How Do Learning Platforms Integrate Resources Today?

Teacher searches LORs one at a time

Teacher installs “LTI resource picker” app into learning platform

Teacher selects a resource

Teacher installs “LTI resource picker” app into learning platform

OR

Teacher selects a resource

Teacher collects resources in “shopping cart”

Teacher copies and pastes URL into learning platform

Resources added to course in learning platform

What’s the problem?

- Inconsistent User Interfaces
- Learning Platform should be the “teacher cockpit”
- LORs have unnecessary development burden for LTI “resource picker” apps
- LTI apps add additional credentialing requirements which aren’t needed in an API search call
So What’s A Better Way?

- Provide a standard REST API for searching LORs
- Let the learning platform or tool own the teacher experience
- Learning platforms get to consume ONE search API
  - And connect to every LOR
- LORs get to implement ONE search API
  - And connect to many LMSes
So Why A New Standard Now?

- Current process is too complicated for teachers to use the digital resources in multiple LORs
- Finally consensus on what a learning object should have as metadata:
  - We built on LRMI/schema.org
- REST APIs are commonplace now
  - Specifically IMS has made some nice progress on REST/JSON APIs with OneRoster that was used as a model
What Do We Care About for Learning Resources?

- resource name and description
- resource type
- publisher or owner of the resource
- license that applies (such as Creative Commons or a publisher’s URL to their license)
- duration (time to consume)
- web link or LTI link to access
- technical format (MIME types such as “text/html”, “video/mpeg”)
- educational audience (student, teacher, administrator, parent, other)

* not in LRMI/Schema.org

- thumbnail image
- subject
- language
- age range (more int’l than grade)
- learning objective (such as a state standard)

CaseItemURI
CaseItemGUID

- author
- publish date
- rating
- relevance
Learning Resource Types

- Work done by CCSSO Communities of Practice to define resource types
- Hierarchical approach enables many types without cognitive overload
- Resources can be tagged with multiple resource types simultaneously
- Examples:
  - Assessment/Formative, Assessment/Interim
  - Collection/Course, Collection/Unit
  - Text/Book, Text/Passage
  - Media/Video
What Does the REST API Look Like?

- An example search
  - https://imsglobal.org/ims/ltisearch/resources?filter=search%3D%27civil%20war%27
  - Note: arguments to filter parameter are URL encoded (hence need for filter parameter)

- Search (filter) data fields:
  - search (searches multiple fields as LOR chooses)
  - name
  - description
  - subject
  - learningResourceType
  - language
  - typicalAgeRange
  - textComplexity
  - learningObjectives
  - author
  - publisher
  - timeRequired
  - technicalFormat
  - educationalAudience
  - accessibilityAPI
  - accessibilityInputMethods
  - publishDate
  - rating
  - relevance
Filtering Options

OneRoster offers powerful searching controls starting with **filter** with two options:

1. **Full predicate logic:**

   ```
   ?filter=<data field><predicate><value>
   OR
   ?filter=<data field><predicate><value><logical><data field><predicate><value>
   ○ Predicates: =, !=, >, >=, <, <=
   ```

2. "attribute=" and "attribute~" shorthands (after "filter=")

   ○ ~ provides OR searching semantics, = provides AND semantics
   
   So... given that resource just has subject1 on it:
   
   - ?filter="subject=subject1" - record not returned
   - ?filter="subject=subject1,subject2" - record not returned
   - ?filter="subject=subject1,subject2,subject3" - record not returned
   - ?filter~"subject=subject1" - record returned
   - ?filter~"subject=subject1,subject2" - record returned
   - ?filter~"subject=subject1,subject2,subject3" - record returned.