Ed-Fi API – Vendor Implementation Models

Overview

Audience assumptions

• Familiarity with K-12 education data (e.g., Student Information System, Assessments)
• Exploring or planning an Ed-Fi API Client implementation
• Using or updating an existing Ed-Fi API Client implementation
• Basic understanding of database concepts, APIs and data transmission formats (e.g., JSON).

Goals

• Understand the basic conceptual framework of the Ed-Fi data model.
• Serve as a fundamental framework prior to starting an Ed-Fi API Client implementation.
• Understand tools and resources available from Ed-Fi to help with further understanding the data standard, accessing and implementing the Ed-Fi Client API, and other useful tools.
Ed-Fi API – Vendor Implementation

Topic Overview

• Getting to know the Ed-Fi Data Standard
• Types and Descriptors (Enumerations)
• Accessing the Ed-Fi APIs (API Client)
• Implementing an Ed-Fi Aligned API (API Endpoints)
• Helpful Tools
• Additional Items
Getting to know the Ed-Fi Data Standard
Ed-Fi Data Standard – UDM

The foundation of the Ed-Fi Data Standard is a Unifying Data Model (UDM), which is an enterprise data model of commonly exchanged and shared K–12 education data. The model is expressed in Unified Modeling Language (UML), and includes entities such as students, teachers, assessment results, attendance, and many others. The role of the Ed-Fi UDM is to harmonize the information model and data types across all facets of Ed-Fi technology.
Ed-Fi Data Standard - UDM

• The purpose of the Ed-Fi UDM is to enable information sharing and reuse of education data. The Ed-Fi UDM provides a standard means for:
  • Data Description - Provides a uniform means to describe education data, its structure (syntax), and meaning (semantics), thereby supporting its discovery and sharing.
  • Data Context - Facilitates discovery based upon the definition of specific education data domains.
  • Data Sharing - Supports the access and exchange of data based upon concrete implementation of the Ed-Fi UDM or subsets of the Ed-Fi UDM that provide semantic interoperability between systems.
  • Data Unification or “Harmonization” - Provides the capability to compare education data artifacts across systems through a well-defined model that unifies the semantics of data artifacts into “common entities.”
Ed-Fi Data Standard - Domains

The scope of education data is large and its organization is complex.

Domains serve to provide views of the Ed-Fi Unifying Data Model to assist in its understanding and its application.

In many cases, a specific Ed-Fi data exchange schema may only deal with data in a single domain, such as assessment or enrollment. In other cases, it may span several domains, such as with a student transfer record.
Ed-Fi Data Standard - Domains

Domains

The Ed-Fi UDM is organized into domains. Domains serve to provide overlapping views of the data model and assist in its understanding and application. The following are the Ed-Fi Data Standard v2.1 domains:

- Alternative/Supplemental Services
- Assessment
- Bell Schedule
- Discipline
- Education Organization
- Enrollment
- Finance
- Graduation
- Intervention
- School Calendar
- Staff
- Student Academic Record
- Student Attendance
- Student Cohort
- Student Identification and Demographics
- Teaching and Learning
Domain Example: Assessment

Excerpt from the Assessment Domain UDM Model v2.1

Overview

The Assessment domain deals with the structure of assessments, the educational standards and objectives measured by assessments, and assessment results. The assessment model is suitable for a wide variety of assessments, including early childhood learning assessments, complex tests like state standardized assessments and college entrance exams, and simple tests like benchmark assessments and course assessments.

Key Entities | Key Concepts | Model Reference | Example Usage

Assessment Model Entities

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assessment</td>
<td>This entity represents a tool, instrument, process, or exhibition composed of a systematic sampling of behavior for measuring a student's competence, knowledge, skills, or behavior. An assessment can be used to measure differences in individuals or groups and changes in performance from one occasion to the next.</td>
</tr>
<tr>
<td>AssessmentItem</td>
<td>This entity represents one of many single measures that make up an assessment.</td>
</tr>
<tr>
<td>LearningObjective</td>
<td>This entity represents identified learning objectives for courses in specific grades.</td>
</tr>
<tr>
<td>LearningStandard</td>
<td>This entity is a sub-element of a learning objective consisting of a precise statement of the expectation of a student's proficiency.</td>
</tr>
</tbody>
</table>
Partial UDM Model: Objective Assessment Entity

This shows the UDM model on top, with the corresponding JSON below. Additionally, the Ed-Fi ODS/API Documentation is included next to the ObjectiveAssessment Entity.

**JSON for the ObjectiveAssessment Entity**

```json
objectiveAssessment {
  "id": "691deb9fc3fd4165bdb15b6e21568ba3",
  "assessmentReference": {
    "academicSubjectDescriptor": "English Language Arts",
    "assessedGradeLevelDescriptor": "Eleventh grade",
    "title": "State Assessment",
    "version": 2007,
  },
  "identificationCode": "State Assessment_ELA11-1",
  "maxRawScore": 8,
  "assessmentItems": [],
  "learningObjectives": [ {
    "learningObjectiveReference": {
      "objective": "Basic Understanding (Reading)"
    }
  } ]
}
```

**JSON for the LearningObjective Entity**

```json
{ "id": "e970a9128f344dc91d62e50e7d8148",
  "academicSubjectDescriptor": "English Language Arts",
  "objective": "Basic Understanding (Reading)",
  "objectiveGradeLevelDescriptor": "Eleventh grade",
  "description": "The student will demonstrate a basic understanding of culturally diverse written texts.",
  "learningObjectiveId": "27",
  "namespace": "http://ed-fi.org/LearningObjective/LearningObjective.xml",
  "learningStandards": []
}
```
Full Objective Assessment Entity in JSON

```json
{
    "id": "691deb9fc3fd4165bdb15b6e21568ba3",
    "assessmentReference": {
        "academicSubjectDescriptor": "English Language Arts",
        "assessedGradeLevelDescriptor": "Eleventh grade",
        "title": "State Assessment",
        "version": 2007,
        "link": {
            "rel": "Assessment",
            "href": "/assessments?academicSubjectDescriptor=English+Language+Arts&assessedGradeLevelDescriptor=Eleventh+grade&title=State+Assessment&version=2007"
        }
    },
    "identificationCode": "State Assessment_ELA11-1",
    "maxRawScore": 8,
    "assessmentItems": [],
    "learningObjectives": [
        {
            "learningObjectiveReference": {
                "objective": "Basic Understanding (Reading)",
                "link": {
                    "rel": "LearningObjective",
                    "href": "/learningObjectives?academicSubjectDescriptor=English+Language+Arts&objective=Basic+Understanding+(Reading)&objectiveGradeLevelDescriptor=Eleventh+grade"
                }
            }
        }
    ],
    "learningStandards": [],
    "performanceLevels": [],
    "_etag": "635696347127000000"
}
```
Types and Descriptors (Enumerations)
Types and Descriptors

- **Types** are intended to be base categories that should not be altered.
- **Descriptors** are a way to further enumerate a type.
- **Descriptors** can be added.
- **Descriptors** require a Namespace (preferably an URI).
Types and Descriptors

• The following are different kinds of descriptors:
  • Descriptors that **must** map back to a type
    • GradeLevelDescriptor
    • AcademicSubjectDescriptor
  • Descriptors that **may** map back to a type
    • LanguageDescriptor
    • TeachingCredentialDescriptor
  • Descriptors that **can not** map back to a type
    • AccountCodeDescriptor
    • AssessmentPeriodDescriptor
Descriptors that **must** map back to a type

Example: gradeLevelDescriptors

---

### gradeLevelDescriptors

This descriptor defines the set of grade levels. The map to known Ed-Fi enumeration values is required.

**GET**

`/gradeLevelDescriptors`

Retrieves resources based on paging capabilities (using the "Get All" pattern).

**Implementation Notes**

This GET operation provides access to resources using the "Get All" pattern. In this version of the API there is support for paging.

**Response Class (Status 200)**

The matching resources were successfully retrieved. If no instances are found will return an empty collection.

**Model**

Example Value

```
gradeLevelDescriptor {
  id (string): The unique identifier of the resource.
  codeValue (string): A code or abbreviation that is used to refer to the descriptor.
  description (string, optional): The description of the descriptor.
  effectiveBeginDate (datetime, optional): The beginning date of the period when the descriptor is in effect. If omitted, the default is immediate effectiveness.
  effectiveEndDate (datetime, optional): The end date of the period when the descriptor is in effect.
  gradeLevelDescriptorId (integer, optional): A unique identifier used as Primary Key, not derived from business logic, when acting as Foreign Key, references the parent table.
  gradeLevelType (string): Key for GradeLevel.
  namespace (string): A globally unique namespace that identifies this descriptor set. Author is strongly encouraged to use the Universal Resource Identifier (http://id, etc.) for the source of the descriptor definition. Best practice is for this source to be the descriptor file itself so that it can be machine-readable and be fetched in real-time, if necessary.
  priorDescriptorId (integer, optional): A unique identifier used as Primary Key, not derived from business logic, when acting as Foreign Key, references the parent table.
  shortDescription (string): A shortened description for the descriptor.
  _etag (string, optional): A unique system generated value that identifies the version of the resource.
}
```

---

{  
  "id": "f43f8860c13c485a86693a511b216ab9",
  "gradeLevelDescriptorId": 1,
  "codeValue": "Eleventh grade",
  "description": "Eleventh grade",
  "namespace": "http://ed-fi.org/Descriptor/GradeLevelDescriptor.xml",
  "shortDescription": "Eleventh grade",
  "gradeLevelType": "Eleventh grade",
  "_etag": "635696150763500000"
}
Descriptors that **may** map back to a type

Example: `languageDescriptors`

```
languageDescriptors
This descriptor defines the language(s) that are spoken or written.

GET /languageDescriptors
Retrieves resources based on the GET pattern. Use the "Get All" pattern.

Implementation Notes
This GET operation supports access to resources using the "Get All" pattern. In this version of the API, support for paging is provided.

Response Class (Status 200)
The matching resources were successfully retrieved. If no instances are found, an empty collection is returned.

Example Value
```
```
Descriptors that **cannot** map back to a type

Example: accountCodeDescriptors

```json
{
  "id": "e78a7fd502a4421e9f851adb06885b76",
  "accountCodeDescriptorId": 50,
  "codeValue": "Account Code2",
  "description": "Account Code2",
  "shortDescription": "Account Code2",
  "accountCodeCategory": "Account Code 2",
  "beginDate": "2012-08-01T00:00:00",
  "endDate": "2012-12-31T00:00:00",
  "_etag": "635696150764600000"
}
```
Types and Descriptors

• There are some Types that do not have a Descriptor
  • Example: AcademicHonorCategoryTypes

• Usually **Types** can be thought of as categories and **Descriptors** as enumerations for those categories
  • Academic Subject Area **Type** of mathematics could be considered a **category**
  • Academic Subject Area Descriptors for the Academic Subject Area Type of Mathematics could be enumerations of:
    • Algebra I
    • Calculus
    • Geometry
    • Trigonometry
    • Etc.
Types and Descriptors: Best Practices

• When accessing or implementing Ed-Fi APIs
  • Do a gap analysis between the core Ed-Fi Descriptors versus the implementation specific code value sets
  • Try to minimize the addition of descriptors
  • Determine a namespace (preferably a URI) and host a list of those descriptors for the implementation at that URI

• The Ed-Fi Enumeration Special Interest Group (SIG) has additional resources and updates on best practices
  • https://techdocs.ed-fi.org/pages/viewpage.action?pageId=28868952
In the Ed-Fi ODS/API Documentation, select Resources, then in the top drop down, select Descriptors. You will then be able to execute a the GET API call on a descriptor resource.
Getting to know the Ed-Fi Data Standard
Links to helpful Ed-Fi resources

- **Ed-Fi Data Standard v2.0**
  - [Unifying Data Model](#)
  - [Ed-Fi API Documentation Site](#) (Swagger UI)

- **Ed-Fi Data Standard v2.1**
  - Released June 2017
  - [Unifying Data Model](#)
    - [v2.1 Model Reference](#)
  - Ed-Fi API Documentation Site (Coming Soon)

- **What’s New in v2.1**
- **Ed-Fi Tracker**
  - Issue tracking system
  - [Introduction to Ed-Fi Tracker](#)
Accessing Ed-Fi API
Implementing an API Client (API Client)
Accessing Ed-Fi ODS/API

Topic Overview

- Accessing Ed-Fi APIs is also referred to as developing an API client
  - API Client Developers’ Guide
- Useful Considerations
  - Dependencies
  - Suggested Phases
  - Surfacing Errors
- Profiles
- Composites
- Helpful Tools
API Client Developer’s Guide

This documentation provides essential information for building client applications that interact with an Ed-Fi ODS / API platform. It includes instructions on how to set up and build a client quickly using the Client SDKs for Java and C#.

**API Client Developers' Guide** Content Overview

- Introduction
- Basics
- Authentication
- Authorization
- Using Code Generation to Create an SDK
- Using the Online Documentation
- Using the Sandbox Administration Portal
- More Information
Useful Considerations

- **Dependency Diagram**
Useful Considerations

Dependencies: Suggested Phases by Resource

• **Phase 1**
  - State Education Agency
  - Education Organization Network
  - Education Service Center
  - Local Education Agency
  - School
  - Feeder School Information
  - Course
  - Session
  - Grading Period
  - Calendar Date
  - Academic Week
  - Course Offering
  - Section
  - Bell Schedule
  - Student
  - Parent
  - Student Parent Association
  - Student School Association
  - Student Section Association
  - Graduation Plan

• **Phase 2**
  - Staff
  - Staff Education Organization Association
  - Staff Education Organization Employment Association
  - Staff Education Organization Assignment Association
  - Staff School Association
  - Staff Section Association

• **Phase 3**
  - Report Card
  - StudentCompetencyObjective
  - StudentLearningObjective

• **Phase 3**
  - DisciplineIncident
  - StudentDisciplineIncidentAssociation
  - DisciplineAction
  - Student Program Association
  - StudentSpecialEducationProgramAssociation
  - StudentSpecialEducationProgramAssociation
  - RestraintEvent
  - StudentCTEPProgramAssociation
  - StudentTitleIPartAProgramAssociation
  - StudentMigrantEducationProgramAssociation

• **Phase 4**
  - StudentInterventionAttendanceEvent
  - StudentProgramAttendanceEvent
  - StudentSchoolAttendanceEvent
  - StudentSectionAttendanceEvent
  - SectionAttendanceTakenEvent

• **Phase 5**
  - StudentAcademicRecord
  - CourseTranscript
Useful Considerations
Surfacing Errors

• Best practices when implementing an Ed-Fi ODS/API Client
  • Capture errors received from the API and surface those errors through the vendor application to a vendor system user
  • Capture all contextual and potentially applicable information in addition to the actual error message

• Error Handling & Best Practices – Additional best practices when developing an Ed-Fi API Client
Profiles

• **API Profiles**
  • An API Profile enables the creation of a data policy for a particular set of API Resources, generally in support of a specific usage scenario (such as for Nutrition or Special Education specialty applications).

• **API Client Developer’s Guide - Authorization**
  • API Profiles enable an Ed-Fi ODS / API platform host to constrain the data exposed from specific resources to meet the needs of individual use cases.
Profiles

- **API Client Developer’s Guide – Authorization**
  - To use a profile, callers must add media type information to their requests
    - For read operations, this takes the form of an Accept header, shown below, which indicates to the server that the caller will accept the profile-based version of the resource.
      - Accept: application/vnd.ed-fi.student.nutrition.readable+json
    - For write operations, the Content-Type header is used
      - Content-Type: application/vnd.ed-fi.student.nutrition.readable+json
This shows how a profile created through inclusion or exclusion can return a subset of elements in the end message.
Composites

- **API Composite Resources**
  - An API composite resource definition enables the Ed-Fi ODS / API to provide subject-oriented data from multiple API resources in a single request, resulting in a simple data integration experience for API consumers.
  - Composites that currently “ship” with the Ed-Fi ODS/API
    - Assessments API
    - Enrollment API
Implementing Ed-Fi Aligned APIs (API Endpoints)
Ed-Fi ODS/API Documentation - Swagger UI

Resources

Resources are the primary entities that most API client applications work with on a regular basis. Students, staff, education organizations, and their related entities are maintained using this area of the API.

Composites

Composites are read-only combinations of resources that address specific business use cases. Using composites reduces the number of calls that an API client application must make to retrieve resources for these use cases.

For example, the Enrolment composite can return all the students in a given class section or all students associated with a particular staff member, and similar student association-related information in a single call. Similarly, the Assessment composite can return all assessments associated with a school, or all assessments associated with a student in a single call. Platform hosts may add their own composite resources.
Ed-Fi ODS/API Documentation

The Ed-Fi ODS/API is divided into several areas by function.

**Resources**

Resources are the primary entities that most API client applications work with on a regular basis. Students, staff, education organizations, and their related entities are maintained using this area of the API.

**Composites**

Composites are read-only combinations of resources that address specific business use cases. Using composites reduces the number of calls that an API client application must make to retrieve resources for these use cases.

For example, the Enrollment composite can return all the students in a given class section or all students associated with a particular staff member, and similar student association-related information in a single call. Similarly, the Assessment composite can return all assessments associated with a school, or all assessments associated with a student in a single call. Platform hosts may add their own composite resources.

Ed-Fi Operational Data Store API

The Ed-Fi ODS API enables applications to read and write education data stored in an Ed-Fi ODS through a secure REST interface. The Ed-Fi ODS API supports both transactional and bulk modes of operation.

**academicWeeks**

This entity represents the academic weeks for a school year, optionally captured to support analyses.

**accounts**

This financial entity represents a funding source combined with its purpose and type of transaction. It provides a formal record of the debits and credits relating to the specific account.

**accountabilityRatings**

An accountability rating for a school or district.

**actuals**

This financial entity represents the sum of the financial transactions to date relating to a specific account.

**assessments**

This entity represents a tool, instrument, process, or exhibition composed of a systematic sampling of behavior for measuring a student's competence, knowledge, skills, or behavior. An assessment can be used to measure differences in individuals or groups and changes in performance from one occasion to the next.

**assessmentFamilies**

This entity represents a logical grouping or association of assessments that share a common purpose, heritage, or content standard. There may be hierarchies of assessment families, characteristics (e.g., Academic Subject) specified for assessment families, by convention, are inherited by the Assessments associated with the AssessmentFamily.

**assessmentItems**

This entity represents one of many single measures that make up an assessment.

**bellSchedules**
Implementing Ed-Fi Aligned API’s

• **Ed-Fi REST API Specifications**
  • API Design & Implementation Guidelines
    • Design
    • Implementation

• **Ed-Fi REST API RFC Specifications**
  • Assessment Data Collection API RFC
  • Assessment API RFC
  • Enrollment API RFC
Ed-Fi REST API Specifications

**API Design & Implementation Guidelines** - Provides design guidelines for an Ed-Fi representational state transfer (REST) application programming interface (API).

These guidelines describe the properties to which an API specification and related implementation must adhere in order to be considered aligned to Ed-Fi technology standards.

The guidelines are organized as follows:

- **Scope**
- **Key Characteristics**
- **Requirement Levels**
- **Design**
- **Implementation**
Assessment Data Collection API RFC

- **Assessment Data Collection API RFC**
  - The RFC describes a REST API for exchange of assessment metadata and assessment results.
  - The RFC pertains directly to the Assessment Domain of the Ed-Fi Data Standard
  - Push (POST and PUT) capabilities
  - Pull (GET) capabilities
  - For updates and questions, contact us via feedback (General Ed-Fi Tracker)

- An [Open API specification](#) is available for this RFC
- [Documentation](#) is available for the RFC (Swagger UI)
Assessment API RFC

• **Assessment API RFC**
  • The RFC describes a REST API for systems to transfer assessment metadata and assessment results from one system to another.
  • Read-only (GET) API
  • Assessment Data management API flows into this RFC
• An [Open API specification](#) is available for this RFC
• [Documentation](#) is available for the RFC (Swagger UI)
Enrollment API RFC

- **Enrollment API RFC**
  - The RFC describes a set of read-only REST APIs for transfer of basic student and teacher demographic data and enrollment data for K–12 education.
  - Read-only (GET) API

- An [Open API specification](#) is available for this RFC
- [Documentation](#) is available for the RFC (Swagger UI)
eTags

- eTags – (Entity Tags) are mechanisms used to support optimistic concurrency and efficient bandwidth handling. If-None-Match – Previously returned eTag header value to prevent the unnecessary data transfer of an unchanged resource (GET)
- If-Match
  - The ETag header value used to prevent the DELETE from removing a resource modified by another consumer
  - The ETag header value used to prevent the PUT from updating a resource modified by another consumer.
- TechDocs Resources
  - Handling Optimistic Concurrency with Etags
  - Handling Web Cache Validation with ETags
  - ETags and Other REST API Conventions and Features
Helpful Tools
Helpful Exercises and Tools

• Swagger UI – Exercises
  • Using the Ed-Fi ODS/API Documentation Site (Swagger UI) - Hands On
  • Using the Online Documentation

• Postman – Exercises
  • Using Postman - Hands On
  • Data Standard and ODS/API 2016 Boot Camp

• SDK Walkthrough
  • 2.x - Using Code Generation to Create an SDK
  • 3.x - Using Code Generation to Create an SDK

• Cloud Deployment Options (available on the Ed-Fi Exchange)
  • AWS
  • Azure
Additional Items
MappingEdu

- MappingEdu
  - MappingEdu is an online system that supports business analysts and other users in mapping one data standard to another.
  - MappingEdu Documentation
Ed-Fi Certification

- Ed-Fi Student Information Systems API v2 Certification
- Assessment Outcomes Management API Certification
- Registry of Ed-Fi Certified Products