ABSTRACT
The vision of the Semantic Web is to make the World Wide Web one large virtual data store that can be accessed by a range of applications. In order for this vision to be realized, tools and techniques must be in place for exposing legacy databases to semantic technology. This paper compares two Semantic Web frameworks, OntoAgent and Ontoprise, which have the capability of connecting to legacy databases. The frameworks are compared on the level of semantic encoding, the amount of data redundancy, supported database platforms, and the amount of automated tool support.

Keywords
Semantic Web, Legacy Databases, OntoAgent, Ontoprise

1. INTRODUCTION
The vision of the Semantic Web [23] is to provide a common framework for sharing data by giving meaning to the data so that it is understandable by computers and people [2]. Adoption of Semantic Web technologies cannot take place without tools and techniques in place to allow for the integration and semantic description of legacy data sources.

This paper reports on the results of a study that compares Semantic Web applications based on the support for the semantic description of legacy data. The Semantic Web frameworks compared are the OntoAgent framework [6] and the Ontoprise framework [18]. The evaluation of the frameworks is performed based on Semantic Web criteria including the level of semantic encoding, data redundancy, supported database platforms, and tool support.

For purposes of this research, the term legacy data will be defined to mean pre-existing data sources that are not described by semantic encodings recommended by the W3C. These data sources could be structured, as in the form of databases, or unstructured in the form of text documents and HTML pages, although this paper will concentrate on databases. It is recognized that a relational database, for example, encodes a form of internal semantics in its schema. However, this schema does not automatically integrate with standard (W3C recommended) Semantic Web languages such as RDF to support inferencing by Semantic Web agents. This paper focuses on the mechanisms by which legacy data will be accessed and/or described for use by the Semantic Web.

Section 2 describes the background and technologies of the Semantic Web. Section 3 discusses methods by which structured data is exposed in the semantic and non-Semantic Web. Section 4 describes the Semantic Web applications compared in this paper. Section 5 outlines the methods of comparison of the two applications under comparison and Section 6 discusses the results.

The conclusion and suggestions for future work are found in section 7.

2. SEMANTIC WEB BACKGROUND
The Semantic Web is the next phase in the development of the World Wide Web (WWW). The Semantic Web will make the content and data of the WWW not only human readable, but machine readable and understandable via the publication of semantic metadata about web resources. Implementations of the Semantic Web will allow agents to read the resource metadata and to reason about it using inference rules and ontologies (domain-specific semantic summaries) [12].

As with the WWW, the research and standardization of technologies required to support the Semantic Web are coordinated by the World Wide Web Consortium (W3C). The W3C creates work groups that develop generic technical solutions and publish them as recommendations (standards). The technologies that the W3C recommends for Semantic Web development include: Web Services, a technology for accessing machine readable data, Simple Object Access Protocol (SOAP), a protocol for invoking Web Services, Resource Description Framework (RDF) a description language for the resource metadata, RDF Schema (RDF-S) for specifying how classes of resources will be described, and Web Ontology Language (OWL) for creating ontologies. All of these technologies are layered on top of Extensible Markup Language (XML) and XML-Schema, also W3C recommendations. Semantic Web research builds upon research in artificial intelligence, knowledge representation, distributed computing, and semantic data integration [1].

3. METHODS FOR EXPOSING LEGACY DATA TO SEMANTIC WEB
Before beginning the description of methods for exposing legacy data to semantic technologies, an example of data display on the non-Semantic Web is briefly mentioned to provide context. Data design in this paradigm is for human-readable representation of human requested data. For example, a human user seeking travel information might request a list of available airline flights by entering parameters in an HTML input form. The travel reservation application performs whatever queries are necessary in order to bring back the data results and display the results using HTML in a form that is readable for a human.

If a software agent was directed to use the travel system above, it would need to interpret the instructions for filling out the form, enter the appropriate parameters on the form, submit the form, and parse the results. All of the above may be difficult if not impossible for a software agent to do, especially if the requirement is to then compare results from multiple travel sites, all with different instructions, forms, and returned data formats. The agent would also need a way to find the list of travel sites to
visit. In contrast, the Semantic Web requires that the kind of data available on a website or web service, the methods for querying that data and the return data formats are all easily accessible and understandable by software agents through the use of standard formats and ontologies. The following subsections will discuss some approaches to making legacy data available to the Semantic Web.

### 3.1 Migration to a Repository in Semantic Format

In order for the Semantic Web to be fully realized, existing data must be exposed in a semantic format readable by software agents or applications. However, most data is not yet in a machine processable format. One solution is to replace the legacy database with one in a semantic format. If an appropriate semantic schema for the new database can be developed, it is not a difficult step to convert the data to the new format and retire the legacy database. However, mapping a database schema to an existing ontology or generating an ontology based on the database schema is not an easy task. Tools from the OntoKnowledge project [17] and KAON project [15] can be used for this purpose. The disadvantage of this approach is that any other applications that to interface with the legacy database will need to change. In the tangled network of databases in a corporation or other information organization, this option may be too costly and disruptive to contemplate in the near term.

Stojanovic, et. al., describe a semi-automated approach, with a formal background, for migrating relational databases to ontological knowledge bases [24]. The parallel processes of database reverse engineering and semantic annotation are discussed. The foundation of the technology relies on work in migrating relational databases to objects. While an implementation of the approach had not yet been done, the authors predicted cost savings and time benefits when compared to traditional annotation methods. As an additional benefit, the increased understanding of the semantics of the data could lead to improved maintenance of data-driven applications and easier migration to other databases.

### 3.2 Semantic Metadata Extraction and Storage in a Separate Repository

Extraction of metadata from the database schema is a common method used by OntoKnowledge [5], Ontoprise, and other example Semantic Web applications and frameworks. The schema of a database can be extracted and converted into a semantic format such as RDF-S. This semantic version of the schema can be mapped to an ontology or published via UDDI or WSDL to make the data available to semantic applications. The semantic metadata and mappings can then be stored in a central repository for the purpose of making queries across multiple data sources.

The Unicorn Systems methodology [21], for example, suggests first creating an Information Model for the enterprise. This Information Model has all of the characteristics of an ontology: classes with inheritance, instances, attributes, business rules, and synonyms. Any database or message format that has a schema can then be mapped to the Information Model.

The advantage for using a central repository is that the number of mappings that must be created and maintained is greatly reduced. Figure 1 shows an example of mappings between information sources in a corporate environment. In this figure, change to any schema will affect multiple mappings across the enterprise. If instead each information source is mapped to one central semantic repository, a change to the schema of an information source would affect the one mapping to the central repository. The introduction of a new data source will entail creating only one mapping to the central repository in order to integrate it with the rest of the enterprise.

### 3.3 Adding Semantic Markup to the Existing Repository

Semantic markup can be provided at the web page or web service accessing the data or on the repository itself. Handschuh et al. provide a deep annotation method for the databases behind web sites [11]. Circumstances of static and dynamic data, and cooperative and uncooperative web site owners are discussed. For the case of the cooperative owner, a web site owner provides server-side web page markup based on the schema of the database. An annotator can then load this markup into automated or semi-automated tools and create and publish an ontology and a collection of mapping rules which can also include pointers to a web service provided by the web site owner for data access. A querying party can then use the ontology and mapping rules to query the database via the web service. The OntoAgent framework [6] provides for creating semantic markup on the repository itself in the form of views.

### 4. DESCRIPTION OF THE SEMANTIC WEB IMPLEMENTATIONS TO BE COMPARED

This study compares two semantic web frameworks, Ontoprise and OntoAgent. These two frameworks were chosen for comparison because they employ two different methods for exposing the legacy data sources: Ontoprise extracts the schema from the database while OntoAgent uses annotation on the database. In this section, an overview of the implementation of these two frameworks will be presented.
4.1 Ontoprise
The On-To-Knowledge project developed a toolset for semantic information processing [10]. Part of the toolset is a module called OntoWrapper that exposes external semi-structured data to an ontology repository [17].

A commercial outgrowth of the On-To-Knowledge project is Ontoprise, a German company that supplies tools for semantic information integration and Knowledge Management. The key products are OntoEdit, OntoBroker, Semantic Miner, and OntoOffice. OntoEdit is a modeling and administration framework for ontology-based integration of heterogeneous structures. OntoBroker is an ontology-based inference engine. Semantic Miner supports knowledge retrieval from heterogeneous sources and has several interfaces, including dynamic visualization, and tree and hyperbolic views. OntoOffice integrates with the Microsoft Office environment to allow for context-sensitive input from the knowledge base and to create annotation to extend the knowledge base.

Ontoprise internally stores the ontology in the F-Logic language [16]. F-Logic is a deductive, object-oriented database language. F-Logic supports object-oriented constructs like classes and inheritance that are used by ontologies. The F-Logic variant employed by Ontoprise provides extensions like built-ins and namespaces [14]. The Ontoprise products can import and export in RDF, DAML+OIL, and OWL Lite formats.

4.2 OntoAgent
The second framework to be compared is the OntoAgent framework. This framework was designed for creating declarative specifications of agent systems [7]. Other goals of the system include mapping this specification onto a runtime system, using mainstream technology as a basis, allowing for shared ontologies, reusing business logic through web services, and seamlessly integrating existing data from relational databases [8].

In order to provide an understanding of how legacy data may be exposed using the Ontoprise framework, we discuss the steps for using the framework to create and use an enterprise-level ontology [13]. A domain ontology is created using the OntoEdit tool to graphically create concepts, attributes, relations, instances and axioms pertaining to the domain. This ontology should be created by a domain expert with awareness of the semantic relationships of the information stored in the enterprise systems (databases, documents, web services, and applications).

OntoEdit can then import a database schema from a database in the form of a lightweight ontology. A plugin component to OntoEdit, called OntoMap, can be used to graphically create mappings between the database ontology and the domain ontology. This must be done by a domain expert. OntoBroker can then use the ontology to reason about the data and Semantic Miner can use the ontology to implement complex queries across the enterprise data sources.
mappings from the database schema to the views must be done manually by a domain expert.

In addition to the facts, the deductive database also stores the Integrity Constraints, and Derivation Rules used by the agent. Integrity Constraints are logical statements used to exclude illegal mental states. Action rules allow the agent to respond to external events and to act spontaneously. Derivation Rules allow deriving new information from the basic facts known by the agent. The derivation rules can access the facts whether they are implemented in tables or as views. OntoSQL is the rule engine employed by OntoAgent [6]. OntoSQL converts Rule Markup Language (RuleML) [20] statements into SQL queries and is designed to work over relational databases. The relational database is thus used as a deductive engine as outlined in [9].

5. METHODOLOGY

The conceptual framework upon which this study is based derives from the theory of distributed computing. The intent of this theory as applied to the Semantic Web is to integrate, synchronize, and merge all data available on the web and make it understandable by both humans and machines [12]. A set of four criteria has been chosen against which the two Semantic Web frameworks described in Section 4 are evaluated. Evaluation consists of the identification of a measurement for each of the criteria and an examination of publications and supporting documentation for each framework to determine the level of support for each of the criteria. Numeric scores are associated with each level of support for each criterion and framework. The total score of each framework is the outcome measurement of comparison between the two frameworks.

The four criteria used for comparison are the level of semantic encoding, the level of data redundancy, supported database platforms, and the amount of tool support. The rest of this section will be devoted to justifications of the use of the criteria, detailed descriptions of these criteria, and the enumerations for the comparison methodology.

Level of semantic encoding was used as a comparison by the TopQuadrant Technology Briefing in their review of semantic technologies [26]. If a system uses a high level of semantic encoding, there will be greater richness and precision in the semantics available to capture the relationships between concepts that the logical reasoning of agents requires.

XML offers a syntax for the writing of structured documents on the web. XML Schema allows for the creation of simple data types within XML documents. RDF is a data model for describing resources on the web, while RDF-S has the capability of creating hierarchies of objects including classes, subclasses, properties, domain and range [1]. The DAML + OIL language was the result of a joint collaboration of United States and European research aimed at defining an ontology modeling language [4]. The W3C used DAML + OIL as a starting point to develop OWL [19]. Due to its expressive power, OWL has been divided into three sublanguages OWL Lite, OWL DL, and OWL Full. OWL Full has the total expressive power of the language, but it can’t be guaranteed to be computationally complete (computable) or decidable (in finite time). OWL DL (Description Logic) has restrictions that give it the maximum power of OWL Full that can be guaranteed to be complete and decidable. OWL Lite is further restricted to allow for easier initial implementation and to provide a simple migration path from thesauri and taxonomies.

The enumerations for the level of semantic encoding appear below. While it is recognized that a true implementation of OWL Full is not practical, it is included here in order to show the full range of semantic encodings available. If the framework employs an internal semantic encoding not directly supported by the W3C, an attempt will be made to match the internal semantic encoding to a W3C equivalent.

Level of Semantic Encoding (from lowest level to highest level):
1. XML
2. XML Schema
3. RDF
4. RDF-S
5. DAML + OIL
6. OWL Lite
7. OWL DL
8. OWL Full

Reducing the amount of redundancy in the data improves the integrity of the data [7]. Some solutions may require complete or partial copying of the data as part of the implementation. For example, extraction of semantic metadata from documents may require that the facts derived from the documents be copied and stored in a database in order to be accessed more efficiently.

Level of Data Redundancy (most redundant to least redundant):
1. Complete data redundancy
2. Partial data redundancy
3. No data redundancy

When dealing with the topic of legacy databases, it is important to realize that a mature information organization is likely to have many databases spanning several eras of information technology. A solution to the problem of exposing legacy databases to semantic technology that is incapable of reaching a broad range of data sources from the full spectrum of data technology will fall far short of the vision of the Semantic Web. The enumerations for supported database platforms appear below.

Supported Database Platforms (most restrictive to least restrictive):
1. One vendor supported
2. Only RDBMS
3. Any database with JDBC and/or ODBC available
4. Any structured database
5. Both structured and non-structured data storage are supported

If a solution does not involve tools to assist in the implementation, that solution is not going to be practical to implement. While it is recognized that adding some kind of
semantic markup or mapping will likely involve a domain or ontology expert, the more tool support for doing this in at least a semi-automated way, the more likely a solution will be accepted and implemented. The enumerations regarding the amount of tool support available for a given framework appear below.

Amount of Implementation Tool Support for Legacy Data Description (from least to most):

1. All changes to implement the solution must be done manually
2. There are tools available (3rd party or within the framework) to handle some implementation details
3. There are tools available (3rd party or within the framework) to handle all implementation details
4. The solution requires no additional tool support (Implementation is totally automated).

6. RESULTS

Table 1. Summary of Comparisons

<table>
<thead>
<tr>
<th>Criteria</th>
<th>OntoAgent</th>
<th>Ontoprise</th>
</tr>
</thead>
<tbody>
<tr>
<td>Semantic Encoding</td>
<td>4</td>
<td>6</td>
</tr>
<tr>
<td>Data Redundancy</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>Supported DBMS</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>Tool Support</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

Table 1 summarizes the comparisons between the two frameworks. The following sections explain the derivations of the scores and discuss the results.

6.1 OntoAgent

Semantic Encoding

OntoAgent stores facts in an RDF triple encoded within a view. The rules are also stored in the database by OntoSQL in the form of views. The documentation of OntoAgent specifically defines the intentions of the design to use the RDF-S level of encoding along with RuleML to encode the axioms [6]. For comparison, the overall encoding of the framework will be assigned to the RDF-S level, level 4, although it is recognized that there may be an easy conversion path to the OWL level of encoding.

Data Redundancy

Semantic annotation on the database in the form of views means that there is no data redundancy and as a result is evaluated at level 3.

Supported Database Platforms

OntoAgent using OntoSQL is designed to work with relational databases only and therefore received a score of 2.

Tool Support

The derivation and creation of the views that support the facts known by the agents must be done manually, therefore the score for tool support gets a value of 1.

6.2 Ontoprise

Semantic Encoding

F-Logic has a high degree of semantic encoding capability as demonstrated by the fact that ontologies can be imported and exported in the OWL Lite format. For this reason, Ontoprise is given a semantic encoding level of OWL Lite, which corresponds to a score of 6.

Data Redundancy

Only the schema of the database is mapped to an ontology. The data is not copied from the database, and therefore receives a score of 3 in this category.

Supported Database Platforms

All databases that support JDBC and ODBC connectivity are available and that earns Ontoprise a score of 3 in this area.

Tool Support

OntoEdit supports the creation and maintenance of ontologies. OntoMap supports the semi-automated mapping of database schemas to ontologies through a graphic interface. Ontoprise will be given a score reflective of some of the implementation details being handled by tools in that mappings must still be done manually by a domain expert with the help of the OntoMat interface. This level of tool support merits a score of 2.

6.3 Summary of Results

Table 1 summarizes the comparisons between the two frameworks. Ontoprise is superior to OntoAgent in terms of semantic encoding (OWL Lite as compared to RDF-S). Ontoprise is also capable of connecting to more kinds of databases than OntoAgent in that Ontoprise is not restricted to relational databases only. Ontoprise provides a tool for the mapping of database schema to ontology whereas OntoAgent expects this to be done manually. The two frameworks are equal in terms of data redundancy in that neither framework expects data to be copied as part of the solution to expose the data to semantic technology.

A major strength of this study was the examination of two different methods of exposing legacy data; thereby allowing for a greater number of perspectives and applications to be assessed. The inclusion of a commercial framework (Ontoprise) allowed for evaluation of a readily accessible product, with implications for real-world business practices. In contrast, the inclusion of a research tool (OntoAgent) provided insight into the merits of an evolving technology.

While broad inclusion provided strength to the study, the comparison of these frameworks became a limitation. OntoAgent was developed in 2002 as a research tool. Among the OntoAgent design goals was the use of current technologies [7]. DAML and OWL were not considered mature enough at that time to use in the OntoAgent design. RuleML has now been combined with OWL [25], so an updated version of OntoSQL could take advantage of this to improve the level of semantic encoding of OntoAgent.
Ontoprise, on the other hand, is a commercial product that has released new versions as recently as March 2004 [18]. As a commercial product, it should be expected that a research and development effort would be able to provide better tools in support of the application and be able to develop technologies to connect to a wider range of data sources. The ongoing development effort would also be expected to incorporate more recent semantic standards as they become mature and accepted. All of these expected differences are expressed in the results of the comparison. The results therefore, should be interpreted with caution as the two frameworks are distinct in their intention and utility.

7. CONCLUSION AND FUTURE WORK

This study compared two semantic web frameworks that expose legacy databases to semantic technology. OntoAgent and Ontoprise were compared in terms of the level of semantic encoding, amount of data redundancy, supported database platforms, and automated tool support. Ontoprise proved to be superior to OntoAgent in the areas of semantic encoding, supported database platforms, and tool support. There was no data redundancy in the solution provided by either framework.

Based on the comparisons made above, we will make some recommendations as to what environment would be an appropriate choice for each framework. Ontoprise would be a good choice when a commercial enterprise-level solution to the problem of exposing not only legacy databases, but documents, web pages, and other data sources to the Semantic Web is needed. OntoAgent, on the other hand, is designed to declaratively create an agent or to expose legacy data via a web service. It could be useful as freeware in a small system. With OntoAgent it is possible, for example, to have the entire system run on one database.

This study is an early effort at identifying comparison criteria for the tools and techniques that expose legacy databases to the semantic web. As the semantic web proliferates, work in this area will escalate in importance.

There is still a great deal of work that can be done in the area of exposing legacy data to the Semantic Web. Studies of the mechanisms used to expose legacy databases to semantic technologies could be used to derive more focused comparison criteria. These criteria could then be used to compare a more comprehensive sampling of semantic web frameworks.

Additional effort could focus on the area of semantic annotation on an existing database by creating views. This work could build on the research of the OntoAgent project [6], the Karlsruhe Ontology project [15], and deductive databases [9]. Work on the creation of semi-automated tools for mapping ontologies to semantic views on databases, including the automated creation of these views, would speed the adoption of this approach. A study using a large number of databases semantically integrated using a combination of annotation and schema extraction could identify best practices and the criteria for adopting one method over the other.

8. REFERENCES


