ED-FI RFC 1 - API PATCH

Ed-Fi Request for Comment: 1
Product: Ed-Fi Data Standard v2.1
Affects: Ed-Fi API Design & Implementation Guidelines (and Ed-Fi ODS / API)
Obsoletes: --
Obsoleted By: --
Status: Active Proposal for Release

Geoff McElhanon, Educuity
Ed-Fi Alliance
June 7, 2016

Synopsis

The current (v2.1) REST API Design & Implementation Guidelines cover partial updates through the HTTP PATCH verb. Partial updates are a way of updating only select fields on an existing entity, as opposed to sending a complete representation. The current guidelines recommend a partial object approach, whereby only those properties requiring update are sent. The proposal author suggests that the currently recommended approach is ambiguous, and that a set of established standards (IETF RFC 6902) proposes an unambiguous JSON solution suitable for use in Ed-Fi implementations.

Detail

A naïve implementation of PATCH might accept a request body whose structure matches exactly that of the target resource, but may only contain a subset of the properties. The server would then process the properties that are present, while leaving those not included as unmodified. While this is definitely simplistic, there are subtle problems with the semantics of this approach.

In strongly-typed languages such as C# and Java, JSON serializers typically have global settings for null and default value handling. For example, the JSON.NET serializer provides the option to include or ignore null values on strongly-typed models being serialized. If the include option is used, then all nullable properties will be included in the JSON output with explicit null values. However, if the ignore option is used, then all properties that have null values will be excluded from the output.

For PUT, where the semantics call for a full replacement of the entire target resource there are no issues introduced for server-side processing. However, using such a non-standard approach for a PATCH implementation using the resource's standard representation would create ambiguity. The process of serializing the message body would not afford the client the ability to explicitly express the intent of the changes. The serialized body will either include or ignore all null value when serializing the request. This makes it impossible for the client to express that a particular property's null value should be set to null by the server, while another property's null value should be ignored because it is not part of the intended PATCH operation. Depending on the serializer's configuration, the server would process the resulting request differently.

Proposed in 2010, RFC 5789 for "PATCH Method for HTTP" provides details and guidance for PATCH implementations. The standard is very clearly written such that a PATCH request consists of a "set of changes represented in a format called a 'patch document' identified by a media type." As opposed to a PUT request (which is a wholesale replacement of an existing resource), the PATCH request's "enclosed entity contains a set of instructions describing how a resource currently residing on the origin server should be modified to produce a new version."

RFC 5789 also indicates that PATCH introduces the potential for collisions that could leave the resource in a "corrupt" state. To guard against this possibility, PATCH implementations on the Ed-Fi REST API should require ETag checks (as described elsewhere in this document) to ensure that a resource isn't being modified in a way that is unexpected for its current state.

While RFC 5789 doesn't define a specific patch format, a JSON-based definition is provided by RFC 6902 for "JavaScript Object Notation (JSON) Patch". Using a request content type of "application/json-patch+json" it describes the patch format in terms of operations performed against parts of a JSON document. Each change to be applied to the target resource must specify an operation (e.g. add, remove, replace, move, copy or test), and must identify where in the document to apply the change using a JSON-Pointer value (as defined by RFC 6901).

The following listing (from RFC 6902) depicts a PATCH request using this approach:

```
PATCH /my/data HTTP/1.1   Host: example.org
   Content-Length: 326
   Content-Type: application/json-patch+json
   If-Match: "abc123"

   [
     { "op": "test", "path": "/a/b/c", "value": "foo" },
     { "op": "remove", "path": "/a/b/c" },
     { "op": "add", "path": "/a/b/c", "value": [ "foo", "bar" ] },
     { "op": "replace", "path": "/a/b/c", "value": 42 },
     { "op": "move", "from": "/a/b/c", "path": "/a/b/d" },
     { "op": "copy", "from": "/a/b/d", "path": "/a/b/e" }
   ]
```

In summary, an implementation of PATCH for an Ed-Fi REST API should adhere to the following proposed Internet Engineering Task Force (IETF) standards:

- RFC 5789 – PATCH Method for HTTP
- RFC 6901 – JavaScript Object Notation (JSON) Pointer
- RFC 6902 – JavaScript Object Notation (JSON) Patch

Specification Content

Suggest the HTTP Verbs section, PATCH item (v2.0 p14) be changed to read as follows:

> **PATCH.** An HTTP PATCH performs a partial update on an existing individual resource using a specialized patch format outlined in RFC 6902. For a partial update, the client submits a set of instructions that are processed by the server to apply changes to the target resource. The entire patch will be applied, or none of it will. The new representation of the entire resource is returned in the response body with an updated ETag header. The PATCH verb *may* be supported for non-read-only Resources.

Suggest the documentation include references to the relevant IETF RFC material and an example similar to the example provided in the Detail section above.