ACHIEVING DYNAMIC INTEROPERABILITY OF COMMUNICATION AT THE DATA LINK LAYER THROUGH ONTOLOGY BASED REASONING

Kenneth Baclawski (College of Computer and Information Science, Northeastern University, Boston, MA; kenb@ccs.neu.edu); David Brady (Department of Electrical and Computer Engineering, Northeastern University, Boston, MA; brady@ece.neu.edu); Mieczylaw M. Kokar (Department of Electrical and Computer Engineering, Northeastern University, Boston, MA; mkokar@ece.neu.edu)

ABSTRACT

Ontology-Based Radio (OBR) is a mechanism for software defined communication nodes to understand and to modify the processing of communication packets. In this paper we describe the application of OBR at the data link layer. In particular, we describe how interoperability can be achieved at run time by using an ontology for data link protocols to deduce the protocol of a data link packet. The inference mechanism is based on the Web Ontology Language (OWL) and the Semantic Web Rule Language (SWRL). OBR presents a number of challenges not faced by other Semantic Web applications such as performance requirements and the highly dynamic nature of communication. We address these challenges by using a Prolog-based SWRL reasoner which provides very fast reasoning, and considerably smaller memory requirements than other Semantic Web theorem provers.

1. INTRODUCTION

Ontology-Based Radio (OBR) [1,2] is a mechanism for software defined communication nodes to understand and to modify the processing of packets during a communication session both at the source and the destination. This mechanism uses an ontology to specify not only the structure of communication packets but also the processing of those packets according to the communication protocol. Nodes have the ability to query both their own capabilities and the capabilities of other nodes. The use of ontologies adds flexibility, inferencing and reasoning features that are not available with ad hoc data structures or database schemas.

In this paper we demonstrate the concept of OBR at the data link layer. A large number of protocols are currently in use at this layer. We demonstrate how data link layer interoperability can be achieved at run-time by using an ontology for data link protocols to deduce the protocol of a data link packet. The inference mechanism is based on the Web Ontology Language (OWL) and the Semantic Web Rule Language (SWRL). However, OBR presents a number of challenges not faced by other Semantic Web applications. (1) Real-time processing demands higher performance for reasoning than an interactive application. (2) The "knowledge base" of a node includes packet information that is continually varying, in contrast with the static knowledge bases required by most ontology-based reasoning systems. (3) The "facts" are not stored in a knowledge base but rather are embedded in the software that implements the data link protocol.

We solve the first two problems by using a Prolog-based SWRL reasoner which provides not only very fast reasoning, but also considerably smaller memory requirements than other Semantic Web theorem provers.

In our prototype we describe how the communication between two OBR nodes can be controlled to achieve dynamic interoperability. We illustrate this with an example in which an OBR node discovers the data link protocol from the structure of packets received from another node. This example demonstrates how deep reasoning can be used to achieve dynamic interoperability.

2. BASICS

An ontology specifies the concepts of a domain, attributes of the concepts and relationships between the concepts. Each concept is expressed using a *class* which may be interpreted as a set of things. A thing that belongs to a class is called an *instance* of that class. Packets, frames and protocol types are all fundamental concepts in radio communication. In the ontology for software radio, these concepts are expressed as classes. For example, Packet is a class whose instances are particular packets. Classes are organized into a *hierarchy* of classes by the *subclass* relationship. For example, Synchronous Data Link Control (SDLC) and Link Access Procedure, Balanced (LAPB) are both special cases of High-level Data Link Control (HDLC). This is expressed by specifying that SDLC and LAPB are subclasses of HDLC.

An *attribute* is a characteristic that something has, such as the number of symbols in an alphabet or the carrier frequency of a waveform. An *attribute value* is a characteristic of a single thing. The value of an attribute is a data value such as a number. A *relationship* is an association among various things. For example, a waveform is used to represent a sequence of symbols from an alphabet. This is expressed by linking the waveform to the symbol sequence. An ontology will generally have many different kinds of attributes and relationships. The size of a data link field might be called *fieldSize,* while the protocol of a frame might be called the *frameProtocol* relationship. The term *property* is used for either an

attribute or a relationship. As with classes, one kind of property may be regarded as a set of instances, called *facts*. For example, when a particular frame *f* is being used by protocol *P*, this fact is the triple *(f, frameProtocol, P)*. Properties can be organized in a hierarchy by the *subproperty* relationship.

2. WEB RESOURCE DESCRIPTION LOGICS

Ontologies are formalized in ontology description languages. OWL (Web Ontology Language [3]) is the most popular language today. It includes three "species" of which OWL-DL language based on Description Logics [4] is used in most applications. OWL Lite differs from OWL-DL in the class constructors that are allowed. For example, in OWL-DL one can specify the *complement* of a class (i.e., all instances that are NOT in the class), but this is not allowed in OWL Lite. OWL-DL allows all class constructors, but it does not allow any mixing of metalevels. For example, classes and their individuals are on different metalevels, so a class cannot be an instance of another class in OWL-DL. By contrast, in OWL Full, a class can be an instance of another class, which itself is an instance of yet another class, and so on to any number of metalevels. Although OWL Full is a very rich ontology language, it is still not as rich as arbitrary first order predicate logic. Table 1 summarizes the features and differences between the various Web-based ontology languages.

Table 1. Web-based ontology languages and their characteristics

Language	**Features**	**Reasoning**
RDF	Binary relationships	None
RDF Schema	RDF plus subclass, subproperty, domain and range	Subsumption
OWL Lite	RDFS plus some class constructors but no crossing of metalevels	Limited description logic
OWL-DL	All class constructors but no crossing of metalevels	General description logic
OWL Full	No restrictions	Limited form of first order logic

Although richer the ontology languages are more expressive, these languages are also more difficult to process. Even OWL Lite requires exponential time in the number of triples for the worst case. OWL-DL is *decidable* meaning that the processing will finish in a finite amount of time, but the amount of time can be more than exponential. Finally OWL Full is *undecidable* which means that some queries cannot be answered in a finite amount of time.

3. QUERYING

Given a data base, one can extract information by using a *query*. Many query languages have been proposed for particular ontology languages like RDF and OWL. The query language that has been proposed for RDF is called RDQL [5]. As is the case with most query languages, RDQL is syntactically and semantically very similar to the SQL query language for relational database systems. RDQL differs primarily in allowing one to specify *patterns*. A pattern is a fact in which some of the components can be variables. For example, the following query retrieves all fields of all data link frames.

```
SELECT ?x
  WHERE (?x, <rdf#type>,
    <datalink#DataLinkFrame>)
  AND   (?y, <http://rdf#type>,
    <datalink#DataLinkField>)
  AND   (?x, <datalink#contains>, ?y)
```

For topographical convenience, we have abbreviated the URIs above. For example, "rdf" should actually be "http://www.w3.org/1999/02/22-rdf-syntax-ns." The variables in the query begin with a question mark. The first pattern limits ?x to be of type DataLinkFrame, the second pattern limits ?y to be of type DataLinkField, while the third pattern requires that ?x contains ?y.

The query language that has been proposed for OWL is called OWL-QL [6]. The RDQL and OWL-QL query languages differ from database query languages. One important difference is that they are web based. While databases are generally restricted to a single server or at least one site, RDQL and OWL-QL effectively regard the entire web as being a single database. However, the most important difference between database queries and query languages like RDQL and OWL-QL is their support for reasoning. In addition to facts that have been explicitly asserted, a query can also retrieve facts that can be inferred. It is for this reason that one refers to RDF and OWL databases as being *knowledge bases*. The set of all currently known or inferred facts in a particular context is called the knowledge base. The reasoning capability of OWL is especially powerful. We elaborate on this feature below.

Radio communication introduces an additional requirement on query languages. Unlike knowledge bases that have a combination of explicitly asserted facts and inferred facts, an OBR knowledge base includes data which is embedded in the software that implements the communication protocols. Extracting such data requires a new software capability known as *self-awareness* or *reflection*.

Reflection is a property which enables software to understand its *runtime structure*. Reflection is a key feature of any software system which is expected to respond to unanticipated queries. Runtime structure includes memory pointers for variables, along with the fields of data structures, the types of the fields and the values of the fields. Reflection is essential if the software should be able to answer queries and execute methods dynamically.

4. REASONING

One of the important features of ontology languages that distinguish them from databases is the ability to make logical deductions. In other words, one can *reason* about the information in a knowledge base. A fact is *deduced* if one can infer that it is true even though the fact has never been explicitly asserted to be true. One of the most important examples of deduction is *subsumption.* For example, if one knows that a frame is an SDLC frame, then one can deduce that it is also an HDLC frame. In general, whenever something is an instance of a subclass, then it is also an instance of all superclasses of the subclass. Subsumption is the basis for reasoning in description logics. For example, all features and axioms applicable to HDLC also apply to SDLC.

While subsumption reasoning is useful, it is not sufficient for all reasoning tasks. When reasoning involves several linked facts, one cannot express the inference using subsumption alone. When database records are linked by common attributes, they are said to be *joined.* To express reasoning involving joins of facts, it is necessary to introduce *rules.* A rule is knowledge in the form of an *if-then* statement. A rule specifies that if its *hypothesis* holds, then its *conclusion* must also hold. The hypothesis is also known as the *antecedent,* and the conclusion is also known as the *consequent.* The rule language that has been proposed for OWL is the Semantic Web Rule Language (SWRL) [7]. An example of a SWRL rule is shown in Section 5 below.

5. ROLE OF ONTOLOGY IN KNOWLEDGE INTENSIVE CHARACTERISTICS

Two-way radio communication introduces a number of challenges not shared by most other Semantic Web applications: (1) Real-time processing demands higher performance for inference and reasoning than an interactive application. (2) The knowledge base of a node includes state information that is continually varying, in contrast with the static knowledge bases required by most reasoning systems. (3) The facts are not explicitly stored in a knowledge base but rather are embedded in the software that implements the communication protocol [2]. If these challenges can be overcome, the Semantic Web can play a number of important roles in radio communication, such as the following (see [8]):

1. Interoperability. Radios can use ontologies to deduce important information such as the protocol being used by other radios.
2. Flexible querying. Information can be queried. Furthermore, such queries can be answered without having any explicit pre-programmed monitoring capability.
3. Run-time modifiability. Protocols and packet structures can be modified at run-time in response to environmental conditions and application requirements.
4. Validation. Formalization allows one to check the consistency of protocols and to validate the correctness of algorithms that implement the protocols.
5. Self-awareness. Communication nodes can understand their own structure and modify their functioning at run-time based on this understanding.

6. DATA LINK LAYER ONTOLOGY

The data link layer is responsible for transmitting frames and for error detection and correction in communication links. There are many *data link protocols*. Each protocol specifies the frame types and structure as well as how the communication link is controlled. Many of the data link protocols specialize and extend other data link protocols, forming a hierarchy as shown in Figure 1. Because of the diversity and complexity of data link protocols, we will not show all of the details of the data link ontology.

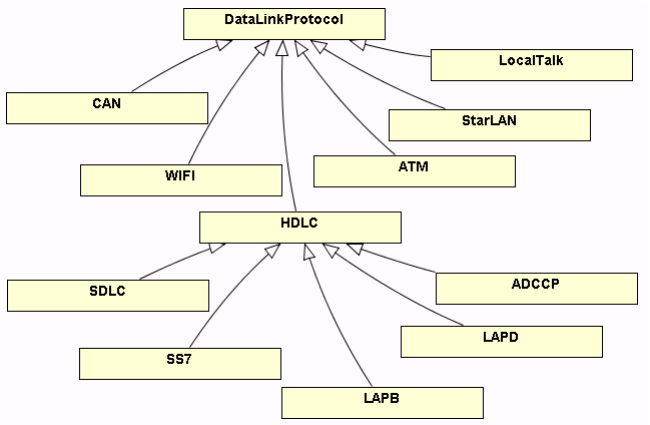

Figure 1. A partial hierarchy of data link protocols

Data link protocols operate in several *modes*. For example, the HDLC protocol has three operational modes: Normal Response (NR) Asynchronous Response (AR) Asynchronous Balanced (AB)

Data link protocols use a large variety of frame types. The names and semantics of the frame types depend on the protocol, but there is considerable overlap among the protocols. Consequently, while each protocol has its own frame type hierarchy, the frame types in different hierarchies can be related by the OWL *sameAs* relationship. The following is the frame type hierarchy for HDLC protocols (see Figure 2):

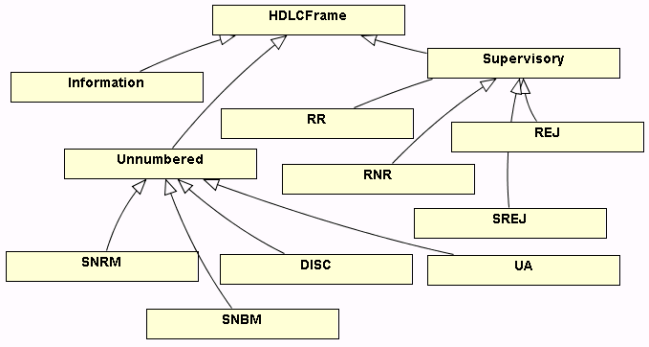

Figure 2. HDLC Frame Hierarchy

Figure 3 shows the top level of the frame type hierarchy of the WIFI protocol. This hierarchy can be further decomposed into more refined subclasses. For instance, the Management frame can be subclassified into Authentication, Deauthentication, AssociationRequest, Disassociation, AssociationResponse, ReassociationRequest, ReassociationResponse, Probe, ProbeRequest, Beacon, ProbeResponse.

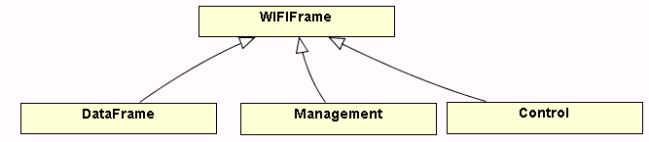

Figure 3. Top level of the WIFI frame hierarchy

A frame consists of a sequence of *fields*. The frame structure is defined by the order of the field types, the number of bits allowed in each field and the values (bit sequences) allowed in each field. The HDLC protocol has the field types shown in Figure 4.

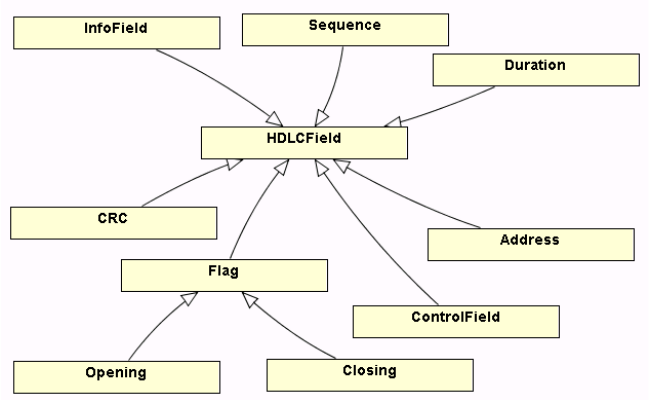

Figure 4. HDLC Field Hierarchy

The WIFI protocol has many of the same field types, except for changes in terminology. For example, FrameControlField *owl:sameAs* ControlField, and ChecksumField *owl:sameAs* CRC field. The data link classes are related to one another as follows (see Figure 5): A data link frame contains an ordered sequence of fields, each of which has a size (in bits) and a value. A data link field also has a mode and belongs to a protocol.

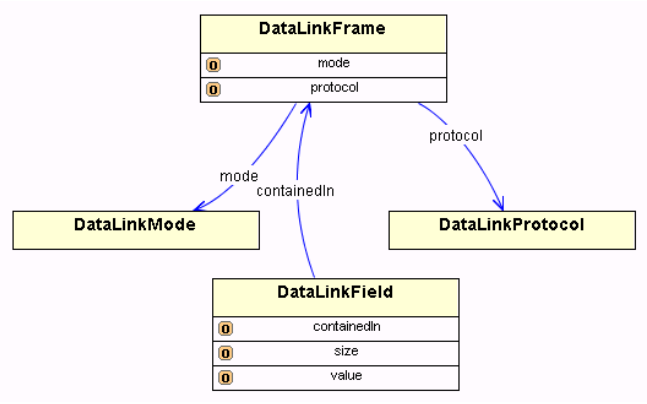

Figure 5. Data link layer ontology

In addition to the hierarchies and relationships described above, the data link ontology specifies a large number of rules that constrain data link fields within the same frame and in related frames. Here are some examples of such rules expressed informally.

If a frame belongs to the SDLC protocol, then the address field has 8 bits.

If a frame belongs to the HDLC protocol, then its opening flag field has value 0x7E (i.e., the bit sequence 01111110).

If a frame belongs to the SS7 protocol, then the address field has 0 bits.

If a frame belongs to the HDLC protocol, then the control field has 8 or 16 bits.

If a frame belongs to the HDLC protocol and the mode is ARM, then the address field has 0 bits.

If a frame belongs to the WIFI protocol then the first two bits of the control field are zeroes. This subfield represents the version number.

All these rules can be represented either in SWRL [7] or in OWL. For instance, the first rule may be represented in OWL as follows

```
<owl:Class>
 <owl:intersectionOf rdf:parseType="Collection">
  <owl:Class rdf:about="#AddressField"/>
  <owl:Restriction>
   <owl:onProperty rdf:resource="#containedIn"/>
   <owl:allValuesFrom>
    <owl:Restriction>
     <owl:onProperty rdf:resource="#protocol"/>
 <owl:allValuesFrom rdf:resource="#SDLCProtocol"/>
```

```
      </owl:Restriction>
     </owl:allValuesFrom>
    </owl:Restriction>
  </owl:intersectionOf>
  <rdfs:subClassOf>
    <owl:Restriction>
     <owl:onProperty rdf:resource="#size"/>
 <owl:hasValue
rdf:datatype="&xsd;integer">8</owl:hasValue>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
..
```

In XML, element tags, attribute names and attribute values can belong to various *namespaces*. The namespace of a tag or an attribute name is specified with a colon. For example, rdf: specifies the RDF namespace, rdfs: specifies the RDF Schema namespace, and owl: specifies the OWL namespace. Within an attribute value, a namespace is specified with an XML *entity*. For example, &xsd; specifies the XML Schema namespace. If no namespace is specified, then the namespace is the current default namespace. In this case, the default namespace is obr, the namespace of ontology based radio.

At the highest level, the rule shown above says that an intersection of two classes is a subclass of another class. This is expressed using the owl:intersectionOf and the rdfs:subClassOf properties. The intersection is used for representing the Boolean AND operator, while subclass is used for presenting the logical IMPLIES or IF-THEN operator. The two classes being intersected are the class of address fields and the class of HDLC fields. The former class is part of the ontology and has a name. The latter class does not have a name. It is specified by two relationships in the ontology: containedIn and protocol. An SDLC field is one that is contained in a frame whose protocol is of type SDLC. The owl:Restriction is used for constructing classes of instances that satisfy a constraint for a particular property. The first restriction specifies "A field that is contained in," while the second restriction specifies "a frame whose protocol has type SDLC." The last restriction specifies the size attribute of the field. Putting all of these together, the rule can be expressed as "IF something is an address field AND is a field that is contained in a frame whose protocol has type HDLC THEN it has size 8. Alternatively, the same rule could be represented in SWRL as shown below.

```
<ruleml:Imp>
  <ruleml:body>
    <swrlx:classAtom>
      <owlx:Class owlx:name="#AddressField"/>
      <ruleml:var>x</ruleml:var>
    </swrlx:classAtom>
    <swrlx:individualPropertyAtom
swrlx:property="#containedIn">
      <ruleml:var>x</ruleml:var>
      <ruleml:var>y</ruleml:var>
    </swrlx:individualPropertyAtom>
    <swrlx:individualPropertyAtom
swrlx:property="#protocol">
      <ruleml:var>y</ruleml:var>
      <ruleml:var>z</ruleml:var>
    </swrlx:individualPropertyAtom>
    <swrlx:classAtom>
      <owlx:Class owlx:name="#SDLCProtocol"/>
      <ruleml:var>z</ruleml:var>
    </swrlx:classAtom>
  </ruleml:body>
  <ruleml:head>
    <swrlx:datavaluedPropertyAtom
swrlx:property="#size">
      <ruleml:var>x</ruleml:var>
      <owlx:DataValue
   owlx:datatype="&xsd;integer">8</owlx:DataValue>
    </swrlx:datavaluedPropertyAtom>
  </ruleml:head>
</ruleml:imp>
..
```

The first part of this rule is the *hypothesis*, and the second part is the *conclusion*. Either part of a rule consists of a sequence of *atoms*. A *class atom* specifies an instance of a class. In this case the class atoms specify that *x* is an instance of the class AddressField, and that *z* is an instance of HDLCProtocol. A *property atom* specifies a fact. Individual property atoms are for properties whose values are *individuals*, and data valued property atoms are for properties whose values are data values. For example, the second atom in the body specifies the fact (*x containedIn y*).

7. DATA LINK LAYER PROTOCOL CONSISTENCY AND SELECTION

The data link ontology specifies the formats and functionality of the data link protocols. In this section we give two examples of how the data link ontology might be used. The first example illustrates how the formal specification of protocols can uncover inconsistencies. This kind of reasoning would normally be performed offline. The second example shows a simple example of how the ontology can be used for interoperability. The first example uses the following two rules from the data link ontology section above:

If a frame belongs to the SDLC protocol, then the address field has 8 bits.

If a frame belongs to the HDLC protocol and the mode is ARM, then the address field has 0 bits.

Since the SDLC Protocol is a subclass of the HDLC Protocol, the two rules imply that an ARM frame belonging to the SDLC protocol has an address field with both 8 and 0 bits. It follows that the SDLC protocol cannot have ARM frames.

One can address this inconsistency in several ways. One could remove the subclass relationship between the SDLC Protocol and the HDLC Protocol classes from the ontology. This has the disadvantage that the SDLC protocol would now have to be completely specified from scratch rather than being derived from a more general protocol. Alternatively, one could modify the first rule above to recognize that the address field for the SDLC protocol will only have 8 bits for modes other than ARM.

The second example concerns the issue of interoperability. Suppose that a radio receives a stream of packets from another radio with which it has not previously communicated. Can the receiving radio deduce the protocol being used by the transmitter? In the case of HDLC versus WIFI, the following two rules allow one to disambiguate the packets:

If a frame belongs to the HDLC protocol, then its opening flag field has value 0x7E (i.e., the bit sequence 01111110).

If a frame belongs to the WIFI protocol then the first two bits of the control field are zeroes.

From these two rules, one can deduce that the first two bits of the packet will distinguish these two protocols. If the first two bits are 01, then it is an HDLC frame. If the first two bits are 00, then it is a WIFI frame. If the bits have some other value, then the frame belongs to neither protocol. This example is somewhat artificial, and one could argue that it is easy to write a small procedure that could make this same determination. However, this is only one example among many possible inferences. The software would quickly become extremely unwieldy if one needed a separate procedure for every possible deduction. Indeed, one would need infinitely many procedures since one can make infinitely many deductions, and every time a new feature or protocol variation was added, one would need to revise all of the procedures.

8. SUMMARY AND CONCLUSIONS

In this paper we demonstrated the concept of OBR at the data link layer. In particular, we described how interoperability can be achieved at run time by using an ontology for data link protocols to deduce the protocol of a data link packet. The inference mechanism is based on the Web Ontology Language (OWL) and the Semantic Web Rule Language (SWRL). OBR presents a number of challenges not faced by other Semantic Web applications such as performance requirements and the highly dynamic nature of communication. We addressed these challenges by using a Prolog-based SWRL reasoner which provides very fast reasoning, and considerably smaller memory requirements than other Semantic Web theorem provers.

---

[1] J. Wang, D. Brady, K. Baclawski, M. M. Kokar and L. Lechowicz. The Use of Ontologies for the Self-Awareness of the Communication Nodes. In Proceedings of the Software Defined Radio Technical Conference, SDR03, Orlando, FL, 2003.

[2] J. Wang, M. M. Kokar, K. Baclawski and D. Brady, "Achieving Self-Awareness of SDR Nodes Through Ontology-Based Reasoning and Reflection." Proceedings of the Software Defined Radio Technical Conference SDR'04, Phoenix, Arizona, November 2004.

[3] F. van Harmelen, J. Hendler, I. Horrocks, D. McGuinness, P. Patel-Schneider and L. Stein, OWL Web Ontology Language Reference, www.w3.org/TR/owl-ref/ M. Dean and G. Schreiber (Eds.), March 2003.

[4] F. Baader, D. Nardi, D. McGuinness, P. Patel-Schneider and D. Calvanese. The Description Logic Handbook: Theory, Implementation and Applications. Cambridge University Press, Cambridge, England, 2003.

[5] A. Seaborne. "RDQL - A Query Language for RDF", www.w3.org/Submission/RDQL, January 2004.

[6] R. Fikes, P. Hayes and I. Horrocks, "OWL-QL: A Language for Deductive Query Answering on the Semantic Web", Journal of Web Semantics, Vol. 2, No. 2, 2005.

[7] I. Horrocks, P. Patel-Schneider, H. Boley, S. Tabet, B. Grosof and M. Dean. "SWRL: A Semantic Web Rule Language Combining OWL and RuleML", http://www.daml.org/2003/11/swrl, 2003.

[8] M. M. Kokar, D. Brady and K. Baclawski. "Roles of Ontologies in Cognitive Radios". In "Spectrum Efficiency and Cognitive Radios," B. Fette (Ed.), Elsevier, in print.