ED-FI WORKING DRAFT 1 - ASSESSMENT ROSTER

Ed-Fi Working Draft 1: Assessment Roster
Technical Suite: Suite 3
Status: Active

By: Ed-Fi Assessment Work Group
Contact: Eric.Jansson@ed-fi.org
Publication Date: February 29, 2020

Synopsis

This Working Draft outlines a proposed new Assessment Roster domain for the Ed-Fi Data Standard. The purpose of the Assessment Roster domain is to capture and communicate data surrounding the registration of a student for an assessment, and includes data that would generally be needed by assessment systems to prepare for the administration of the assessment for the student. The proposal assumes a parallel REST API binding and an aggregate read-only API, though the final API specification is not included.

Contents

General Discussion

Education agencies often need to provide roster data to assessment vendors in advance of the administration of an assessment, particularly for interim benchmark and summative assessments. This roster data differs significantly from roster data provided to learning tools. This model captures the concept of an assessment registration, while learning tool rosters generally focus on the composition of class sections, and contain some course metadata and academic session calendaring metadata.

The domain as proposed here does not model the actual administration of the assessment nor will it model the reporting of the results; both of those are already represented in the core Ed-Fi Assessment domain and have API bindings described by ED-FI RFC 21 - ASSESSMENT OUTCOMES API.

The specific APIs for data management are not detailed below, but it is assumed that there are CRUD bindings similar to those for the current Ed-Fi Unifying Data Model. It is also assumes that there is a read-only composite API that provides simpler access for providers who need to load this assessment roster data.

Background & Field Research

Through working sessions, the Ed-Fi Assessment Work Group identified two different types of rosters — exclusion rosters and inclusion rosters, and the intent of this model is to support both though the exclusion roster is seen as the priority.

- Exclusion is when criteria are used to include all students meeting the criteria (e.g. "all 5th graders must take the statewide assessment"), and then students are identified for exclusion (e. g. "except when those 5th graders are judged by the LEA to need an accommodation").
- Inclusion is when students are individually included in ("registered for") the assessment, often based on business logic derived from other student characteristics

In both cases, the community wish to see a standard implementation of a roster which provides only the necessary data required to include a student in an assessment. As an API, the roster should be simple and not require fetching data from multiple endpoints.

Use Cases

- **Use Case 1.** Exclusion use case. In this use case, the SEA determines which students should be included in the roster. LEAs may submit data which results in the exclusion of a student from the assessment, but the LEA does not submit data to specifically include a student in the assessment. These lists are largely developed by applying varying sets of criteria to student data contained in SEA systems.
  - State policy mandates an assessment for all 3rd graders, but allows for exceptions when special accommodations are necessary. The SEA enrolls all students enrolled in a public school in third grade to take the mandated statewide exam. LEAs then communicate exclusions to the SEA for students identified as requiring an alternative assessment. This data is then provided to the assessment provider, and removes the need for the LEA manually to transmit spreadsheets to the provider.
- **Use Case 2.** Inclusion use case. In this use case, the SEA is not directly responsible for identifying students. Rather, the LEA identified the students to be administered the assessment.
  - A state wished to support LEAs in registering for statewide assessments, and so allows LEAs to transmit and manage over time via API the student registrations for the annual assessments. This data is then provided to the assessment provider, and removes the need for the LEA manually to transmit spreadsheets to the provider.

Workflow Examples

A few examples show how the assessment roster would work:
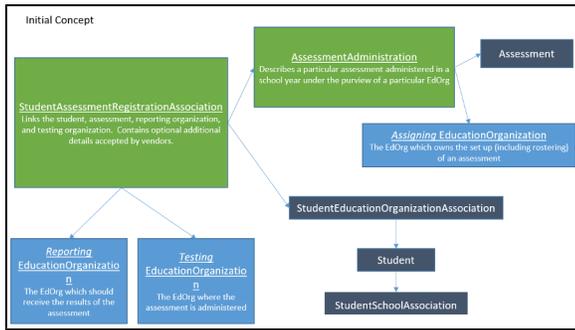
Example 1. WI Workflow

1. Wisconsin Office of Student Assessment (WI OSA) works with vendors to establish a test window. Once determined, Wisconsin Department of Public Instruction (WI DPI) publishes Assessment and Assessment Administration.
   a. EducationOrganizationReference in AssessmentAdminstration is WI DPI.
2. Prior to the test window, WI DPI publishes StudentAssessmentRegistrationAssociation daily.
   a. TestingEducationOrganizationReference in StudentAssessmentRegistrationAssociation is the school the student is currently enrolled in.
   b. ReportingEducationOrganizationReference in StudentAssessmentRegistrationAssociation is the district the student is currently enrolled in.
3. As test window approaches, Vendor retrieves roster for assessment via Assessment Roster Composite. Expectation is that students get added to a roster automatically, but not deleted.
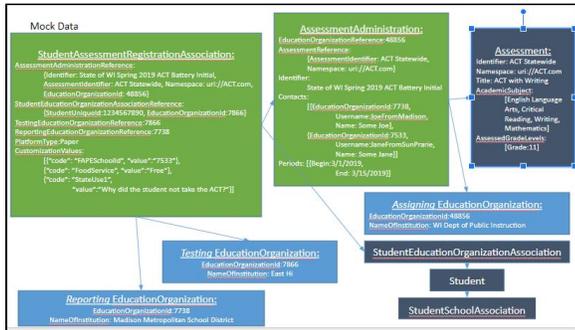
Example 2. DE Workflow

1. Delaware State Assessment Office works with vendors to establish a test window. Once determined, Delaware Department of Education (DE DOE) publishes Assessment and Assessment Administration.
   a. EducationOrganizationReference in AssessmentAdminstration is DE DOE.
2. Prior to the test window, SIS reads the Assessment Administration Data and provide a way within the SIS for district users to send StudentAssessmentRegistrationAssociation data (daily?).
   a. TestingEducationOrganizationReference in StudentAssessmentRegistrationAssociation is the school the student is currently enrolled in.
   b. ReportingEducationOrganizationReference in StudentAssessmentRegistrationAssociation is the district the student is currently enrolled in.
3. As test window approaches, Vendor retrieves roster for assessment via Assessment Roster Composite. Expectation is that students get added to a roster automatically, but not deleted.

Working Draft Details

Assessment Roster Model

An example showing mock data in the initial concept model:

Workflow Examples

Assessment Roster Workflow at the SEA Level (in WI)

1. WI OSA works with vendors to establish a test window. Once determined, WI DPI publishes Assessment and Assessment Administration.
   a. EducationOrganizationReference in AssessmentAdminstration is WI DPI.
2. Prior to the test window, WI DPI publishes StudentAssessmentRegistrationAssociation daily.
   a. TestingEducationOrganizationReference in StudentAssessmentRegistrationAssociation is the school the student is currently enrolled in.
   b. ReportingEducationOrganizationReference in StudentAssessmentRegistrationAssociation is the district the student is currently enrolled in.
3. As test window approaches, Vendor retrieves roster for assessment via Assessment Roster Composite. Expectation is that students get added to a roster automatically, but not deleted.

Assessment Roster Workflow at the LEA Level (in WI)

The LEA/School workflow should be the same as for WI DPI other than that the school/LEA would be the assigning organization and should push data to StudentAssessmentRegistrationAssociation through their SIS as changes are made.

If automatic removal of students from an Assessment Roster is desired, this would need to be negotiated between LEA/School and the Vendor.

Assessment Roster Setup Data

The information required to populate the model for data load and integration scenarios includes:

Assessment (from the core Ed-Fi data model)

- AssessmentIdentifier (e.g., ACT Statewide, ACT Aspire, DRC Forward)
- Namespace (vendor namespace from credentials)
- Title
- AcademicSubjects [ ]
- AssessedGradeLevels[ ]

AssessmentAdministration

- EducationOrganizationReference (SEA, LEA, or School)
- AssessmentReference (e.g., ACT Statewide)
- AdministrationIdentifier (e.g., State X Spring 2019, The ACT Battery Initial…)

- AdministrationContact[ ] (optional, not in MVP scope)
    - EducationOrganizationReference
    - ElectronicMailAddress
    - FirstName
    - LastName
    - LoginId
- Period (test window or dates offered)

StudentAssessmentRegistrationAssociation

- AssessmentAdministrationReference (required)
- StudentEducationOrganizationAssociationReference (required)
- TestingEducationOrganizationReference (may be required by vendor)
- ReportingEducationOrganizationReference (may be required by vendor)
- PlatformTypeDescriptor
- AssessmentCustomization[ ]

Note that AssessmentCustomization values are intended to account for the variability in requirements between vendors and implementations. Rather than code:value pairs, descriptor:value pairs could be used, but the complexity of managing the descriptors seemed beyond the value provided.

Example AssessmentCustomization:

```
"AssessmentCustomization": [
  {"code": "FAPESchoolId", "value":"7533"},
  {"code":"FoodService", "value":"Free"},
  {"code":"StateUse1", "value":"Why did the student not take the ACT?"}
]
```

Assessment Roster Composite

A later of the domain definition will include an Assessment Roster Composite from which the provider can read the roster data. Expected entities and elements include:

- Student
    - StudentUniqueId
    - FirstName
    - LastName
    - BirthDate
- StudentEducationOrganizationAssociation
    - Addresses[ ]
    - ElectronicMails[ ]
    - HispanicLatinoEthnicity
    - LimitedEnglishProfieciencyDescriptor
    - Races[ ]
    - SexDescriptor
    - StudentIdentificationCodes[ ]
- StudentSchoolAssociation
    - SchoolId
    - EntryDate
    - ExitWithdrawDate
    - EntryGradeLevelDescriptor
    - PrimarySchool
- StudentAssessmentRegistrationAssociation
    - ReportingEducationOrganizationReference
    - TestingEducationOrganizationReference
    - PlatformTypeDescriptor
    - AssessmentCustomization[ ]
- AssessmentAdministration
    - AssigningEducationOrganizationReference
    - Identifier
    - Periods
    - Contacts
- Assessment
    - Identifier
    - Namespace
    - Title
    - AcademicSubjects [ ]
    - GradeLevels[ ]