Integrating Environmental Information: Incorporating Metadata in a Distributed Information Systems Architecture

Stephen Cranefield and Martin Purvis
Information Science Department
University of Otago
Dunedin, New Zealand
Email: {scranefield, mpurvis}@infoscience.otago.ac.nz

Abstract
An approach is presented for incorporating metatata constraints into queries to be processed by a distributed environmental information system. The approach, based on a novel metamodel unifying concepts from the Unified Modelling Language (UML), the Object Query Language (OQL), and the Resource Description Framework (RDF), allows metadata information to be represented and processed in combination with regular data queries.

1 Introduction

In order to make progress on the difficult task of understanding complex systems in the natural environment, it is necessary to take advantage of whatever information may be available. Although large stores of environmental information are becoming increasingly physically accessible, they are often distributed across various remote sites, stored on different platforms and in different formats, and organised according to differing organisational schemas and semantic models. To assist in this matter, distributed information systems are being developed that can help access these scattered sources of environmental data. The most difficult problem that these systems must address is not so much the problem of physically accessing the data, but rather the problem of integrating the various sources of information that have been accessed into a coherent scheme. With environmental information systems, in particular, the data collections that are assembled are often based on mental models of associated physical processes that are often not stored explicitly with the data. When multiple data sources, each based on a separate physical model, are involved, it may be difficult to integrate them into a common framework.

The New Zealand Distributed Information Systems (NZDIS) project (Purvis *et al.*, 2000b) seeks to address these difficulties by employing a software architecture based on a loosely coupled collection of distributed software agents. Each individual agent is presumed to be a specialist for a particular task, and the expectation is that complex projects can be undertaken by a collection of agents, no one of which has the capability of performing all the required tasks of the project. An advantage of an open agent architecture like this one is that individual agents can be replaced by improved models over time, thereby enabling the system to improve gradually, grow in scope, and generally adapt to changing circumstances (Purvis *et al.*, 2000a; Purvis *et al.*, 2000b). Agents receive and reply to requests for services and information by means of a high-level declarative agent communication language, such as FIPA ACL (FIPA 1998), whose message contents may be expressed in terms of formal ontologies that describe the vocabularies of various domains. Software interoperability is supported by encapsulating existing software tools and information sources as agents (by using a 'wrapper' layer of agent code) so that all interactions are expressed in terms of the common agent communication language. Further discussion concerning the overall merits of agent-based architectures for heterogeneous

distributed information systems is provided in (Genesereth & Ketchpel, 1994; Cranefield *et al.* 1994; Cranefield & Purvis, 1997; Purvis *et al.*, 2000b) In this paper the discussion is primarily concerned with how the NZDIS agent-based system is designed to process environmental *metadata* in order to respond to queries directed across heterogeneous data sources.

2 The NZDIS Architecture

The NZDIS agent architecture has several types of specialized agents that provide various types of service within the system: a *user agent* is the interface between a user and the other agents in the system and provides the user interface for the system; an *ontology agent* provides answers to queries about ontologies and metadata; a *broker agent* accepts registrations from other agents and then uses the registration information to respond to incoming queries from agents that wish to locate a certain type of service provider; a *resource agent* (such as a data source agent or a computational agent) provides an interface between the other agents and the specific command or query language associated with a data resource or information processing module, and *query processing agents* are part of a specialized subsystem that deal with requests for information from user agents and organize a task plan that will respond to those requests. A schematic illustration of the NZDIS architecture is shown in Figure 1. Each agent is depicted by a rectangle. Links with black arrowheads indicate agent messages and links with white arrowheads indicate that an object reference is accessed.

Within the NZDIS system, a query, encoded as the content of an agent communication language message, is expressed in terms of the Object Database Management Group's Object Query Language (OQL) (Cattell, 1997). A Data Source Agent (see Figure 1) provides the interface to a particular database and translates the query into the appropriate query language for the database.

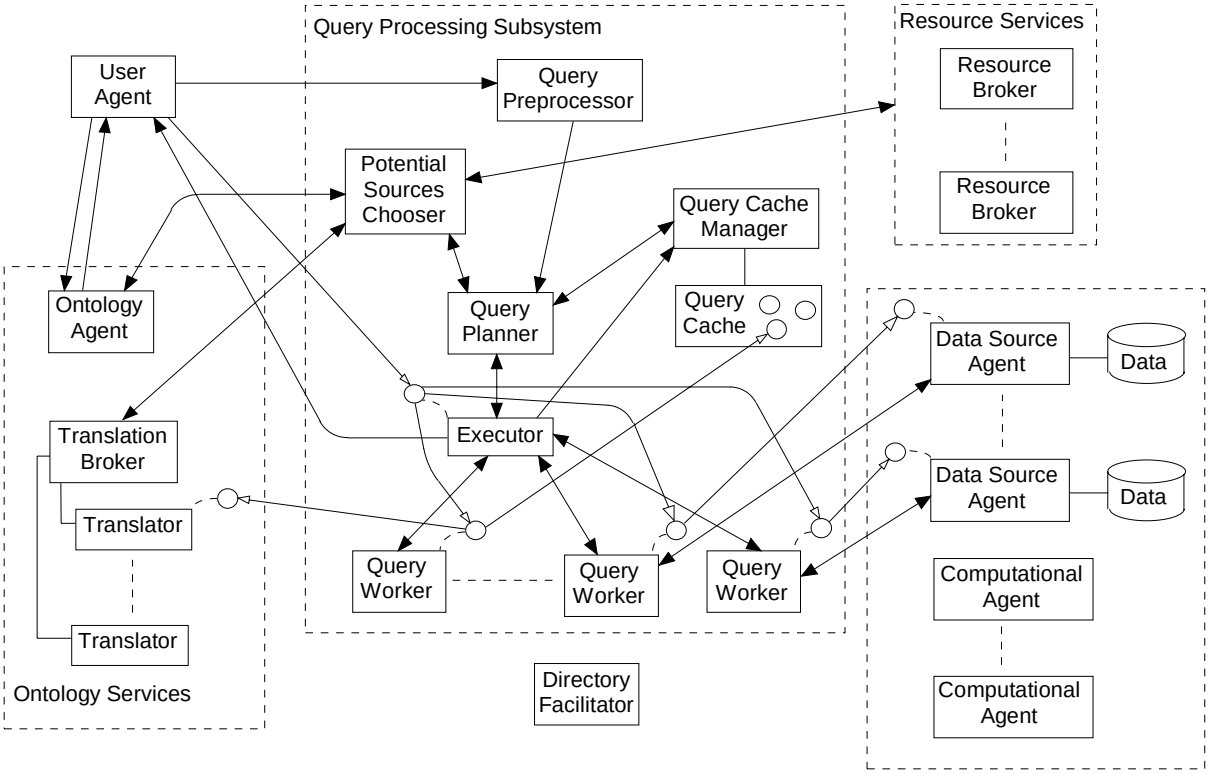

Figure 1. The NZDIS agent-based architecture.

We illustrate further aspects of the system by means of an example query. Suppose, for example, there are two separate data collections available:

- ✦ one collection containing questionnaire responses pertaining to asthma incidence and including geographical location in NZ Map Grid coordinates of respondents.

- ✦ and another collection containing climatological information with respect to geographical location.

A user may be interested in collecting questionnaire records for all the people who suffer asthma and who live in areas with average humidity greater than 70%. The system provides software utilities that access ontologies services (see Figure 1) enabling the user to browse the relevant ontologies in order to formulate a suitable query. For our example we assume that there are separate ontologies describing the domains of the two data sources, one for the asthma survey (AsthmaOnt) and one for the climate database (ClimateOnt). In the NZDIS system, ontologies are represented by the Unified Modelling Language (UML) from the Object Management Group (OMG).

Figure 2 shows the AsthmaOnt ontology represented in a UML class diagram, which describes the names and relationships among elements of the asthma survey domain.

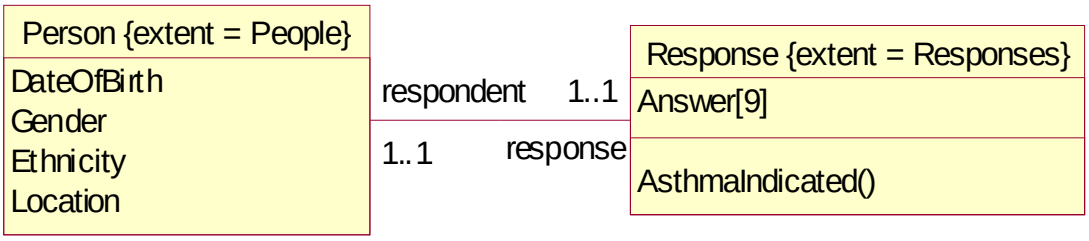

Figure 2. The AsthmaOnt ontology in UML.

The climate ontology is shown in Figure 3. A meshblock is the smallest geographical unit for which statistical data is collected by the New Zealand government.

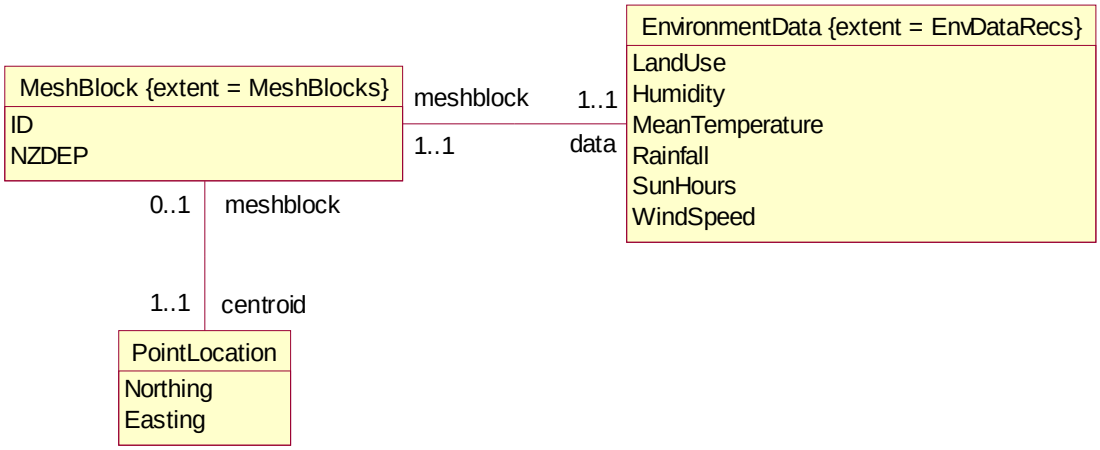

Figure 3. The ClimateOnt ontology for climate data.

With access to these ontologies, the user can interact with the user agent to formulate a query which will be sent to the *Query Preprocessor.* The message content, expressed in terms of the Object Query Language (OQL) from the Object Data Management Group (ODMG), may look something like the following:

```
select p
from AsthmaOnt:Person as p, ClimateOnt:Meshblock as m
where p.Location = m.ID
  and m.data.Humidity > 70
  and p.Response.AsthmaIndicated()
```

Ultimately a query plan is generated by the *Query Planner* agent, and the *Executor* agent generates *Query Workers* which obtain data from individual data sources. The *Executor* agent then combines results from the individual *Query Worker* agents and informs the *User Agent* of the results.

3 Representing and Processing Metadata

The query discussed above is phrased in terms of information that is directly represented in the datasets, e.g. humidity is represented by a field in a relational database (and is presented to the rest of the NZDIS system by the associated data source agent as an attribute of the EnvironmentData class). However, there is often additional information available about a dataset that is not explicitly encoded in its data model. Such information might include the owner of the data, a description of the dataset, its format and the spatial region and/or time period that the data relates to. Such additional information is known as metadata (data about data) and the increasing interconnectivity of computer networks has led to widespread efforts to promote the encoding of metadata in machine-readable formats and the development of standard sets of metadata elements that are common to particular application domains (ANZLIC 1999, DESIRE 1999, Dublin 2000, GILS 1999, GSDI 1999, ISO/TC 2000, LBLNL/EPA 1998, MDC 1999, W3C 1999). In addition, a restricted set of metadata elements has been identified as likely to be useful across many domains. This is known as the Dublin Core (Dublin, 2000).

To date, most uses of metadata have focussed on resource discovery, whereby resources satisfying particular requirements are located by searching repositories of metadata records describing the available resources. In this paper we demonstrate how the NZDIS architecture incorporates metadata records within its query mechanism and allows a single query to include constraints on both the data to be retrieved and the metadata describing that data. This will be illustrated by considering an extension to the example query discussed above.

3.1 Expressing metadata using the Resource Description Framework

Suppose that the asthma questionnaire data has been collected over a long period of time as funding became available incrementally and that we now wish to modify the query to exclude any data older than January 1st, 1995. The asthma questionnaire dataset (and therefore the ontology in Figure 2) does not include an explicit date field associated with each person/questionnaire pair. However, let us assume that this data is provided by an organisation that upholds a strict policy of documenting each dataset in terms of the Dublin Core element set. It is then possible to determine the date on which the data was collected by reading the record for the temporal coverage of the data. We assume that the Dublin Core metadata is encoded using the Resource Description Framework (RDF) (RDF 2000) — an abstract model and associated

XML-based syntax for representing metadata. This is increasingly likely to be the case, as RDF is becoming widely supported in tools by numerous software vendors including Microsoft, IBM and Netscape.

Figure 4 shows how the RDF encoding of the Dublin Core element set can be used to describe the temporal coverage of a dataset ds.

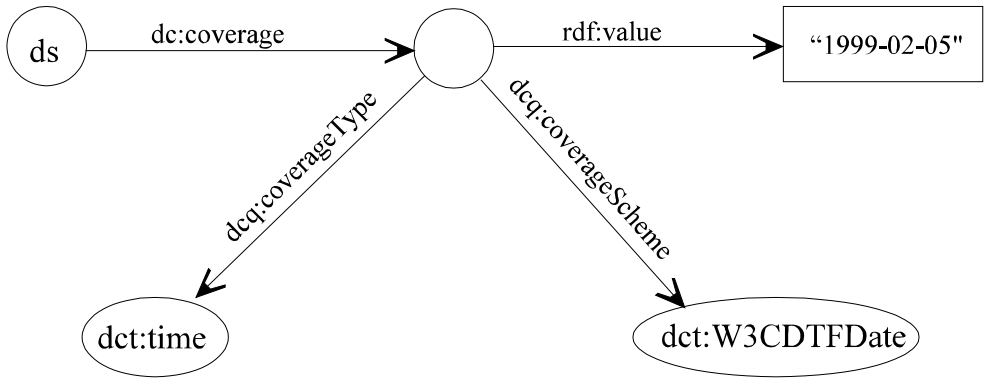

Figure 4. RDF encoding of the Dublin Core representation for temporal coverage of a data set.

The semantic model for RDF is based on the concepts of *resource* (anything that can be referenced by a Uniform Resource Identifier (URI) (URI 2000)), *property* (a resource representing an element of some publicised metadata vocabulary) and *literal* (a value having no persistent existence). An RDF statement consists of three parts: the *predicate* (a property), the *subject* (a resource) and the *object* (either a literal or another resource). In Figure 4, statements are represented by arrows, with the predicate appearing as a label above the arrow, the subject being the source of the arrow and the object the arrow's target. Resources are denoted by ellipses and literals by rectangles.

To encode the temporal coverage of a dataset, the Dublin Core element 'coverage' is used. To further specify the type of coverage described (temporal rather than spatial) and the vocabulary used to describe the value (a string in W3C Date/Time Format (W3C 1997)), the Dublin Core RDF encoding specifies that the object of the coverage statement should be an intermediate resource, with three further RDF statements used to decribe the coverage type (time) and coverage encoding scheme (W3CDTF format). In the XML syntax for RDF, the XML namespace mechanism is usually used to associate a prefix (such as "dc", "dcq" and "dct") with a URL prefix for a standard set of properties (Dublin Core elements, qualifiers and terms respectively in this example). For ease of reading, Figure 4 uses this shorthand notation without defining any namespace prefixes. Note that a standard set of Dublin Core terms (possible values for the objects of qualifier statements) has not yet been defined, so the resources shown with the "dct" prefix are speculative.

3.2 A metamodel ontology incorporating RDF concepts

The NZDIS architecture is built around two object-oriented technology standards: the Unified Modelling Language (UML), used for representing ontologies, and the Object Data Management Group's Object Query Language (OQL), used for encoding queries generated by the user interface

agent and passed on to the query processing subsystem. The entry point to data retrieval in an OQL query is the notion of an *extent*: the set of all objects of a particular type. An object-oriented database system must provide a way for the system administrator to define the names for the extents that should be maintained by the system. This means that as objects are created or deleted from the database, the system must update the collection of references representing that type's extent. Note that the ontologies in Figures 2 and 3 use the UML tagged value notation to specify the name of extents that should be maintained for all classes except PointLocation.

In theory, RDF metadata can be attached to anything identifiable by a URI: even individual objects. However, in practice, most metadata describes complete datasets. In order to add the ability to refer to datasets (and therefore their associated metadata) within OQL, a way is needed of navigating within a query from an extent to a dataset that includes objects of that extent's type. This is done by associating a type's "universal extent" with a set of "component extents", each of which represents the set of all objects of a given type within a particular dataset. This is derived from the bags of extents associated with each type in the DISCO system (Tomasic *et al.*, 1998), where each extent in the bag corresponds to one data source. However, as Tomasic *et al.* do not present the metamodel used in DISCO (only an informal description is given of the extensions made to the ODMG model), it is not possible to make a precise comparison of our

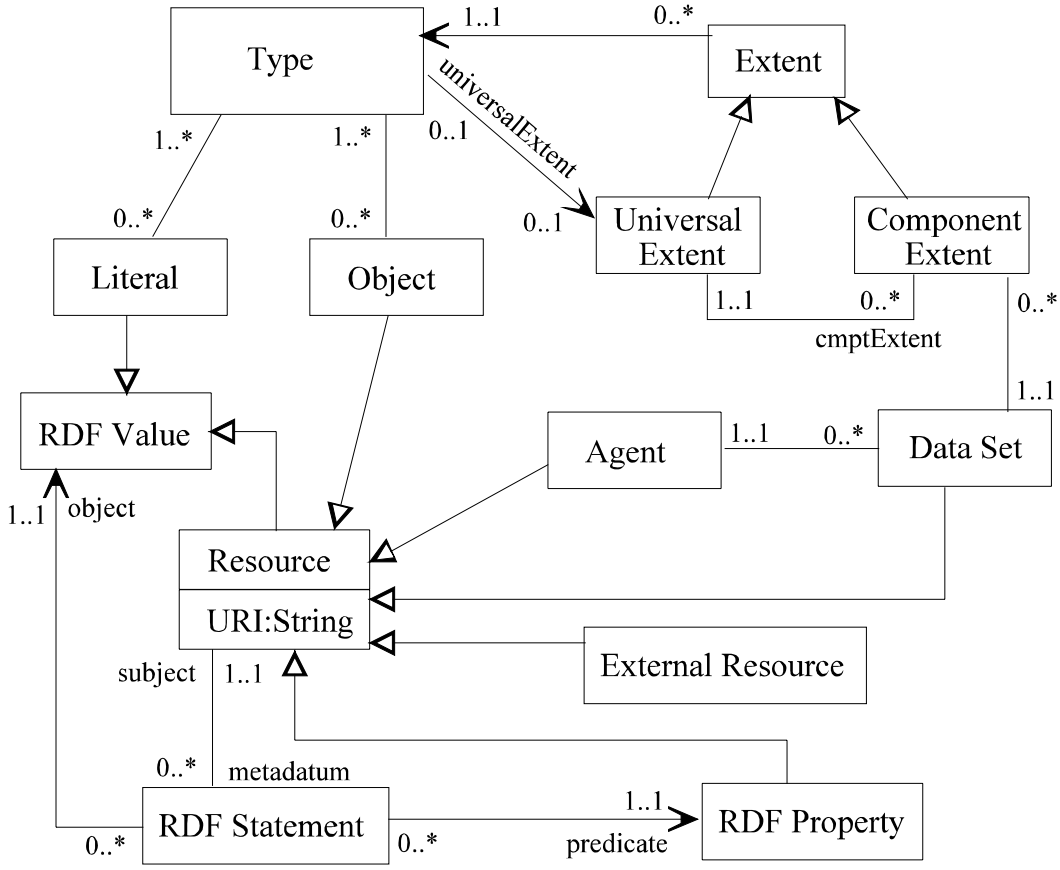

Figure 5. A metamodel ontology incorporating OMG, ODMG, and RDF concepts.

solution to theirs.
Figure 5 presents a "metamodel ontology" that is used in the NZDIS system to integrate concepts from the UML, ODMG and RDF metamodels, and thus to allow constraints on RDF metadata to be included in OQL queries. We do not include the ODMG constructs related to metadata in this metamodel. These are a set of interfaces that only define the functionality of a repository for storing database schemas but not metadata in general. The model of metadata defined by RDF is more general and better suited to our requirements. Note that a full version of this metamodel would include additional concepts such as class inheritance and attributes and operations of classes — these enhancements can be taken directly from the UML metamodel.

The RDF diagram shown in Figure 4 can now be represented as an instance of this metamodel as shown in Figure 6. Figure 7 then shows how, based on Figure 6, a condition on the coverage time of the Asthma Questionnaire dataset can be added to the original OQL query from Section 2. Lines 1 to 3 and 11 to 13 of this query are unchanged from the original query. The new variables declared in lines 4 to 10 of the `from` clause are used to define a pattern of RDF statements similar in topology to that shown in Figure 6, while lines 14 to 26 assert the required values for the predicates and objects of these statements. In particular, line 26 adds the condition that the dataset *ds*'s temporal coverage (now represented by the expression `value_stmt.object`) is a date after 1 January 1995. This is done by constructing a Date object initialised from the object of the `rdf:value` statement (which is known to be a string if the condition on lines 22 and 23 succeeded). The Date class's `after` method is then used to check this value against a String literal representing 1 January 1995.

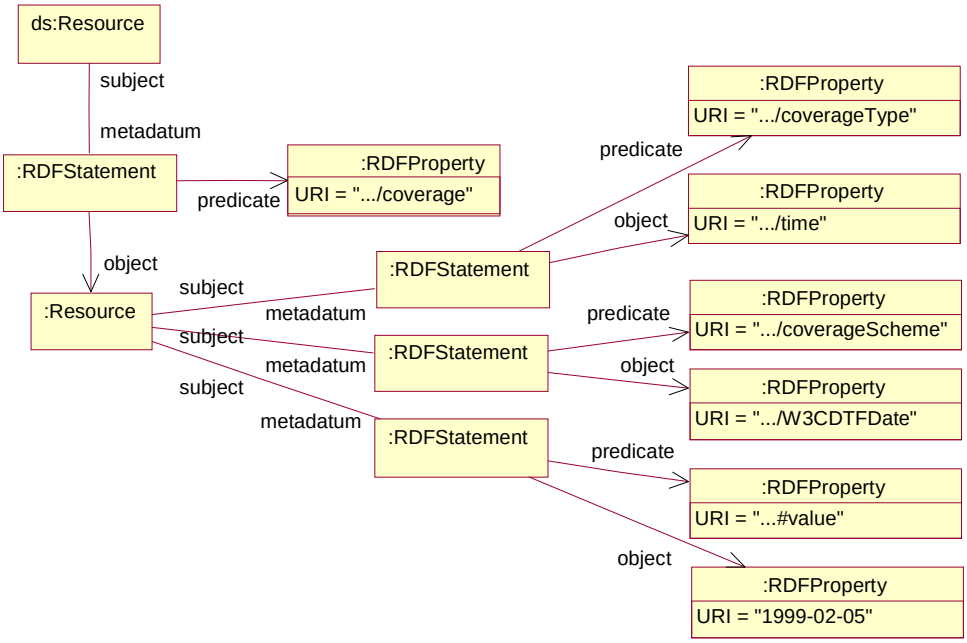

Figure 6. UML object diagram showing the RDF metadata of Figure 4 expressed in terms of the metadata ontology (Figure 5).

```
1   select p
2   from  AsthmaOnt:People as p,
3         ClimateOnt:Meshblock as m,
4         p.cmptExtent as ce,
5         ce.dataset as ds,
6         ds.metadatum as coverage_stmt,
7         coverage_stmt.object as obj,
8         obj.metadatum as qual_type_stmt,
9         obj.metadatum as qual_scheme_stmt,
10        obj.metadatum as value_stmt
11  where p.Location = m.ID
12    and m.data.Humidity > 70
13    and p.response.AsthmaIndicated() and
14    and coverage_stmt.predicate.URI =
15        "http://purl.org/dc/elements/1.0/coverage"
16    and qual_type_stmt.predicate.URI =
17        "http://purl.org/dc/qualifiers/1.0/coverageType"
18    and qual_type_stmt.object.URI =
19        "http://purl.org/dc/terms/1.0/time"
20    and qual_scheme_stmt.predicate.URI =
21        "http://purl.org/dc/qualifiers/1.0/coverageScheme"
22    and qual_scheme_stmt.object.URI =
23        "http://purl.org/dc/terms/1.0/W3CDTFDate"
24    and value_stmt.predicate.URI =
25        "http://www.w3.org/1999/02/22-rdf-syntax-ns#value"
26    and Date((String)value_stmt.object).after(Date("1995-01-01"))
```

Figure 7. An OQL query that includes metadata in the query.

Figure 8 shows the skeleton of the Date ontology used in this query. Constructing Date objects in the query using constructors required an extension to the OQL, which does not include the concepts of user-specified constructor operations.

While this query may seem complicated, it should be remembered that this is not intended as the format used by the user to construct queries, rather it is constructed by the user interface agent based on the user's interactions with a graphical user interface. What this example query demonstrates is how the metamodel ontology allows metadata concepts expressed using RDF to be incorporated into a standard object-oriented database query language such as OQL. This allows a uniform treatment of data and metadata within the query processing system which has considerable benefits in terms of simplicity of system architecture, ease of system modification and reuse of design and code.

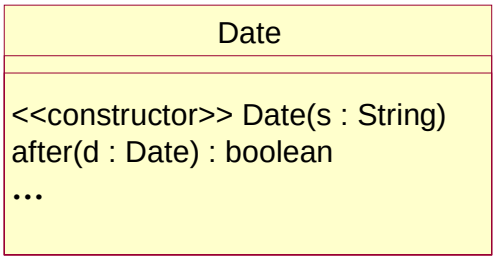

Figure 8. The Date ontology referenced in Figure 7.

4 Conclusion

The NZDIS project is based on the idea of wrapping heterogeneous environmental information sources with agents that exchange information using a standard agent communication language. This paper describes how environmental metadata is included in this scheme to enhance the power and efficiency of general queries across these distributed information sources. The approach described provides a manner in which queries explicitly referencing metadata can be systematically represented in an OQL format. It is based on a novel metamodel that unifies concepts from UML, OQL, and RDF (shown in Figure 5).

References

[ANZLIC, 1999] ANZLIC Metadata Working Group home page, http://www.anzlic.org.au/metagrp.htm, 1999.

[Cattell *et al.*, 1997] R.G.G. Cattell, D. Barry, D. Bartels, M. Berler, J. Eastman, S. Gamerman, D. Jordan, A. Springer, H. Strickland, and D. Wade, editors. *The Object Database Standard: ODMG 2*.0. Morgan Kaufmann, 1997.

[Cranefield *et al.*, 1995] Cranefield, S. J. S., Gorman, P., and Purvis, M. K., "Communicating Agents: An Emerging Approach for Distributed Heterogeneous Systems", *New Zealand Journal of Computing*, 6:1B (1995) 337-343.

[Cranefield & Purvis, 1997] Cranefield, S. and Purvis, M., "An agent-based architecture for software tool coordination", *Intelligent Agent Systems: Theoretical and Practical Issues*, L. Cavedon, A. Rao, W. Wobcke (eds.), Springer-Verlag Lecture Notes in Artificial Intelligence, vol. 1209,(1997) 44-58.

[DESIRE, 1999] DESIRE: Development of a European Service for Information on Research and Education, http://www.lub.lu.se/desire/, 1999.

[Dublin, 2000] Dublin Core Metadata Initiative home page, http://purl.org/dc/, 2000.

[FIPA, 1998] FIPA 98 Specification Documents, http://www.fipa.org/spec/fipa98.html, 1999.

[Gateway, 2000] The Gateway to Educational Materials home page, http://www.geminfo.org/, 2000.

[Genesereth & Ketchpel, 1994] Genesereth, M. R. and Ketchpel, S. P. "Software Agents Software agents. *Communications of the ACM*, 37(7):48–53, July 1994.

[GILS, 1999] Global Information Locator Service home page, http://www.gils.net/locator.html, 1999.

[GSDI, 1999] Global Spatial Data Infrastructure home page, http://www.gsdi.org/, 1999.

[ISO/TC, 2000] ISO/TC 46 Subcommittee on presentation, identification and description of documents home page, http://www.nlc-bnc.ca/iso/tc46sc9/index.htm, 2000.

[LBLNL EPA, 1998] LBNL EPA Scientific Metadata Standards Project home page, http://pueblo.lbl.gov/~olken/epa.html, 1998.

[MDC, 1999] Meta Data Coalition home page, http://www.mdcinfo.com/, 1999.

[Purvis *et al.*, 2000a] Purvis, M., Cranefield, S., and Nowostawski, M., "A Distributed Architecture for Environmental Information Systems" to appear in Proceedings of the *International Symposium on Environmental Software Systems*, IFIP Series, Kluwer Academic (2000).

[Purvis *et al.* 2000b] Purvis, M., Cranefield. S., Bush, G., Carter, D., McKinlay, B., Nowostawski, M., and Ward, R., "The NZDIS Project: an Agent-based Distributed Information Systems Architecture", *Proceedings of the Hawai`i International Conference on System Sciences (HICSS-33)*, R. H. Sprague, Jr. (ed.), (CD ROM) IEEE Computer Society Press, Los Alamitos, CA (2000).

[RDF 2000] Resource Description Framework, W3C Technology and Society Domain, http://www.w3.org/RDF/, 2000/.

[Tomasic *et al.*, 1998] Tomasic, A., Raschid, L., and Valduriez, P., "Scaling Access to Heterogeneous Data Sources with DISCO", *IEEE Transactions on Knowledge and Data Engineering*, 10(1), January 1998.

[URI 2000] Uniform Resource Identifiers Working Group, Internet Engineering Task Force, http://www.ics.uci.edu/pub/ietf/uri/, 2000.
[W3C, 1997] “Date and Time Formats”, W3C Technical Report, http://www.w3.org/TR/NOTE- datetime.html, 1997.
[W3C, 1999] W3C Technology and Society Domain, Metadata Activity Statement, http://www.w3.org/Metadata/Activity.html, 1999.