. Medical information is stored as Python objects. The Indivo APIs provide a flexible way to aggregate a longitudinal health record from various component data sources, and to provide the ability to share components of an aggregated record with third parties [16]. This allows our tool to be customized to the individual user using his/her own health records and for a community site to be developed.

Other researchers have efforts underway to integrate Indivo with other tools. ONC (the Office of the National Coordinator for Health Care Technology) [18] has funded the Substitutable Medical Apps Reusable Technologies (SMART) project, and its extension, the SMART-enabled i2b2 platform [19] so that substitutable SMART applications (apps) can interact with health data within i2b2 (Informatics for Integrating Biology and the Bedside) [20]. The i2b2 Center plans to provide a scalable informatics framework that will enable clinical researchers to use existing clinical data for discovery research, [20]. Our extension may be considered a health data analytics application with a user customizable interface.

V. Conclusions

Management of a chronic disease, such as diabetes, is influenced by latest research findings and disease-centric community interaction, which can augment the private interaction between the patient and his/her physician. We wish to provide such a web-based tool. We expect to demonstrate our evolving tool at the conference. Other researchers have shown that an interactive, personalized and user-friendly web application may reduce user attrition rate and improve health outcomes. Our web application based on open source tools will allow other software developers and medical researchers to explore ways to improve the patient's web experience. This has potential to improve health outcomes and reduce health care cost.

REFERENCES

[1] S. Islam, G. Freytag, and R. Shankar, "Intelligent Health Information System to Empower Patients with Chronic Diseases", IEEE IRI 2012, August 8-10, 2012, Las Vegas, Nevada, USA.

[2] Kaiser: US Health Care Costs, http://www.kaiseredu.org/issue-modules/us-health-care-costs/background-brief.aspx2012.

[3] Christensen, C.M., The Innovators' Prescription, McGraw Hill, New York, 2009

[4] Wager, K.A., Lee, F.W., and Glaser J.P., Health Care Information Systems, 2nd Edition, Jossey-Bass, Wiley, 2009.

[5] Yu, C.H., Bahniwal, R., Laupacis, A., Leung, E., Orr, M.S., and Straus, S.E., Systematic Review and evaluation of web-accessible tools for management fo diabetes and related cardiovascular risk factors by patients and healtcare providers, J. Am Med Inform Assoc., Vol. 19, Issue 4, pp. 514-522, July 2012.

[6] Eclipse IDE for Java Developers, 2012, http://www.eclipse.org/downloads/packages/eclipse-ide-java-developers/junosr1

[7] M. Horridge, "A Practical Guide To Building OWL Ontologies Using Protege 4 and CO-ODE Tools Edition 1.3," The University Of Manchester, March 2011.

[8] Indivo: The Personally Controlled Health Record, http://indivohealth.org/

[9] WindowBuilder User Guide - Java Developer Tools – Google Developers, 2012, https://developers.google.com/java-dev-tools/wbpro/

[10] Trail: Creating a GUI with JFC/Swing, 2012, http://docs.oracle.com/javase/tutorial/uiswing/index.html.

[11] OWL API, 2012, http://owlapi.sourceforge.net/documentation.html.

[12] Documentation – Code Examples, 2012, http://owlapi.sourceforge.net/documentation.html.

[13] Hermit OWL Reasoner, 2012, http://hermit-reasoner.com.

[14] I. Horrocks, B. Motik, and Z. Wang, "The HermiT OWL Reasoner", Proceedings of the 1st International Workshop on OWL Reasoner Evaluation (ORE-2012), Manchester, UK, July 1st, 2012.

[15] OWL 2 and SWRL Tutorial, 2012, http://dior.ics.muni.cz/~makub/owl/

[16] Mandl, K.D., Simons, W.W., Crawford, W. CR, and Abbett, J.M., Indivo: : A personally controlled health record for health information exchange and communication, BMC Medical Informatics and Decision Making, September 12, 2007, pp. 25-34

[17] Indivo Wiki, http://wiki.chip.org/indivo/index.php/HOWTO:_write_an_Indivo_app_using_Python, retrieved on November 10, 2012

[18] ONC website: http://healthit.hhs.gov/portal/server.pt/community/healthit_hhs_gov__home/1204, retrieved on November 10, 2012

[19] SMART i2b2 wiki site: https://community.i2b2.org/wiki/display/SMArt/SMART+i2b2, retrieved on November 10, 2012

[20] i2b2 site: https://www.i2b2.org/, retrieved on Novemer 10, 2012