Recommendations for Adopting External Data Models into caBIG: Tales from the Trenches

Tahsin Kurc, Patrick McConnell, Xin Zheng, and Brian Davis

VCDE and Architecture Workspace

Abstract

A number of external groups have developed and are developing standards for representation of information in their respective domains. These standards may define the structure of data elements (data objects) and/or information model(s) for the domain. There are benefits to adopting external data standards in caBIG such as bringing in established data models, facilitating interaction with legacy systems, and enabling external groups to leverage caBIG information sources. In this paper, we examine some examples of external standard adoption efforts in caBIG. We investigate the issues that were faced in these efforts, what approaches were taken, and what caBIG and external tools were employed. Based on our investigation, we see two two basic approaches to bringing an external standard into caBIG. The first is to try to model and/or semantically annotated the standard in its entirety. This generally allows for maximal interoperability with legacy systems since the standard data model is maintained. An alternative is to model a subset of the data model in UML and then model the entire data object as a single attribute. This allows for interoperability at the semantic level required by caBIG while maintaining some compatibility with legacy systems. A key conclusion learned in our effort is that when adopting a standard, the team should acquire at least a basic understanding of the usefulness of the external standard and its intended use in caBIG. They should not blindly follow onerous adoption strategies in a one-size fits all manner.

1. Introduction

A key characteristic of the caBIG program is its effort towards syntactic and semantic interoperability across resources in the caBIG environment. This is critical to achieving programmatic access to resources (syntactic interoperability), while ensuring that information exchanged between two entities can be interpreted correctly (semantic interoperability). To this end, the caBIG program is developing standards, including

- Common data elements standards: agreement and standardization on data elements that represent information captured and referenced by applications and studies,
- Vocabulary and ontology standards: agreement and standardization on basic concepts, terminologies, and definitions related to cancer research, and
- Information models standards: agreement and standardization on associations between data elements.
- Aligning with messaging standards

The caBIG community has also produced compatibility guidelines that describe the requirements for a data or analytical resource to be caBIG compatible and how they should adopt the caBIG standards[1].

Standardization efforts are not unique to the caBIG program. Several groups within the cancer research community as well as in other areas of biomedical research have developed and are developing standards to facilitate unified access to and exchange of information in their respective fields. Examples of these efforts include mzXML for proteomics data[2], BioPAX for biological pathways[3], MAGE-OM/MAGE-ML for microarray data[4], DICOM[5] for radiology, and HL7[6] for clinical, financial, and administrative information. In order to achieve a wider application of caBIG technologies and better interoperability with external groups, it is desirable that caBIG have mechanisms and tools in place to be able to adopt (and adapt) external data standards as needed.

caBIG has adopted processes to address the semantic and syntactic interoperability requirements[7]. The processes lead to a set of artifacts that are required from each software system to enable semantic and syntactic compatibility (eg, UML models, XMI files, and CDEs, all stored in the caDSR, APIs). The issue is that standards (such as standard UML models, standard domain-specific vocabulary and ontology, and data exchange standards) developed outside of caBIG (eg, mzXML, BioPAX) do not

[1]https://cabig.nci.nih.gov/guidelines_documentation/caBIGCompatGuideRev2_final.pdf
[2] http://sashimi.sourceforge.net/software_glossolalia.html
[3] http://www.biopax.org/
[4] http://www.mged.org/Workgroups/MAGE/mage-om.html
[5] http://medical.nema.org/
[6] http://www.hl7.org/
[7] https://cabig.nci.nih.gov/guidelines_documentation

necessarily construct the artifacts required for caBIG compatibility, nor is their generation trivial.

In this working group effort, we focus on adoption of external information/data standards in caBIG. *In this document, we use the term "information standard" or "data standard" to refer to standards developed by a community in order to represent information captured in a particular domain; an information standard can be used to store, search for, and exchange information in a unified way.* We do not cover messaging standards in this work. In the rest of the document, we use "external information standard" and "external standard" interchangeably. We investigate how the current set of processes and artifacts required for caBIG compatibility should be used to support the adoption of external information standards, as well as what issues remain to be addressed. We look at three example cases of adoption of external information standards, namely mzXML, BioPAX, and MAGE-OM. We present the experience and problems faced in these example cases and develop an initial set of guidelines for adoption of external standards.

1.1. Goals in Adopting External Standard Data and Information Models

There are several reasons for and benefits in adopting or adapting external standards in caBIG:

- Bringing established domain models into caBIG.

  Information modeling is not a trivial process. It requires good understanding of the domain and the data management, data access, data exchange, and data analysis requirements of domain users. Developing comprehensive data and information models to represent information captured in an application domain involves contributions of a community of users, oftentimes by forming a standards body/group. The caBIG community can leverage such standardization and modeling efforts by adopting external standards in caBIG.

- Interfacing with legacy systems.

  In many cases there are a (large) number of legacy systems developed around standard data models. These legacy systems have been in use for many years. Adoption of external standards could make it easier to interact and interface with legacy systems.

- Enabling integration of data represented in external standards and caBIG datasets.

  In most cases an external standard is used to represent information captured in a particular application domain and information model. Adoption of external standards with caBIG data models would allow researchers using external data types to semantically integrate their datasets with other datasets in caBIG.

With these motivating factors in mind, this whitepaper provides recommendations and processes to adopt standards external to caBIG.

1.2. Background

The caBIG community has developed compatibility guidelines that define the requirements for resources (e.g., applications, tools, and services) to enable and ensure compatibility with other resources. These guidelines describe four different levels of maturity (Legacy, Bronze, Silver, and Gold) that quantify the degree of syntactic and semantic interoperability of a resource[1]. Silver Level maturity is a critical requirement that significantly reduces obstacles to programmatic access to the functionality of the resource and correct interpretation of the information it serves. Silver Level maturity indicates that the resource can produce and/or consume objects, whose definitions and semantic meanings draw from information models, common data elements, and controlled vocabularies and ontologies that have been accepted by the caBIG community and are publicly available.

The compatibility process involves 1) development of structured UML domain models as described in the caCORE Software Development Kit (SDK) documentation, 2) annotation of the UML domain model definitions with publicly available vocabularies, and 3) registration of the annotated model (as XMI and CDEs) in the caDSR. In addition, XML schemas corresponding to the classes registered in the caDSR should be published in the GME so that the objects that are instances of the data types can be exchanged in the Grid environment in a caBIG compliant way (i.e., as XML documents conforming to XML schemas registered in the GME).

1.2.1. mzXML

mzXML is a data standard for encoding raw mass spectrometry data. The primary experimental data are the m/z (mass to charge ratio) and intensity values stored as a two-dimensional array that is encoded in base 64. In addition, there are a number of metadata elements related to the instrumentation used to produce the data, algorithmic processing steps, the technique to prepare the sample for injection into the mass spectrometer, instrumentation parameters used when running the experiment, and data integrity. mzData is an alternate standard to mzXML and has many of the same fields, and work is underway to merge the two standards.

mzXML has a number of characteristics that make it interesting for consideration for caBIG adoption. First, mzXML has a data model described by XML Schema but no object model. This makes it challenging to actually bring the standard into caBIG because the caBIG compatibility process requires an object model in UML to be used to register CDEs and concepts. Second, mzXML itself has a number of attributes that are semantically interesting to consumers (such as the metadata-related elements), but the actual experimental results (m/z-intensity pairs) are not semantically interesting being that they are simply a binary blob. Querying into such data in a grid setting is neither practical nor useful. Finally, though not strictly related to the caBIG compatibility process, it is interesting to note that most applications find it difficult to work with mzXML documents because of the unwieldy nature of their vast size, a typical experiment potentially generating gigabyte-size mzXML document. This should be considered in the standardization process in case a modification of the standard would make it more useable.

1.2.2. BioPAX and OWL

BioPAX (Biological Pathway Exchange) is a community effort to create a standard format for biological pathway data. The BioPAX format is defined in OWL (Web Ontology Language) and currently consists of two levels (Level 1 Ontology and Level 2 Ontology). These ontologies include support for expressing metabolic pathways, signaling pathways, protein-protein interactions, and molecular interactions. The Level 2 Ontology is the most recent release, which extends Level 1 Ontology with support for expressing molecular interactions and protein post-translational modifications. Several pathway databases and software systems use BioPAX as their data exchange format (e.g., BioCyc, Reactome, BioModels database, Cytoscape, Patika, p-tools, and PathCase).

BioPAX uses OWL to represent its Ontology. OWL is based on the RDF (Resource Definition Framework), which provides a model for objects and relationships between objects. OWL extends RDF by incorporating additional vocabulary to describe classes, properties, relations between classes, cardinality, equality, etc. Unlike XML Schema, which defines a hierarchy among objects based on parent-child relationship, OWL and RDF can describe graphs, which represent objects and relations between them. An introduction to OWL in the context of biomedical applications can be found at http://www.medicalcomputing.net/owl1.html.

1.2.3. MAGE

MicroArray and Gene Expression (MAGE) is a widely used/accepted standard for representing gene expression microarray data. MAGE was proposed in November 2000 by the Microarray Gene Expression Data (MGED) Society to facilitate the exchange of microarray information between different data systems. It was adopted by the Object Management Group (OMG) in January 2002.

MAGE consists of two primary parts: an object model (MAGE-OM) defined in UML and an XML document exchange format (MAGE-ML) defined by a DTD (Document Type Definition) derived directly from the object model. Additionally, an MGED ontology has been developed by the MGED Society to provide standard terms for the annotation of microarray experiments.

Many institutions and vendors provide MAGE-compliant microarray data management systems and data acquisition/analyzing tools. Most of these systems/tools import or export microarray data in XML format, using tags defined by MAGE-ML DTD. Many caBIG applications are designed to interface with such MAGE complaint systems. Therefore, it was very important to adopt MAGE as a caBIG data standard. The UML object model, XML data exchange format, and well defined vocabulary also made MAGE an attractive candidate for caBIG adoption, because they fit into caBIG's notion of using XML Schema, CDEs, and concepts to describe data structures and semantics.

Despite the fact that MAGE seems to have the artifacts (annotated UML model, DTD, and XML stream as message) needed to become a caBIG data standard, there are a number of challenges for MAGE to be adopted into caBIG. First, despite the fact that most of the microarray user use MAGE-ML DTD as the tool for MAGE standard, caBIG best practices require the adoption team to start with MAGE-OM object model. A MAGE-ML DTD equivalent XML schema could be generated later using the caCORE SDK for data exchange purpose. However, most microarray data are exchanged in XML files compliant with MAGE-ML DTD, therefore a separate tool has to be developed to utilizing the MAGE-ML DTD for data validation. This step is not a caBIG standard practice, but it enables caBIG to interface with other non-caBIG/legacy systems. Second, a well annotated object model with good class/attribute naming conventions is needed to create and register CDEs in the caDSR. In addition, each class must have at least one attribute. The MAGE OM has 132 classes that conform to best practice naming conventions; however, the model contains 51 classes without single attribute and 16 classes inherit from classes among the above mentioned 51 classes, resulting total 66 classes without any attribute. Furthermore, 9 MAGE OM classes were modeled with 2nd or 3rd level inheritance and the multi-level inheritance can not be handled by current caDSR loading tools. Finally, 4 MAGE-OM class attributes were not successfully registered in caDSR due to the names of those attributes, resulting in 2 MAGE-OM classes missing. As a result, 77 out of 132 MAGE-OM classes were not loaded. Some MAGE classes and attributes have their own annotation, which is different from definitions of concepts in the EVS-NCI Thesaurus (the public repository for annotating UML models). This situation leads to ambiguous definitions of similar or same objects in caBIG. If the original MAGE annotation were to be kept, the routine caBIG semantic integration practice would need to be changed.

2. Methods

Information and object models must meet the silver level compatibility requirements in order to be caBIG compliant. So, we have performed a cursory and abbreviated silver level compatibility review on each of the example cases. We investigated how the external data standards in these example cases could be adopted with existing caBIG compatibility processes. The examples represent three different classes of external standards efforts:

- mzXML is a de-facto community standard for mass-spectrometry data. It has a standard XML schema representation. It represents the class of standards where an XML schema exists.
- BioPAX is an evolving standard for representing biological pathway data. It is represented in OWL. This example represents the class of standards expressed in an ontology language.
- MAGE is a standard for MicroArray and Gene Expression data. MAGE has an XML DTD representation as well as an object model. It represents the class of standards that have an existing object model.

In these examples, we have looked at 1) how each standard was adopted and what approaches were taken, 2) what caBIG and external tools were used, and 3) what issues arose during the adoption process, which of these issues were addressed, and which issues remain to be addressed. For mzXML, which has an XML schema representation, and MAGE, which has an object model representation, we have also investigated what issues arose as far as the caBIG compatibility guidelines and best practices are concerned.

3. Results

3.1. mzXML

The goal of the mzXML standardization effort was to bring the mzXML data standard into the caBIG community. The approach taken is to create an object model that both mimics the data model and is semantically well-defined. To these ends, a number of rules for mapping XML Schema constructs to UML classes and attributes (in XMI) are defined, and a Java program is written to execute these rules (see Figure 1). The resulting XMI file is registered in the caDSR with concepts registered in the EVS- NCI Thesarus.

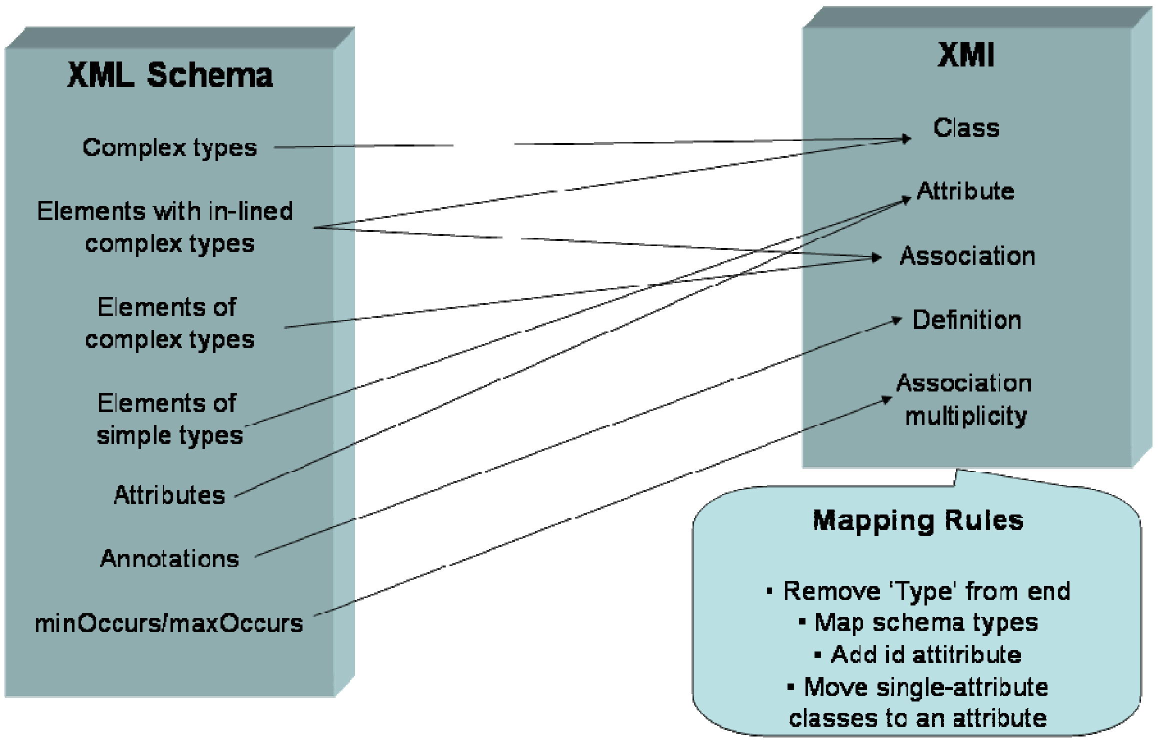

Figure 1. Summary rules used for mapping the mzXML XML Schema to an object model.

In order to minimize the impedance mismatch between the data model and the generated object model, a number of caBIG modeling best practices were broken or not fully met:

- **Empty classes**: Because of the hierarchical nature of XML, it is often the case that a complex type contains only complex types. This generates a class with only associations to other classes (and no attributes), which is not semantically useful in caBIG.
- **Naming**: There are a number of best practices related to the naming of classes, attributes, and associations that were not met because the names of types and attributes in the XML Schema were kept. These include:
    - Names should be singular, not plural
    - Attribute names should not include the name of the enclosing class
    - Abbreviations should not be used (unless widely accepted)
    - Names should have good semantic meaning
- **Cardinality**: In XML Schema, it is common for simple attributes to have a cardinality of maxOccurs to be unbounded. This is easy to represent in UML with a cardinality of *; however, this is not easy to represent in the caDSR using Java-based types. Thus, attributes with unbounded cardinality are mapped to arrays.

In order to follow a number of caBIG modeling best practices, a number of mismatches to the data model are introduced:

- **Naming**: Class names are changed to start with an upper-case letter, and attribute names were changed to start with a lower-case letter. Complex types that ended with the word type are stripped of that word. In order to meet caCORE SDK constraints, the names of associations with cardinality greater than 1 are appended with the word "Collection".
- **Data types**: A number of XML Schema data types do not exist in the caDSR and so are mapped to some other basic type that does exist (e.g. xs:duration is mapped to String). Any restrictions implemented by the extension of a simple type are lost.
- **Structure**: Classes with a single attribute are removed, and the attribute is added to the classes that associated its encapsulating class. An attribute named value is added to a class that is derived from a complex type that extends a simple type. In order to meet caCORE SDK compatibility, an attribute named id is added to each class.

The result of this mapping is an object model that is both semantically acceptable as well as close enough to the original XML Schema that data which validates to the schema can be mapped into the object model.

3.2. BioPAX

BioPAX has an OWL representation. It is expressed in XML, but has a number of specific tags to be able express classes and semantic relationships between classes.
The overall goal of the review is to determine how to represent OWL and therefore BioPAX in caBIG using the compatibility process and create artifacts (UML model, etc.).

Tools exist to create a UML model from an OWL representation. An example set of steps to convert OWL to UML is provided at http://www.biopaxwiki.org/cgi-bin/moin.cgi/Converting_OWL_To_UML. These steps are described for the Protege 2.1.2 software (an open-source ontology editor and knowledge base framework, http://protege.stanford.edu/) with a UML plug-in (http://protege.stanford.edu/plugins/uml). Protege with the UML plug-in automates MOST of the process for converting from OWL to UML; however, some manual editing of the resulting UML model needs to be done in order to remove default OWL/RDF classes created during conversion.

Manual editing of the UML model (generated from an OWL document) is needed for caBIG adoption. Some of the issues faced by the developers during the UML modeling of BioPAX are listed below. We should note that the annotation of BioPAX with caBIG data types has not been completed; that is, the models have not been loaded to caDSR yet.

1. The UML model generated from the OWL-to-UML mapping may contain associations with no directionality. Two or more associations may contain the

same source and target classes. In those cases, the associations should be edited manually to ensure that the role names are different in each association.

2. The meanings and names of classes and attributes generated through the OWL-to-UML mapping as well as those edited manually should be verified to ensure that the concepts in EVS fully and correctly describe these meanings.
3. Enumerated value domains needed to be created in the model. This was a problem in earlier versions of caCORE SDK, but the Semantic Integration Workbench (SIW) now can help with enumerated types.
4. The current version of caCORE SDK code generator cannot generate APIs and object classes from the models in the state they are generated through the OWL-to-UML mapping process. The problem is that the generated class attributes and names may contain hyphens. It is suggested that the naming guidelines defined in the Programmer's Guide be followed.
5. Each class should contain an "id" attribute, which is used by the API.

The initial effort for adoption of BioPAX aimed to register the entire BioPAX representation in caDSR. The cPath project in caBIG has taken a different approach (cPath is an open-source pathway database software and uses BioPAX as one of its data exchange formats). Their approach has stemmed from the fact that BioPAX does not have an XML schema and developing a schema for the BioPAX ontology is not trivial. Moreover, the UML model developed for BioPAX did not satisfy several of the query use cases. Thus, instead of attempting to model the entire BioPAX model, a subset of attributes and ontology elements have been selected; the subset has data elements that are more likely to be queried by clients. These selected data elements have been registered in caDSR and the corresponding XML schema has been generated. In addition to the selected subset of data elements, the registered object contains a "string data element". This data element is used to store the BioPAX document so that when a client queries the data using the harmonized subset of data elements, it can retrieve the entire pathway information represented in BioPAX ontology.

The cPath approach is acceptable as far as the compatibility requirements/guidelines are concerned. An object is registered in caDSR and clients can access a cPath data source using this registered object. However, this approach assumes the client will be able to decode and consume the "string" data element that stores the BioPAX document. The approach also limits the types of queries that can be issued against the data, since only a subset of data elements are harmonized. However, this is a good approach to quickly bring data sources that use an external standard into the caBIG environment. It reduces the effort for adoption since it involves adoption of a selected subset of the data standard instead of the entire standard.

3.3. MAGE

The motivation for introducing MAGE standard into caBIG is to enable all microarray data managing tools and analysis services to interface with other non-caBIG/legacy systems that compliant with MAGE standard. The approach is to register MAGE-OM into caDSR and generate all CDEs and XML schema necessary for semantic interoperability and data exchange. To accomplish this goal, the following steps were executed:

- MAGE OM version 1.1 was selected for caBIG adoption (XMI file can be downloaded from OMG web site http://www.omg.org/technology/documents/formal/gene_expression.htm)..
- MGED ontology, concept for describing microarray experiments, was loaded into EVS for MAGE model semantic annotation.
- In case of class inheritance, the attributes in parent class were inherited in all children classes. However, multiple-level inheritance was not handled properly.
- Annotate each MAGE-OM class attribute manually using MGED Ontology and other EVS concepts.
- Some attributes were added into original empty classes.
- Created CDE for each MAGE-OM class attribute. However, 4 attributes with name "value" were missing
- Created data model according to MAGE-OM and manually mapped the tables to classes in a XML file.
- Developed a separate application, caArray MAGE ML Loader (caAMEL), to validate the in-coming MAGE-ML file according to MAGE-ML DTD and load data into caARRAY database and create a similar web application to extract MAGE-ML file from the caBIG system.

The MAGE standard adoption team made following decisions, while registering MAGE model into caDSR to exercise the caBIG best practices. The final outcome is a altered MAGE model.

- Some CDE names are different from their corresponding attribute names.
- 66 MAGE-OM classes are not registered in caDSR, therefore can not be reused by the caBIG community.
- Attributes are added to certain classes in order to register them into caDSR.

The following issues were identified during the MAGE-OM adoption process and the team planned to reload the model to solve these issues.

- Class attributes were not inherited by grand- and great-grand-children classes. 9 MAGE classes were currently missing due to this reason and they should be recovered during next load.
- No CDE was created for attributes with name "value", even the attributes were well annotated in UML model. This may due to the limitation of the tools used for the process.
- Necessary attributes, such as ID, should be added to current empty MAGE-OM classes.

3.4. Lessons Learned

In a given external standard adoption effort, there are likely to be mismatches between the external standard representation and the caBIG object model. This should be expected as external standards bodies will develop their standards to meet the requirements of their respective communities and using the data modeling approaches accepted or employed by those communities. This is an important issue that should be taken into consideration when seeking to adopt an external standard, especially if data will be exchanged between caBIG compliant systems and legacy systems that expect data in the external standard representation. *It is important to note that a large mismatch between the external data standard and the caBIG harmonized object model may require implementation of complex transformation tools and services.*

In the case of adoption of an OWL ontology, the availability of tools that support mapping from OWL to UML reduced the amount of effort. However, these tools should be used with care, as the resulting UML output may not be compatible with caBIG tools and may require manual editing to clean up the model. In addition, care should be taken to eliminate/correct directionless associations and to ensure that class names and class attributes do not contain symbols such as hyphens, since the current set of caCORE SDK tools cannot generate code if such symbols are part of the class and attribute names.

Even with the availability of tools the overall adoption process can be lengthy because of the need to manually edit the UML output and to modify the model to harmonize with existing caBIG object models. It also may require considerable effort to generate an XML schema, if the original ontology representation did not have one. An alternative approach might be to select a subset of ontology terms, classes, and attributes (relationships) and harmonize them with caBIG common data elements. In the harmonized object model, an "external data element" entry (e.g., a String value) can be included to encode the entire result in the external data standard representation.

A critical question in any effort to adopt an external standard is to ask is how the standard is to be used in caBIG. For example, is mzXML to be used as a data exchange standard, or is it meant to provide a well structured UML model that can be re-used by other caBIG applications? is BioPAX to be used as a UML model (for object re-use), or as a standard ontology (for defining pathway information), or as data exchange standard (e.g., to facilitate interaction with external/legacy applications)? We think that there may be different artifacts and guidelines for each different type of external information standard (e.g., data exchange standard, vocabulary or ontology standard).

4. Recommendations

4.1. Incorporating an XML Schema

The result of the mzXML mapping effort is a workflow for mapping an XML Schema to a UML model (see **Error! Reference source not found.**). This workflow attempts to minimize the syntactic differences between the data and object models while maximizing the semantic quality of the object model:

1. Map XML Schema to UML based on mzXML mapping rules
2. Perform an evaluation of object model based on interoperability guidelines, best practices, and adoption

3. Modify the mapping rules and/or the model, keeping in mind the consequences of mismatch with the data model
4. Evaluate mismatch between object model and data model based on the work necessary to perform a mapping
5. If mismatch too great:
   a. Define a new data model
   b. Define a translation service between new data model and standard data model
6. If mismatch acceptable
   a. Implement translation between object model and data model
7. Register object model in the caDSR, concepts in EVS, and XML Schema(s) in GME – Please note that registration of XML schemas in GME is not required for the Silver level compatibility, but for the Gold level compatibility.

The first key decision step is 2, where the modeler must evaluate the object model. He must determine whether the model meets the caBIG semantic and syntactic interoperability guidelines and best practices. If it does not, he must either change the model directly or change the mapping rules. The second key decision step is 4, where the modeler must evaluate the mismatch between the data model and the object model. He must determine whether a translation between the models is possible or even makes sense. If not, then he must define a new data model and then create a service to translate between the data standard and the new data model. This course of action is not as desirable because caBIG tools and services will not be directly compatible with legacy applications because the translation service must be used as a go-between. In either event, the end result is a model defined in the caDSR.

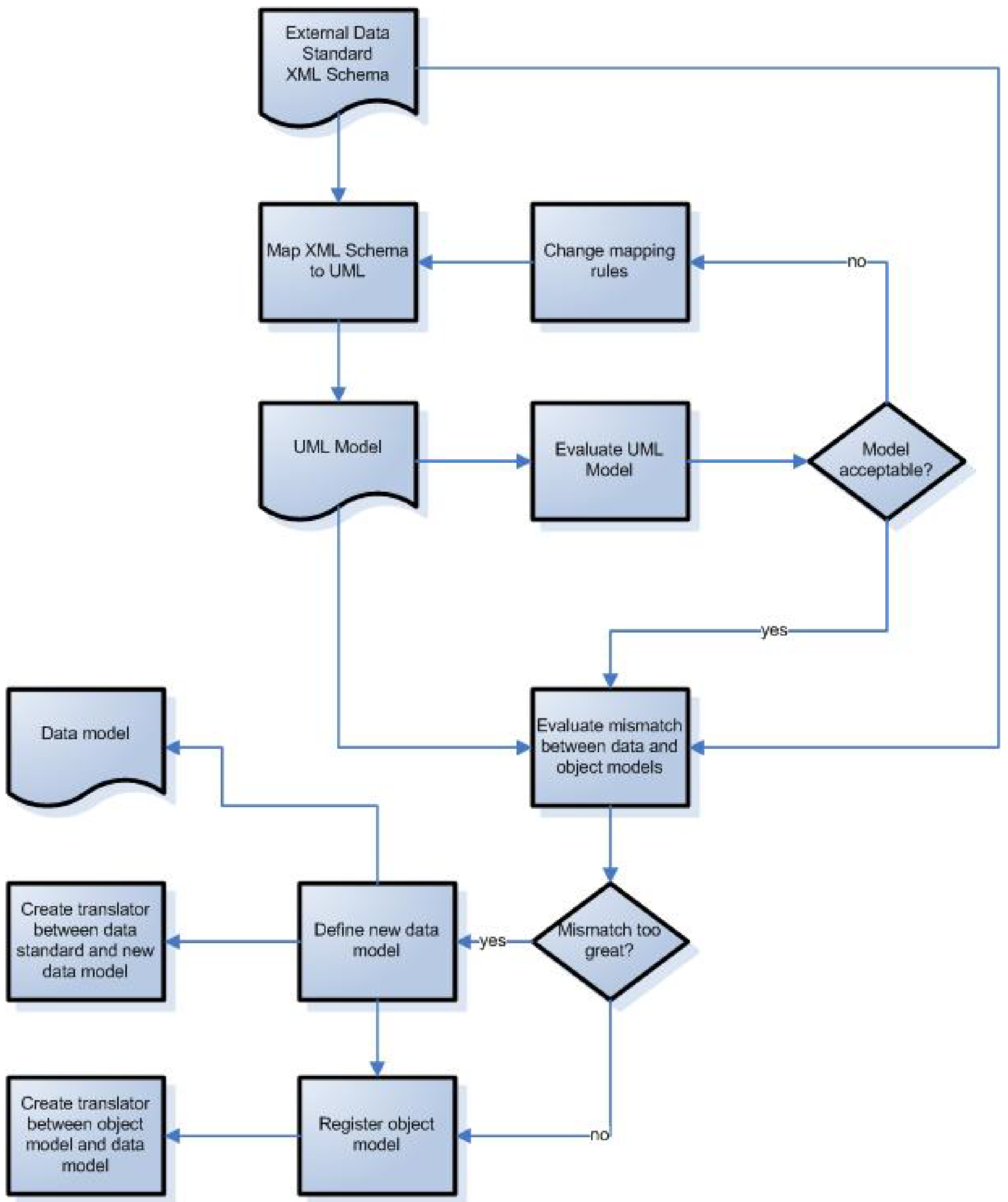
External Data Standard XML Schema
Map XML Schema to UML
Change mapping rules
no
UML Model
Evaluate UML Model
Model acceptable?
yes
Data model
Evaluate mismatch between data and object models
Create translator between data standard and new data model
Define new data model
yes
Mismatch too great?
Create translator between object model and data model
Register object model
no

Figure 2. Workflow for mapping an XML Schema to an object model. The workflow starts at the top and ends in the bottom with a number of produced artifacts and the registering of a model in the caDSR.

4.2. Incorporating an OWL ontology

1. Use existing tools to create UML models
2. Do manual editing of the UML models for semantic integration with other existing models in the caDSR.
3. Try to minimize mismatch between OWL ontology based model/representations and the model/representation for harmonization based on interoperability guidelines and best practices.
4. Be careful about associations. They should be directional.
5. Create enumerated value domains as needed based on the original OWL ontology.
6. Be careful about attribute names and class names. The caCORE SDK does not handle hyphens in names when generating code mappings.

The cPath approach described in Section 3.2 can be employed as an alternative solution (possibly as a mid-term solution).

1. It allows relatively quick adoption of an external standard.
2. It maintains the original representation through a "string" attribute.
3. However, it does not generate a representation as strongly-typed as might be desired. A client program is expected to understand the format of the information stored in the string attribute.
4. Queries are limited by the subset of data elements that are harmonized.

4.3. Incorporating an Object Model

The work flow for incorporating an existing object model into caBIG system is no different from the routine practices adopted by caBIG, except for the fact that we do not have the full freedom to modify the object model in the semantic integration steps. However, some alternations are necessary to register MAGE OM in the caDSR. The approach for adopting an existing external object model is the same as for adopting an existing external XML schema or OWL ontology: attempt to minimize the differences between the "caBIGfied" object model and the original models, while maximizing the semantic quality of the final object model. The following practices are suggested for incorporating an existing object model into caBIG:

1. Evaluate the original object model (manually or with the help of SIW) based on compatibility guidelines and best practices. Make necessary changes to ensure that the format and the quality of the model meets caBIG best practices (naming convention for ClassName, attributeName, and direction and naming of the associations). The final outcome of this step is a "caBIGfied" object model.
2. Document the discrepancy between the original object model and "caBIGfied" object model and the resulting difference in the data model.

4.4. General Recommendations

There are two basic approaches to bringing an external standard into caBIG. The first is to try to model and/or semantically annotated the standard in its entirety. This generally allows for maximal use of the standard in terms of representation of information and facilitates better interoperability with legacy systems, since the standard data model is maintained. However, one viable alternative, as demonstrated by the BioPAX effort, is to model a subset of the data model in UML and then model the entire data object as a single attribute. For example, key metadata attributes necessary to query for the object and decode the underlying data model should be modeled. Then, the data standard itself would be modeled as a single attribute with a primitive data type, such as string. This allows for interoperability at the semantic level required by caBIG while maintaining some compatibility with legacy systems. Both these approaches to modeling existing standards are not mutually exclusive; however, if they are both employed, then a translation service should be developed so that the two different models can be used interchangeably in workflows.

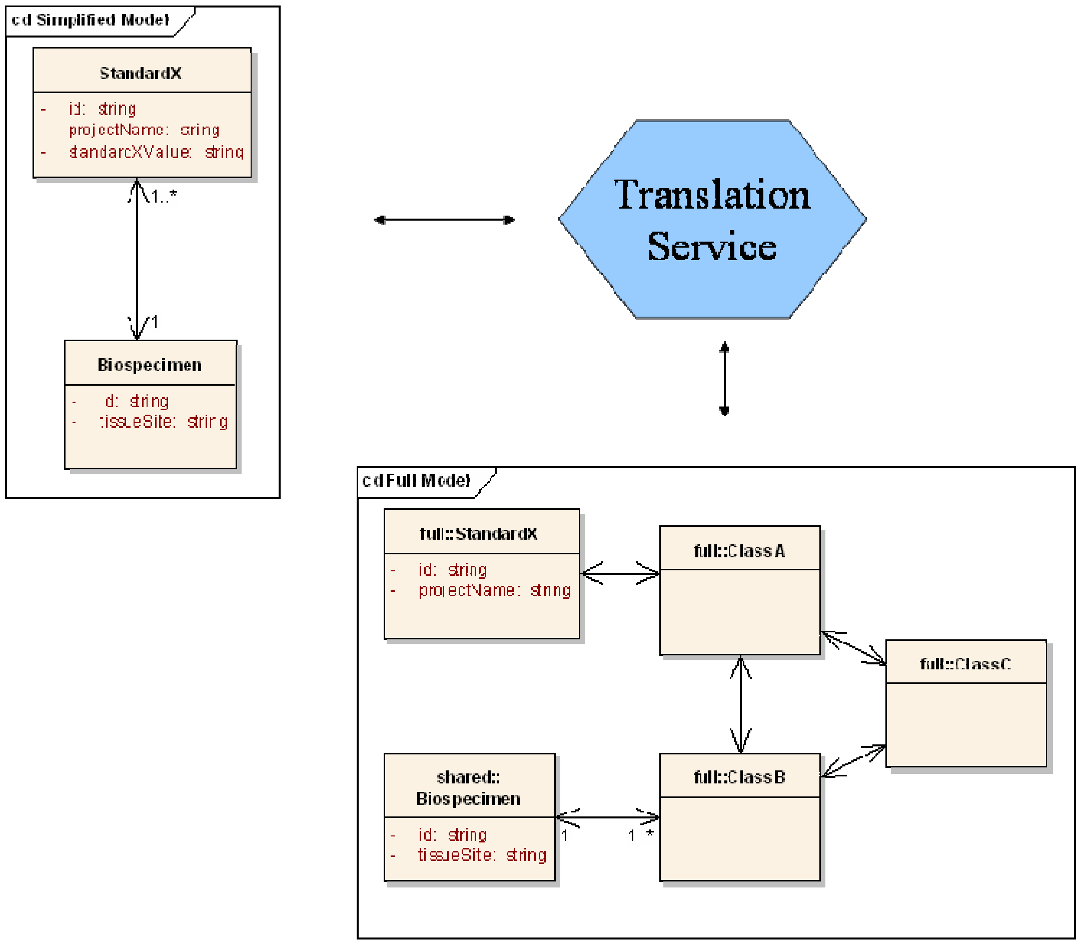

Figure 3. Example simplified and full models with translation service.

Consider an example external standard, StandardX. As illustrated in Figure 3, when adopting this standard, it is possible to generate a simplified model of the standard as a single class called StandardX with an attribute called standardXValue. The standardXValue attribute contains the actual standard encoded in a string. Other metadata attributes are also included, such as id and projectName. These along with the associated metadata object of Biospecimen can be used to provide information for querying and decoding of the adopted model. This strategy is similar to the approach taken by the cPath project in the BioPAX effort. In this case, the translation service is relatively simple, since it only needs to extract the content of the standardXValue attribute. The disadvantage of this approach is that only a subset of the standard model is made available for querying and manipulation by caBIG clients. The alternate strategy is to model the entire standard in an object model. The root class here happens to be StandardX, and associations to ClassA, ClassB, ClassC, and Biospecimen exist, which comprise the actual data elements found in the standard. This strategy allows us to capture the entire information model represented by the standard. However, it is possible that the adopted model as registered in caDSR/EVS will not match the external standard, due to modifications for harmonization with other existing caBIG common data elements and controlled vocabularies. In that case, the translation service will likely be more complicated, since it now needs to operate on a finer level of mapping (e.g., at the attribute level) between the original standard and the adopted model.

4.5. Recommendations for Possible Infrastructure Extensions/Enhancements

A number of tooling and infrastructure extensions and enhancements could be made to ease the adoption of external information standards into caBIG. One key area of enhancement is in the tooling for conversion of external standards to UML. For example, tools were created to convert the XML Schema for mzXML into a caBIG-compatible UML. This tool could be extended for more general use. There are a number of tools for converting OWL into UML, but currently they all require manual editing to meet caBIG modeling best practices. These open source tools could be enhanced to include many of the caBIG criteria used for evaluating compatibility. At the infrastructure level, enhancements to more natively support translation could be created. For example, there is an application currently being developed named GeneConnect that builds upon the caGrid data service infrastructure and will translate between the various genomic identifiers. This idea could be extended to translating between models, in addition to identifiers. For example, a translation service framework could be developed to support translation between the model in caBIG of an adopted external standard and the original external standard, if the adoption process modifies the external standard.

5. Conclusions

In this paper, we have examined three example efforts to adopt external information standards into caBIG. These efforts span a range of standards, including one based on XML schemas, one based on OWL ontology representation, and one with an object-model.

A key conclusion and lesson learned is that when adopting a standard, the team should acquire at least a basic understanding of the usefulness of the external standard and its intended use in caBIG. An adoption effort should be examined in the context of use cases, which are intended to be addressed by adopting the external standard.

Our investigation of these examples showed that there exist caBIG tools as well as external tools that can help reduce the effort in the adoption process. However, it still takes considerable effort to adopt complex, large standards. The potential mismatch between the external standard and the caBIG harmonized object model should also be taken into account when adopting the external standard, as mapping between the caBIG object representation and the external standard representation may require support for complex translation tools and services.

A feasible, albeit less than ideal, approach is to adopt only a portion of the standard and include a data element (referred to here as an "external data element") attribute in the resulting object model to store the data in the external data standard form as well. The external data element attribute can be used to communicate data between caBIG and legacy applications that expect external data representation, without needing to implement a translation tool or service. If the portion of the external data standard to be adopted is carefully chosen, a sufficient amount of information encoded in the external data representation can be made available to caBIG clients.