Chapter 3

METADATA CONCEPTS TO SUPPORT A NET-CENTRIC DATA ENVIRONMENT

Kenneth J. Laskey[*]
The MITRE Corporation, 7515 Colshire Drive, McLean VA 22102

Abstract: The term *metadata* is often defined as "data about data" but that circular reference does little to describe what constitutes metadata and how it is used. Here, we will focus on metadata as conceived to support the concepts of a service-oriented architecture and, in particular, as it relates to the DoD Net-Centric Data Strategy and the NCES core services; more specifically, what types and structure of metadata are implied by current use cases, what functions are implied to support creating maintaining, and using such metadata, and what is implied about a metadata infrastructure that would support such metadata and its related functions.

Key words: metadata, service-oriented architecture, net-centric, GIG ES, GES, NCES

1. INTRODUCTION

The term *metadata* has expanded beyond the point of conveying a definitive meaning when the word is used. The data model in a database is traditionally looked at as metadata because it describes the structure of the database. Similarly, information included before a table in a data file can identify the variables represented by the values in the rows and columns, and this is often described as metadata.

When used in the context of a service-oriented architecture (SOA), metadata typically serves a much wider purpose. For the myriad of capabilities with which metadata has been connected in an SOA context, it

[*] The author's affiliation with The MITRE Corporation is provided for identification purposes only, and is not intended to convey or imply MITRE's concurrence with, or support for, the positions, opinions or viewpoints expressed by the author.

would be more accurately described as that subset of the data related to an entity that provides some critical descriptive information which is especially useful in some context for identifying, using, or otherwise interacting with the entity. Context is especially important. The entity may be a physical object or a computational object, such as a data set or an application, or anything else to which there is a need to apply a description. Any subset of data (*i.e.*, any information associated with or comprising the entity) may be identified as metadata if it satisfies the needs for some context, and there may be multiple metadata sets corresponding to any number of contexts.

Admittedly, this is quite an expansion over the traditional use of the term. As an example of the expanded use for different contexts, consider the ways in which metadata for a book may be defined and used. For a librarian, the Library of Congress classification number is likely an important metadata element. Conversely, for a bookseller, the classification number is not likely to be as important but the current sales price would be (while this price may not be of interest to the librarian). The text in the book is unlikely to be identified as metadata, but specific quotes from the book may be metadata for someone advertising the book.

2. CONSIDERATIONS FOR NET-CENTRICITY

The Global Information Grid (GIG) Core Enterprise Services (CES) Strategy [1] calls for “robust [GIG] enterprise services (GES) [to] provide visibility and access to data, enabling the end user to execute an intelligent pull of mission-tailored information from anywhere within the network environment.” Moreover, “the CESs provide the basic ability to search the DoD enterprise for desired information and services, and then establish a connection to the desired service/data.”

This vision describes an environment where the interaction between the providers and consumers of resources must be flexible and readily configurable across entities (consumers, providers, and resources) that had no previous knowledge that the others existed. This implies a number of capabilities that go beyond the traditional data and processing paradigm.

- Consumers must be able to search for resources without knowing the details, such as specific APIs, of the resource beforehand. This implies that the description of the resource must be expressed in a universally accessible format and, though it will be associated with the resource, the description will be external to the resource so it can be accessed without reading or otherwise invoking the resource itself.
- The external description must contain sufficient detail so the consumer can decide if the resource will satisfy the current need.

- If the resource is appropriate, the consumer must be able to access the resource content or invoke the resource processing without knowing the APIs or other details of the resource.
- If the consumer attempts to access the resource, sufficient information must be available about the consumer so that the provider or an agent acting for the provider can determine if the access is authorized.
- The producer and consumer must share a common format for the description and must also agree on how to interpret the description content. This may be accomplished by indicating a common vocabulary or distinct vocabularies for which services exist to mediate a translation.

The DoD Net-Centric Data Strategy [2] lays out a path for accomplishing this through the use of metadata. As a notional example of metadata enabling net-centric capabilities, consider a user looking for meteorological data in Afghanistan. Metadata associated with a data resource that could support this includes

1. general document metadata with the name of the data resource and the geographic locations from where it can be accessed; metadata specific to the function of the data resource, such as the date, time, and location where the data was collected,
2. access control restrictions which must be satisfied (or possibly licensing terms if it is a commercial source) and a pointer to the service interface (*e.g.* WSDL [3]) to retrieve the data,
3. a pointer to pedigree information describing the quality of the data as evaluated based on how the data was collected and processed and the accuracy of the measurements.

The request for the meteorological data may generate a log file detailing the services invoked and resources used to satisfy the request, and the log file could be archived using a network storage service. Associated with the stored log would be metadata containing a log ID, the date of the request, and the identity of the requester. Note, in this example, the log file itself is not considered metadata but information describing the log file is. A pointer to the log metadata would be returned with the requested data so the requester would both know how the request was fulfilled and be able to point to the log as a repeatable means to satisfy a similar request in the future.

In this example, the distinction between what is only data (the log file) and what is data used as metadata (*e.g.*, when the log file was created) is unimportant (and is likely to change in other contexts). What is important is that subsets of the information space surrounding the meteorological data were available as needed for various services in order to locate, access, and evaluate the suitability of the resource before the resource was ever used. In fact, using the resource was possible because metadata could directly supply or point to information that the service needed to complete its function. This is the role played by metadata in a service-oriented architecture and the context for the present discussion.

3. DEFINITION OF METADATA

To support and enable the capabilities required of a service-oriented architecture and the GIG CES vision of net-centricity, we offer the following definition:

Metadata is that set of descriptive properties which serves one or more of the following functions

1. uniquely characterizes an entity and for which values associated with the descriptive properties allow a user (human or machine) to discriminate between one entity and another,
2. describes how the entity and its contents can be accessed (both procedurally and the terms of access) in either a read or write mode or executed if the entity comprises processing instructions,
3. contains pointers to information not explicitly part of a given metadata set but which is required as processing or control inputs by other applications or services.

Metadata often includes what the entity is, where it is located, and how to make use of it. It may describe entity properties such as format, structure/organization, context, business rules, or any other chosen elements of its integral or associated data or capabilities. It may include the calling argument to methods, invocation of services, or similar executable commands that act on the content of an instance of the entity, including accessing it from its native storage format.

As noted in both the book example in the Introduction and the weather example in the previous section, what constitutes the appropriate metadata set depends on the context of the user and the current needs to be satisfied. Thus, it is less important to have defined the perfect metadata set than to ensure that the combined metadata available can provide or support access to the critical information at the critical time.

4. NET-CENTRIC EFFORTS TO IDENTIFY METADATA CONTENT

As noted in the previous section, what constitutes metadata can be quite variable and the only real test to see if one has the “right” set of metadata is to ask whether that set satisfies the task at hand. To provide more structure to the description of metadata, numerous efforts have attempted to organize metadata into classes, sometimes forming a metadata taxonomy. This section will look at several such efforts that are particularly relevant to GIG Enterprise Services. Later sections will discuss specific results from one of the efforts and will attempt to provide some clarity as to how a consolidated view of these efforts support the operational needs of GES and the net-centric vision.

4.1 DoD Net-Centric Data Strategy perspective

The DoD Net-Centric Data Strategy describes the DoD data vision and specifically, the Net-Centric Data Goals. These goals are listed in Table 3-1.

Table 3-1. DoD Net-Centric Data Goals

Goals that increase data that is available to communities and the Enterprise

- **visibilit**y: descriptive metadata about the data asset has been provided to a catalog that is visible to the Enterprise
- **accessibilit**y: data is stored such that users and applications in the Enterprise can access it
- **institutionalizing**: data approaches are incorporated into DoD processes and practices

Goals that ensure data is usable by both anticipated and unanticipated users and applications

- **understandability**: through strong emphasis on Community of Interest (COI)-level consensus as made visible through various DDMS (DoD Discovery Metadata Standard) metadata
- **trustworthiness**: through mechanisms such as providing defined pedigree and security information and then having COI mark what is "authoritative"
- **interoperability**: resulting from compliance with metadata standards (i.e., DDMS) and data exposure standards (e.g., GES discovery interface standards)
- **responsiveness to users**: perspective of users through involvement in COIs and evaluating data sources, catalogs, or services, and content metadata usability

In discussing the goals, the Data Strategy alludes to but does not further define the following classes of metadata:

- Structural: how data assets are physically composed (*e.g.* type of file: GIF, JPEG, ...) and relationships between specific parts of the data asset
- Discovery: key attributes and concepts of a data asset used for discovery; this includes the means to enable a user to discriminate between individual elements of a data asset or across data assets
- Service: defines the capabilities of the service, the necessary inputs to use the service, and a description of what the service provides
- Content: provides topics, keywords, context, and other content-related information to give users and applications (including search engines) insight into the meaning and context of the data
- Security: information (*e.g.*, security and privacy markings consistent with applicable standards) through which systems will be able to control

access to assets based on classification metadata and enable typically inaccessible assets to be available to users and applications that have appropriate access needs
- Pedigree: allow for identification of the author, publisher, and sources contributing to the data, allowing users and applications to assess the derivation of the data
- other: vocabularies, taxonomic structures used for organizing data assets, interface specifications, mapping tables, ...

4.2 DDMS perspective

The DoD Discovery Metadata Standard (DDMS) [4] was developed as a standard to support the net-centric goal of visibility across the Department of Defense. Its intent is to establish a common specification for the description of data assets[1] and thus enable the capability to locate all data assets across the Enterprise[2], regardless of format, type, location, or classification. To facilitate data asset discovery, DDMS is developed as a common set of descriptive metadata elements, including a core set that is identified as mandatory to enable a basic level of visibility.

The DDMS logical model contains a core layer as defined in the specification and an extensible layer intended to support domain-specific or Community of Interest discovery metadata requirements. The core layer is composed of four category sets:

- Security Category Set: describes security classification and related fields needed to support access control, but not intended to support comprehensive resource security marking; the Net-Centric Data Strategy directly references this category set in describing its security metadata
- Resource Category Set: describes aspects of a data asset that support maintenance, administration, and pedigree of the data asset; the Net-Centric Data Strategy directly references this category set in describing its pedigree metadata
- Summary Content Category Set: describes concepts and additional contextual aspects of the data asset and is intended to aid in precision discovery; the Net-Centric Data Strategy directly references this category set in describing its content metadata
- Format Category Set: describes physical attributes of the data asset, including elements such as file size, bit-rate or frame rate, and MIME

[1] The DOD Net-Centric Data Strategy defines a *data asset* as any entity that is composed of data. The DDMS considers the term to include services that provide access to data.

[2] In the DDMS context, the *Enterprise* refers to the Department of Defense, its organizations and related agencies.

type; the Net-Centric Data Strategy directly references this category set in describing its format/structural metadata

A further breakdown defines *primary categories* of the core layer, each with its own set of constituent elements. The primary categories, shown in Table 3-2, are considered mandatory if they contain at least one mandatory element and are otherwise optional.

Table 3-2. DDMS Primary Category Sets

Core Layer Category Set	Primary Category	Obligation
The Security elements enable the description of security classification and related fields	Security	Mandatory
Resource elements enable the description of maintenance and administration information	Title	Mandatory
	Identifier	Mandatory
	Creator	Mandatory
	Publisher	Optional
	Contributor	Optional
	Date	Optional
	Rights	Optional
	Language	Optional
	Type	Optional
	Source	Optional
The Summary Content elements enable the description of concepts and topics	Subject	Mandatory
	Geospatial Coverage	Mandatory unless not Applicable
	Temporal Coverage	Mandatory unless not Applicable
	Virtual Coverage	Optional
	Description	Optional
The Format elements enable the description of physical attributes of the asset	Format	Optional

4.3 NCES Analysis of Alternatives (AoA) Use Cases

Net-Centric Enterprise Services (NCES) is a program created to provide the services and capabilities that are key to enabling the ubiquitous access to reliable decision-quality information that is envisioned by GES. The initial scope and requirements for GES were defined through the NCES Analysis of Alternatives (AoA) [5]. In support of the AoA activity, an initial set of core enterprise services were identified and further defined by inter-Service, inter-Agency teams, and then the AoA effort defined a set of use cases corresponding to these core services, with the use cases representing typical scenarios that an early NCES deployment might support.

In addition to the AoA effort to define services, it was widely recognized that there had been no detailed presentation of what metadata must be created and managed, how it would be managed, and by whom. Thus, a subsequent effort was chartered to fill that gap by providing a concept of operations (CONOPS) for metadata. In order to provide continuity with the work of the AoA, the metadata CONOPS effort [6] analyzed a subset of the AoA use cases (Table 3-3) to determine

- what types of metadata were implied by the use cases;
- what functions were implied if such metadata was to be created, maintained, and used;
- what was implied about a metadata infrastructure that would be needed to support this metadata and the related functions.

Table 3-3. AoA use cases analyzed for Metadata CONOPS

NCES Core Services	Corresponding Use Cases
Discovery	Generalized combination of discovery of persons, content, services, and metadata use cases
Enterprise Services Management	Integrated Service Management
Mediation	Dissemination by channel General data access
Messaging	E-mail Notification Mailing/distribution lists Newsgroups/message boards Instant messaging

The first stage of the analysis was to consider each step from each use case in Table 3-3 and to identify the likely metadata needed to support the step. The second stage was to look across use cases and collect the individually identified metadata into common metadata sets and then to look for further commonalities in structure and use. The full analysis considered the following points:

- a common defined purpose for the set
- notional elements that would be included in the metadata set
- other defined metadata sets that would serve as components of a composite set (discussed below)
- life cycle aspects and other points to consider about the metadata set

The analysis included one or more interviews with the relevant task lead for each core service in order to ensure an in-depth understanding of the use case details and how metadata was a part of the scenario. For some of the services, use cases were combined into a single generalized use case because the required metadata and metadata processes were the same across most or all of the use cases; this was most notably done for Discovery and, to some extent, Messaging.

Note, the intent of the metadata analysis was to be wide-ranging but not necessarily to be complete or definitive. For example, aspects of a logging function seemed to naturally arise during the analysis even though this functionality was not directly included in the use cases. Thus, logging was considered with respect to potential needs and uses of metadata, but defining full details of the logging function were out of scope for a metadata effort. Conversely, while not all of the AoA use cases were included in the analysis, the investigation followed a systematic process and covered a large enough range of metadata activities to provide insight into the demands on metadata and the systems that would support it.

It should also be noted that services described in the context of the metadata analysis, especially those beyond the core services as defined in the AoA, and the registry capabilities indicated as needed to support metadata are notional and there is no NCES commitment to build these services or to build these as described.

5. FINDINGS FROM METADATA ANALYSIS FOR NCES

The AoA use case analysis considered a select number of use cases but the results of analyzing each use case step produced a significant amount of data, making presentation of the entire metadata analysis beyond the scope of this discussion. However, several instances of the analysis will be presented to demonstrate the process and the results. This will lead to observations on how to categorize metadata, conclusions on the purposes specific metadata types will likely need to serve, and suggestions for infrastructure capabilities to support these metadata needs.

5.1 First stage of analysis: examining the individual AoA use case steps

The analysis of each use case step typically yielded one or more types of metadata. For example, one step of the General Data Access use case stated

> Data Access Service (DAS) invokes Find Service to search repository of Data Access Methods (DAM) for candidate DAMs that can support current data request.

In the full use case, the Data Access Service is described as a single Mediation service that receives data access requests and can invoke any Data Access Method. In turn, the DAM is a service that is specific to a given data resource and possibly the specific data requested. By design, every DAM responds to the DAS in a standard, prescribed manner, and the DAS coordinates delivery of results back to the requester.

The question is then what metadata is needed for this step to successfully occur. DAS will search a repository for candidate DAMs, thus indicating the need for DAM metadata (later combined into service metadata). From the remaining context, it is known that multiple DAMs may exist and so the DAM metadata must include information (some expressed through constraint metadata) to support choosing among the candidates. Once a choice is made, DAS will require the DAM WSDL (Web Service Description Language interface definition) to invoke DAM processing. Continuing the thought process also leads to the need for source metadata, with the combined results for this use case step being shown in Table 3-4. Note, the metadata, as notionally defined, supports the discovery and choice of DAM and corresponding data source before the source is ever accessed.

Table 3-4. Metadata types associated with Mediation/General Data Access use case

DAM metadata and notional elements

- DAM WSDL
- who responsible for WSDL (person/organization metadata)
- when it was last changed (date metadata)
- source from which DAM can retrieve data (pointer to source metadata)
- data that DAM can retrieve (including pointer to vocabulary description from which these data names are taken)
- assumptions/limitations that support deciding among DAMs (likely specified through constraint metadata)

source metadata and notional elements

- what source is (name and/or ID)
- who maintains it (person/organization metadata)
- pedigree metadata (describing previously evaluated data quality)
- index of DAM WSDLs (assuming more than one access is likely from a given source)

Systematic analysis of the use cases indicated in Table 3-3 resulted in many other types of metadata but also in a frequent duplication of metadata sets or the appearance of ones similar to previously identified sets. For example, one step from the ESM (Enterprise Service Management) use case states

> ISM (Integrated Service Management) correlates status data across CESs and provides resultant relevant operational status, performance, configuration, and security information to potential users.

and this implies the metadata types shown in Table 3-5.

Note, the analyses of the two examples result in the common appearance of date metadata and person/organization metadata. Such commonality is not unexpected because the underlying assumption for a metadata schema registry is that interoperability will be facilitated by reuse of common schema elements. However, the analysis highlights the granularity at which reuse is most likely to occur and the extent to which commonalities can be leveraged to further the goals and ultimate value of metadata creation, maintenance, and use.

Table 3-5. Metadata types associated with Enterprise Services Management use case

report metadata with notional elements

- who/what generated report (person/organization metadata)
- when generated (date metadata)
- link to directive requiring report
- type of report (linked to vocabulary of report types)
- subject of report (and link to vocabulary from which subject term derives, *e.g.* for ISM, if subset of management domain, link to definition of domain subsets)
- status of report (linked to vocabulary of report status)
- how/when report disseminated (possibly service link or service metadata)
- history (what did this supersede, what superseded this, when (actual or scheduled))

5.2 Second stage of analysis: forming conclusions across the use cases -- the modularization of metadata

The complete analysis of all the Table 3-3 use cases uncovered many commonalities and a factoring across the use cases indicated metadata sets may be grouped into three categories based on their structure, their patterns of reuse, and the granularity of the concepts represented. The introduction of these categories is a fundamental difference in the way we look at metadata because instead of defining distinct, complex metadata structures for specific purposes, we introduce a modular approach of defining complex metadata in terms of more elementary metadata building blocks. This is consistent with the current paradigm for building software, but metadata has often been more compartmentalized, and this has hindered reuse in the same way as it hindered reuse in early software development. The DoD Metadata Registry similarly seeks to facilitate reuse, but metadata developers must search existing schemas and then extract useful parts. A more effective approach should be to define generic parts and support the developer in assembling the pieces.

As described in the following, the names chosen for the categories are *concepts*, *functions*, and *resources*. These names are less important than their use to convey the needs of metadata providers and consumers and the implications for a metadata system that will satisfy these needs. The immediate sections describe the characteristics of each category and the perspective implied by a modular approach. While references to the constituent metadata elements are introduced as needed to clarify the discussion of metadata categories, the detailed discussion of individual metadata sets is deferred until Section 5.3.

5.2.1 Concept metadata

Concept metadata is generally a set of information elements that convey a single elementary concept which is reused frequently as part of other metadata sets. The concept may require more than one element but it is likely to be a schema fragment (but still well-formed in the XML sense) rather than a complete schema. The following are a limited number of examples of concept metadata:

- name – the textual label by which an entity is identified, whether it be a physical object (such as a truck or a computer), a computational object (such as a schema, a data resource, or a service), or any other entity.
- person_name – possibly a special case of the general name; likely a collection of fragments representing formats for names of persons as these names are structured in different cultures, but with catalogued mappings between what are seen as common parts of the name variations
- datetime – formats representing date and time, likely built from the ISO date and time standard [7]
- pointer/reference/link – a standard means to point to other network accessible objects, most likely using the URI syntax for the target object
- keywords - textual terms defined within a referenced vocabulary (possibly defined by an XML namespace) that provides descriptive associations
- identifiers - unique means to identify an entity (possibly defined by an XML namespace), including a reference to documentation defining the identifier format and use.

Note that both the keywords and the identifiers include a reference to a defining vocabulary. The need to make such references a common part of the metadata space will be reiterated and expanded below.

The benefit of concept metadata is that it is focused and concise. If variations are required (see for example the HR-XML [8] work on a standard format for person names), it is far simpler to create a mapping (or indicate non-mapped elements) between variations of, say, a name type than it is to map schemas that are several (or dozens of) pages long. Reusing concise concept metadata and their associated mappings provide immediate interoperability over those elements even if there is not total understanding of a complex schema that incorporates the concept metadata.

5.2.2 Functions metadata

A review of the AoA use cases shows a strong dependence on processes and the recurring need to identify mechanisms and constraints that enable use of an entity in a manner consistent with needs and requirements of both users and resources. Function metadata combines concept metadata sets, simpler function metadata sets, and additional unique metadata to capture descriptive and access information needed to support such reusable functions. For example, access/invocation metadata collects information to support data access or service invocation; pedigree metadata describes pedigrees that have been established for various resources. The functions themselves may be fairly elementary, such as the person/organization metadata, or a more complex combination of concept metadata and more elementary function metadata, such as the access/invocation and pedigree examples. The following are a number of frequently occurring function metadata sets and a brief description of the function each provides:

- Person/Organization – identity and contact information for a person or organization (using concept metadata such as name, address, email)
- Title/Position - identity and contact information based on specific role (*e.g.* Director of IT) rather than current person in the role; may redirect to instance of Person/Organization metadata
- Creation/Modification - critical information about latest change to an identified resource; information would include contact information (using Person/Organization or Title/Position metadata) of who made change, datetime (concept) metadata of when change was made
- Access/Invocation - means to access a service or other resource; includes the WSDL interface, constraints and policies for access, and assumptions/constraints associated with the processing that will be performed or data that will be provided; references identified using pointer/reference/link (concept) metadata
- Constraint – means to identify rules that define constraints, limitations, and assumptions related to any entity; includes party responsible for definition and maintenance, means to access, and recommended associated processing of; references identified using pointer/reference/link (concept) metadata
- Pedigree - documented level of “goodness” as qualified by vocabulary through which pedigree level is defined, associated constraint set with details of pedigree criteria, means of evaluating entity against criteria; references identified using pointer/reference/link (concept) metadata
- Log - means to identify (including responsible party) and describe access to logs for tracking use and modification of a resource; assume logs maintained external from but linked to the entity being tracked

Section 5.3 contains a detailed description and identifies notional elements for many function metadata sets. The notional element list for each set can be considered a baseline but the expressivity of the baseline can be easily expanded by adding other concept or function metadata. By considering existing metadata sets as building blocks, a scalable mechanism is defined to incorporate previously defined semantics. With current schemas, some of the constituent elements are optional when a metadata producer creates metadata instances; for modular metadata, the inclusion of additional concept or function metadata is the optional extensibility mechanism. By reusing building blocks, a metadata producer can exercise the established context to fully describe the entity at hand.

The modular construction is important for immediate interoperability and can provide enhanced capability as the quality, completeness, and sophistication of the metadata increases. For example, consider having the metadata sets expressed as ontologies, where these ontologies would capture not only the class-subclass structure but also the axioms relating the classes. Then, if we capture mappings between variations of a metadata type (such as mentioned above for the name concept) as additional axioms, these axioms can be combined and processed by available inference engines to establish broader understanding. Adding a new variation would not require mapping to every existing one because existing relationships would be leveraged to establish the meaning of the new variation within the existing context.

5.2.3 Resource metadata

While concept metadata describes elementary concepts and function metadata describes the information related to common activities, resource metadata combines these to describe the assets that can be utilized to respond to user needs. Unlike concepts and functions, the types of resources tend to be more coarsely defined and more limited in number. An SOA environment has data and processing resources, and to these a GES discussion adds others, specifically entities requiring content metadata and structural metadata. The description and relation between these resources are the focus of this section.

A *data resource* is a source of content. It accepts a request and returns a value or set of values in response. The return can be an entity (such as a particular schema), an attribute of an entity (such as when the schema was last modified), or any numerical or textual value or set of values. The content can be static objects stored in some repository or dynamically generated through the use of a processing resource. Data about a missile that is stored in a database is content. The weather forecast for tomorrow in Iraq is content generated from a weather simulation. In a net-centric environment, the requester does not know the format from which the response is retrieved or how it is generated.

A *processing resource* is one that accepts a task and return a status indicating the extent to which the task was completed and information on how the state of entities changed as a result of the processing. One or more processing resources may be invoked as part of a process of submitting a query and being returned a response. From the standpoint of a user (either human or machine), it is unimportant what combination of data and processing resources are invoked as long as the request is satisfied.

Content metadata as described for DDMS comprises metadata to "aid in precision discovery" and includes such specialized metadata as that describing geospatial coverage. While such a description is consistent with the findings of the AoA analysis, a broader description may be more useful. Table 3-6 shows a comparison of the notional metadata elements for content, data, and service (*i.e.*, a processing resource) metadata. (The rows are solely for convenience in comparing like elements.) The interesting point is that the notional elements for content and data resource metadata were collected at separate times (during the overall analysis) but give very similar results. During the analysis, a data resource was considered the asset from which information is retrieved while content was thought of as the retrieved information. This leads to minor differences in the metadata elements, such as content metadata includes the creation/modification function metadata while the data resource metadata assumes there may be an update policy to be referenced. However, one element of the "update cycle" is "last update", a direct parallel to and possible use of creation/modification function metadata. Furthermore, while not explicitly noted, version and status metadata for services implicitly include creation/modification information on when and by whom the version or status was assigned. The conclusion is that while content metadata may be a useful grouping, it is not important whether we classify the metadata associated with an entity as data resource metadata or content metadata as long as the component metadata makes use of and references the same common building blocks. As with update cycle *vs.* creation/modification, it is not the *a priori* classification that is important but rather providing the metadata that is most appropriate in facilitating eventual use of the entity.

Table 3-6. Comparison of notional elements for content, data resource, and service metadata

Content	Data Resource	Service
- name of content - description (text) - formal descriptors/keywords indicating function - pointer/link to vocabulary defining descriptors/keywords	- name - description (text) - formal descriptors/keywords indicating function - pointer/link to vocabulary defining descriptors/keywords	- name - description (text) - formal descriptors/keywords indicating function - pointer/link to vocabulary defining descriptors/keywords
- pointer to content (where content exists/is stored)	- unique identifier (could be URI)	- unique identifier (could be URI)
- creation/modification metadata	- update cycle -- description of update policy -- refresh cycle (may be "continuous") -- last update	- version (format for defining insignificant, minor, major changes) - status (*e.g.* current version, beta test, superseded; status definitions to be referenced)
- type of content (log, data, processing, ...) - format (MIME type)		
- responsible party -- type (Person/Org, Title/Position, ...) -- Person/Organization metadata <or> Reference by title/position metadata	- responsible party -- type (Person/Org, Title/Position, ...) -- Person/Organization metadata <or> Reference by title/position metadata	- responsible party for service maintenance -- type (Person/Org, Title/Position, ...) -- Person/Organization metadata <or> Reference by title/position metadata - responsible party for service operation -- (same as service maintenance)
- access/invocation metadata sets	- Access/invocation metadata - prequalified list (individuals, organizations (individual who have association with), roles) of who can invoke	- Access/invocation metadata - prequalified list (individuals, organizations (individual who have association with), roles) of who can invoke - Service Level Agreement metadata
- Constraints/assumptions metadata - pedigree metadata sets	- Constraints/assumptions metadata - pedigree metadata sets	- Constraints/assumptions metadata - pedigree metadata sets
- Security metadata (including access privileges required)	- Security metadata (including access privileges required)	- Security metadata (including access privileges required)

Structural metadata can be considered a subset of data resources (or alternately, content) but it has typically been given more prominence because it is seen as the prerequisite resource in the build *vs.* runtime perspective for developing and using metadata systems. For example, the DoD Metadata Registry Guide [9] describes Information Resources, Data Assets, and Data Services, where Information Resources refer to XML schema, XML style-sheets, document type definitions, attributes, data structures and other types of structural metadata. From the AoA analysis, specific metadata types that could be considered structural metadata include metadata for schemas, message holders, message objects, and possibly other network and device descriptions. However, several conclusions emerge from the AoA analysis that suggest a less prominent role for structural metadata as a special category. In an SOA environment, integration is done through service interfaces rather than the traditional wiring together of components. Thus, the need for detailed format information is encapsulated in the creation of the service interface, a task generally performed by those already knowing the format details. Secondly, the discussion of schema metadata in Section 5.3 suggests that the build time *vs.* runtime distinction may not be as useful as a query *vs.* populate paradigm. The AoA analysis identifies analogous metadata functions across build and runtime activities, and a query *vs.* populate perspective emphasizes how common tools and techniques are more natural if structural components are considered as another resource with metadata similar to that shown in Table *3*-6. Following this perspective, supporting metadata, such as statistics on where a schema is used or by whom, is equally relevant to nonstructural entities, and effective reuse would be facilitated by having such common functions available as part of any metadata and supported by metadata registries.

5.3 Discussion of select metadata sets – paradigms for using modular metadata

The previous section emphasized the modular definition of metadata sets, introduced the concept, function, and resource metadata categories, and provided some detail on specific metadata sets in each category. Recall that the metadata sets are the result of first identifying metadata types and constituent elements that were implied by AoA use cases and then collecting similar metadata sets across use cases. The result is a collection corresponding to the three metadata categories, with the associated metadata sets described through their notional metadata elements and the conclusions that emerge from considering the functions that the metadata must support.

There was no concerted effort to make the constituent elements completely consistent across all metadata sets because different functions were suggested by different use cases and one of the perceived benefits of a modular structure is that, once defined, different elements can be used where deemed necessary by a metadata developer. Thus at this stage, it is more important to introduce a range of ideas than to definitively attach any given idea to a specific metadata set. The remainder of this section describes details of several metadata sets that are expected to have high reuse. In addition, the descriptions provide a context for suggesting additional functionality and useful perspectives that may be enable the broad range of GES expectations.

Access/Invocation metadata

The Access/Invocation metadata set is a prominent example because, in a SOA, the details of access of any information resource or invoking any processing resource should be hidden from the user. This is most commonly seen as the function of the resource's WSDL. However, access in a composable environment requires more than just the details of the interface; it includes the information a user needs to decide if the resource is appropriate for an intended use. Thus, the Access/Invocation metadata should include items such as

- a description of the interface corresponding to this metadata
- the type of access (read, write, delete) supported
- WSDL description
- who is responsible for the interface
- when and by whom the interface was last changed
- details on constraints (including security and intellectual property rights), assumptions, and pedigree
- details on service level agreements (SLAs)
- what permissions are necessary to use the interface
- who is prequalified to use the interface
- who has certified the interface for use

The prequalified list is a notional mechanism by which users who have met all necessary criteria can be granted expedited access. Possibly, this could be done by a service that checks whether the criteria (*e.g.*, policies, terms of use, access category definitions) identified as part of this metadata set has been satisfied and registers with the criteria to be informed of changes that might affect continued prequalification status. The prequalification service would maintain a list of the entities it has qualified and revalidate the applicable members of the list if a criteria changes.

The certified list is similar but in this case a Community of Interest (COI) could certify a resource as having an authoritative status (per its documented definition of authoritative) or be preferred for use. The resource would note who has certified it (a possible factor in whether someone outside the certifying organization wanted to use it) and the COI would maintain a list of its pre-certified resources. The certification process could also be done through a service that ensured the certification lists for resources and the COI remain consistent.

Constraint and Pedigree metadata sets

In a SOA environment, constraints will describe a host of assumptions, restrictions, and conditions related to a resource, not only to determine whether a prospective user should be permitted access but for the prospective user to decide whether the resource is appropriate for the immediate tasking needs. Notional elements include:

- name and description of the constraint set,
- version number and link to the definition of the version terminology,
- Access/Invocation metadata for reading the constraint set,
- Access/Invocation metadata for the preferred processing agent for evaluating an entity against the referenced constraint set
- pointer/link to entities that are evaluated against this constraint set.

Constraint definitions are a precursor to establishing pedigrees. Pedigree metadata is most often thought of as that information that would be useful in evaluating the pedigree of an entity. On further analysis, it becomes clear that such supporting information, rather than being separately identified as pedigree metadata, is interspersed throughout other metadata sets, such as the responsible party for a service access or the date a resource was created or modified. Moreover, the vital metadata is less what goes into evaluating a pedigree and more which pedigrees have been satisfied and how has that been determined. This leads to the following notional elements:

- the pedigree which describes the status of the resource,
- a pointer/link to the constraint set which specifies the conditions satisfied or not satisfied by an entity with this pedigree,
- a pointer/link to the processing engine used to evaluate the constraint set and establish the pedigree,
- when the pedigree was established,
- if applicable, when the pedigree expires.

An entity can have multiple pedigrees corresponding to different constraint sets or different degrees of satisfying a constraint set. A pedigree may be as straightforward as to say a metadata instance has been validated against a schema or it may capture a partial validation which in and of itself has merit.

Aspects of pedigree are similar to prequalification described as part of Access/Invocation metadata. Establishing pedigree could be done through a separate service that performs certification by evaluating the entity with respect to an identified constraint set and then appending the pedigree metadata set to the entity's existing metadata. Thus, an entity's metadata would not just be a static set submitted by someone during a registration process but could also be modified by authorized parties during the life cycle of the metadata. The pedigree evaluation engine would not only write to the entity's metadata but would also register with the constraint set and the evaluating engine so the pedigree could be revalidated should the constraint or the evaluation mechanism change.

Schema metadata

Schemas serve in several distinct roles to enable metadata functions that are an integral part in the typical build time and runtime scenarios. In particular, for querying metadata, schema elements provide the available search parameters for the query submitter. Someone querying to identify an entity supplies target values for some subset of the schema elements for metadata describing the entity, and the query results indicate those instances whose metadata values best match/approximate the target values. The query process is the same for all queries but uses different schemas as the basis for queries of different entities. For populating metadata, schema elements provide the descriptive parameters for which a metadata producer provides descriptive values. The populating process is the same for populating any metadata instance, again using the schema appropriate to the entity at hand.

During the traditional build time, a schema developer will search metadata describing existing schemas to find one(s) to reuse as the basis for a new schema. If we assume there is descriptive information about schemas and the query provides more than a string match to schema elements, then the metadata template for both the query to find the existing schemas and the template to populate to describe the eventual new schema is a schema-for-schemas[3].

During the runtime activity of a metadata producer needing to create metadata for some new entity, the producer will search metadata describing existing schemas to find one to use as the template for a new metadata

[3] One class of resources requiring descriptive metadata are the schemas that serve as the structure for metadata instances. Thus, there is a schema for describing schemas that is likely produced by those organizing and maintaining a metadata registry. This schema-for-schemas follows all the rules for schemas and its metadata description is an instance of itself. While this logic appears circular, it is consistent with descriptions in the XML Schema specification. The power of this construction is that the metadata for describing schemas is no different from the metadata describing any other class of entities, and thus the metadata can be created, organized, and searched by common mechanisms.

instance. The metadata template for the query is the schema-for-schemas and the metadata template to populate to describe the new entity is the schema identified by the query.

For a metadata consumer needing to find an entity to support a runtime need, there is an initial query or browse phase to identify a schema that describes the required entity (in the other scenarios just described, the required entity is a schema and the corresponding schema is the schema-for-schemas). Using the schema found from the initial search/browse, the consumer will search metadata instances describing the required entity to find one to satisfy the current runtime need. The metadata template for this query is the one from the initial search/browse and there is no populate metadata phase.

Note that the above scenarios for the three user types follow similar processes. When one is looking for an entity to meet their needs, they assume the role of someone querying to identify resources. This could be a schema developer looking for schema fragments upon which to build, a metadata producer looking for a schema to populate to describe their current resource, or a metadata consumer looking for some entity relevant to a COI task. When one needs to create metadata instances, those instances are created by providing values to the elements of the relevant schema. For the schema developer, the organizing schema is the schema-for-schemas and the metadata produced is that describing new schemas. For the producer of metadata for resources other than schemas, the organizing schema is any of the other schemas developed by schema developers. The process of creating metadata for schemas or metadata for any other entities is the same.

A conclusion of the analysis is that a major distinction in the scenarios is not build *vs.* run time but query (including use of query results) *vs.* populate. To support this, the notional elements for schema metadata could include:

- schema name
- schema description
- schema keywords and link to keyword vocabulary definition
- who created schema and when created
- how to access (e.g. WSDL, if through service)
- which schemas incorporate this schema (*i.e.*, use as a building block)
- which schemas are incorporated in this schema (*i.e.*, used as building blocks)
- how many instances use this schema
- list of entity owners with largest number of instances using this schema
- list of domains which recommend using this schema

Note the last five elements provide information to describe a context for this schema and facilitate reuse. The statistics are likely collected by the metadata registry and their values would be maintained by the registry or delivered through a registry service. The specifics of those metadata elements and their eventual use should be the subject of further design.

Log metadata

Especially as relates to security, there is considerable discussion of NCES or any service framework being able to trace and audit transactions. In addition, in a composable environment, it is not enough for a user to submit a request and get back an answer if the answer does not include information specifying how the answer was generated and from where input data was obtained. This is important not only for immediate documentation but also for repeatability and efficiency in executing later requests. For example, if a user submits the exact same request on two consecutive shifts and is returned different responses, the user must know whether the difference is due to a change in the input data or a change in the processing or data resource. In addition, considerable compute and communications resources could be used in determining how to satisfy a complex request, and it is advisable to be able to repeat a previous established process rather than reinventing it for every request.

Log metadata assumes that the processing steps and utilized resources are captured through an auditing process and the resultant log will be stored and catalogued for future reference and use. The notional elements chosen to support such activity include:

- link to the log
- link to the entity for which the log applies
- type of log (*e.g.*, access, update, processing steps)
- access/invocation for reading log
- access/invocation for executing log

Note, one access/invocation elements is defined for reading the log contents, and the other, assuming the log exists in a form that can be considered an executable resource, defines the invocation of that resource.

6. CONSOLIDATED VIEW OF METADATA CLASSES

The discussion in Section 4 detailed goals that metadata is meant to empower and the metadata groupings that have been derived to enable realization of those goals. Although the authors of each effort were familiar with the preceding results, the various groupings were conceived somewhat independently, taking a different perspective on framing the problem. This should not be thought of as duplicated effort because the critical role assigned to metadata in a service-oriented architecture has many facets which have only just begun to get attention both with respect to NCES and in the general Web community. Indeed, the different perspective have helped to build a more complete metadata picture. The focus of this section is to begin to assemble that larger picture.

The Net-Centric Data Strategy defines seven DoD data goals and these can be considered the benchmarks by which any metadata strategy would be measured. In discussing approaches to achieving these goals, the Data Strategy introduces high level metadata types and functions, and this provides an initial set of metadata categories. DDMS focuses primarily on one of the net-centric goals, Discovery, and begins building the metadata tagging framework to capture information by which existing communities discriminate among entities that can satisfy their user needs. The AoA analysis derives metadata specifics from the more general perspective of use cases covering a number of NCES core services, including Discovery. For the AoA analysis, the focus is on enabling functionality implied by each step of the use cases and this often requires simultaneously satisfying several of the Data Strategy goals.

The different perspectives lead to identifying different metadata categories and specifying different levels of detail. With the broader perspective, the AoA analysis generated less specificity at the element level than that provided by the DDMS focus on Discovery. However, there is significant commonality at the basic concepts level, such as name, description, or contact information, and in most cases, a more complete solution is a combination of the two sets of results. For example, DDMS dedicated significant effort in specifying security details while such details are lacking from the AoA analysis. Security is of vital importance to NCES and all Web services but the AoA analysis time frame did not afford the opportunity to fully analyze security concerns for which DMMS provides guidance.

While there is significant agreement among the identified metadata categories, there are also some differences in structure and content. With respect to structure, the AoA analysis found significant benefit in a modular framework where small schema fragments were readily reused in building more complex but also reusable structures. For example, the DDMS Resource Set specifies metadata structures for the roles of Creator, Publisher, and Contributor. There are identical elements within each of these structures but the most visible equivalence is implied by nomenclature and any formal relationship is embedded in the DDMS schema. Using more transparent metadata building blocks, such as Person/Organization, would support a common structure defining many roles, possibly with a metadata element being added to identify the role itself. This also highlights the importance of identifying the vocabulary from which the roles or other terms are defined. By making a vocabulary identifier an integral part of the metadata structure, the framework is more extensible, reusable, and interoperable in the future because new roles can be added at the instance level rather than having to add to and modify the schema structure itself. DDMS has several metadata constructs where the Qualifier tag is used to identify vocabulary, but there may be significant benefit to making this a standard part of the infrastructure.

With respect to metadata content, one area where there is a difference is in the perceived need for Format metadata. Format details are critical in a traditional integration because this is the level at which developers needed to wire together the often diverse components from which their standalone systems would take form. Consistent with this approach, DDMS highlights Format as one of the core layer categories. However, the emphasis of a SOA is on the Web service interface, reducing the need for format detail because this is hidden by the standard definition of the service interface. Thus, the format detail is now limited to service developers who are likely part of the project teams responsible for the resource being exposed by the service. The format detail would be available internally to the team and will be of less interest to most of the community who directly or indirectly uses metadata to enable other service capabilities.

While interest in format details may be reduced, the composable aspects of a SOA environment elevates the need for resource pedigree, both in terms of the information needed to establish pedigree and the means and results of evaluating this information. DDMS follows a more traditional approach by identifying information likely to be useful in evaluating pedigree and collecting these in the Resource Set elements. The AoA analysis found that the relevant information is naturally distributed across a number of metadata categories and there was limited value in collecting these under one structure. This is because most information will have multiple uses and higher quality metadata is likely to result by allowing the metadata provider to use (and reference) a local vocabulary rather than extract information to an imposed structure. In addition, the information useful in establishing pedigree is likely to expand and evolve over time, resulting in use of information that had not been previously associated with pedigree. Moreover, the importance lies not in collecting the possible information bits required as input but in documenting how pedigree has been evaluated, what context defines the criteria, and what is the result of the evaluation. Thus, the emphasis shifts to metadata that describes the rules and constraint sets which define any particular pedigree and identifying the processing resources used to evaluate entities against these rules. Pedigree and also logging are examples of functions with greater importance in a SOA environment, and these merit in-depth consideration in the future.

Finally, there are several considerations that apply across all the efforts to describe and categorize metadata. First, the semantics of the metadata tags must be clear and unambiguous. In general, this is done but the Qualifier tag in DDMS is one example where a tag is overloaded and its meaning can be very different depending on context. If there are basic information items that are specific to only a few metadata contexts, these should be defined in terms of metadata building blocks and then consistently reused across all relevant metadata sets. Flexibility in assigning names and terms can be accommodated by emphasizing separate, either NCES or COI defined, vocabularies from which terms can be referenced. This again provides flexibility at the operational level without requiring changes to the infrastructure to accommodate changes in the mission. XML Namespaces are a valuable example in providing a degree of clarity and flexibility. The namespace identifies a unique vocabulary but does not specify the descriptive resources at the indicated URI. Thus, the resources retrieved by dereferencing the URI can be tailored to the entities being described. Defining what resources support the user needs and NCES mission may be a useful area of further investigation.

A final consideration is life cycle issues. The emphasis up to now has been on encouraging metadata production by the resource owners, and while the metadata is not necessarily static over time, the assumption was that changes in the metadata would remain the responsibility of the owners. The AoA analysis uncovered several scenarios where metadata may be modified and augmented over the resource life cycle and these changes will be made by authorized entities other than the resource owner. For example, if one organization has established a pedigree for a resource, this may be vital information for another organization considering the same resource. The pedigree is not under the control of the resource owner and the owner should not be involved in augmenting the metadata to reflect the someone else’s pedigree. Distributed, authorized modifications and additions to metadata have not been adequately considered in the past and may be a vital capability in the future.

7. CONCLUSIONS

Metadata is an important enabler for any service-oriented architecture, and is especially critical in support of GIG Enterprise Services and the Net-Centric Data Strategy goals. The discussion compared several efforts to describe metadata and introduced the benefits of a modular approach to metadata structure. It also highlighted supporting capabilities that could be implemented through metadata registries. These capabilities include

- providing a standard way to link any term to a defining vocabulary
- providing services to augment metadata in a consistent manner and as required to introduce or update descriptive information that is outside the control of the associated resource, *e.g.* to track certified and prequalified use of resources
- collecting and making available statistics that describe the use and reuse of schemas and other resources.

The discussion is not meant as a definitive specification of particular metadata types or sets, but to provide insight into the requirements for creating, maintaining, and using metadata in a SOA environment. The reference to NCES Analysis of Alternative use cases demonstrates the aspects of metadata that directly impact the GIG ES and accomplishing the Net-Centric Data Strategy goals.

REFERENCES

[1] ASD(NII) CIO Global Information Grid Core Enterprise Services Strategy, Draft version 1.1a, 9 July, 2003, http://www.defenselink.mil/nii/org/cio/doc/GIG_ES_Core_Enterprise_Services_Strategy_V1-1a.pdf

[2] DoD CIO Memorandum, DoD Net-Centric Data Strategy, Version 1.0, 9 May 2003, http://www.defenselink.mil/nii/org/cio/doc/Net-Centric-Data-Strategy-2003-05-092.pdf

[3] Web Services Description Language (WSDL): version 1.1, http://www.w3.org/TR/wsdl; version 2.0, http://www.w3.org/2002/ws/desc/ .

[4] DoD Discovery Metadata Specification (DDMS), Version 1.0, 29 September 2003, DASD (Deputy CIO), http://diides.ncr.disa.mil/mdreg/user/DDMS.cfm.

[5] Net-Centric Enterprise Services (NCES) Analysis of Alternatives (AoA) Report, 4 May 2004.

[6] DISA Concept of Operations for DoD Metadata, draft, September 2004.

[7] Data elements and interchange formats - Information interchange - Representation of dates and times, ISO 8601 : 2000.

[8] HR-XML Consortium, Person Name 1.2 Recommendation, 26 February 2003.

[9] DoD Metadata Registry Guide, draft for public comment, http://diides.ncr.disa.mil/mdregHomePage/mdregHome.portal