# Intelligent Resource Discovery using Ontology-Based Resource Profiles

## 19th International CODATA Conference
## 7-10 November 2004, Berlin, Germany

J. Steven Hughes[(1)], Dan Crichton[(1)], Sean Kelly[(1)], Jerry Crichton[(1)], Thuy Tran[(1)]

[(1)]Jet Propulsion Laboratory
4800 Oak Grove Drive
Pasadena, California 91109
USA
{steve.hughes@jpl.nasa.gov, dan.crichton@jpl.nasa.gov, sean.kelly@jpl.nasa.gov,
gerald.crichton@jpl.nasa.gov, thuy.tran@jpl.nasa.gov}

**ABSTRACT**

Successful resource discovery across heterogeneous repositories is strongly dependent on the semantic and syntactic homogeneity of the associated resource descriptions. Ideally, resource descriptions are easily extracted from pre-existing standardized sources, expressed using standard syntactic and semantic structures, and managed and accessed within a distributed, flexible, and scaleable software framework. The Object Oriented Data Technology task has developed a standard resource description scheme that can be used across domains to describe any resource. It uses a small set of generally accepted, broad scope descriptors while also providing a mechanism for the inclusion of domain specific descriptors. In addition this standard description scheme can be used to capture hierarchical, relational, and recursive relationships between resources. In this paper we will present a intelligent resource discovery framework that consists of separate data and technology architectures, the standard resource description scheme, and illustrate the concept using a case study.

**INTRODUCTION**

The Object Oriented Data Techonolgy task was funded in 1998 by NASA's Office of Space Science to develop a national software framework for sharing data across heterogeneous, distributed data repositories. The resulting framework consists of separate but complementary data and software architectures that enable data sharing across multiple science and engineering disciplines.

The software architecture provides reusable software components with common interfaces, allows new components to be easily integrated into the framework, and provides a mechanism to wrap legacy data system components with minimal impact. The complementary data architecture is domain dependent and can be used either within a single domain or across different domains. Its effectiveness is primarily dependent on the maturity of the data model provided for the target domains. The framework also supports location independence in that the user describes what they want, not how or where to get what they want. The framework can also be scaled to meet increases in the number of interconnected repositories.

As part of the data architecture, the design and development of a standard resource description has enabled intelligent resource discovery across distributed heterogeneous resource repositories. This paper focuses on this capability and describes the data architecture, summarizes the supporting technology architecture, and provides an example case study.

## DATA ARCHITECTURE

Architecture is a term applied to both the process and the outcome of thinking out and specifying the overall structure, logical components, and the logical interrelationships of a computer system. We define data architecture to be the application of this concept to the data components involved in a computer system.

A key assumption in the development of a data architecture is that the data components to some degree model a specific problem domain. For example, in the space science domain, an image collected by a spacecraft instrument is modeled as a 2-dimensional structure of lines and line_samples that was collected within a context defined by the states of the instrument and spacecraft at the time the image was collected. To capture this model and specify the overall structure, logical components, and logical interrelationships within the domain, we need a formal mechanism. For this discussion we will focus on ontologies.

An ontology is a set of concepts - such as things, events, and relations - that are specified in some way in order to create an agreed-upon vocabulary for exchanging information within a domain. Historically there are many methodologies for defining and collecting various aspects of domain concepts into a domain model. For example the Entity-Relationship (E-R) model is a widely accepted data modeling technique that focuses on the definition of domain entities and their relationships. It defines an entity as something that exists either in concept or in actuality. Entities are defined using properties, often called attributes, and relationships that relate two or more entities. Subsequently an E-R model is typically used to implement a database application using a specific record level methodology such as the Relational Model. Other methodologies involve the creation of taxonomies and controlled vocabularies. For this discussion we assert that ontologies subsume the domain concept information collected by any of these methodologies, specifically that which is necessary for intelligent resource discovery. However in the following we will often use familiar technique such as the E-R model in explaining specific aspects of a domain model. In addition, an ontology allows the creation of a knowledge base through the ingestion of instances of the domain concepts being described. A knowledge base is an important extension since it allows a more complete description of the domain, helps validate the model or schema, can be used to develop domain classifiers that classify new instances, as well as supporting more advance search.

As examples, the image, instrument, and spacecraft mentioned earlier are all considered entities in the space science domain. The image entity is defined using attributes that describe its logical structure, namely the 2-d structure of lines and line_samples and attributes that describe the context within which the image was collected, such as instrument_id, filter_name, and exposure_duration. The spacecraft and instrument entities also help provide context through their relationship to the image entity. For example the filter_name and exposure_duration are actually attributes of the instrument but are associated with the image through the relationship “<instrument> produces <image>” in the data model.

## DATA DICTIONARY

A data dictionary or controlled vocabulary is a basic component of a data architecture and is the key mechanism for defining entity attributes. It is a list of terms, their definitions, permissible values, and other characteristics that together are called a data element. For example, the attribute exposure_duration mentioned earlier is a data element in the planetary science data dictionary and is defined as the period of time over which data is collected by an instrument. It takes on a floating point value and is measured in units of milliseconds.

At the element level, special attributes are needed to define data elements. These attributes address such characteristics as the “name”, “definition”, and “permissible values” of an element. Also called meta-attributes, these attributes are used to formally collect element attributes and create a domain data dictionary.

As part of the information architecture, we have chosen the ISO/IEC 11179 standard framework for the specification and standardization of data elements [6]. It provides an accepted base set of attributes needed

to describe data elements. As an international standard it also provides a global basis for data element definition and classification and supports data dictionary interoperability. The specification classifies the base set of attributes into four categories namely identifying, definitional, representational, and administrative as briefly summarized in Figures 1 and 2.

| | ISO/IEC 11179 Attributes |
|---|---|
| **Attribute** | **Value** |
| ***Identifying Attributes*** | |
| Name | Single or multi word designation assigned to a data element. |
| Identifier | A language independent unique identifier of a data element within a Registration Authority. |
| Version | Identification of an issue of a data element specification in a series of evolving data element specifications within a Registration Authority. |
| Registration Authority | Any organisation authorized to register data elements. |
| Synonymous name | Single word or multi word designation that differs from the given name, but represents the same data element concept. |
| Context | A designation or description of the application environment or discipline in which a name and/or synonymous name is applied or originates from. |
| Definition | Statement that expresses the essential nature of a data element and permits its differentiation from all other data elements. |
| ***Relational Attributes*** | |
| Classification scheme | A reference to (a) class(es) of a scheme for the arrangement or division of objects into groups based on characteristics which the objects have in common, e.g. origin, composition, structure, application, function etc. |
| Keyword | One or more significant words used for retrieval of data elements. |
| Related data reference | A reference between the data element and any related data. |
| Type of relationship | An expression that characterizes the relationship between the data element and related data. |

**Figure 1 - ISO/IEC 11179 Attributes**

| ***Representational Attributes*** | |
|---|---|
| Representation category | Type of symbol, character or other designation used to represent a data element. |
| Form of representation | Name or description of the form of representation for the data element, e.g. 'quantitative value', 'code', 'text', 'icon'. |
| Datatype of data element values | A set of distinct values for representing the data element value. |
| Maximum size of data element values | The maximum number of storage units (of the corresponding datatype) to represent the data element value. |
| Minimum size of data element values | The minimum number of storage units (of the corresponding datatype) to represent the data element value. |
| Layout of representation | The layout of characters in data element values expressed by a character string representation. |
| Permissible data element values | The set of representations of permissible instances of the data element, according to the representation form, layout, datatype and maximum and minimum size specified in the corresponding attributes. The set can be specified by name, by reference to a source, by enumeration of the representation of the instances or by rules for generating the instances. |
| ***Administrative Attributes*** | |
| Responsible organization | The organization or unit within an organization that is responsible for the contents of the mandatory attributes by which the data element is specified. |
| Registration status | A designation of the position in the registration life-cycle of a data element. |
| Submitting organization | The organization or unit within an organization that has submitted the data element for addition, change or cancellation/withdrawal in the data element dictionary. |
| Comments | Remarks on the data element. |

**Figure 2 - ISO/IEC 11179 Attributes (cont)**

The “identifying” category is used for the identification of a data element. For example, exposure_duration would be the value of the attribute “name”. The “definitional” category is used to describe the semantic aspects of a data element and consists of a textual description that communicates knowledge about the data element that typically is not captured by any of the basic attributes. The “relational” category describes associations among data elements and/or associations between data elements and classification schemes, data element concepts, objects, or entities. For example relating exposure_duration to an instrument entity provides critical information about how exposure_duration is to be interpreted. The “representational” category describes representational aspects of data element such the list of permissible data values and their type. For example, exposure_duration would be typed as floating point. Finally the “administrative” category provides management and control information.

## COMMON DATA ELEMENTS

With the advent of the web and the resulting explosion of electronic resources available for online access, there was a compelling need for standard attributes for electronic resources. Since the information contained in these electronic resources span the breadth of human knowledge, the set of standard attributes that span all electronic resources would be necessarily limited in number and each attribute would be very broad in scope.

The Dublin Core initiative specifically addressed this issue and developed the 15 data elements or attributes briefly summarized in Figure 3 below. They were defined using the ISO/IEC 11179 framework and their complete definitions are available on the web.

| | Dublin Core Data Elements |
|---|---|
| Title | A name given to the resource. |
| Creator | An entity primarily responsible for making the content of the resource. |
| Subject | The topic of the content of the resource. |
| Description | An account of the content of the resource. |
| Publisher | An entity responsible for making the resource available |
| Contributor | An entity responsible for making contributions to the content of the resource. |
| Date | A date associated with an event in the life cycle of the resource. |
| Type | The nature or genre of the content of the resource. |
| Format | The physical or digital manifestation of the resource. |
| Identifier | An unambiguous reference to the resource within a given context. |
| Source | A Reference to a resource from which the present resource is derived. |
| Language | A language of the intellectual content of the resource. |
| Relation | A reference to a related resource. |
| Coverage | The extent or scope of the content of the resource. |
| Rights | Information about rights held in and over the resource. |

**Figure 3 - Dublin Core Elements**

As mentioned previously, the Dublin Core (DC) attributes are by definition very general and when used as search constraints do not always produce precise results. They were developed as common attributes to describe Internet electronic resources across all possible domains and this generality limits their ability to partition the search space into easily managed subsets. For example a search for electronic resources that have a Format of value “image/jpeg” will typically result in a large number of images from almost any repository. The addition of additional DC attributes as search constraints help to further partition the search space, however their effectiveness is strongly dependent on the mapping of DC attributes to the specific domains and the adherence to this mapping during repository ingestion. The solution to this problem as suggested by the DC Subject attribute is the addition of concepts from the specific domains. This presupposes the existence of domain models at least in the form of taxonomies if not complete domain ontologies.

The existence of a domain model, say at the level of an E-R diagram, works well for resource discovery in the simple cases where only a few resource types exist. For example, the search for specific images from among the approximately 49,000 Viking images of the planet Mars is easily accomplished by designing a relational table for the Viking image attributes and loading 49,000 records. However, other Mars missions such as Mars Global Surveyor, Mars Odyssey, and the Mars rover missions, are providing thousands more Mars images that should be available for searching in a single session. In all cases however, additional image attributes have been added to capture new information that was not available in prior missions. Traditional design approaches suggest that either separate mission databases be created or that single database be created that spanned all missions. The former results in additional management overhead while the second results in a database that is sparsely populated. In reality, missions to Mars will continue and in fact, several thousand product types exist in the planetary science archive. Neither approach is a general solution for the simple or correlative search across the entire archive. The solution to this problem is a single resource description sufficient for describing all resource types. In the following we will describe this resource description and describe efforts to implement a search capability based on this construct.

**GENERALIZED RESOURCE DESCRIPTION – RESOURCE PROFILE**

In the data architecture, an electronic resource will be described by a profile, an XML document that uses both domain specific attributes and the Dublin Core attributes to concisely define a resource. A profile has three sections: the profile attributes, the resource attributes, and domain specific attributes called the profile element section. The first section, the profile attributes, simply describe the profile using attributes such as identifier, type, and status. The identifier attribute is typically implementation dependent and could be an Object Identifier (OID), Universal Resource Identifier (URI), or sequence numbers. The type attribute is typically 'profile' but 'data dictionary' has been used when the XML document was used to simply capture data element information.

The second section, the resource attributes, generically describes the resource using the Dublin Core (DC) attribute set. All DC attributes are allowed but only Identifier is required. For the data architecture, three additional resource attributes have been added to identify the resource's local domain (resContext), classification (resClass) , and location (resLocation). The valid values assigned to the DC attributes are typically taken from selected domain specific attributes. For example, the DC attribute Title, a "label" for the resource, could take values from a domain specific resource identifier.

Finally, the profile element section encodes domain specific attributes associated with the resource. These attributes are extracted from a formal domain data model. The XML DTD for the profile is provided in Figure 4 below.

```
<!ELEMENT profiles
 (profile*)>

<!ELEMENT profile
 (profAttributes,
  resAttributes,
  profElement*)>

  <!ELEMENT profAttributes
   (profId, profVersion?, profType,
    profStatusId, profSecurityType?, profParentId?, profChildId*,
    profRegAuthority?, profRevisionNote*, profDataDictId?)>

  <!ELEMENT resAttributes
   (Identifier, Title?, Format*, Description?, Creator*, Subject*,
    Publisher*, Contributor*, Date*, Type*, Source*,
    Language*, Relation*, Coverage*, Rights*,
    resContext+, resAggregation?, resClass, resLocation*)>

  <!ELEMENT profElement
   (elemId?, elemName, elemDesc?, elemType?, elemUnit?,
    elemEnumFlag, (elemValue* | (elemMinValue, elemMaxValue)),
    elemSynonym*,
    elemObligation?, elemMaxOccurrence?, elemComment?)>
```

**Figure 4 - Profile Schema - DTD**

For example, if a profile were used to describe the planetary science image mentioned above, the profile element section would encode the attribute names, instrument_id, filter_name, and exposure_duration, and their values to ensure a precise characterization of the image. Figure 5 below illustrates this concept.

```
<profile>
 <profAttributes>
  <profId>1.3.6.1.4.1.1306.2.11480003140</profId>
  <profType>profile</profType>
  <profStatusId>active</profStatusId>
 </profAttributes>
 <resAttributes>
  <Identifier>GO-J/JSA-SSI-2-REDR-V1.0:24I0131</Identifier>
  <Title>GO-J/JSA-SSI-2-REDR-V1.0:24I0131</Title>
  <Format>image/pds</Format>
  <resContext>NASA.PDS</resContext>
  <resClass>data.product</resClass>
  <resLocation>http://product_server_query_to_return_actual_product...
 </resAttributes>
 <profElement>
  <elemName>TARGET_NAME</elemName>
  <elemType>CHARACTER</elemType>
  <elemValue>IO</elemValue>
 </profElement>
 <profElement>
  <elemName>INSTRUMENT_ID</elemName>
  <elemType>CHARACTER</elemType>
  <elemValue>SSI</elemValue>
 </profElement>
 <profElement>
  <elemName>FILTER_NAME</elemName>
  <elemType>CHARACTER</elemType>
  <elemValue>RED</elemValue>
 </profElement>
 <profElement>
  <elemName>EXPOSURE_DURATION</elemName>
  <elemType>REAL</elemType>
  <elemValue>62.50</elemValue>
 </profElement>
```

**Figure 5 - Data Product Profile - Example**

## TECHNOLOGY ARCHITECTURE

The Object Oriented Data Technology (OODT) framework that supports the data architecture previously described consists of a set of cooperating, distributed peer software components. The major components of the OODT framework implement a metadata (profile) and data (product) model using profile servers and product servers. In addition, a query service directs queries by traversing a network of connected profile and product servers, providing the veneer of a peer-to-peer network. The distributed services provide for the location and description of resources (profile queries) and retrieval of resources (product queries) leveraging the profile metadata model. In the following we briefly describe the OODT framework. A detail description of the framework is provided in [1, 2, 3, 4].

## DISTRIBUTED FRAMEWORK COMMUNICATION

OODT is a distributed system, wherein components may be dispersed geographically across a standard TCP/IP network, such as the Internet. Connectivity between components utilizes a standards based distributed systems implementation such as Java Remote Method Invocation (RMI) or the Internet Inter-ORB Protocol (IIOP) for CORBA-based communication.

The OODT components support plug-ins that extends the implementation by performing the work of querying both the metadata catalogs and the data repositories themselves. In this way, the OODT software is a framework in that application programmers extend and implement prescribed software objects and interfaces that directly integrate into the framework. This is in contrast to normal software implementation efforts that may not specify ubiquitous interfaces. More work necessarily falls upon the developers of the framework to support the prescriptive interfaces.

OODT's framework provides three major components:

- Profile servers serve scientific metadata and can tell whether a particular resource can provide an answer to a query.
- Product servers serve data products in a system-independent format.
- Query servers accept profile and product queries and traverse the network of profile and product servers, collecting results. It is possible to access the query service through direct interfaces with the distributed computing interfaces (such as RMI and CORBA invocations), or through an HTTP interface.

As described earlier, profiles are metadata descriptions of resources; that is, they "profile" a resource by describing its inception and composition using the common data elements of the data architecture. Profile servers enable discovery of resources by providing the ability to search resource collections. Profile servers answer the question, "Where can I go to find out about X?"

A profile server's primary responsibility is to provide a way to run a query against the server's set of profiles. Although users may access a profile server directly via its remote interface, it is far more common for queries to enter the system through the query server, which directs them transparently to and along graphs of appropriate profile servers (when a profile describes another profile server).

Upon receiving a query, the profile server's backend interprets the query passed in a way appropriate to the implementation. For example, a backend that stores information in a relational database may convert parts of the query into a database SQL query. For each matching profile, the backend constructs a list of matching profiles and returns them.

### Product Servers

Product servers exist to provide a way to retrieve specific data products. Product servers accept the same query structure as profile servers, but instead of returning a list of matching profiles, they return matching products. Data products in this sense can be individual data granules, datasets, or collections of datasets, depending on the backend implementation in the product server and the way it handles queries and results.

When constructing a query, the user may indicate preferred MIME types. For example, a user wanting PNG images may list image/png as the only acceptable MIME type. A user preferring PNG images but willing to have JPEG images would list image/png, image/jpeg in that order. A user preferring PNG images but willing to accept any image type would list image/png, image/*. If the user doesn't specify a MIME type when creating the query, the software generates a default list of acceptable MIME types, namely */*, meaning that any type is acceptable. Sophisticated product servers can convert between data types. One mechanism for handling interoperability of legacy data systems is to deploy product servers that convert

between file formats that are native to the local data system and the common data formats supported by the larger data grid system.

**Query Servers**

Query servers manage queries across distributed resources and are the point of entry into an OODT framework installation. Query servers contain the algorithms necessary to traverse the logical P2P model, executing queries at appropriate servers and gathering results. Query servers also simplify the interaction with the user, who is freed from the knowledge of accessing the remote interfaces of profile servers and product servers. Users instead call upon a query server for all profile and product interaction.
The OODT implementation supports several different interfaces to the query service to ensure that it supports both cross platform and cross language interoperability. This includes not only interfaces for programming languages such as Java, but interfaces using the web standard HTTP.

The OODT framework components in a network architecture are illustrated in Figure 6 below.

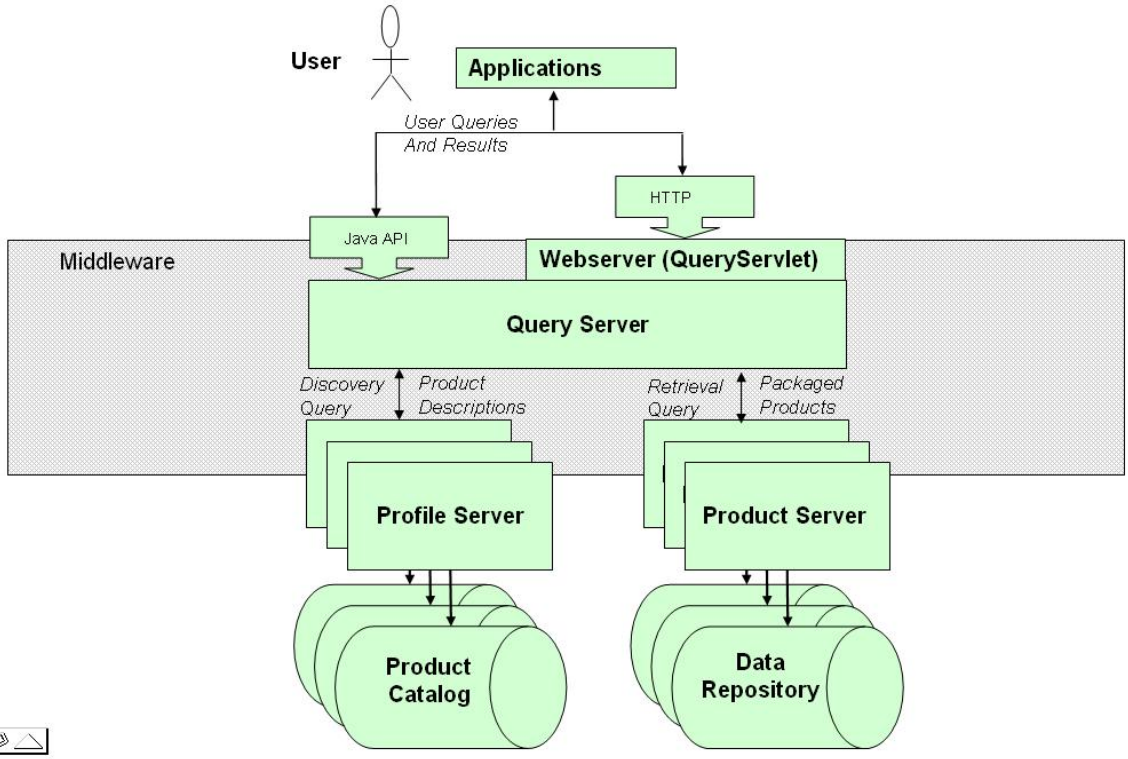


**Figure 6 - Framework Component Architecture**

**CASE STUDY – NASA'S PLANETARY DATA SYSTEM**

The Planetary Data System (PDS) is the official science data archive for NASAs planetary science community. As such, it contains several terabytes of data collected from over thirty years of solar system exploration. At its inception in the late 1980's, the PDS developed an ontology that guides the capture of the information necessary to describe the data and ensure that the data remain scientifically useful for future scientists. Collected and validated using the ontology, this information and the science data was submitted to peer review and then distributed to funded scientists in the planetary science community on CD and DVD media.

The combination of several factors including the advent of the Internet, requests to support correlative search across instruments, and huge increases in the volume of data returned from missions necessitated the development of an online system that supports search and retrieval of data products from across the distributed heterogeneous data repositories of the PDS. In October of 2002, the PDS released the first version of its online distribution system in support the 2001 Mars Odyssey mission. Using the OODT framework and product servers at each distributed data repository, data products were available to planetary scientists as soon as they were released from the mission. The distribution system has since been augmented to include the majority of all data products in the PDS archive. The current system implements a two level search. First the set of collections of products (data sets) is searched, and then products within the collection are searched. Development is now focused on global product level search capability using a hierarchy of OODT resource profiles and profile servers.

The key to the success of the PDS search capability is the fact that each data product in the archive has a header that provides identification, descriptive, and contextual information. This information has been validated against the PDS data model and relates data products to the other entities of the model such as targets, instruments, spacecraft, and missions. In other words, the data product headers contain the information necessary to support data product search.

The global product search capability is designed to have a single point-of-entry interface and a search scope that encompasses the entire geographically distributed PDS archive and its thousands of product types and millions of data products. To implement this search, a hierarchy of resource profiles has been designed. At the leaf nodes of the hierarchy, each data product is represented by a single resource profile created from data product labels. For example, Figure 5 above illustrated a portion of a profile for a Galileo, Solid State Imaging System (SSI) image product where the domain attributes, target_name, filter_name, and exposure_duration have been encoded into the profile elements section. The resource attributes Identifier and Title have been set to the value of the concatenated domain attributes data_set_id and image_id, and the profile Identifier has been set to a unique object id, generated by the system.

All data products collected within a single data set are represented by a single product type and therefore have the same set of data elements in their product headers. This allows for the creation of data set profiles by aggregating the leaf node data product profiles. For the PDS this results in several thousand data set or level two profiles. (For this discussion we assume that data sets have only one product type. However this is not actually the case with many data sets having multiple product types.) The permissible value allowed the data elements are summarized as in a data dictionary. For example, a range of values bounded by minimum and maximum values is used for numeric valued data elements such as exposure_duration. Data elements with discrete values such as target_name are typically enumerated. Figure 7 below illustrates a portion of the Galileo image data set profile that contains the imaging data product mentioned above.

```
<profile>
 <profAttributes>
  <profId>1.3.6.1.4.1.1306.2.1148</profId>
  <profType>profile</profType>
  <profStatusId>active</profStatusId>
 </profAttributes>
 <resAttributes>
  <Identifier>GO-J/JSA-SSI-2-REDR-V1.0</Identifier>
  <Title>GO-J/JSA-SSI-2-REDR-V1.0</Title>
  <Format>image/pds</Format>
  <resContext>NASA.PDS</resContext>
  <resClass>system.profileServer.dataSet</resClass>
  <resLocation>http://starbrite.jpl.nasa.gov/servlet/
 </resAttributes>
 <profElement>
  <elemName>TARGET_NAME</elemName>
  <elemType>CHARACTER</elemType>
  <elemValue>GANYMEDE</elemValue>
  <elemValue>IO</elemValue>
  <elemValue>JUPITER</elemValue>
  <elemValue>...</elemValue>
 </profElement>
 <profElement>
  <elemName>INSTRUMENT_ID</elemName>
  <elemType>CHARACTER</elemType>
  <elemValue>SSI</elemValue>
 </profElement>
 <profElement>
  <elemName>FILTER_NAME</elemName>
  <elemType>CHARACTER</elemType>
  <elemValue>CLEAR</elemValue>
  <elemValue>GREEN</elemValue>
  <elemValue>RED</elemValue>
  <elemValue>...</elemValue>
 </profElement>
 <profElement>
  <elemName>EXPOSURE_DURATION</elemName>
  <elemType>REAL</elemType>
  <elemMinValue>0.0</elemMinValue>
  <elemMaxValue>51195.8</elemMaxValue>
 </profElement>
</profile>
```

**Figure 7 - Data Set Profile Example**

Data set profiles are in turn aggregated to create higher level profiles. This process continues until a single all-inclusive root profile is generated. This root profile is the starting point for searches initiated from the single-point of entry user interface. The root profile in Figure 8 below illustrates values for target_name and instrument_id taken from data sets in addition to the Galileo image data set.

```
<profile>
  <profAttributes>
    <profId>1.3.6.1.4.1.1306.2.9999</profId>
    <profType>profile</profType>
    <profStatusId>active</profStatusId>
  </profAttributes>
  <resAttributes>
    <Identifier>ALL-ALL-ALL-N-ALL-V1.0</Identifier>
    <Title>ALL-ALL-ALL-N-ALL-V1.0</Title>
    <Format>text/xml</Format>
    <resContext>NASA.PDS</resContext>
    <resClass>system.profileServer.all</resClass>
    <resLocation>http://starbrite.jpl.nasa.gov/servlet/profileprofile...
  </resAttributes>
  <profElement>
    <elemName>TARGET_NAME</elemName>
    <elemType>CHARACTER</elemType>
    <elemValue>GANYMEDE</elemValue>
    <elemValue>IO</elemValue>
    <elemValue>JUPITER</elemValue>
    <elemValue>VENUS</elemValue>
    <elemValue>...</elemValue>
  </profElement>
  <profElement>
    <elemName>INSTRUMENT_ID</elemName>
    <elemType>CHARACTER</elemType>
    <elemValue>SSI</elemValue>
    <elemValue>NIMS</elemValue>
    <elemValue>SAR</elemValue>
    <elemValue>...</elemValue>
  </profElement>
</profile>
```

**Figure 8 - Root Profile Example**

Each resource profile includes a resource location that provides a link to the resource. A resource location in a data product profile will typically provide a URI for a data product, typically a link to a data product server that will retrieve the product when invoked. A resource location for a data set profile provides a link to the profile server that serves the associated data product profiles. Similarly all higher level aggregated profiles provide links to profile servers that serve their associated lower level profiles. For the initial PDS prototype, the root profile and all data set profiles are managed by a single profile server of profile servers. Product level profile servers are managed by a two or three product level profile servers. These were configured based on performance and maintenance requirements.

This hierarchy of profiles provides a map of all profile servers and product servers in the framework and characterizes each server in terms of the attributes that it can handle. For example, a query presented through the single-point-of-entry interface is first handled by the root profile server. The root profile server

finds all resource profiles that match the query constraints and returns them to the interface client. The client is then able to determine where to broadcast the original query to continue the search. Ultimately data product profiles are returned allowing the client to decide what products should be retrieved based on a perusal of the data product information in the profiles.

This hierarchy of profiles also provides a means to create dynamic user interfaces. For example, the root profile provides the information necessary to create a single-point-of-entry interface since all global data elements and their value ranges or enumerated values are provided. This root interface is presented to the user and the user selects one or more attributes as query constraints. The user then has the option of either having the system return all matching profiles or the user can request that the system refine the interface based on the matching profiles. In the latter case, the user can iteratively refine the interface until sufficient attributes have been identified to constrain the query. Ultimately the user is provided with data product profiles.

For example, the root profile for the PDS archive contains Galileo as a value of Mission_Name. The selection of “Galileo” in the root interface would match on all Galileo data set profiles in the root profile server. If the user had asked for a refinement of the user interface, the resulting interface would include the Galileo image attribute Filter_Name and its permissible values, including RED. A subsequent selection of RED for Filter_Name and a request to return all profiles would return all Galileo imaging data products that were taken through the RED filter.

## CONCLUSION

The Object Oriented Data Technology (OODT) task has developed a standard resource description scheme that can be used across domains to describe any resource. It uses a small set of generally accepted, broad scope descriptors while also providing a mechanism for the inclusion of domain specific descriptors. The use of this resource description scheme in an OODT technology framework of profile and product servers provides a powerful, flexible, and scaleable resource discovery capability across distributed resource repositories. Intelligent resource discovery is provided by the addition of domain ontologies and semantically homogeneous resource descriptions. The Planetary Data System (PDS) has been able to implement intelligent resource discovery across its distributed data repositories and its thousands of data product types and millions of individual data products.

## REFERENCES

[1] Crichton D, Hughes, J.S., Kelly, S. A Science Data System Architecture for Information Retrieval. Clustering and Information Retrieval. Kluwer Academic Publishers. June 2003. - Book Chapter on OODT

[2] Crichton D, Hughes, J.S., Kelly, S, Rameriz, P. A Component Framework Supporting Peer Services for Space Data Management. 2002 IEEE Aerospace Conference. Big Sky, Montana. March 2002.

[3] Crichton D, Downing G, Hughes J. S, Kincaid H, Srivistava S. An Interoperable Data Architecture for Data Exchange in a Biomedical Research Network. 14th IEEE Symposium on Computer-Based Medical Systems. July 2001.

[4] Crichton, D., Hughes J. S, Kelly S, Hyon J. Science Search and Retrieval using XML. Second National Conference on Scientific and Technical Data, Washington D.C., National Academy of Sciences. March 2000.

[5] DCMI, "Dublin Core Metadata Element Set, Version 1.1: Reference Description," Dublin Core Metadata Initiative, 1999.

[6] ISO/IEC, "Framework for the Specification and Standardization of Data Elements 11179-1," Specification and Standardization of Data Elements 11179, International Organization for Standardization, Geneva, 1999.