inBloom

Learning Map and
inBloom Index System

Data Model and Server

inBloom Index
Common Core Middleware
API Specification

V0.9 22 Jan 2013

Applied Minds, LLC

1209 Grand Central Ave Glendale CA 91201

Contents

Background

The Learning Map Data Model

The Bill & Melinda Gates Foundation (the Foundation) supports the implementation of the Common Core State Standards for US K-12 education. The Foundation awarded a contract to Applied Minds, LLC (AMI) to develop a Learning Map Data Model (LMDM) with the goal of it becoming a standard for educational technology infrastructure. The Learning Map will provide the organizing framework that maps the relationship between learning objectives, including dependencies and higher level groupings. It will also allow educational media resources, such as courses, books and web content to be linked to learning objectives. Curricula aligned to the standards will exhibit great diversity in highlighting paths through the Learning Map, and a suite of tools will allow authoring and visualization of the Learning Map. We believe that this data model will eventually enable the creation of online learning tools that are more responsive to the individual needs of a student. The LMDM is inspired by and based on the philosophy of the more general Knowledge Web (See: http://edge.org/conversation/aristotle-the-knowledge-web).

inBloom Technology

The Foundation has, in collaboration with the Carnegie Corporation, initiated an ambitious effort, Shared Learning Infrastructure (SLI), now inBloom Technology, to provide a new technology infrastructure that supports the Common Core Standards and the Foundation's vision, to be implemented by inBloom. AMI has been contracted by inBloom to build an implementation of a Learning Map and inBloom Index System, suitable for third-party software developers to populate content and develop applications.

inBloom Index Common Core Middleware API Specification

As part of the project, AMI developed a data model and API specification for the InBloom Index, formerly the Learning Registry Index (LRI), and is now delivering the inBloom Index Developer Server. To assist developers, and to handle the specific needs of the Common Core State Standards, AMI has implemented a sophisticated middleware layer that will enable easier access to the inBloom Index Developer Server.

This report presents the developer version (hence v0.9) of the inBloom Index Common Core Middleware API Specification. This document should be read in conjunction with the "inBloom Index Data Model and API Specification" report.

inBloom Index Common Core Middleware API

Introduction

The inBloom Index Common Core Middleware sits between the applications developers will write and the inBloom Index server. It provides a RESTFUL abstraction of inBloom Index queries.

The Middleware is at `http://knowledgeweb.appliedminds.com:9000`.

GET

Get all initiatives

```
/ccss/initiatives
```

Get all frameworks

```
/ccss/frameworks
```

Get all sets for a framework

```
/ccss/sets?framework=FRAMEWORK_ID
```

Get all grade_levels for a set (Math)

```
/ccss/grade_levels?set=MATH_SET_ID
```

Get all domains for a grade_level (Math)

```
/ccss/domains?grade_level=MATH_GRADE_LEVEL_ID
```

Get all standards for a domain (Math)

```
/ccss/standards?domain=MATH_DOMAIN_ID
```

Get all clusters for a domain (Math)

```
/ccss/clusters?domain=MATH_DOMAIN_ID
```

Get all standards for a cluster (Math)

```
/ccss/standards?cluster=CLUSTER_ID
```

Get all standard_components for a standard

```
/ccss/standard_components?standard=STANDARD_ID
```

Get all strands for a set (ELA)

```
/ccss/strands?set=ELA_SET_ID
```

Get all anchor_standards for a strand

`/ccss/anchor_standards?strand=ELA_STRAND_ID`

Get all domains for a framework (ELA)

`/ccss/domains?framework=ELA_FRAMEWORK_ID`

Get all domains for a strand (ELA)

`/ccss/domains?framework=ELA_STRAND_ID`

Get all grade_levels for a domain (ELA)

`/ccss/grade_levels?domain=ELA_DOMAIN_ID`

Get all anchor_standards for a strand (ELA)

`/ccss/anchor_standards?strand=STRAND_ID`

Get all standards for an anchor_standard (ELA)

`/ccss/standards?anchor_standard=ANCHOR_STANDARD_ID`

Get all anchor_standards for a standard (ELA)

`/ccss/anchor_standards?standard=ELA_STANDARD_ID`

Get all standards for a grade_level (ELA)

`/ccss/standards?grade_level=ELA_GRADE_LEVEL_ID`

Get all competency_paths

`/ccss/competency_paths`

Get competency_paths by author

`/ccss/competency_paths?author=AUTHOR_NAME`

Get specific competency_path

`ccss/competency_paths?competency_path=PATH_ID`

Get all learning_resources

`/ccss/learning_resources`

Get learning_resources by competency

`/ccss/learning_resources?standard=STANDARD_ID`

Get specific learning_resource

`/ccss/learning_resources?learning_resource=RESOURCE_ID`

Get all anchor_standard_sections

`/ccss/anchor_standard_sections`

Get all properties for an LRI entity type

`/ccss/TYPE/property_names`

- Example: `/ccss/initiative/property_names`

Parameters for GET

`format`	Specifies return format. Values: `"xml", "json"` Default value: `"json"`
`sort`	Sort returned data. Values: `"true", "false"` Default value: `"true"`
`property`	Return the specified property (and `urn:lri:property_type:id`) Values: Any `LRI property_type` May be repeated. Sorts by first property.
children	For standards, also returns standard_components. Values: "true", "false" Default value: "false"

Headers returned by GET

- `Access-Control-Allow-Origin`
- `Connection`
- `Content-Type`
- `Date`
- `Server`
- `Transfer-Encoding`
- `Webapp`
- `X-Powered-By`

Status codes returned by GET

- `200 OK`
- `404 Not Found`
- `500 Internal Server Error`

Example: Get all initiatives

```
http://knowledgeweb.appliedminds.com:9000/ccss/initiatives
```

Response


```
{
   "response": [
      {
         "initiatives": [
            {
               "props": {
                  "urn:ccss:property_type:ccid": "CCSS",
                  "urn:lri:property_type:contains": [
                     "urn:ccss:framework:CCSS.ELA-Literacy",
                     "urn:ccss:framework:CCSS.Math"
                  ],
                  "urn:lri:property_type:creator": "LRI_ADMIN_USER_0",
                  "urn:lri:property_type:guid":
"8c65411b42dcd41156dd961683feb539",
                  "urn:lri:property_type:id": "urn:ccss:initiative:CCSS",
                  "urn:lri:property_type:name": "Initiative CCSS",
                  "urn:lri:property_type:timestamp": "2012-12-
06T00:41:21.188750",
                  "urn:lri:property_type:types": [
                     "urn:ccss:entity_type:initiative",
                     "urn:lri:entity_type:competency",
                     "urn:lri:entity_type:competency_container",
                     "urn:lri:entity_type:learning_objective",
                     "urn:lri:entity_type:thing"
                  ]
               }
            },
            {
               "props": {
                  "urn:lri:property_type:contains":
"urn:inBloom:navigatorApp:initiative:bestInit:bestSub",
                  "urn:lri:property_type:creator": "LRI_ADMIN_USER_0",
                  "urn:lri:property_type:description": "bestInit",
                  "urn:lri:property_type:guid":
"f8b90444d8ec7225fe2b774ebcad28d8",
                  "urn:lri:property_type:id":
"urn:inBloom:navigatorApp:initiative:bestInit",
                  "urn:lri:property_type:name": "bestInit",
                  "urn:lri:property_type:timestamp": "2013-01-
19T08:42:20.315278",
                  "urn:lri:property_type:types": [
                     "urn:ccss:entity_type:competency_container",
                     "urn:ccss:entity_type:initiative",
                     "urn:lri:entity_type:competency",
                     "urn:lri:entity_type:learning_objective",
                     "urn:lri:entity_type:thing"
                  ]
               }
            },
            {
               "props": {
                  "urn:lri:property_type:contains":
"a8b7eaa865bd2bdc467093584f51b6d3",
```

```
                  "urn:lri:property_type:creator": "LRI_ADMIN_USER_0",
                  "urn:lri:property_type:description": "testInit",
                  "urn:lri:property_type:guid":
"10cc316a630a018fd701d3af624e2a82",
                  "urn:lri:property_type:id":
"urn:slc:navigatorApp:initiative:testInit",
                  "urn:lri:property_type:timestamp": "2013-01-
19T07:56:05.950174",
                  "urn:lri:property_type:types": [
                     "urn:ccss:entity_type:competency_container",
                     "urn:ccss:entity_type:initiative",
                     "urn:lri:entity_type:competency",
                     "urn:lri:entity_type:learning_objective",
                     "urn:lri:entity_type:thing"
                  ]
               }
            }
         ]
      }
   ],
   "status": "normal"
}
```


Example: Get all frameworks using "property" and "format" parameters

```
http://knowledgeweb.appliedminds.com:9000/ccss/frameworks?property=urn:lri:prop
erty_type:name&format=xml
```

Response


```
<?xml version="1.0" ?>
<root>
 <pair>
  <key>
   status
  </key>
  <value>
   normal
  </value>
 </pair>
 <pair>
  <key>
   response
  </key>
  <value>
   <pair>
    <key>
     frameworks
    </key>
    <value>
     <pair>
      <key>
       child_type
      </key>
      <value>
       urn:ccss:entity:type:domain
      </value>
     </pair>
     <pair>
```

```
      <key>
       props
      </key>
      <value>
       <pair>
        <key>
         urn:lri:property_type:name
        </key>
        <value>
         Framework CCSS CCSS.ELA-Literacy
        </value>
       </pair>
       <pair>
        <key>
         urn:lri:property_type:id
        </key>
        <value>
         urn:ccss:framework:CCSS.ELA-Literacy
        </value>
       </pair>
      </value>
     </pair>
    </value>
    <value>
     <pair>
      <key>
       child_type
      </key>
      <value>
       urn:ccss:entity:type:set
      </value>
     </pair>
     <pair>
      <key>
       props
      </key>
      <value>
       <pair>
        <key>
         urn:lri:property_type:name
        </key>
        <value>
         Framework CCSS CCSS.Math
        </value>
       </pair>
       <pair>
        <key>
         urn:lri:property_type:id
        </key>
        <value>
         urn:ccss:framework:CCSS.Math
        </value>
       </pair>
      </value>
     </pair>
    </value>
    <value>
     <pair>
      <key>
```

```
      child_type
     </key>
     <value>
      urn:ccss:entity:type:set
     </value>
    </pair>
    <pair>
     <key>
      props
     </key>
     <value>
      <pair>
       <key>
        urn:lri:property_type:name
       </key>
       <value>
        dd
       </value>
      </pair>
      <pair>
       <key>
        urn:lri:property_type:id
       </key>
       <value>
        urn:inBloom:navigatorApp:initiative.dd.dd
       </value>
      </pair>
     </value>
    </pair>
   </value>
   <value>
    <pair>
     <key>
      child_type
     </key>
     <value>
      urn:ccss:entity:type:set
     </value>
    </pair>
    <pair>
     <key>
      props
     </key>
     <value>
      <pair>
       <key>
        urn:lri:property_type:name
       </key>
       <value>
        f
       </value>
      </pair>
      <pair>
       <key>
        urn:lri:property_type:id
       </key>
       <value>
        urn:inBloom:navigatorApp:initiative.f.f
       </value>
```

```
       </pair>
      </value>
     </pair>
    </value>
   </pair>
  </value>
 </pair>
</root>
```

Example: Get all framework properties

```
http://knowledgeweb.appliedminds.com:9000/ccss/frameworks/property_names
```

Response


```
{
   "response": [
      {
         "property_names": [
            "urn:ccss:property_type:ccid",
            "urn:lri:property_type:ancestors",
            "urn:lri:property_type:assessed_by",
            "urn:lri:property_type:comment_plain",
            "urn:lri:property_type:contained_by",
            "urn:lri:property_type:creator",
            "urn:lri:property_type:description",
            "urn:lri:property_type:guid",
            "urn:lri:property_type:id",
            "urn:lri:property_type:is_path_step",
            "urn:lri:property_type:label",
            "urn:lri:property_type:name",
            "urn:lri:property_type:properties",
            "urn:lri:property_type:required_by",
            "urn:lri:property_type:specified_by",
            "urn:lri:property_type:supertypes",
            "urn:lri:property_type:taught_by",
            "urn:lri:property_type:timestamp",
            "urn:lri:property_type:types",
            "urn:lri:property_type:uri",
            "urn:lri:property_type:url"
         ]
      }
   ],
   "status": "normal"
}
```


Example: Get all frameworks in iPython console

```
ipython
import json
import requests
url = 'http://knowledgeweb.appliedminds.com:9000/ccss/frameworks'
response = requests.get(url)
```

Response

```
print(response.status_code)
200
print(response.reason)
OK
print(response.headers)
{'date': 'Sat, 19 Jan 2013 21:06:06 GMT', 'connection': 'keep-alive', 'x-
powered-by': 'Python', 'transfer-encoding': 'chunked', 'webapp': 'ccss',
'access-control-allow-origin': '*', 'server': 'nginx/0.8.54'}
print(json.dumps(response.json, indent=3))
{
   "status": "normal",
   "response": [
      {
         "frameworks": [
            {
               "child_type": "urn:ccss:entity:type:domain",
               "props": {
                  "urn:lri:property_type:contained_by":
"urn:ccss:initiative:CCSS",
                  "urn:lri:property_type:name": "Framework CCSS CCSS.ELA-
Literacy",
                  "urn:lri:property_type:guid":
"56c8cb1d15865a9280721b2b352771e2",
                  "urn:ccss:property_type:ccid": "CCSS.ELA-Literacy",
                  "urn:lri:property_type:timestamp": "2012-12-
06T00:37:12.770659",
                  "urn:lri:property_type:creator": "LRI_ADMIN_USER_0",
                  "urn:lri:property_type:types": [
                     "urn:ccss:entity_type:framework",
                     "urn:lri:entity_type:competency",
                     "urn:lri:entity_type:competency_container",
                     "urn:lri:entity_type:learning_objective",
                     "urn:lri:entity_type:thing"
                  ],
                  "urn:lri:property_type:contains": [
                     "urn:ccss:domain:CCSS.ELA-Literacy.L",
                     "urn:ccss:domain:CCSS.ELA-Literacy.RF",
                     "urn:ccss:domain:CCSS.ELA-Literacy.RH",
                     "urn:ccss:domain:CCSS.ELA-Literacy.RI",
                     "urn:ccss:domain:CCSS.ELA-Literacy.RL",
                     "urn:ccss:domain:CCSS.ELA-Literacy.RST",
                     "urn:ccss:domain:CCSS.ELA-Literacy.SL",
                     "urn:ccss:domain:CCSS.ELA-Literacy.W",
                     "urn:ccss:domain:CCSS.ELA-Literacy.WHST",
                     "urn:ccss:set:CCSS.ELA-Literacy.CCRA"
                  ],
                  "urn:lri:property_type:id": "urn:ccss:framework:CCSS.ELA-
Literacy"
               }
            },
            {
               "child_type": "urn:ccss:entity:type:set",
               "props": {
                  "urn:lri:property_type:contained_by":
"urn:ccss:initiative:CCSS",
                  "urn:lri:property_type:name": "Framework CCSS CCSS.Math",
```

```
                  "urn:lri:property_type:guid":
"937de90e191bf7cb0aa4d009df199609",
                  "urn:ccss:property_type:ccid": "CCSS.Math",
                  "urn:lri:property_type:timestamp": "2012-12-
06T00:37:16.019032",
                  "urn:lri:property_type:creator": "LRI_ADMIN_USER_0",
                  "urn:lri:property_type:types": [
                     "urn:ccss:entity_type:framework",
                     "urn:lri:entity_type:competency",
                     "urn:lri:entity_type:competency_container",
                     "urn:lri:entity_type:learning_objective",
                     "urn:lri:entity_type:thing"
                  ],
                  "urn:lri:property_type:contains": [
                     "urn:ccss:set:CCSS.Math.Content",
                     "urn:ccss:set:CCSS.Math.Practice"
                  ],
                  "urn:lri:property_type:id": "urn:ccss:framework:CCSS.Math"
               }
            }
         ]
      }
   ]
}
```


POST

Use POST to create new entities and update existing entities. XML templates are provided for create and update.

Example: Get the XML template for creating an initiative

```
http://knowledgeweb.appliedminds.com:9000/ccss/create/template/initiative/xml
```

Response

```
<xml>
  <pair key="uid">
    <value>UNIQUE_ID</value>
  </pair>
  <pair key="urn:lri:property_type:contains">
    <value>COMPETENCY_CONTAINER_ID</value>
  </pair>
  <pair key="urn:lri:property_type:creator">
    <value>CREATOR_NAME</value>
  </pair>
  <pair key="urn:lri:property_type:description">
    <value>DESCRIPTION</value>
  </pair>
  <pair key="urn:lri:property_type:id">
    <value>ID</value>
  </pair>
  <pair key="urn:lri:property_type:name">
    <value>NAME</value>
  </pair>
</xml>
```

Create an initiative

/ccss/initiative/create

Create a competency_path

/ccss/create/template/competency_path/xml
/ccss/competency_path/create

Create a learning_resource

/ccss/create/template/learning_resource/xml
/ccss/learning_resource/create

Templates for other types

/ccss/create/template/framework/xml
/ccss/create/template/set/xml
/ccss/create/template/competency_container/xml
/ccss/create/template/competency/xml

Create other types

/ccss/initiative/create
/ccss/framework/create
/ccss/set/create
/csss/competency_container/create
/ccss/competency/create

XML for updating an entity

All entity types are updated using the same XML template. The Middleware uses the provided ID and property names to look up the property GUIDS in the LRI.

```
http://knowledgeweb.appliedminds.com:9000/ccss/update/template/update/xml

<xml>
  <pair key="urn:lri:property_type:id">
    <value>ENTITY_ID</value>
  </pair>
  <pair key="updates">
    <value>
      <pair key="property">
        <value>PROPERTY_1_NAME</value>
        <value>NEW_PROPERTY_1_VALUE</value>
        <value>PROPERTY_2_NAME</value>
        <value>NEW_PROPERTY_2_VALUE</value>
      </pair>
    </value>
  </pair>
</xml>
```

Update existing entity

`/ccss/update`

Body of POST request must contain

`data`	XML of new entity
`format`	Format of data sent to the server Values: "`xml`"

Headers returned by POST

- `Access-Control-Allow-Origin`
- `Connection`
- `Content-Type`
- `Date`
- `Server`
- `Transfer-Encoding`
- `Webapp`
- `X-Powered-By`

Status codes returned by POST

- `201 Created`
- `404 Not Found`
- `500 Internal Server Error`

Example: Create an initiative in iPython console

```
ipython
import requests
url = 'http://knowledgeweb.appliedminds.com:9000/ccss/initiative/create'
# Using XML template from
http://knowledgeweb.appliedminds.com:9000/ccss/create/template/initiative/xml
# Fill in <value></value> tags with your data
postData = {
  'format': 'xml',
  'data': '<xml>
  <pair key="uid">
    <value>Example</value>
  </pair>
  <pair key="urn:lri:property_type:contains">
    <value>COMPETENCY_CONTAINER_ID</value>
  </pair>
  <pair key="urn:lri:property_type:creator">
    <value>CREATOR_NAME</value>
```

```
  </pair>
  <pair key="urn:lri:property_type:description">
    <value>DESCRIPTION</value>
  </pair>
  <pair key="urn:lri:property_type:id">
    <value>ID</value>
  </pair>
  <pair key="urn:lri:property_type:name">
    <value>NAME</value>
  </pair>
</xml>'
}
response = requests.post(url, data=postData)
```

Example: Update an existing initiative in iPython console

```
ipython
import requests
url = 'http://knowledgeweb.appliedminds.com:9000/ccss/update'
# Use XML template from
http://knowledgeweb.appliedminds.com:9000/ccss/update/template/update/xml
# Fill in <value></value> tags with your data
postData = {
  'format': 'xml',
  'data': '<xml>
    <pair key="urn:lri:property_type:id">
      <value>ENTITY_ID</value>
    </pair>
    <pair key="updates">
      <value>
        <pair key="property">
          <value>PROPERTY_NAME</value>
          <value>NEW_VALUE</value>
        </pair>
     </value>
   </pair>
</xml>'
}
response = requests.post(url, data=postData)
```