# Gap Analysis Report Draft Phase 1.1

## Executive Summary

This report provides a comprehensive analysis of the similarities, differences, and integration challenges between OneRoster 1.2 and Ed-Fi Student API standards (Data Standard v5.x/ODS API v7.x). Both standards serve K-12 education data exchange needs but differ in scope, structure, and implementation approaches.

The analysis reveals that while core rostering concepts align well between the standards, gaps exist in data elements, structural requirements, and conceptual approaches. Integration is feasible in principle but requires careful mapping of identifiers, descriptors, and data flows.

## 1. Overview of Both Standards

### OneRoster

OneRoster is an interoperability standard focused on exchanging roster data (students, teachers, classes, courses, enrollments) and related academic information between K-12 education systems. It defines a simple data model for people, courses, classes, terms, and enrollments, with standardized roles and statuses.

**Key Characteristics:**

- **Primary Use Case**: Synchronizing data from Student Information Systems (SIS) to learning platforms like Learning Management Systems (LMS)
- **Design Philosophy**: Simplicity and consistency through predefined enumerated values
- **Implementation Formats**: RESTful API with JSON data and/or bulk CSV file exchange
- **Data Flow**: Primarily read-oriented (SIS as provider, LMS as consumer), with limited grade pass-back capabilities

### Ed-Fi Student API

Ed-Fi is a comprehensive data standard and technology suite for K-12 data. The Ed-Fi Student API refers to the RESTful web service endpoints (part of the Ed-Fi Operational Data Store/API) that manage student information and roster data in an Ed-Fi data model.

**Key Characteristics:**

- **Scope**: More extensive normalized data model covering roster data and many other K-12 data domains

- **Design Philosophy**: Comprehensive, detailed, extensible model with descriptor-based flexibility
- **Implementation**: RESTful API with both transactional and read-optimized composite endpoints
- **Data Flow**: Designed for both reading and writing data in real-time, supporting detailed transactional updates

In summary, OneRoster is a *lightweight exchange format* purpose-built for rostering, whereas Ed-Fi is a *broader data standard and platform* in which rostering is one component.

# 2. Data Model Mappings

An initial mapping is available here. Both standards cover similar fundamental roster entities but organize them differently. This section highlights key aspects.

### Users (Students, Teachers, etc.)

| **OneRoster Approach** | **Ed-Fi Approach** | **Mapping Considerations** |
|---|---|---|
| "Users" entity with a list of role fields to categorize user type | Separate entities: Student, Staff, and Parent (Contact) | Role is implicit in which Ed-Fi API endpoint is used |
| Identified by sourcedId | Separate identifiers per domain (StudentUniqueId, StaffUniqueId) plus global GUIDs | One possible approach is to use Ed-Fi GUIDs with type prefixes for OneRoster sourcedIds |
| Personal fields: givenName, familyName, etc. | FirstName, LastSurname, etc. on respective person entities | Direct field mapping possible |
| Limited demographics in core model | Detailed demographics (race, ethnicity, etc.) | Gap: OneRoster lacks many demographic fields present in Ed-Fi |

### Organizations (Orgs)

| OneRoster Approach | Ed-Fi Approach | Mapping Considerations |
|---|---|---|
| Single "orgs" entity with type field (school, district) | EducationOrganization hierarchy (School, LocalEducationAgency) | Explicit mapping based on org type, OneRoster represents relationships using an optional parent/child relationship |
| Flat structure with minimal metadata | Requires certain classification descriptors | Ed-Fi requires additional metadata OneRoster doesn't provide |
| Organization hierarchy not explicitly modeled | Links schools to district via references | OneRoster supports hierarchical relationships through parent-child references, but does not enforce a predefined hierarchy object. The full hierarchy can be derived by traversing relationships, but implementations vary. Mapping requires ensuring relationships are properly constructed to align with Ed-Fi's structured hierarchy. |

### Courses and Classes

| OneRoster Approach | Ed-Fi Approach | Mapping Considerations |
|---|---|---|
| Distinguishes between Course (catalog offering) and Class (specific section) | More granular: Course, CourseOffering, and Section | OneRoster "class" typically aligns with Ed-Fi "Section" |
| Class has courseSourcedId, title, classCode, etc. | Section has sectionIdentifier, links to CourseOffering and Course | Some OneRoster fields have no direct Ed-Fi equivalent (e.g., classType) |
| Status field (active/toBeDeleted) | No explicit status flag; instead, status is inferred | Status is typically determined in Ed-Fi using start and end dates (e.g., `beginDate` and |

| OneRoster Approach | Ed-Fi Approach | Mapping Considerations |
|---|---|---|
| | based on dates or data presence. | `endDate` in `Section` or `StudentSectionAssociation`). If an end date is present and in the past, the record is considered inactive. If missing or in the future, the record is active. Additionally, if a section has no linked enrollments, it may be inferred as inactive. |

### Enrollments (Student and Teacher Membership)

| OneRoster Approach | Ed-Fi Approach | Mapping Considerations |
|---|---|---|
| Single Enrollment entity links user to class with role | Separate associations: StudentSectionAssociation and StaffSectionAssociation | Different association types based on role |
| Primary flag for teacher role | ClassroomPositionDescriptor in StaffSectionAssociation | Requires mapping boolean to descriptor value |
| Contains dates, status, and references | Similar concepts but in different structures | Core concepts map well despite structural differences |

### Academic Sessions (Terms)

| OneRoster Approach | Ed-Fi Approach | Mapping Considerations |
|---|---|---|
| Global, shared sessions across dataset | School-specific sessions tied to school years | Ed-Fi model allows more variation but harder to map to OneRoster's simpler approach |
| Hierarchical (parent/child) relationships | No direct parent/child modeling | Hierarchy may be lost in translation to Ed-Fi |
| Small set of district-wide terms | Many session records (each school-year-term combination) | Integration must decide how to handle this structural difference |

## 3. Technical and Conceptual Differences

Despite covering similar domains, the standards differ significantly in architecture, philosophy, and implementation.

## Data Model Structure and Scope

| **Aspect** | **OneRoster** | **Ed-Fi** | **Integration Impact** |
|---|---|---|---|
| Structure | Simplified, flattened model | Normalized relational model exposed as granular REST end points | Requires denormalization/normalization during conversion |
| Scope | Focused on roster and grades | Full student profiles and K12 data | Ed-Fi contains much more data than OneRoster can express |
| Purpose | Interface specification for data exchange | Central operational datastore | Different architectural assumptions |

### Vocabulary and Data Classification Systems

| **Aspect** | **OneRoster** | **Ed-Fi** | **Integration Impact** |
|---|---|---|---|
| Value representation | Fixed enumerated vocabularies with predefined values<br><br>(Many vocabularies in OR 1.2 are extensible using a defined pattern. The pattern is to prepend the item with "ext: " That is described here) | Flexible descriptors that can be extended | Requires mapping between fixed codes and dynamic descriptors |
| Flexibility approach | Limited set of standardized values; All first class objects in OneRoster have a metadata property | Extensible descriptor pattern with local customization | Ed-Fi implementations may have values not expressible in OneRoster and vice versa |

| Aspect | OneRoster | Ed-Fi | Integration Impact |
|---|---|---|---|
| | that is a container to put additional data in | | |
| Example types | User roles (student, teacher, parent), organization types, enrollment status | ClassroomPositionDescriptor, AcademicSubjectDescriptor, TermDescriptor | Mapping tables needed to translate between systems |
| Control over values | Centrally defined by standard (Additionally, extensible vocabularies gives integration specific flexibility for data that OR doesn't natively support). | Locally configurable within standard patterns | Complex mapping for non-standard descriptor values |
| Semantic richness | Simple categorical values | Hierarchical descriptor system with metadata | Information may be lost when mapping complex descriptors to simple enumerations |

## API Design and Data Exchange

| Aspect | OneRoster | Ed-Fi | Integration Impact |
|---|---|---|---|
| Overall approach | Supports bulk data retrieval via getALL endpoints on all first class objects. E.g. getAllStudents<br>Supports individual retrieval via get1 endpoints on all first class objects, e.g. getStudent | Provides a granular RESTful API, where each resource is accessed individually. | OneRoster is optimized for efficient bulk synchronization, making it well-suited for large-scale data exchanges.<br><br>Ed-Fi's transactional model allows for real-time, fine-grained updates but requires more API calls for full dataset retrieval. Integration may involve **balancing bulk vs.** |

| Aspect | OneRoster | Ed-Fi | Integration Impact |
|---|---|---|---|
| | Also includes predefined query endpoints for common use cases<br><br>Predefined query endpoints for specific use cases, | | real-time needs based on use case requirements |
| Formats | JSON REST API and standardized CSV | JSON REST API only | OneRoster CSV generation required if Ed-Fi is source |
| JSON style | Uniform flat list style with minimal nesting<br>Example:<br>`json { "users": [ { "sourcedId": "S12345", "givenName": "John", "role": "student" } ] }` | CamelCase with complex nesting relationships embedded directly within objects<br>Example:<br>`json { "students": [ { "studentUniqueId": "S12345", "name": { "firstName": "John" } } ] }` | Schema transformation needed when converting between models. OneRoster requires multiple lookups to resolve relationships, while Ed-Fi allows a **single request** for full entity details. |
| Query capabilities | Supports endpoint payload filtering for all collection responses.<br><br>Required filters include support for:<br>- role<br>- gradingPeriod<br>- school<br>- term<br><br>Example:<br>`GET /enrollments?role=student&schoolSourcedId=1234&dateLastModified=2023-01-01` | Supports searching and filtering on resources, including:<br>- GET All Queries (Retrieve full collections)<br>- GET by ID (Retrieve a specific entity using its identifier)<br>- Filtering on root-level properties<br>- Querying by natural key values (e.g. `GET /ed-fi/students?firstName=John&lastSurname=Smith)`<br>- Supports multi-field filtering using & operators | Both standards offer structured query capabilities but differ in approach. OneRoster provides standardized, predictable filtering on predefined attributes, making implementation straightforward. Ed-Fi allows more flexible, multi-field searches and natural key lookups, which support advanced data retrieval but may vary across implementations.<br><br>OneRoster is well-suited for bulk data retrieval, while Ed-Fi excels in |

| Aspect | OneRoster | Ed-Fi | Integration Impact |
|---|---|---|---|
| | | - Uses flattened parameter names (e.g., `?schoolId=5678` instead of `?schoolReference.schoolId=5678`) | transactional and fine-grained filtering scenarios. |
| **Composites** | Not applicable | The Ed-FI Enrollment API and other API Composites provide pre-joined, subject-oriented data views. These enable retrieving related records (e.g., students, enrollments, sections) in a single API request instead of multiple calls | Ed-Fi's composite API resources help reduce API calls by retrieving related data in a single request, making integration more efficient. While OneRoster does not have a direct equivalent, its flat structure and bulk retrieval model serve a similar purpose for batch processing. |

## Data Flow and Update Mechanisms

| Aspect | OneRoster | Ed-Fi | Integration Impact |
|---|---|---|---|
| Primary purpose | Data export (SIS to others) | Full CRUD operations | Different assumptions about data flow |
| Change tracking | dateLastModified field in responses | Built-in Change Query feature | Different approaches to delta updates |
| Change handling | Synchronization by replacement or via incremental changes | Transactional updates<br><br>Built-in Change Query API for tracking changes | Different update paradigms |
| Write capabilities | Gradebook service only | Comprehensive write support | Ed-Fi supports full CRUD operations (read/write), enabling full bi-directional data sync, whereas |

| Aspect | OneRoster | Ed-Fi | Integration Impact |
|---|---|---|---|
| | | | OneRoster is primarily designed for one-way rostering data export with limited write-back (grades) |

## Authentication and Authorization

| Aspect | OneRoster | Ed-Fi | Integration Impact |
|---|---|---|---|
| Auth method | OAuth 2.0 (Client Credentials Grant | OAuth 2.0 (Client Credentials Grant) | None |
| Authorization model | Simple: credential scoped to dataset | Claim sets and data scopes for fine-grained access | More complex authorization mapping needed |
| Multi-tenancy | Limited multi-tenant support | Strong multi-tenant capabilities | Ed-Fi's design supports more complex multi-school or multi-tenant scenarios than OneRoster's |

## Identifiers and References

| Aspect | OneRoster | Ed-Fi | Integration Impact |
|---|---|---|---|
| Primary keys | Opaque sourcedId for each object | Natural keys or compound keys plus internal GUIDs | Need to map between different ID approaches |
| Reference style | Direct sourcedId references | Compound key references or descriptor references | Translation required between reference styles |

| Aspect | OneRoster | Ed-Fi | Integration Impact |
|---|---|---|---|
| Uniqueness scope | Globally unique within dataset | Unique per entity type | Potential for ID collisions when mapping |

# 4. Gap Analysis and Interoperability Challenges

Building on the data model mappings and technical differences identified above, key gaps and challenges are identified that need to be addressed for successful integration between OneRoster and Ed-Fi.

## Data Element Gaps (Missing Fields)

Building on the field mappings outlined in Section 2 under "Notable Field-Level Gaps," there are several areas where one standard contains data elements that have no direct equivalent in the other:

- **OneRoster concepts not directly represented in Ed-Fi**: Academic session hierarchies (parent-child relationships between terms), course grade level targets, and class types (scheduled, homeroom, etc.)
- **Ed-Fi data outside OneRoster scope**: Demographics (race, ethnicity, economic status), program enrollments (special education, English Learner), detailed staff assignments beyond teaching roles, Attendance, Discipline, Finance, and more.

Note: An evaluation of the 1EdTech Assessment Results Profile for Gradebook Service and Ed-Fi Assessment Domain is outside the scope of this phase of analysis.

These missing elements present implementation challenges when trying to provide complete data exchange between systems using different standards.

## Descriptor vs. Enumeration Mismatch

As detailed in Section 3 under "Vocabulary and Data Classification Systems," OneRoster and Ed-Fi take fundamentally different approaches to codifying values:

- **OneRoster uses fixed enumerated vocabularies** with predefined values (e.g., specific role types, organization types). Vocabularies are extensible using a defined pattern.
- **Ed-Fi employs flexible descriptors** that can be extended while maintaining a standardized structure

This philosophical difference creates significant integration challenges:

- Mapping OneRoster's strict codes to Ed-Fi's extensible descriptors requires careful configuration
- Special handling is needed for custom descriptor values that may not fit into OneRoster's enumerations
- Different semantic meanings may exist for seemingly similar values across the standards

## Academic Session Alignment

The different approaches to academic terms (discussed in Section 2 under "Academic Sessions (Terms)") create a specific integration challenge:

- **OneRoster expects** a small set of district-wide, hierarchical academic sessions
- **Ed-Fi provides** school-specific sessions tied to school years, potentially creating many more distinct sessions

This structural difference forces implementers to either:

1. Compress Ed-Fi's numerous school-specific sessions into fewer OneRoster district-wide sessions (potentially losing school-specific calendar information)
2. Output many school-specific OneRoster sessions (which may confuse applications expecting a simpler structure)

Furthermore, the parent-child relationships explicitly modeled in OneRoster (e.g., a semester's parent is the school year) have no direct equivalent in Ed-Fi, potentially losing important hierarchical information during integration.

## Role and Enrollment Definitions

The different approaches to representing people and their roles (detailed in Section 2 under "Users" and "Enrollments") create specific challenges:

- **Ed-Fi's teacher-class relationships** include detailed positioning information through ClassroomPositionDescriptor
- **OneRoster uses** a simpler boolean "primary" flag for identifying lead teachers

Staff who don't teach classes present a particular challenge. In Ed-Fi, these individuals have clear roles through StaffEducationOrganizationAssignmentAssociation, while OneRoster has limited representation for non-teaching roles. This can result in staff like counselors, administrators, or specialists being either:

- Excluded from OneRoster exports entirely
- Included with generic roles that don't capture their specific function

This misalignment makes it difficult to maintain consistent staff role information across both standards.

## Identifiers and Matching

The different identifier approaches discussed in Section 3 under "Identifiers and References" create significant integration challenges:

- **Ed-Fi uses** natural keys and composite keys for entity identification, with separate identifier namespaces per entity type (StudentUniqueId, StaffUniqueId)
- **OneRoster requires** globally unique sourcedIds across all entity types in the dataset

This fundamental difference means that when mapping between systems:

1. Direct use of Ed-Fi IDs as OneRoster sourcedIds could create collisions (e.g., a student "12345" and a staff "12345" would conflict in OneRoster)
2. Using composite keys from Ed-Fi requires transformation logic to create single unique strings for OneRoster
3. Creating new GUIDs specifically for OneRoster requires maintaining a persistent mapping/crosswalk between the two identification systems

One recommended approach (using Ed-Fi GUIDs with type prefixes) adds complexity but ensures consistent cross-system identification. However, this requires robust identifier management and synchronization between systems, particularly for bidirectional data flows.

## Structural Requirements

The normalized data model of Ed-Fi (described in Section 3 under "Data Model Structure and Storage") imposes structural requirements that have no counterpart in OneRoster's flatter model:

- **Ed-Fi requires** certain relationship chains to exist before others can be created. For example:
    - A Student must have a StudentSchoolAssociation (enrollment in a school) before they can have a StudentSectionAssociation
    - Organizations must have specific descriptors (e.g., grade levels served by a school)
- **OneRoster‘s** model has fewer required dependencies
-

When integrating from OneRoster to Ed-Fi, this gap requires:

1. Inferring or creating the missing required Ed-Fi relationships based on context
2. Providing default or "filler" values for required Ed-Fi fields that OneRoster doesn't supply
3. Mapping OneRoster's simpler terms and types to Ed-Fi's more specific descriptors

For example, if OneRoster doesn't explicitly provide a student's enrollment in a school (only their enrollment in classes), the integration must create the StudentSchoolAssociation in Ed-Fi based on which school those classes belong to.

These structural differences significantly complicate the transformation logic needed when moving data between the systems.

## Session Timing and Updates

The different update mechanisms explained in Section 3 under "Data Flow and Update Mechanisms" create specific challenges for maintaining data consistency:

- **OneRoster typically uses** synchronization by replacement or incremental changes based on lastModified timestamps
- **Ed-Fi supports** granular transactional updates and has a built-in Change Query API

These differences affect how systems handle data lifecycle events:

1. **Deletions/Inactivations**:
    - OneRoster might omit an enrollment in the next sync
    - Ed-Fi would explicitly have an endDate or deletion for that association

2. **Real-time Updates**:
    - Ed-Fi's Change Query provides a standardized way to get only changed records
    - OneRoster implementations vary in how they handle change tracking

When integrating these systems, synchronization logic must account for these different approaches. For example:

- When an enrollment ends in Ed-Fi, should it be marked inactive or removed in OneRoster?
- When an enrollment disappears from OneRoster, should it receive an endDate in Ed-Fi or be completely removed?
- How frequently should synchronization occur to balance timeliness against system load?

These timing and update pattern differences require careful business logic in the integration layer to ensure both systems maintain a consistent view of the data.

## Custom Extensions Alignment

Building on the different data models and technical approaches highlighted in previous sections, the standards have fundamentally different extension mechanisms:

- **Ed-Fi uses** a formal extension schema with well-defined patterns for adding fields, entities, and descriptors
- **OneRoster employs** a metadata approach, allowing custom properties to be added within a designated container

These different extension approaches create several integration challenges:

1. When mapping from OneRoster to Ed-Fi, determining whether to create formal Ed-Fi extensions or map to existing Ed-Fi structures
2. When mapping from Ed-Fi to OneRoster, implementers must decide which Ed-Fi extensions to represent as OneRoster metadata extensions
3. Maintaining consistency in extension naming and values across implementations

The extension challenge is compounded by the fact that extensions are, by nature, outside the standard. A district using Ed-Fi extensions might create OneRoster exports with metadata extensions that represent the same concepts, but without careful coordination, different districts might use different naming conventions for the same extended data.

For successful implementation, there must be agreement on:
- Which extensions are necessary across both standards
- How they should be named and formatted
- Which system is authoritative for extended data

# 5. Recommendations for Alignment and Integration

Based on the gaps and challenges identified in Section 4, a number of strategies can be considered to achieve effective integration between OneRoster and Ed-Fi standards:

## Establish Clear Mapping Conventions and Align Descriptor Values and Enumerations

**Addressing gaps in:** Data elements, identifiers, and extension alignment

- Create a canonical mapping guide between OneRoster 1.2 and Ed-Fi v7.x fields
- Adopt a consistent identifier strategy, possibly using Ed-Fi GUIDs with type prefixes for OneRoster sourcedIds
- Document field-by-field mappings, including handling of fields without direct equivalents

## Align Descriptor Values and Enumerations

**Addressing gaps in:** Descriptor/enumeration mismatch, academic session alignment, role definitions

- Configure Ed-Fi ODS with descriptor values that align with OneRoster's terminology (e.g., Term descriptors that match academicSession types)
- Define how to map OneRoster's primary teacher flag to Ed-Fi's ClassroomPositionDescriptor
- Establish convention for handling staff without class assignments in rostering data

## Address Authentication and Authorization Holistically

**Addressing gaps in:** Different security models between standards

- Ensure consistent data visibility rules across both APIs by aligning security scopes and permissions
- Define clear authorization profiles for combined usage that respect the data access limitations in both systems
- Document security implementation details for cross-standard integration

## Develop a Strategic Approach to OneRoster Profiles vs. Ed-Fi APIs

The gap analysis in Section 4 highlighted significant alignment challenges.. This section examines the strategic choice between creating a OneRoster Profile or leveraging Ed-Fi's comprehensive API structure to address data needs beyond basic rostering.

### Strategic Considerations: OneRoster Profiles vs. Ed-Fi APIs

The gap analysis in Section 4 identified notable alignment challenges between systems relying on custom data approaches and standardized frameworks. This section evaluates the strategic decision between adopting OneRoster Profiles to meet data needs beyond basic rostering and leveraging Ed-Fi’s comprehensive API structure, considering their respective strengths and limitations.

#### The OneRoster Profiles Approach

OneRoster, designed to serve a global audience, offers a rostering standard adaptable to diverse educational contexts. Unlike custom extensions, OneRoster Profiles are nationally defined specifications that inherit and remain synchronized with the core standard, enabling tailored requirements while preserving interoperability. Governed by a global vote of the Technical Advisory Board, Profiles provide a structured mechanism for coordination, mitigating the risks of fragmentation seen with ad-hoc extensions.

- **Strengths:** The global applicability of OneRoster ensures relevance across varied regulatory and operational environments, unlike Ed-Fi’s U.S.-centric focus. Profiles maintain alignment with the core specification, reducing the need for bilateral agreements by establishing a consistent, internationally recognized framework. This

governance model supports scalability and adaptability without sacrificing the standard's integrity.

- **Challenges:** While Profiles offer more structure than extensions, their national scope may still introduce variability in implementation, requiring systems to accommodate multiple profile definitions. The focus on rostering as a primary function limits the depth of data coverage compared to broader models, potentially necessitating supplementary solutions for complex data needs.

### The Ed-Fi API Approach

Ed-Fi provides a comprehensive, education-specific API framework tailored primarily to the U.S. education system. Its standardized data model extends beyond rostering to encompass entities such as demographics, program enrollments, and assessments, offering a holistic approach to data integration.

- **Strengths:** Ed-Fi's extensive data model reduces the need for custom additions by addressing most district requirements within its standard entities and relationships. Its community-driven governance process ensures new fields are incorporated systematically, fostering consistency across implementations. The use of skill descriptors—a deliberate design choice—allows controlled flexibility within the model, enabling districts to define values without altering the core structure, though this approach is neither inherently superior nor inferior to other methods.

- **Challenges:** Ed-Fi's U.S.-specific design limits its applicability in global contexts, potentially misaligning with international requirements. The broader scope of the model introduces greater complexity, which may increase implementation effort compared to the more focused OneRoster standard. The reliance on descriptors, while structured, may still require additional configuration to meet unique local needs.

## Comparative Analysis

Both approaches offer distinct advantages depending on organizational priorities. OneRoster Profiles provide a globally governed, rostering-focused solution that balances flexibility and standardization, making them well-suited for organizations operating across diverse regions or requiring alignment with an international standard. The use of Profiles, rather than extensions, avoids the interoperability pitfalls of fragmented metadata fields, though it may not fully address data needs outside rostering without additional integration efforts.

Ed-Fi, conversely, delivers a robust, U.S.-oriented framework with semantic richness and comprehensive coverage, ideal for districts seeking a unified data model under a single standard. Its design choice of skill descriptors reflects a structured yet adaptable approach,

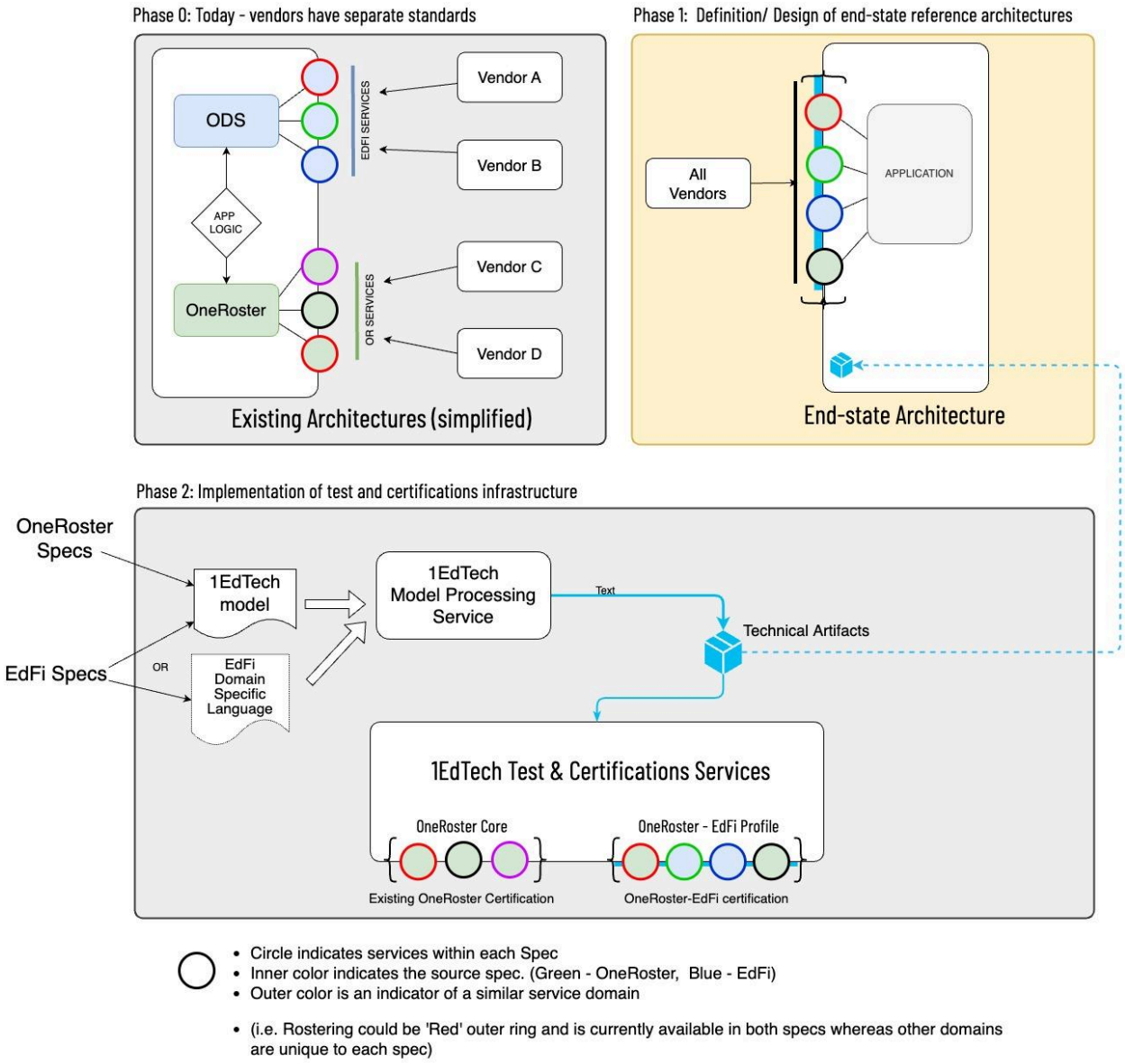
though its domestic focus and broader complexity may pose challenges for global or narrowly scoped implementations.

### Strategic Implications

The choice between OneRoster Profiles and Ed-Fi APIs hinges on the scope of data needs and geographic context. For organizations prioritizing global interoperability and rostering efficiency, OneRoster Profiles offer a streamlined, governed path that aligns with international norms. For those requiring a deeper, U.S.-specific data model with reduced integration overhead, Ed-Fi presents a compelling alternative. A balanced strategy may involve leveraging OneRoster Profiles for core rostering functions while evaluating Ed-Fi for supplementary data domains, ensuring alignment with both operational goals and interoperability requirements.

# 6. Proposed Implementation Architecture

Building on the recommendations in Section 5, we propose a process flow to support implementations and address the identified gaps while leveraging the strengths of both standards. This architecture creates a hub-and-spoke model with Ed-Fi ODS/API at the center, enabling both OneRoster data distribution for rostering and direct Ed-Fi API interactions for richer bidirectional data exchange:




This proposed architecture demonstrates:

1. A simplified description of the current state (phase 0), highlighting that each vendor (SIS or LMS etc) may be required to implement one or both standards to meet their needs.
    a. Each standard has a unique set of services and capabilities with some overlap (rostering etc).
2. The End-state architecture would comprise a combination of services from both standards. The next phase of development will include the definition and mapping of those services.
3. Phase 2 proposes a future-state where the 1EdTech infrastructure that maintains and supports the 1EdTech ecosystem for certification and diagnostics, can be applied to support the OneRoster-EdFi joint profile.

This architecture leverages both Ed-Fi's comprehensive data model and API capabilities for rich integration and OneRoster's standardized approach for rostering, creating a best-of-both-worlds solution for districts.

# Conclusion

This analysis reveals that while OneRoster 1.2 and Ed-Fi Student API standards overlap in their core rostering concepts, they represent different philosophies in K-12 data exchange.

OneRoster delivers a widely-adopted framework designed for global interoperability, prioritizing simplicity and consistency, whereas Ed-Fi offers a rich, normalized data model with extensive coverage beyond basic rostering.

The gaps we've identified in data elements, vocabulary approaches, term structures, and implementation requirements create integration challenges, but these can be overcome through the mapping conventions and implementation strategies outlined in this report. By developing clear mapping guidelines and exploring the potential for a joint profile that aligns key aspects of both standards, districts can benefit from improved interoperability while streamlining integration efforts for vendors.