Unifying Data Model - Design Principles

Design Principles

The building blocks of the Ed-Fi Unifying Data Model are entities (classes), attributes, and relationships (associations). Even with such simple building blocks there are a multitude of design choices. The following design principles were used in creating the Ed-Fi Unifying Data Model.

Core Principles

The Ed-Fi Unifying Data Model embodies the following core principles:

- Is easily understood
- Is independent of any specific application or physical implementation
- Implements a common core that provides consistency across applications
- Easily extends and accommodates change and evolution
- Logically separates the data from the handler mechanisms and implementation-specific information

Additional Principles

The Ed-Fi Data Standard was developed in accordance with the following key principles.

Stay Within the Defined Scope

The following questions are used to select what is included in the Ed-Fi Data Model:

- Does the entity, attribute, or relationship exist within the defined K–12 education scope?
- Is the entity, attribute, or relationship important for a specific educational purpose?
- Is the entity, attribute, or relationship available to be collected?
- Under what scenarios would the entity, attribute, or relationship be shared or exchanged?

When an education data element is considered, it is not included in the Ed-Fi Data Model without some evidence that:

- The data element is being electronically captured in an application or information system used by an education organization.
- The data element has a use that requires sharing or interchange, such as for dashboard visualizations or state/federal reporting.

Consider the “Natural State and Structure” of the Data

The Ed-Fi Data Model is constructed with easily understood semantics:

- Entities are naturally the most important “things” in the domain that require representation in data.
- Attributes naturally identify, describe, characterize, or classify entities.
- Relationships (associations) between entities are not transient and typically persist over time.

Include Generalizations Sparingly

The Ed-Fi data Model includes generalizations that contribute critical inheritances and represent important generalization concepts in the domain. The following questions are used to determine whether a generalization should be included:

- Is the generalization a commonly used term within the domain?
- Does the generalized entity have common important attributes that should be inherited?
- Does the generalized entity have associations that should be inherited?
- Are all of the specialized entities (subclasses) of common purpose and structure?
- Do source systems commonly identify the lower-level abstractions or store and display them in the same structure and context?

The NEDM model has a rich taxonomy of generalizations. However, most of these generalizations are not important when creating a reference model used for data storage, exchange, visualization, and reporting. For example, the entities for TeachingLearningResource, Assessment, and AssessmentResult are subclasses of the generalized InstructionArtifact. Note that the subclasses are of different purpose. Further examination reveals that the entities do not share common attributes or relationships outside of perhaps a name, description, and ID. As a result, we conclude that TeachingLearningResource is a poor choice for the reference model.
By contrast, the generalization—EducationOrganization—has both important attributes and associations that are meaningfully inherited by School (campus), LEA (district), and Regional Educational Service Center. Note that all are of a similar common purpose and structure.

Create Classes to Abstract Cohesive Groups of Attributes

To improve understandability, the Unifying Data Model abstracts cohesive collections of attributes into classes. These become complex types in the Ed-Fi XML Schema. For example, a student’s name could be represented as a flat structure with attributes FirstName, MiddleName, LastName, and NameSuffix. A new class (in XML, a new complex type) is created for Name that includes the attributes above. This reduces the number of attributes directly shown for Student without loss of understandability.

Examples where the Unifying Data Model applies this technique are as follows:

- Address
- BirthData
- Telephone

Create Classes to Group Attributes that Are Multi-Valued Records

Some attributes are expressed as multi-valued records rather than create a separate domain class, define the record attributes as a separate class, and then reference a multi-valued attribute. These become complex types in the Ed-Fi XML Schema. For example, students may have multiple disabilities with each specified by a disability type, a description, and an order of severity (primary or secondary).

Examples where the Unifying Data Model applies this technique are as follows:

- Disability
- EmploymentPeriod

Consider the Direction of Associations in the Context of Possible Interchanges

When associations do not have their own attributes, the ReferenceType embodying the association is contained in the complex type for the source entity. It is important to consider which entity is the source for the association, based on its likely use in the interchange.

Consider the associations for the AttendanceEvent shown in the diagram below.

![AttendanceEvent Associations](image)

As shown, the association relating the student to the attendance event is contained in the source AttendanceEvent. It is reasonable to assume that there could be an interchange loading only attendance events. Note that if the direction of this association were reversed, then the loading of attendance would always need to be accompanied by a load of Students. This same rationale applies to the Section and Program associations.

Create New Descriptor Entities for Important Context-Specific Enumerations
There are cases where enumerated values cannot be standardized and are determined by their context, these are referred to as *Descriptors* in the Ed-Fi Data Model. For example, the PerformanceLevel values (e.g., Met Standard, Commended, College Ready) for an Assessment, which are typically determined by meeting cut scores, are custom to each Assessment and cannot be standardized. The PerformanceLevelDescriptor holds the CodeValue and Description for each of the performance-level enumerations specific to an Assessment as well as other important attributes. The StudentAssessment attribute PerformanceLevel contains references to the predefined performance levels in the PerformanceLevelDescriptor.

**Use Real Data to Validate the Practicality of the Ed-Fi Unifying Data Model**

Data profiling is a recognized technique for designing database schemas and data warehouse structures. Similarly, analyzing real data as it is housed in student information and other education systems provides significant validation of the Ed-Fi Unifying Data Model.

In addition, the process and rigor of representing the UDM as both an XML schema and a database schema significantly adds to the depth and quality of the model.