Technical Report NEESgrid-2004-01

www.neesgrid.org

(Whitepaper Version: 1.0)

Last modified: February 18, 2004

A Brief Review of Data Models for NEESgrid

Jun Peng and Kincho H. Law[1]

Feedback on this document should be directed to junpeng@stanford.edu

[1] Department of Civil and Environmental Engineering, Stanford University, Stanford, CA 94305-4020

Acknowledgment: This work was supported primarily by the George E. Brown, Jr. Network for Earthquake Engineering Simulation (NEES) Program of the National Science Foundation under Award Number CMS-0117853.

1 Abstract

This document provides a brief review of several data model representations, data modeling tools, and example data models, in order to assess their usability for defining a standard approach for the NEESgrid data/metadata effort. The data/metadata task force plans to produce end-to-end solutions that integrate across site specifications database, project level model, domain specific data models, and common elements.

2 Overview

The primary goal of NEESgrid data/metadata effort is to work collaboratively with NEESgrid team and the NEES community and to help define data requirements and needs for the George Brown Jr. Network for Earthquake Engineering Simulation instigated by the National Science Foundation. The NEESgrid is designed as a distributed virtual laboratory for earthquake experimentation and simulation. The "collaboratory" will allow researchers gain remote, shared access to experimental equipment and data. This draft document gives a brief discussion of the data models reviewed as an initial investigation towards the development of a standard approach for data modeling for NEESgrid data/metadata effort.

As pointed out by Pekcan [1],"A **data model** is the grammar, vocabulary and content that represents all types of "information" stored in one format or another in a "system". The **grammar** defines the relationships between **elements** in the system; the **vocabulary** defines the terminology used to describe these element; **content** defines what is to be included in the system." A data model is in essence a representation of the data and their relationship and provides a conceptual or implementation view of the data. Ideally, the data model should be independent of hardware/software platforms so that its implementation can be universal.

There are many existing data modeling techniques and tools that are available to help design and structure the data. Within NEES community, there have been some ongoing works to develop data dictionary and data models. The purpose of this document is not to provide an exhaustive review on the works related to data modeling but to briefly review and evaluate some of the relevant approaches. A brief summary provides a discussion on the approach currently undertaken to develop a project data model for NEES experimentations.

3 Data Modeling Representation

The process of designing a data model begins with identifying necessary data and the relationships among components of that data. The structure of a data model can be described in different form, such as relational model, object-oriented model, and hierarchical structure. This section briefly reviews some popular data modeling approaches.

3.1 Entity-Relationship (E-R) Model

Entity Relationship (E-R) model is a graphical data modeling approach that organizes data into entities and defines the relationships among the entities [2]. The E-R modeling approach is particularly useful and widely adopted for relational database design. The principal components of E-R model are as follows:

- **Entity sets**, which are the "things" real or abstract) about which we would like to store the data; for examples, these may include project, event, specimen, researcher, sensor, measurement, time, etc. Specific occurrences of an entity are called **instances**, e.g. researcher John Smith, sensor at the northeast corner of the second floor, etc.

- **Relationships**, which are the associations among two or more entities, e.g. a project consists of events, or sensors are attached to a specimen. **Cardinality** defines the number of occurrences of one entity for a single occurrence of the related entity, e.g. a project may have multiple events, which may include pre-experimental tests, phases of the actual experiment, computer simulations, and post-experimental tests, etc.
- **Attributes**, which are the values describing the properties of an entity, e.g. projectID, title, description, keywords, and objective are all attributes of a project entity. An attribute or a combination of attributes that uniquely identifies one and only one instance of an entity is called a **primary key** or **identifier**. For example, projectID uniquely identifies project.

An E-R model is easy to understand because of its graphical representation and can be directly mapped to relational database system. Often, domain (attribute) and referential integrity constraints are included when defining the entities and relationships to ensure database consistency and correctness.

3.2 Object-Oriented Model

In an object-oriented data model, information is modeled as objects, which can be any sorts of (real or abstract) entities. An object encapsulates certain related data as attributes. An object connects with other related objects via some relationships. The relationship types commonly used include *classification, association, aggregation* and *generalization*. These relationships types may in turn impose certain "object-oriented" features (such as inheritance) and integrity constraints to help maintain consistency and correctness of the data in the database.

There are two popular object-oriented data model representations: object definition language (ODL) and the unified modeling language (UML). ODL is a proposed standard language for specifying the structure of databases in object-oriented terms. The primary purpose of ODL is to allow object-oriented design of databases to be written and then translated directly into declaration of an object-oriented database system [3]. UML is the industry-standard language for specifying, visualizing, constructing, and documenting the artifacts of software systems [4]. Although UML is primarily designed for object-oriented software design, it can also be used to design object-oriented data models.

3.3 XML-based Model

The eXtensible Markup Language (XML) (see http://www.w3.org/XML/).has been fast becoming a de facto standard for data exchange because of its extendibility, hierarchical (object) structure and it vast support by computer software and hardware vendors. XML is a meta-markup language that consists of a set of rules for creating semantic tags used to describe data [5]. An XML element is made up of a start tag, an end tag, and content in between. The start and end tags describe the content within the tags, which is considered the value of the element. In addition to tags and values, attributes are provided to annotate elements. XML has wide acceptance from the computer industry, from application and database vendors such as Microsoft, AutoCAD, IBM and Oracle. XML has also been widely used for defining (product) data models. The key advantages of XML are its object-oriented structure and readability extensibility.

These are many XML-based data model representations. XML Schemas (http://www.w3.org/XML/Schema) provide a means for defining the structure, content and semantics of a data model. An XML Schema consists of components such as type definitions and element declarations. These can be used to assess the validity of well-formed attributes, and furthermore may specify default values for attributes and the types of attributes. The schema-validity assessment checks the constraints on attributes, and thus can be used to model the constraints imposed on the data model.

Resource Description Framework (RDF) (http://www.w3.org/RDF/) is a framework for describing and interchanging metadata [6]. The resource is modeled as individual objects of a data model. Properties are used to describe and define a resource. The properties of RDF can easily be expressed in XML. Figure 1 shows segment of an example RDF document.

An XML-based markup language, NEESML [7], has been developed to populate a NEESgrid metadata repository with objects and definitions of object types. NEESML is a general-purpose metadata description language that can be used to archive metadata to files. NEESML includes many features from RDF. It provides syntax for defining object types and specifying objects, the values of their properties, and the relationships between objects. NEESML can also be used to share type definitions and objects between repositories, and as an interface between other applications and repositories. There is a NEESgrid metadata ingestion tool that accepts NEESML files and uses them to popular a repository with objects and type definitions.

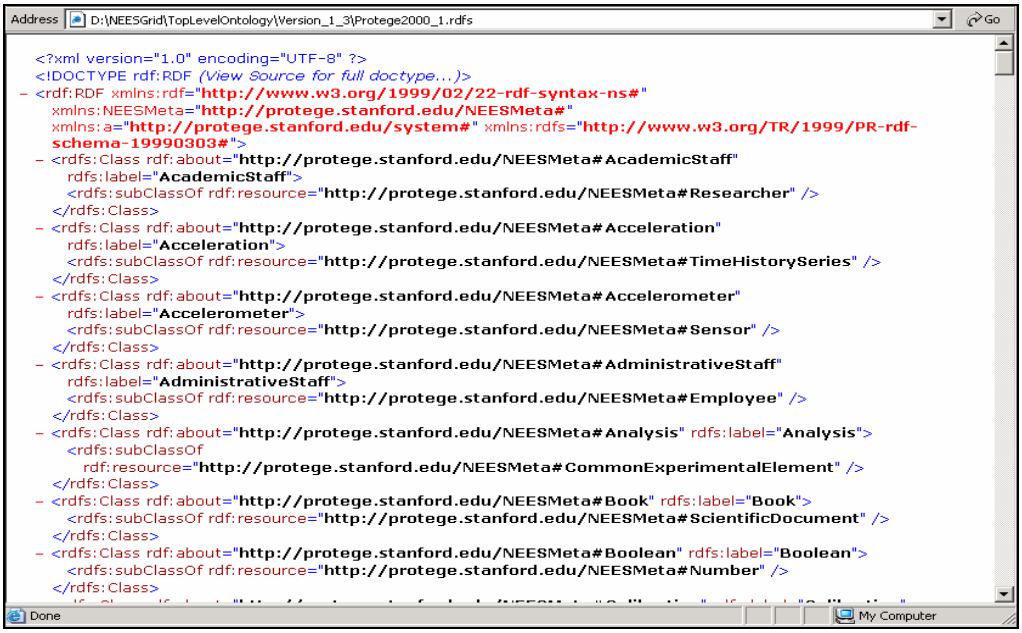

Figure 1 – Segment of an Example RDF Document

4 Data Modeling Tools

There are many data modeling or software design tools that can be used to facilitate the design of a data model for specific application.[2] In this section, we focus our discussion on Protégé-2000, which is an open-source software package designed to help developing knowledge-based systems [8].

Protégé-2000 (http://protege.stanford.edu) is a useful tool to build ontology for knowledge based systems. As an open source software, Protégé-2000 has attracted a wide variety of plug-ins from around the world to enhance its capability. Some of these software plug-ins allow a model developed in Protégé-2000 to be exported in many formats, including UML, OWL (Web Ontology Language, http://www.w3.org/2001/sw/WebOnt/), XML Schema, and RDF.

In Protégé-2000, a graphical user interface (GUI) is provided to facilitate ontology development. The interface enables the modeling of an ontology of classes to describe a particular subject with a set of concepts and their relationships. The interface also allows direct entering of specific instances of data and the creation of a knowledge base. Figure 2 shows an example of the GUI, with classes view shown in the left window, and detailed attributes view of a class (Project) shown in the lower right window.

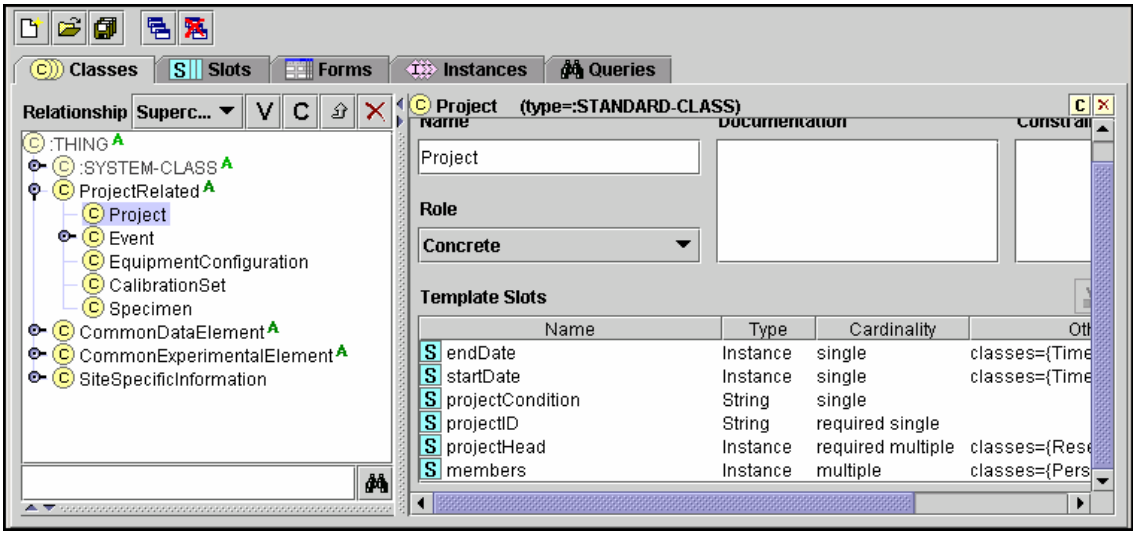

Figure 2 – Protégé-2000 Interface

[2] Examples of Data modeling tools to create ER diagrams include ER Creator (by Model Creator, Inc.), and ERwin (by Computer Associates, Inc.). Examples of tools for generating UML include Argo UML (by Tigris, Inc.) and Rational Rose (by IBM, Inc.). Direct data modeling tools for database design include DataArchitect (by Sybase, Inc.), OR-Compass (by Logic Works), and others. Several drawing packages can also be used for creating data model views, such as SmartDraw (by SmartDraw) and Visio (by Microsoft).

5 Relevant Data Models

To support NEESgrid experimentations, the envisioned data model must include different levels of information, ranging from project management, experimental models, sensor elements, etc.. There have been several relevant developments that can benefit the development of the data model for NEES experiments. The following discussion focuses on reviewing some of these models and their applicability for the data/metadata efforts.

5.1 Oregon State Model

Probably the first attempt to develop a data model for NEES was by the Oregon State University and the Northwest Alliance for Computational Science and Engineering (NACSE) for describing laboratory tsunami experiments (http://nees.orst.edu/IT/data.model/) [9]. The model contains relationships among projects, experiments, researchers, equipment, experimental results, etc. The model is designed for storing the experimental data in a relational database. Figure 3 shows a high-level E-R diagram for the data model [9]. The diagram shows that a project may have multiple experiments, an experiment may have multiple configurations, a configuration may have multiple trials, etc. The model consists of a small number of entities to make the structure of the model simple and to keep the number of tables manageable. A flexible scheme is used to assign attributes to entities. For instance, the Equipment table may include entries for any number of physical equipments used in an experiment – from strain gauges to wave basins – rather than requiring the assignment of each sensor or gauge to a particular slot. Another feature of the model is its extendibility; for instance, the model has the ability to incorporate new types of measurement instruments. The model has been tested and commented independently by researchers at University of Minnesota [10]. Presently, the model is designed primarily for tsunami wave basin experiment and it needs to be extended to other types of earthquake experiments; for example, the current model also lacks details on some common elements, such as time, location, and ground motion. The OST model does provide many insights that are valuable for the development of other NEESgrid data models.

5.2 Ontology of Science

There have been similar efforts in the Science community to develop standard model for science projects. An Ontology of Science, which is modified from the KA² ontology developed earlier by Knowledge Annotation Initiative of the Knowledge Acquisition Community [11,12]. The science ontology is available at http://protege.stanford.edu/ ontologies/ ontologyOfScience/ontology_of_science.htm.

The Ontology of Science is designed for modeling scientific events and educational events, such as a scientific conference, a research project, or a software development project. The ontology has a high-level project entity. As shown in Figure 4, the project has relationship with other entities, including people, organization, product and events. The project also has a start date and an end date, which can be used to calculate the duration of the project. Besides the high-level modeling, the Ontology of Science also provides detailed modeling of several common elements, such as people, scientific document, location, and time. The ontology provides a good organizational view that could be used to organize a collection of experimental projects.

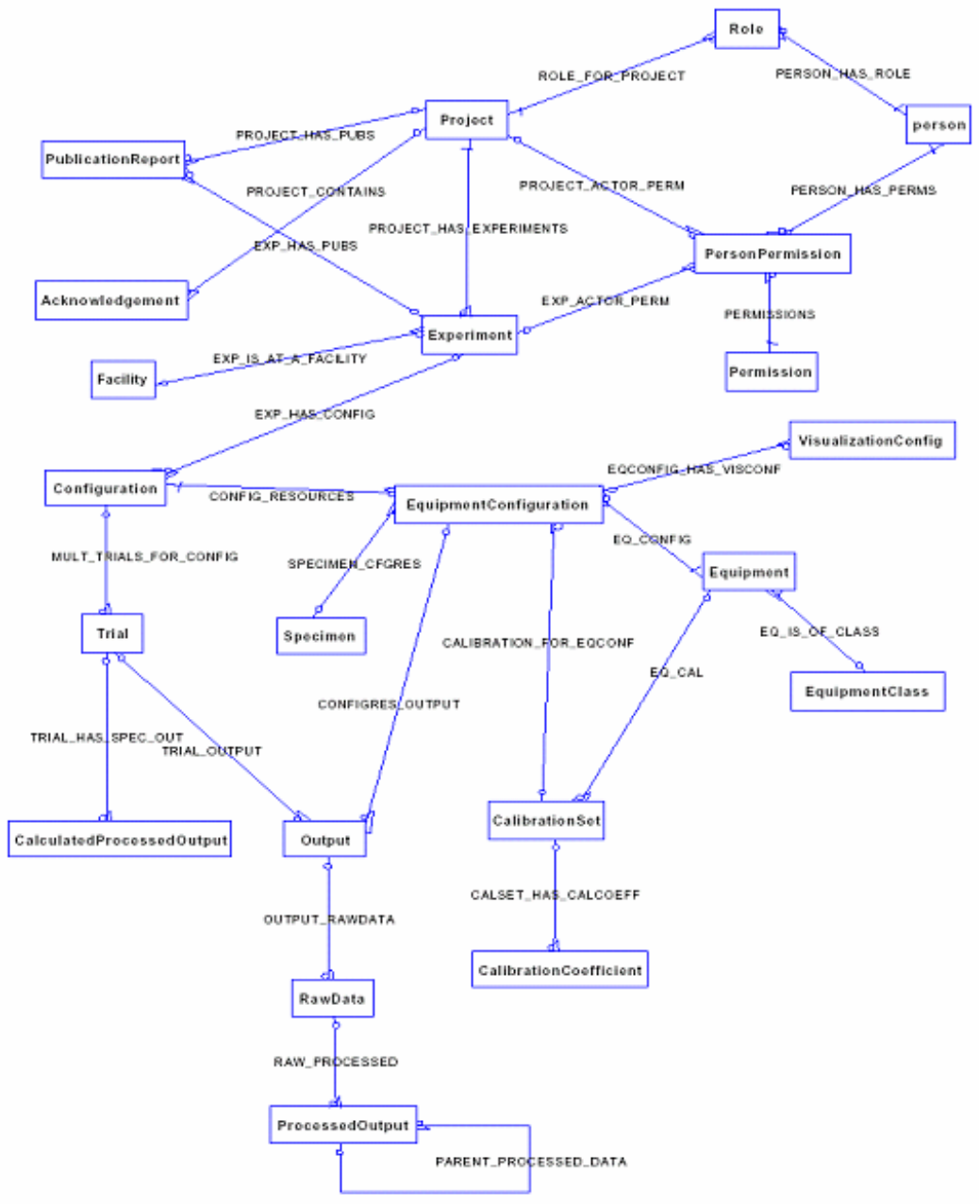

Figure 3 – E-R Diagram of the Oregon State Model [4]

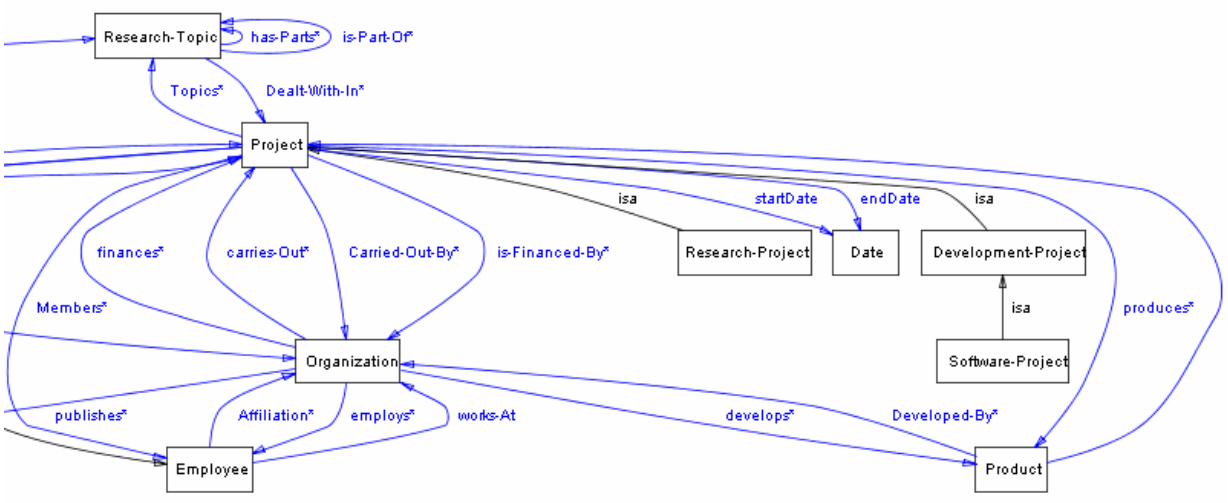

Figure 4 – Project Model of the Science Ontology

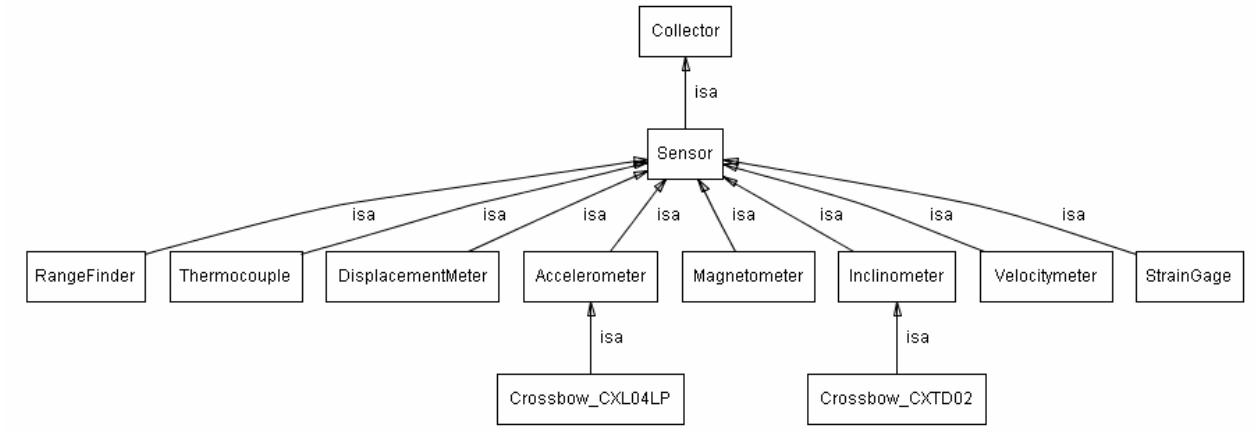

Figure 5 – Sensor Model of the Berkeley CUREE/Kajima

5.3 Berkeley CUREE/ Kajima

In the structural engineering domain, Prof. Bozidar Stojadinovic of UC Berkeley and his collaborators has created a framework for the integration and visualization of structural state data (see http://www.ce.berkeley.edu/~boza/research/Curee-Kajima/). The goal of this project is to conceptualize, develop and implement a framework for gathering, classifying and integrating structural data collected in and around a structure, and to enable effective visualization and fusion of such data to define the state of a structure.

Presently, the model focuses on the collection of observed data. The model has a comprehensive modeling on collector and data type. An abstract class, Collector, is a general object that collects or generates data about a structure. A Collector can be any type of sensor, a camera, numerical analysis, or even a person. Each collector type is capable of collecting different types of data and handling different types of input and output data. The observed

or generated data may include numerical data, descriptive textual data, or visual and graphical data (such as visual imagery from a camera, hand sketch, drawings, etc..).

The sensor model of Berkeley CUREE/Kajima provides a good representative class of different types of sensors commonly used for structural monitoring. Individual models of sensors are defined as separate objects to allow special calibration information and permits creation of separate instances of each sensor with individual serial numbers or characteristics. As shown in Figure 5, different types of sensors, such as accelerometers, strain gages, thermocouple, are modeled as subclasses of Sensor. The model has been developed using Protégé-2000 and can produce different representation formats.

5.4 Sensor ML

Sponsored by OpenGIS Consortium, SensorML (http://vast.uah.edu/SensorML/) is proposed toward preserving part of the vital sensor data required for both real-time and archival observations [13]. The purpose of SensorML is to provide general sensor information, to support the processing and analysis of sensor measurements, to describe performance characteristics, and to archive properties and assumptions regarding the sensors.

SensorML provides an XML schema for defining the geometric, dynamic, and observational characteristics of a sensor. The root for all SensorML documents is Sensor, which represents a device for the measurement of physical quantities. Key components of a sensor modeling include sensor identification, sensor location, constraints, platform attached by the sensor, coordinate reference system, sensor description, and measurement characteristics.

An important concept of SensorML is SensorGroup. A SensorGroup can be of two types, sensor package and sensor array. A sensor package is composed of multiple sensors that operate together to provide a collective observation or related group of observations. For example, a collection of sensors can be used in a combined fashion to create a sensor that measures wave velocity and direction. A sensor array is a set of sensors of the same type at different locations. These locations may be within a single sensor frame, a different location on a single mount, or on different platforms. A sensor array produces observations that are used to build a spatial coverage.

SensorML provides a good prototype and a rich model for describing sensor information. It is particularly useful for in-situ experimental applications involving spatially distributed information.

5.5 Specimen Models

There have been many attempts to construct universal data (product) model to capture detailed descriptive information about building structures. The purpose for most of the existing models is to develop data exchange and sharing standards for CAD systems.

The CIMSteel Integration Standards (CIS/2) (http://cic.nist.gov/vrml/cis2.html) is the logical product model and electronic data exchange format for structural steel project information [14]. The CIS/2 standard is based on STEP (STandard for the Exchange of Product Data [15]) and is proposed to describe structures and engineering information, testing procedures and industry specific information. CIS/2 has been implemented for describing steel framed structures, from nuts and bolts to materials, loads to frames and assemblies. Adopted by the

American Institute for Steel Construction, many steel structure software packages now provide CIS/2 import and/or export capabilities.

Another industry standards related to building structure is the Industry Foundation Classes (IFC) (http://www.eccnet.com/step/) that is developed by a consortium, the International Alliance for Interoperability (IAI), to provide data exchange and sharing capabilities for the building and construction industry [16]. The IFC is a data representation standard and file format for defining architectural and constructional CAD graphic data. The IFC uses text-based structures for storing the definitions of objects encountered in the building industry.

While various domain models have been developed for the building industry, they are either too specific (CIS/2 for steel, for example) or too general (IFC for all phases of a building project) to be useful for describing a specimen or a test model typically used in earthquake engineering experiment and simulation. Furthermore, these models could share many insights to describe a structural component, they may not necessary be useful for the experimental tests application.

6 Summary and Discussions

In this document, we provide a brief review of several data model representations, data modeling tools, and example data models. The objective is to assess their usability for defining a standard approach for the NEESgrid data/metadata effort. The data/metadata task force members have decided to produce end-to-end solutions that integrate across site specifications database, project level model, domain specific data models, and common elements. The overall data model for the NEESgrid is summarized as shown in Figure 6.

Protégé-2000 is chosen as the data modeling tool. This is not only because Protégé-2000 is open source software with extensive plug-ins, but also due to its proven ability for ontology and data modeling. Protégé-2000 provides a uniform GUI to facilitate the modeling of entities and instances. Data models developed using Protégé-2000 can easily be exported in different representational formats, such as E-R diagrams or XML-based models such as RDF. Currently, NEESML is the *de facto* standard for ingesting data to the NEES data repository. There are many similarities between RDF and NEESML. The use of Protégé-2000 to generate elements in RDF, which can then be extracted to generate NEESML documents, appears to be a natural approach to develop data model for NEES's experiments.

To develop a schema for project description and management, the model developed at Oregon State University provides a good basic model for further developments and extensions. The relationships among projects, experiments, trials, researchers, equipment, and experimental results can be retained and extended to incorporate other entities. Extensive definition of some of the entities and detailed data dictionary are needed. In addition, the project-level information can be enhanced by incorporating some of the modeling details provided by the Ontology of Science.

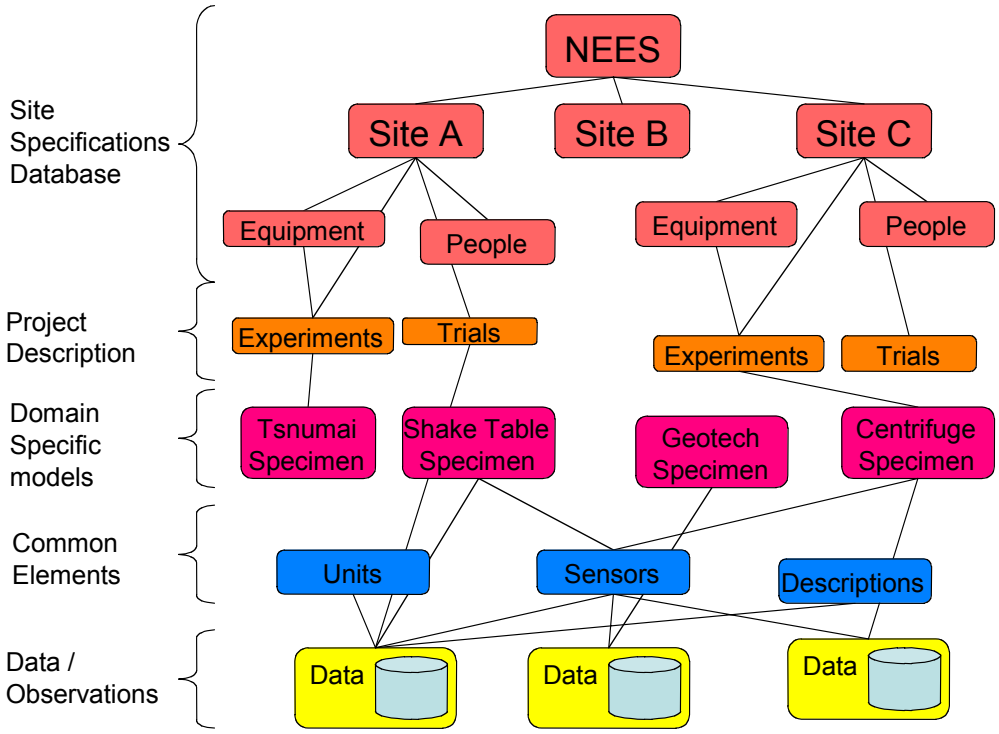

Figure 6 – Overall Data Model for NEESgrid (Courtesy of Chuck Severance)

The models for various common elements, such as units, sensors, locations and time can be extracted and extended from many existing developments. For instance, SensorML has detailed models on sensors and recorded data. The concept of SensorGroup could be useful for the NEESgrid data models because typical earthquake experiments use large number of sensors of the same type, and defining SensorGroups can avoid the duplication of sensor description. The Berkeley CUREE/ Kajima model provides a comprehensive modeling on data types, quantities, and collectors, which can be adopted in the NEESgrid data model. The Ontology of Science model has effective means of representing location and time. Effort is currently underway to consolidate some of these elements from the different models and to develop other essential elements to represent units and geometry.

Universal modeling of specimen for all experiments is very difficult if not impossible. Not only are there different types of experiments (such as shake table, pseudo-dynamic tests, centrifuge, and tsunami) and different materials (such as concrete, steel, wood, etc.), but also the geometry of specimen may be too complicated and cumbersome to model. Current standard building models are not suited for such dynamic experimental applications. Therefore, it is recommended that the development of specimen model be focused on tools and methodologies that can capture and organize CAD drawings, sketched drawings and notes, photos, narrative descriptions, electronic notes, sensor data, etc.

Last but not least, we must bear in mind that in this effort, the data models are to be designed to support the experimental tests in earthquake engineering, the needs and feedbacks from the NEES community/stakeholders are of paramount importance to the acceptance and usability

of the data models. The data models to be developed will likely be evolving as research and development of NEES projects progress and as the models are tested and validated over time.

7 Acknowledgments

This report is drafted as an interim report by the authors as part of the NEES's System Integration effort, WBS No. 2.4 Data and Metadata Management. The authors would like to acknowledge the collaboration and contributions of the NEESgrid's Data/Metadata task committee members:

Gokhan Pekcan (Task Group Coordinator)	University of Nevada at Reno
Bill Spencer	University of Illinois at Urbana-Champaign
Charles Severance	University of Michigan
Jean-Pierre Bardet	University of Southern California
Jennifer Swift	University of Southern California
Andrei Reinhorn	State University of New York at Buffalo
Lelli Van Den Einde	University of California at San Diego
Kincho H. Law	Stanford University
Jun Peng	Stanford University

The authors would like to thank Jim Eng of University of Michigan and Joe Futrelle of UIUC for their time and discussions related to NEESgrid developments. The authors would also like to thank Dr. Patrick Laplace of University of Nevada, Reno and Professors Steve Mahin, Bozidar Stojadinovic and Greg Fenves of University of California, Berkeley for their time to discuss the data and metadata issues related to earthquake engineering experiments and simulations. Any opinions, findings, and conclusions or recommendations expressed in this material are, however, those of the authors and do not necessarily reflect the views of others and the National Science Foundation.

8 References

1. G. Pekcan, Private Communication, December, 2003.
2. Chen, P.P.-S., "The Entity Relationship Model – Toward a Unified View of Data," *ACM Transactions on Database Systems,* 1(1):9-36, 1976.
3. J. Ullman and J. Widom. *A First Course in Database Systems*, Prentice Hall, Inc., 1997.
4. J. Arlow and I. Neustadt. *UML and the Unified Process: Practical Object-Oriented Analysis and Design*, Addison-Wesley Pub Co., Boston, MA, 2001.
5. M.J. Young, *Step by Step XML*, Microsoft Press, 2001.
6. Lassila and R. R. Swick (eds.). *Resource Description Framework (RDF) Model and Syntax Specification*. Recommendation, W3C, Feb. 1999. (http://www.w3.org/TR/1999/REC-rdf-syntax-19990222.)

7. J. Futrelle, *Overview of NEESML*, Technical Report NEESgrid 2003-12, v1.0 National Center for Supercomputing Applications, Urbana-Champaign, IL, 2003.. (http://www.neesgrid.org/documents/NEESgrid_TR_2003-12.pdf.)

8. J. Gennari, M. A. Musen, R. W. Fergerson, W. E. Grosso, M. Crubézy, H. Eriksson, N. F. Noy, S. W. Tu *The Evolution of Protégé: An Environment for Knowledge-Based Systems Development*., Technical Report SMI-2002-0943, Stanford Medical Informatics, Stanford University, 2002. (http://www.smi.stanford.edu/pubs/SMI_Reports/SMI-2002-0943.pdf)

9. Oregon State University and Network Alliance for Computational Science and Engineering. *NEES Database and Metadata Structure*, Version 1.3, white paper, Network for Earthquake Engineering Simulation, 2003.

10. B. Erkman, et.al., *Multi-Axial Subassemblage Testing System (MAST): Oregon State University Data and Metadata Model adapted to Quasi-Static Large-Scale Testing*, MAST Laboratory Report No. MAST-03-03, University of Minnesota, Minneapolis, Minnesota, September, 2003 (available at http://nees.umn.edu).

11. R. Benjamins, D. Fensel and A.G. Perez, "Knowledge Management through Ontologies," *Proc. Of the 2nd International Conference on Practical Aspects of Knowledge Management*, Basel, Switzerland, 1998.

12. F.L.G. Freitas and G. Bittencourt, "An Ontology-Based Architecture for Cooperative Information Agents," (available at http://www.ime.usp.br/~kvd/Otros/final.pdf)

13. M. Botts (ed.) *Sensor Model Language (SensorML) for In-situ and Remote Sensors*, OpenGIS Interoperability Program Report, OGC 02-026, Version 0.7 Open GIS Consortium Inc, 2002.. (http://vast.uah.edu/SensorML/OGC-02-026_SensorML_0.07.doc)

14. C.M. Eastman, *Building Product Models: Computer Environments, Supporting Design and Construction*, CRC Press, 1999.

15. ISO, *Product Data Representation and Exchange, Part 1: Overview and Fundamental Principles*, No. 10303-1, International Organization for Standardization, 1994.

16. IAI, *Industry Foundation Classes*, Specification Volumes 1-4, International Alliance for Interoperability, Washington, D.C., 1997.