



Metadata[®] Ontology Language

Metadata[®] has developed a new type of language – the Metadata[®] Ontology Language. The language was invented by Jack Myers who also coined the term “Metadata.” Metadata[®] is now a registered trademark of The Metadata[®] Company. Considerable advantages result from the creation and use of such a language, especially in the areas of data integration, semantics, ontology development, data analysis, and software/hardware generalization. The main feature of the Metadata[®] Ontology Language is that it closely replicates the structure of human thought so that the various interpretations of human ‘reality’ that exist within the brain-mind can be represented and processed in an accurate, useful way, and without the loss of original meaning. This makes the Metadata[®] Ontology Language especially useful for the creation and uses of ontologies.

DATA LANGUAGE PRINCIPLES

The Metadata[®] Ontology Language has incorporated a number of principles found in other types of ‘language,’ including natural language, mathematics, logic, and computer programming language. The Metadata[®] Ontology Language supplements the capabilities of these other languages, filling in the much needed semantic hole in data processing. And, since the Metadata[®] Ontology Language is based on common language principles, it can easily interface with the other languages, greatly increasing data processing possibilities. The Metadata[®] Ontology Language can be used to model all other types of language, thereby inter-relating them. This is especially valuable when integrating data from disparate sources, particularly when they are a mix of structured and semi-structured data sources.

Natural Language

Natural language, e.g., English, evolved for social discourse, not data structuring or data processing purposes. Because of its inherent structure, natural language creates considerable ambiguity of meaning. Natural language structure is not suitable for direct use in computer processing. Variations of natural language, with all of the inherent problems, are still found in modern data processing methodologies, for example, relational database management systems (RDBMS), XML, RDF, OWL, etc.

The use of parts-of-speech (nouns, adjectives verbs, etc.) in natural language is what enables words to be structured in an infinite number of ways using a relatively small number of relations between them. The Metadata[®] Ontology Language also uses parts-of-speech patterned after the functions the data performs. There are fewer parts-of-



speech in the Metadata[®] Ontology Language than in natural language and they perform different functions because of the special structuring needs of data vs. words.

The use of natural language words (names) for data identification poses another major problem in data structuring. Names are not the things being talked about, even though they appear to be using natural language. The concepts referred to by words are automatically identified and linked within the brain-mind during discourse. There is a need to externally represent concepts without the use of names in any computer system that presumes to incorporate semantics. The Metadata[®] Ontology Language does this by using Data Element Numbers (DENs) for concept identification.

The natural languages, apart from pidgin language, have very rich and complex grammars. English, for example, has over ten-thousand rules, the vast majority of which its speakers are completely unaware. The sheer number of rules is what gives natural language its extreme ability to express thoughts, but it also creates an excessive amount of ambiguity, both for individual concepts and for concept relations, which the brain-mind can somehow largely decipher. Any data language, such as the Metadata[®] Ontology Language, must be based on a simple, well-defined grammar that eliminates, or at least identifies when ambiguity exists, using a few explicit rules.

Mathematics

Mathematics is known for its precision in defining variables, rules, and relations. It allows previously established mathematical statements to be combined with new statements. This power of mathematics has permitted the exponential growth of science. Unfortunately, mathematics does not address semantics to any extent. The use of a variable and what the included data means are two different things. However, mathematics does not address meaning. What mathematics does very well, and in a very precise way, is express relationships of data without the use of verbs, which permits its statements (formulas) to be explicit, unambiguous, and general.

Mathematical relationships include the logical The Metadata[®] Ontology Language uses these three mathematical operators “<”, “>”, and “=” in place of natural language verbs to express all data relationships. It should be noted that these data relationships may be named by verbs, but verbs do not participate in the data structure. The use of verbs in defining data relations creates an infinite number of possible data relationships. By limiting the number of possible relations to just “<”, “>”, and “=,” and by using three parts-of-speech, just 28 general data relations can be used to structure all data. This also permits one logical database design to be used for all data.



Logic

Logic is the science of correct reasoning, not data processing. It studies the possible ways of making sense out of object-events, for example, by providing a systematic way of investigating conversational discourse. Logic is concerned with the creation and relation of assertions and with the validity of inferences, i.e., the criteria of validity of thoughts. It is only possible because the human brain-mind is capable of holding general notions, or concepts (represented in the Metadata[®] Ontology Language using DENs).

Logic enables the purposeful and lucid interface between concepts through generalization, abstraction, relation, form, system, etc. It is a valuable tool of philosophical thought and reason and an important component to clarify thinking. Logic can be used to reveal inconsistencies in our thoughts and how we express and represent them, such as in data structure. As such, it is an important consideration in any data language.

Creating a data language based on human thinking is particularly difficult because it must deal with both concepts and the relation of concepts. Logic deals with reason extremely well. However, it does not deal well with the structure of concepts. It necessarily hinges on the capacity of the human brain-mind to designate, or refer to, concepts by common names without any reference to meaning, something that is accomplished in the Metadata[®] Ontology Language using DENs.

Logic has a close connection with natural language. Yet, in natural language, the structuring of words into sentences requires that one of the words in the sentence represent a relation. The fact that the relation may be named by a preposition or other kind of word requires an extra auxiliary verb to assert the relation. While verbs perform this function in natural language in a very complex and ambiguous way, simple mathematical operators are used in the Metadata[®] Ontology Language.

Unless we appreciate that different data may take the same logical form in our heads, we have no effective or efficient way of relating data. This point is not addressed by traditional data processing. The general proposition structure used by the Metadata[®] Ontology Language eliminates the typical confusion of elements and relations found in open-ended propositional structures, especially when several propositions are contracted into one. Also, different levels of abstraction are often obscured and mixed. This is a major issue of the XML/RDF approach to data structuring since there is no semantic consistency.

To solve this problem, the Metadata[®] Ontology Language uses a simple 5-tuple propositional structure to cover all data relations. The elementary propositions are combinations of DENs structured using just 28 general data relations, which assert that a



meaningful relation holds among the concepts represented. This propositional structure can represent any *state of affairs* (real or imaginary) and is expressible using any ordinary declarative language, such as English.

Computer Language

Computer languages must provide several features not found in other languages. One is the explicit use of data types and domains; another is the use of meta-data. There currently is no common ground for the content or structure of meta-data, and both features go unexpressed in natural language, although natural language is reflexive in that it can talk about itself. Data types, domains, and meta-data are explicitly handled in the Metadata[®] Ontology Language in a common, interrelated way. It is important to note that the Metadata[®] Ontology Language can be used to define other external meta-data, thus providing a general solution of a very complicated problem.

LANGUAGE OVERVIEW

A key feature of the Metadata[®] Ontology Language is that it is based on the use of a Data Element Number (DEN) to represent concepts. Each DEN can be separately related to any other DEN, or combination of DENs. Combinations of DENs, in the form of 5-tuple propositions, form semantic 'truth statements' that are each semantically complete. The 5-tuple proposition is composed of up to five DENs and one relational operator, where the combination of meaning represented is both holistic and interdependent. The structural relationship which existing among DENs always adheres to the predefined set of grammatical rules (28) of the Metadata[®] Ontology Language, which are not dependent on the use of words for expression or manipulation.

This contrasts most other computer-based data structuring methodologies. Typically, they rely on uniquely naming data columns in tables/records and relating these tables/records using natural language words (e.g., teaches/taught, owns/owned by, rides/ridden by, sell/sold by, etc.). This makes the resulting data structure dependent on the varied and complex natural language rules known only by the brain-mind.

The Metadata[®] Ontology Language's data relations can represent far richer meaning, and in a significantly simpler way, than can the triple used by RDF, or other common data processing constraints. Complex semantic structures (nets) can be created by simply merging individual data relations while preserving each individual data relations representation of meaning.



Data Elements

The DEN represents what is perceived by people. Rather than trying to give each concept a name, which opens the concept to potential conflicting interpretations, each concept is given a unique, non-significant number to represent it. This is a very subtle, but significant, difference between the Metadata[®] Data Language and all other data structuring techniques. DENs are surrogates for concepts and thus given unique explicit definitions using text, formulas, etc. DENs may be given any number of textual names which refer to the DENs representing concepts. Thus data structure is based on DENs (concepts), not names. The brain-mind automatically links words with concepts, which is something computers are unable to do since they have no concepts. The goal is to represent concepts as precisely as possible, and in such a way that can easily show any relationships among them.

Parts of Speech

The Metadata[®] Ontology Language uses the symbols I, M, and D to identify the role of each DEN in a relation. This is comparable to the use of parts-of-speech in natural language grammar. Any DEN used for subject identification is classified as an Identifier. In essence, Identifiers are used as subjects about which the other types of DENs are related. For example, the DEN representing the concept 'person' may be an Identifier because it is a subject about which we want to collect and relate other data, such as family role, physical attributes, employment, etc.

The M may be assigned to any DEN which is functionally used as a Modifier. A Modifier (M) represents the unique variations and occurrences of an Identifier (I). It is also used to categorize similar data via codification. For example, a "person" (I) has many things happen over a lifetime – at different times or "date" (M).

The D is used as a predicate attached to Identifier-Modifier combinations including Identifier-Modifier to Identifier-Modifier combinations.

Data Relations

Relationships are expressed between subjects (Identifiers) using the operators ">", "<", and "=". The ">" means that one subject is somehow superior to another subject in a given data relation; the "<" means that one subject is somehow inferior to another subject in a given data relation; and the "=" means that two subject is somehow equivalent in a given data relation.



The operator may be given a name in a given data relation. For example, “>” may be named “owns”, “contains”, or “instructs”, the relation “<” may be named “is owned by”, “is contained in”, or is “taught by.”

The combination of the three “>”, “<” and “=” operators together with three I, M, and D functional data classes (roles) creates 28 possible data relationships, which hold for all data.

EXAMPLE

In the following example, a person I (DEN 100) on a given data M (DEN 200) has a hair color D (DEN 300) and has a given role M (DEN 500) within a family. These four data relations allow the tracking of a person’s hair color and family role over time (at a later date the hair color may be grey).

The hair color in the example is brown when it is associated with Bob on date 1-1-95, and the family role of Sue to Bob is wife on 1-1-05. As shown, certain data relations are logical opposites such as IM>IM and IM<IM.

<u>Relation</u>	<u>DEN</u>	<u>Name</u>	<u>Data Instance</u>	<u>English Translation</u>
<u>Relation</u>	<u>DEN</u>	<u>Name</u>	<u>Instance 1</u>	<u>Instance 2</u>
I	100	Person	Bob	Bob
IM	200	Date	1-1-95	1-1-05
IMD	300	Hair Color	Brown	
IM>I	100	Person		Sue
IM>IM	500	Role		wife
I	100	Person		Sue
IM	500	Role		wife
IM<I	100	Person		Bob
IM<IM	200	Date		1-1-05

Metadata[®] Toolset Description

Based on the Metadata[®] Ontology Language, we have developed a powerful suite of data management tools that together, provide unprecedented capabilities for users. The Metadata[®] Toolset includes tools that manage data more like the way that users think and use data. Unlike other data management technology, users’ data needs are not constrained by the limitations of the particular data management tool that they have



available to them. The following describes key characteristics of the toolset along with a description of the components that comprise the Metadata[®] Toolset.

Data Integration

The toolset provides the capability to integrate disparate databases, transparent to the user. So, while a user may need data from three different databases, each having its own design, and maybe even using different storage structures (e.g., relational, network, hierarchical), the Metadata[®] Toolset offers a means to provide data from all these sources without the user needing to pull data from each source and then somehow integrate them before the data are useful to the user. The result is a fully integrated, unified database that:

- Contains all the data and data relationships contained across all the database sources;
- Ensures common meaning for each data element and relation in the database; and,
- Ensures data value integrity when using the data in different applications.

Self-Defining Database

The Metadata[®] Toolset provides a single database design that can be used by all applications. Therefore, when new data requirements are identified, the self-defining Metadata[®] Toolset database incorporates them into the existing design. The Metadata[®] Toolset automatically links the new data requirements to all the relations already in the existing database design. Reexamining different parts of the design to determine impacts is not necessary. The toolset accomplishes this automatically.

Another difference between the Metadata[®] Toolset and other data management technologies is important to note. Most database technology manages data principally at the record level. As a result, the relations between data are aggregated from the level that users think and manipulate data which is at the attribute level. Metadata[®] manages data at the attribute level. This offers substantial improvement regarding the ways users can access, manipulate and analyze data.

This attribute-level data management is also key to why the Metadata[®] Toolset is self-defining as noted above. When new data requirements are added to the Metadata[®] Toolset, the new requirements have relations to data already in the platform. As these are linked, all the relations to the existing data are automatically associated with the new data. Often these relations are not immediately thought of by the analyst or user. But, due to the platform's self-defining capability, they become immediately available to that user.



Benefits of the Metadadata[®] Toolset

There are numerous benefits provided by the Metadadata[®] Toolset. A few examples include:

- Unprecedented data interoperability
- Integration of content, data relations, and data meaning from disparate databases
- Virtual data retrieval
- Non-programming-based data migration and interoperability
- User-oriented non-programming-based data queries
- Non-destructive database design enhancements
- Data design reusability
- Data reusability
- Reduced data management costs

Further Information

For additional information regarding the Metadadata[®] Toolset, please contact:

Long Beach, California

Steve Peacock

562-495-2011

SPeacock@Metadadata.com

Reston, Virginia

Sean Yanosh

703-476-9700

SYanosh@Parallax.net