Simple LIS Specification

v0.2.1

Michael Sofaer
Chief Architect, Inigral, Inc.
msofaer@inigral.com

Introduction

This specification is designed to be as similar as possible to the draft IMS LIS standard, within the confines of a REST interface, and easy implementation. It is not an official IMS product, and there is no guarantee that anything implemented based on this document will be in any way interoperable with an IMS system. Special thanks to Dr. Charles Severance for guidance on this document.

Architecture

This specification is envisioned to be used within a REST web service environment. It is designed to support one-way data integration from an SIS to an LMS. It is envisioned that the requests could also be combined in a file and run as a batch, whether for synchronization or for initial provisioning, if a web service implementation is not desired.

This specification is designed to use HTTP as fully as possible, and leverage the tools HTTP provides as much as possible.

All text fields are UTF8. The SIS is responsible for providing text in the correct language, there are no language fields.

Security

Security can be handled at a higher level (SSL and authentication), or custom security can be added by an implementer.

Requests

The SIS may make PUT and GET requests directly to any resource, ex: http://root/meetings/
The SIS may make GET and DELETE requests with a sourced_id to any resource handler, ex: http://root/people/mg332

Resources

The LMS will accept, and the SIS will provide, the following persistent resources:
Person
CourseTemplate
Term
Group
CourseOffering
CourseSection
Membership
Meeting

Every resource must have a *sourced_id* field, which must be unique with in the scope of the resource.
A put request with a new sourced_id will cause the record to be persisted by the LMS. A put request with an existing sourced_id will replace the current record.

Data Models

blue fields are required

orange fields are optional

Person
names
  given
  family
contact_info
  email

CourseTemplate
title
code This is the short human-readable identifier. e.g. MATH101
description

Term
title
begins_on
ends_on

Group
title
category
sub_category
description
parent_sourced_id

pre-defined categories are:
AcademicUmbrella (sub-categories like college, school, campus, department)
AcademicProgram (sub-categories like concentration, minor, degree)
Residence (sub-categories like dormitory, fraternity house, university-operated apartments)
Enterprise
Administration
StudentOrganization (sub-categories like intramurals)
Experience

sourced_id 'Application' is reserved for an Enterprise group for application administration.
Putting or Deleting a group with that sourcedId has undefined behavior.

CourseOffering
term_sourced_id
course_template_sourced_id
group_sourced_id This is intended for the department offering the course

CourseSection

course_offering_sourced_id
label A short label for display after the course code.
description

Membership
person_sourced_id
target_type
target_sourced_id
role
term_sourced_id

term_sourced_id should be provided for sections memberships, and must be
pre-defined roles to sections are 'Instructor', 'Student' and 'TeachingAssistant'
pre-defined roles to StudentOrganizations are 'Member', 'Fan' and 'Officer'
pre-defined roles to Administration groups are 'Staff' and 'Ambassador'
pre-defined roles to Enterprise groups are 'Administrator', 'Staff', 'Moderator' and 'Analyst'

Meeting
target_type
target_sourced_id
i_calendar
See external resource for information on the icalendar object to put in here. e.g.
http://tools.ietf.org/html/rfc2445#section-4.6.1

Foreign Key Constraints

The LMS is expected to use foreign key constraints, and return 403 Forbidden when an attempt is made to delete a record that is still needed.
Polymorphic targets (memberships and meetings) cannot have foreign key constraints, and care must be taken that a multi-threaded system does not create data corruption.

The LMS is expected to cascade deletion of a user's memberships when the user is deleted, rather than returning 403

#TODO: Should there be cascade on sections as well, of both meetings and memberships?

Sample Requests

1)
URL root/people/
METHOD PUT
BODY

```
<person>
  <sourced_id>bjones8</sourced_id>
  <names>
    <given>Bob</given>
    <family>Jones</family>
  </names>
  <contact_info>
```

```
    <email>bob@your_school.edu</email>
  </contact_info>
</person>
```

RESPONSE:
HTTP 201 CREATED
URI: root/people/bjones8

2)
URL root/memberships.xml
METHOD PUT
BODY

```
<membership>
  <sourced_id>bjones8-Application</sourced_id>
  <person_sourced_id>bjones8</person_sourced_id>
  <target_type>Group</target_type>
  <target_sourced_id>Application</target_sourced_id>
  <role>Moderator</role>
</membership>
```

HTTP 201 CREATED
URI: root/memberships/bjones8-Application

3)
URL root/people/
METHOD PUT
BODY

```
<person>
  <sourced_id>bjones8</sourced_id>
  <names>
    <given>Bob</given>
  </names>
  <contact_info>
    <email>bob@your_school.edu</email>
  </contact_info>
</person>
```

RESPONSE:
HTTP 422 UNPROCESSABLE ENTITY

```
There is no field with the name 'family' in
<names>
    <given>Bob</given>
  </names>
```

4)
URL root/people/bjones
METHOD GET

RESPONSE
HTTP 200 OK

```
<person>
  <sourced_id>bjones8</sourced_id>
  <names>
    <given>Bob</given>
    <family>Jones</family>
  </names>
  <contact_info>
    <email>bob@your_school.edu</email>
  </contact_info>
</person>
```

5)
URL root/people/bjones8
METHOD DELETE

RESPONSE
HTTP 204 NO CONTENT

6)
URL root/people/bjones8
METHOD GET

RESPONSE
HTTP 404 NOT FOUND

7)
URL root/memberships/bjones8-Application
METHOD GET

RESPONSE
HTTP 404 NOT FOUND

8)
URL root/people/
METHOD PUT
BODY

```
<people>
<person>
  <sourced_id>bjones8</sourced_id>
  <names>
    <given>Bob</given>
    <family>Jones</family>
  </names>
  <contact_info>
    <email>bob@your_school.edu</email>
  </contact_info>
</person>
<person>
  <sourced_id>bjones9</sourced_id>
  <names>
    <given>Bob</given>
    <family>Jones</family>
  </names>
  <contact_info>
    <email>bob2@your_school.edu</email>
  </contact_info>
</person>
</people>
```

RESPONSE:
HTTP 201 CREATED
URI: root/people/bjones8
URI: root/people/bjones8
Additional test cases are available in the prototype test code, send me an email!