Conceptual Model for Ecological Data Management

This draft compares OBOE and several other conceptual models (CM), OGC O&M model and EQ in catching the semantics of observations and measurements of ecological data. The comparison shows that OBOE is highly compatible with other conceptual models and also can accommodate many important domain ontologies, which are widely used in the ecology domain.

1 Introduction

How to evaluate the quality of a conceptual model (CM)
This question is posted based on the need on evaluating which conceptual model (e.g., OBOE, O&M, etc.) is a good one, or is better than the other. Unfortunately, the literature [20] shows that there is no well-formed standard way to evaluate the quality of a conceptual model. On the contrary, most of the conceptual models are evaluated in an *ad hoc* manner. However, this article highlights a major principle in evaluating a conceptual model: a conceptual model is valuable only it is used in practice.

Following this principle, we consider several factors in building our final model (either OBOE or an extended OBOE) in SONET project. The several factors include:

1. How compatible is a CM with other conceptual models?
2. How easy it is for a CM to accomodate other domain knowledge?
3. How easy it is to perform operations on the model?

In what follows, we do analysis of OBOE from the above three perspective.

2 Terminology analysis of CMs

To start with, we first analyze what are the "things" or "objects" that each model is representing. Then, we show their term correspondences. Finally, we sketch the algorithms to make them compatible.

2.1 OBOE

In OBOE, an *observation* represents any *measurement* of some *characteristic* (attribute) of some real-world entity or phenomenon. A *measurement* consists of a realized value of some characteristic of an *entity*, expressed in some well-specified units (drawn from a measurement standard) One observation can provide *context* for other observations. E.g., observations of spatial or temporal information often provide context for some other observation. Using the OBOE concepts, we can describe the type level or instance level relationships. E.g., an entity type has characteristic types. Or, an entity instance has characteristic instances.

HP: more to come.

2.2 OBOE and EQ

In EQ (Entity Quality) system [1], the key terms are as follows.

- Entity: describes some object in the real world. (e.g., dorsal fin)
- Quality: describes an entity's attribute and its attribute value. (e.g., shape = round, means dorsal fin's shape is round.).

- Character: is composed of Entity and Quality Attribute to represent the meaning of which entity's which attribute. E.g., dorsal fin's shape.
- Character State: Quality value (e.g., round, to represent dorsal fin's shape is round).

2.3 OBOE and O&M

To compare OBOE and O&M, we focus on several key terms.

First, O&M also uses the terminologies *Observation* and *Measurement*. But their definitions show the different meaning. In OBOE, these two terms refer to something, but in O&M, these terms denote some action.

- An **Observation** is an action with a result which has a value describing some phenomenon. (p1 [5]) or *an act of observing a property.* ([13] Clause 4.10)
- *Measurement is a set of operations having the object of determining the value of quantity.* ([13] Clause 4.9).

Second, terms used to describe real world object (OBOE) or phenomena (O&M). In OBOE, an *Entity* can have many *Characteristics*. Parallel to this, in O&M, a *feature type* can have many common *characteristics* (i.e., *property-type*s) shared by feature instances. So, the corresponding relationship at this level is: *Entity* in OBOE corresponds to *feature type* (or more specifically, *feature of interest*. *Characteristic* in OBOE corresponds to *property type* (or more specifically, *observed property* in O&M.

- The **featureOfInterest** is a feature of any type (ISO 19109, ISO 19101), which is a representation of the observation target, being the real-world object regarding which the observation is made. (p12 [5])
- The **observedProperty** identifies or describes the phenomenon for which the observation result provides an estimate of its value. It must be a property associated with the type of the feature of interest. (p12 [5])

Third, terms used to describe the observation and measurement that are made on the objects of phenomena. OBOE uses the terms *observation* and *measurement* to denote these. O&M uses *observation result* to denote the observation made. So, here, we can see that the $measurement_{OBOE}$ corresponds to $observation\ result_{O\&M}$. In OBOE, a measurement is on a characteristic; in O&M, a result is on a property, which corresponds to OBOE characteristic. The process used to get the measurement (or result) is called $protocol_{OBOE}$ and $observationprocedure_{O\&M}$.

- The **procedure** is the description of a process used to generate the result. It must be suitable for the observed property. (p12 [5]) Or, *method, algorithm or instrument, or system of these which may be used in making an observation.* ([13] Clause 4.11)
- The **result** contains the value generated by the procedure. The type of the observation result must be consistent with the observed property, and the scale or scope for the value must be consistent with the quantity or category type. (p13 [5]). Or, *Observation result* if an estimate of the value of a *property* determined through a known procedure. ([13] Clause 4.13)

Fourth, terms used to describe the measurement (result) values. The values can be categorical or numerical.
$ObservationContext_{O\&M}$ ([13] Fig 2, Clause 6.2.4) is equivalent to $context_{OBOE}$.
The summarized term correspondences used in both models can be found in Table 1.

2.4 Techniques comparison used in OBOE and O&M

OBOE uses OWL DL (Wed Ontology Language Description Logic) to describe its model while $O\&M$ utilizes the UML (Unified Modeling Language) to represent its conceptual schemas. In what follows, we briefly compare these two techniques.

UML defines in a shared package a common core set of language structures. These constructs focus on the **representation** of **static** structural information, i.e., "class diagram" [18]. UML is generally used together with Object Constraint Language (OCL [11]) to complement to its static modeling feature. Both of these are used in O&M. In what follows, when we use the term UML only to represent UML without the OCL. However, some study also use UML Full to represent UML+OCL.

There are a lot of similarities [14, 16, 15] between UML (+OCL) and OWL DL. UML uses a diagrammatic representation and an XML representation called XML Metadata Interchange (XMI) [12]. OWL uses XML syntax-ed RDF and OWL language. Both UML and OWL define object-centered, intention-based representation of knowledge about a system [18]. Both languages have two layers of knowledge representation, instance level and type level. Some similarities are

- Represent an instance individuals: *owl:individual* in OWL and *InstanceSpecification* Define class membership, OWL uses the RDF *type* relation or by using the name of the classifier. UML uses *instanceOf* or the colon-based naming convention.
- Define Class. Using both languages, we can define the sub class relationship (*rdfs:subClassOf* and *A subClassOf B*), equivalent class relationship (*owl:equivalentClass* and *A equivalentClass B*), disjoint class (*owl:disjointWith* and *A disjointWith B*). It also support *owl:unionOf*
- Property values are defined using the the name of a property or relating it to an object or data value.
  - *owl:ObjectProperty, owl:DatatypeProperty* are mapped to binary, unidirectional UML associations.
  - *owl:inverseOf, owl:FunctionalProperty* can be represented in UML using bidirectional associations to combine two inverse, binary and unidirectional associations.

However, these two techniques also different a lot. The basis difference is on its underlying **assumption** that OWL takes open world assumption while UML uses close world assumption. This difference affects a lot of interpretations of the constructs.

In addition, UML does not support **synonyms** while OWL DL supports it. OWL allows the definition of synonyms for classes, properties and individual descriptions. It has explicit constructs to define equivalent classes, properties, and individuals. This way, the same real-world element may have be referred to as different names. So, OWL provides the flexibility to describe the same real world system by using different kinds of terminologies based on people's preferences. Besides in OWL, the identifiers for classes, properties, and individuals are distinct. I.e., the same name always refer to the same real-world element. To summarize, in OWL, the function from class,property and individual names to real-world element is non-injective because different names to refer to the same real-world element. UML on the other hand, does not support synonyms. I.e., if two different names represent two different interpretations. Also because of this, UML has the Unique Name Assumption (UNA). So, the function from name to real-wold element is injective. I.e., the same name refers to the same elements and different names have different interpretations. Since UML does not have the Unique Name Assumption (UNA), it has *owl:sameAs*, *owl:differentFrom*, and *owl:AllDiffereent* to state the identity of individuals.

Besides the above modeling differences, several studies [18] in comparing UML (+OCL) and OWL show that

- UML itself is generally used as a representation-oriented set of paradigm. This is similar to the Entity-Relationship modeling paradigm in database development. Since UML provide intuitive diagrams to describe concepts, at a higher analysis level, UML is more appropriate than OWL DL for people to discuss ideas in the design.
- OWL DL provides many constructs to represent knowledge by a vocabulary and logical definitions.
- Due to its static nature, UML itself cannot represent a lot of constraints that are needed in describing web semantic services. A rough list can be retrieved from [18] as follows.
  - *owl:oneOf,owl:intersectionOf, owl:complementOf* OWL allows to define a class by constraining it's instances be in an enumeration list by using *owl:OneOf*. However, UML only allows the the enumeration of data types. OWL allows to directly define a class to be intersection of other classes using *owl:intersectionOf*. In UML, we cannot directly define the intersection class. But we can define a class to the sub-class of several classes. So, an instance of this class is implicitly instances of its super-classes.
  - *owl:allValuesFrom, owl:someValuesFrom, owl:hasValue,owl:maxCardinality, owl:minCardinality, owl:cardinality*
  - *rdfs:domain, rdfs:range, owl:TransitiveProperty, owl:SymmetricProperty*
- OCL can represent the equivalent semantics:

  - *owl:allValuesFrom, owl:someValuesFrom,,owl:maxCardinality, owl:minCardinality,owl:cardinality.*
  - *rdfs:domain, rdfs:range, owl:InverseFunctionalProperty, owl:TransitiveProperty, owl:SymmetricProperty*

- Even though OCL enriches the representation of UML a lot, it still cannot fully represent some OWL DL constructs: *owl:hasValue, owl:subPropertyOf, owl:equivalentProperty*

On one hand, we can see how UML (+OCL) represent the same semantics which can be expressed using the OWL DL constructs. To summarize, if UML is not used together with OCL, it can best serve as a graphical representation of the model. But it cannot support most of the more precise constraints that can be defined using OWL DL. On the other hand, OWL does not have the counterpart concepts for *UML's aggregation and composition relationships.*

With OWL DL, reasoning can be supported using some tools such as Pe UML together with OCL also support reasoning. In addition, there exists tools to support this. E.g., OCLE [4], Oclarity: Plugin for Rational Rose [3], etc. (A more complete list can be found at `http://www.jordicabot.com/research/OCLSurvey/index.html`).

NOTE from Huiping: not sure yet how the OCI reasing can be done. What's the capabiliy difference between the OCI reasing and OWL DL reasoning.

NOTE: from discussion with Mark

(1) Add the content of the Annotation content with observatioin type, measurement type, etc.

(2) OBOE + annotation language ==¿ OBOE. This need to be discussed with Shawn. What OBOE specification should include.

(3) What are the different expressive power of the implementation techniques for O&M and OBOE?

Need to add:

(1) Annotation specification? Relationships that can be caught by OBOE, can they be expressed using O&M? E.g., one observation can have multiple measurements?

2.5 Term correspondences

Table 1 summarizes the above analysis with term correspondences.

ER	OBOE	O&M	EQ
	Entity	Observation::featureOfInterest	Entity
Entity	Observation		
Characteristic	Measurement	OM_Observation	Quality value or Character State
	Standard	Result type	
	Characteristic	Observation::observedProperty	Quality attribute or Character State
Relationship	Context	ObservationContext	
Value	Characteristic Value	Result	
	Protocol	Observation:procedure	
		Observation::phenomenonTime	
		Observation::resultTime	

Table 1: CM comparison

Relationships between the concepts:

- *OWL functional property* is convered to many to one mapping cardinality in UML. E.g., OBOE *characteristicOf* is a fFunctional property, i.e., it infers that $Characteristic_{OBOE}$ is for EXACTLY ONE $Entity_{OBOE}$. So, it is many to one relationship.
  O&M uses class *OM_Observation* to connect *featureOfIntereste* and *observedProperty.* Since the relationship from *featureOfIntereste* to *OM_Observation*

- *OWL transitive property.*

- SOME restriction *owl:someValuesFrom* . If a class A has

```
<rdfs:label>Characteristic</rdfs:label>
    <owl:equivalentClass>
```

```
            <owl:Restriction>
                <owl:onProperty rdf:resource="#hasCharacteristicValue"/>
                <owl:someValuesFrom rdf:resource="#CharacteristicValue"/>
            </owl:Restriction>
        </owl:equivalentClass>
```

. . . It means that at least one of the *hasCharacteristicValue* property of a *Characteristic* must point to an individual that is a *CharacteristicValue*.

- ONLY restriction *owl:allValuesFrom*

HP:
Q1: SOME and ONLY for hasCharacteristic, etc. ObjectProperty.
Q2: ER paper: observation and measurement should be one to many relationship, but not many to many relationship.

3 Model compatibility

This section, we show the algorithms on how to convert data complying with different models and illustrate the with several examples.

3.1 Conversion between OBOE and O&M

In what follows, we describe how to convert OBOE to O&M compliant file and also the conversion in the opposite direction.

The brief algorithm to convert an O&M-compliant document to OBOE model.

- For each *om:featureOfInterest*, generate an *OBOE::Entity* instance e_i such that o_i has property *ofEntity* e_i
- For each Observation in O&M, denoted using *om:observation*, generate an *OBOE:observation* instance o_i.
- For each *om:result*, generate an *OBOE:Measurement* instance m_i such that m_i has the property *measurementFor* o_i.
- For each *om:procedure*, generate an *OBOE:protocol* instance p_i such that m_i has the property *usesProtocal* p_i.
- For each *om:observedProperty*, generate an *OBOE::Characteristic* instance ch_i such that m_i has property *ofCharacteristic* ch_i.

More to come on the sampling and specialized observations.

3.2 Mappings in other domains

We generated an example for mappings in other domains using O&M and OBOE. Table 2 shows the mappings in the Earth observation domain.

4 Use cases: compatibility of OBOE with domain ontologies

We have several use cases that we can test the different data models. In what follows, we would show how compatible PATO is with OBOE.

4.1 PATO

Phenotype And Trait Ontology (PATO) [6, 7] is a phenotype quality ontology proposed by Ashburner and Lewis. This ontology is presented with the purpose of capturing qualitative and quantitative information about phenotypes in a species-neutral way.

EO	example	O&M	OBOE
observation value, measurement value, observation	$35\mu g/m^3$	Observation:: result	measurement: *hasValue* 35 with unit $\mu g/m^3$
method, sensor	ASTER, U.S.EPA Federal Reference Method for $PM_{2.5}$	Observation:: procedure	protocol: measurement *usesProtocal*
parameter, variable	Reflectance, Particulate Matter 2.5	Observation:: observedProperty	Characteristic: measurement is *ofCharacteristic*
2-D swath or scene	Sampling grid	Observation:: featureOfInterest: Sampling Surface	Entity: measurement is for some observation (*forObservation*), which is of some entity (*ofEntity*).
Earth surface		SamplingSurface: sampledFeature	Entity: sub class of an entity
3-D sampling space	Sampling grid	Observation:: featureOfInterest: SamplingSolid	Entity: a sub class of an entity
media (air, water, $\cdots$), Global Change Maser Directory "Topic"	troposphere	SamplingSolid:: sampledFeature	Entity: a sub class of an entity

Table 2: CM comparison with Earth Observations (EO)

PATO is recommended by Open Biomedical Ontologies (OBO) [2] . PATO is used in several research groups. The entity, attribute, value (EAV) model relies on PATO. Nottingham Arabidopsis Stock Center (NASC) database uses the EAV model to describe mutant phenotypes and natural variants in Arabidopsis [17]. Some other EAV model organism databases, e.g., ZFIN [9, 21] and FlyBase [10] also uses PATO. In addition, Phenoscape [8] also uses PATO (and other ontologies) to link natural phenotypic diversity to zebrafish mutants. [19] also used PATO as their controlled vocabulary list. Because of the wide usage of PATO, it is very important to see that PATO is compatible with the OBOE model. In what follows, we discuss how to convert PATO to OBOE.

PATO does not provide the detailed information of entities in OBOE. In real world, domain scientists generally use PATO and other ontologies together to annotate datasets. This will not be in our discussion focus in this section.

PATO has six different slims to denote the different views that people can use to categorize the classes. In these six slims, four of them (cell_quality, abnormal_slim, absent_slim, relational_slim) are for specific domain usage. The other two slims attribute_slim and value_slim are at a more general level to denote whether the classes (concepts) are attributes or are attribute values.

Based on this meaning, the classes denoted with attribute_slim can be mapped to *characteristics* in OBOE. While the classes denoted with attribute_slim can be mapped to *Standard Characteristic Value* in OBOE, which is a sub-class of 'Characteristic Value'. These standard characteristic values are for a measurement standard. This measurement standard then has object property 'forCharacteristic' only for the newly characteristic and property 'hasStandardValue' only in the newly created standard characteristic value.

For example, class *intensity* in PATO denoted with 'attribute_slim' can be mapped to as a characteristic *PATO:intensity*. The sub-classes of intensity in PATO are {mild, moderate, severe, 'increased intensity', 'decreased intensity', remittent} are denoted with 'value_slim'. Then they can mapped to a Standard Characteristic value *PATO:intensity value*. To impose the constraint that characteristic *PATO:intensity* can only take values in *PATO:intensity value*. We can create a measurement standard *PATO:intensityStandard* it has two object properties 'forCharacteristic' only for the characteristic *PATO:intensity* and 'hasStandardValue' only *PATO:intensity value*.

Formally, this can be done in a bottom-up manner.

- Create a sub-class CH_i of *Characteristic* for a PATO class C_i denoted with 'attribute_slim'. If C_i has a sub-class C_j marked with '*attribute_slim*' in PATO, then in OBOE, we create a corresponding characteristic CH_j for

class C_j and treat CH_j as a sub-class of CH_i in OBOE.

- Create a sub-class $ChSV_i$ of *Characteristic Standard Value* and map all the direct children of C_i with 'value_slim' to sub-classes of $ChSV_i$. Here $\langle CH_i, ChSV_i \rangle$ is a corresponding Characteristic-value pair. If C_i has a sub-class C_j marked with '*attribute_slim*′ in PATO, then we create a sub-class $ChSV_j$ of $ChSV_i$ and map all the direct children of C_j with 'value_slim' to sub-classes of $ChSV_j$.
- For each corresponding Characteristic-value pair $\langle CH_i, ChSV_i \rangle$, create a measurement standard MSi with two object properties: *'forCharacteristic' only* CH_i and *'hasStandardValue' only* $ChSV_i$.

4.2 EnvO

4.3 Trait Ontology

This will be provided by Marie-Angelique.

References

[1] EQ for character matrices. https://www.phenoscape.org/wiki/eq_for_character_matrices.

[2] OBO: http://www.obofoundry.org/.

[3] Oclarity: Plugin for rational rose. (http://www.empowertec.de/products/rational-rose-ocl.htm).

[4] Ocle http://lci.cs.ubbcluj.ro/ocle/overview.htm.

[5] Open geospatial consotium inc (OGC): OpenGIS observations and measurements encoding standard (o&m) http://www.opengeospatial.org/standards/om.

[6] PATO phynotypic quality: http://www.obofoundry.org/cgi-bin/detail.cgi?id=quality.

[7] PATO wiki: http://obofoundry.org/wiki/index.php/pato:main_page.

[8] Phenoscape: https://www.phenoscape.org/wiki/main_page.

[9] ZFIN the zebrafish model organism database: http://zfin.org/.

[10] FlyBase Consortium: The flybase database of the drosophila genome projects and community literature. *Nucleic Acids Res.*, 30(1):106–108, 2002.

[11] Object management group, "ocl 2.0 adopted specification". OMG Specification, Oct. 2003.

[12] Object management group, xml metadata interchange, v2.1. OMG Specification, Sep 2005.

[13] Iso tc 211/sc. geographic information - observations and measurements. review draft., 2010.

[14] S. Brockmans, R. Volz, A. Eberhart, and P. Löffler. Visual modeling of owl dl ontologies using uml. In S. A. McIlraith, D. Plexousakis, and F. van Harmelen, editors, *International Semantic Web Conference*, volume 3298 of *Lecture Notes in Computer Science*, pages 198–213. Springer, 2004.

[15] A. Calì, D. Calvanese, G. D. Giacomo, and M. Lenzerini. A formal framework for reasoning on uml class diagrams. In M.-S. Hacid, Z. W. Ras, D. A. Zighed, and Y. Kodratoff, editors, *ISMIS*, volume 2366 of *Lecture Notes in Computer Science*, pages 503–513. Springer, 2002.

[16] S. Cranefield and M. K. Purvis. Uml as an ontology modelling language. In *Intelligent Information Integration*, volume 23 of *CEUR Workshop Proceedings*, 1999.

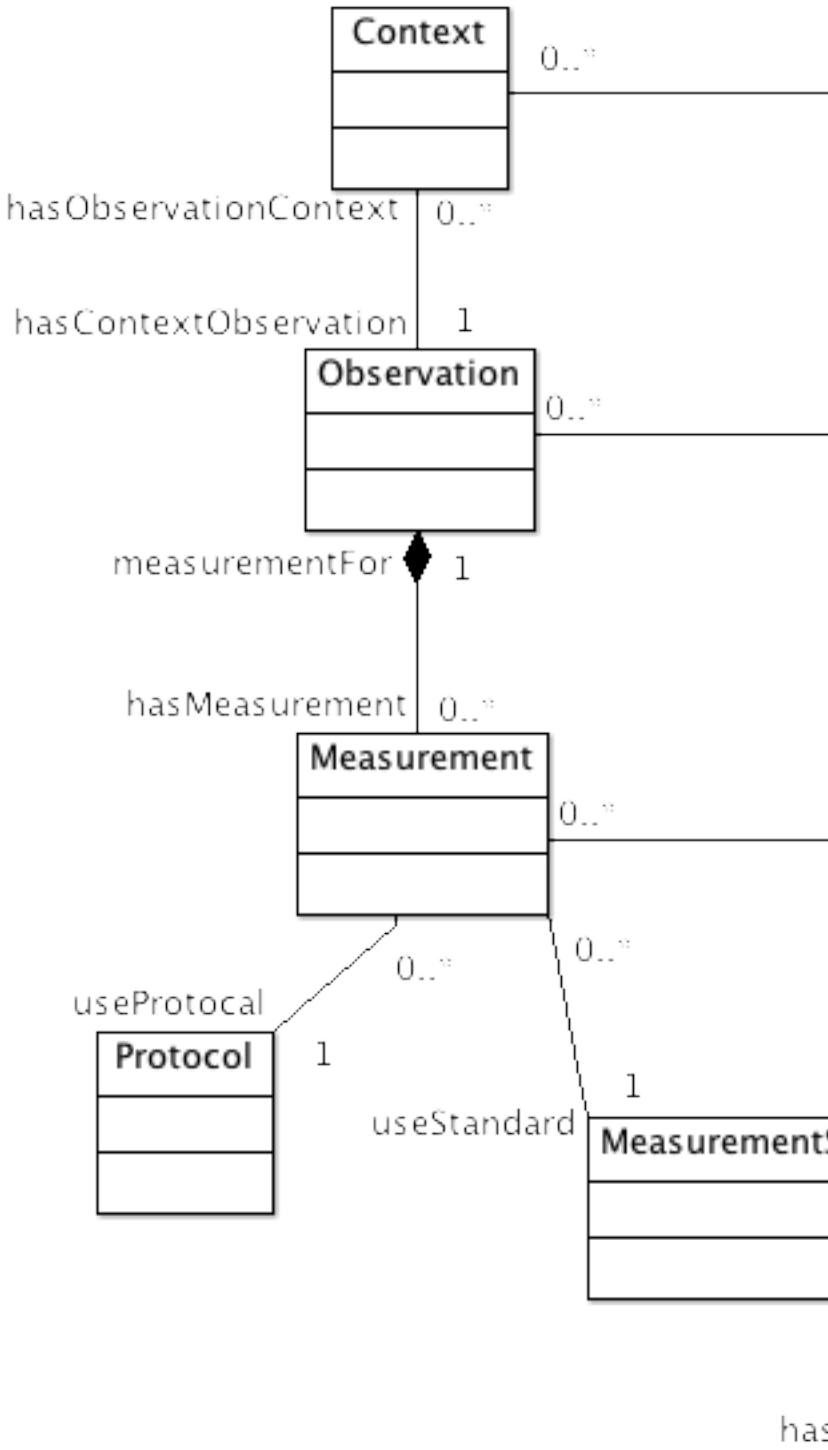
[17] K. Ilic, E. A. Kellogg, P. Jaiswal, F. Zapata, P. F. Stevens, L. P. Vincent, S. Avraham, L. Reiser, A. Pujar, M. M. Sachs, N. T. Whitman, S. R. McCouch, M. L. Schaeffer, D. H. Ware, L. D. Stein, and S. Y. Rhee. The plant structure ontology, a unified vocabulary of anatomy and morphology of a flowering plant. http://www.ncbi.nlm.nih.gov/pmc/articles/pmc1803752/. *Plant Physiol*, 143(2):587–599, 2007.

[18] K. Kiko and C. Atkinson. A detailed comparison of uml and owl. Technical Report TR-2008-004, Department for Mathematics and Computer Science, University of Mannheim, Germany, 2005.

[19] I. Mikó and A. R. Deans. Masner, a new genus of ceraphronidae (hymenoptera, ceraphronoidea) described using controlled vocabularies. *ZooKeys*, 20:127–153, 2009.

[20] D. L. Moody. Theoretical and practical issues in evaluating the quality of conceptual models: current state and future directions. *Data Knowl. Eng.*, 55(3):243–276, 2005.

[21] J. Sprague, D. Clements, T. Conlin, P. Edwards, K. Frazer, K. Schaper, E. Segerdell, P. Song, B. Sprunger, and M. Westerfield. The zebrafish information network (ZFIN): the zebrafish model organism database. *Nucleic Acids Res.*, 31(1):241–243, 2003.


Context
0..*
hasObservationContext
0..*
hasContextObservation
1
Observation
0..*
measurementFor
1
hasMeasurement
0..*
Measurement
0..*
0..*
0..*
useProtocal
Protocol
1
1
useStandard
Measurement
has