The Learner Record Domain Model

A Holistic Examination

Jeff Merriman, Tom Coppeto, Scott Thorne

DXtera is often tasked with integrating systems or aligning with standards around various aspects of "Learner Records". To achieve this, DXtera turns to its holistic models to help align functionality and data between formal and informal educational systems, interoperability standards and open or proprietary data schemas.

This document examines the models related to the Learner Record and how they fit with other models found across formal and informal learning environments, and educational and academic processes. If you wish to skip the preamble and just dive into the models, feel free to go directly to the Chronicle Model, the starting place for this analysis.

The Models of the Open Service Interface Definitions

For its holistic model DXtera turns to the domain models of the Open Service Interface Definitions (OSID) which define the broadest set of entities that describe educational and learning functions.[1] The complete OSID model currently comprises functional and informational definitions across over 400 entities found across various kinds of educational organizations and processes, and draws cardinality across models to provide a holistic domain representation of educational learning and business systems.

Due to the sheer size and number of OSID models it can be daunting to take in all at once. It is more useful to explore them from the perspective of a concrete problem being addressed; like a software development process, integration requirement, development of a particular data standard, or simply studying a discrete information domain or process to gain deeper understanding.

The "Learner Record" is a good example of a domain that can be readily examined, but also leads to other models based on the interconnectedness of the overall holistic model.

A Matter of Perspective

This document begins its examination from the perspective of a learner record, but by definition everything is interconnected in a holistic model. Like the proverbial onion, it has layers. We choose the domain we want to investigate, and then layer after layer are peeled back to uncover the supporting sub-models, and for any particular investigation, we must eventually decide where to stop. In this document we will peel back the "learner record" on an entity by entity basis until it feels like we've gone far enough, for now.

[1] The complete OSID Specification is published and openly licensed at http://osid.org

The same can be said for any other perspective from which we might start. For instance, attempting to examine “program requirements” or “educational pathways” will require peeling back the models from those perspectives. Such an investigation may ultimately arrive at the core models for a learner record if we choose to peel back enough layers.

Entities and Functions

Most of today’s standards efforts focus on data models or schemas, and perhaps basic get, post, and put kinds of functions depending on whether the goal is to define a data packaging standard or a protocol. This approach is often incomplete. The more interesting aspects of a holistic model for the purposes of integration are found in defining the broadest set of required functional operations. For example, in the case where entities tend to live in hierarchies, special functions need to be defined for designing and traversing those hierarchies. These functional areas also support a key part of the authorization[2] model required for access control, but which will not be covered here.

Each section of this document outlines both the data/information aspects of the models as well as the functional aspects. Functional operations make up a large percentage of the OSID definitions and essentially define interactions across systems that will result in meaningful real-time integration. They are agnostic to underlying data persistence or transport technology. Interface definitions or protocol standards derived from these functional models should not care whether information is stored in a database, in filespace or on a blockchain distributed ledger.

There are a number of functional patterns that are common across all the entities examined, and to list out the common patterns for each would make this document unnecessarily long. The following table outlines these common functions and will be referenced in each section:

Read Functions Common to all Entities

getEntity (by Id)
getEntities (by list of Ids, attribute query, by Catalogs, or just get all Entities)

Write Functions Common to Entities

createEntity updateEntity (by Id), deleteEntity (by Id)
aliasEntity (assign an alternate Id with which to reference a given Entity, by Id)

Notification Functions Common to all Entities

registerForNewEntities, registerForChangedEntities, registerForDeletedEntities
registerForChangedEntity (by Id), registerForDeletedEntity (by Id)

Cataloging Functions Common to all Entities

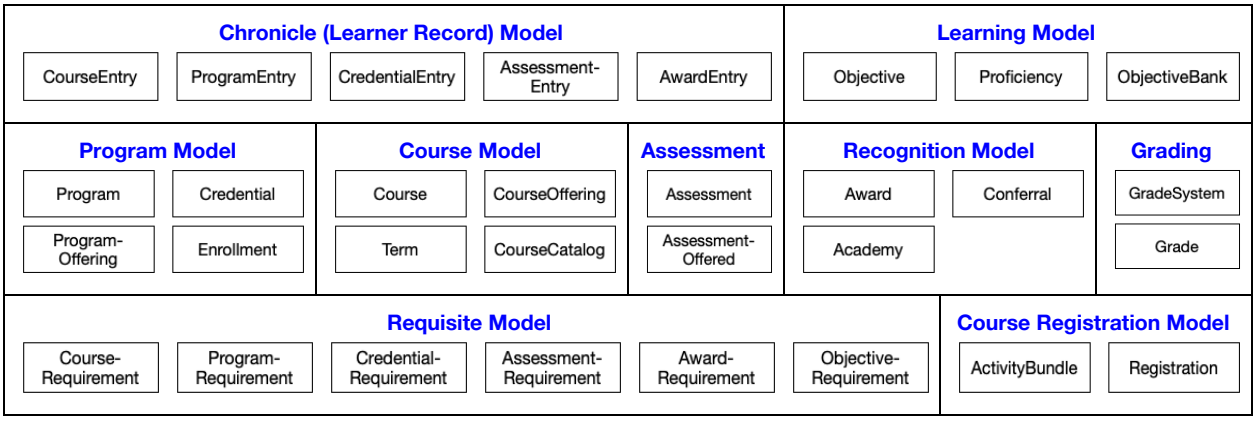
getCatalogsForEntities (by Id list of Ids. Exposes organization structures)
assignEntityToCatalog, unassignEntityFromCatalog, reassignEntityToCatalog (by Entity and Catalog Id)

[2] The Authorization model will not be covered in this document. For more information please reference the OSID Authorization specification at http://osid.org/specifications/osid/authorization/package.html

Getting Started

The following sections provide definitions at the “sub-domain” and entity level, starting with the Chronicle model that defines “Learner Record” information from the perspective of a formal educational record, followed by the Learning model which deals with learning outcomes, competencies and skills as well as learner Proficiency from a strictly skills or outcome based perspective.

Following those models are sub-domains and entities required to be defined to support key attributes of the learner records; things like Courses, Programs, Assessments, Awards. Following these are more tertiary models related to Requirements and Registrations that are also referenced in the primary and secondary models, but may or may not be necessary for supporting learner record integration, depending on the interoperability goals.



Each model includes one or more entities. These entities in turn define *Attributes*[3], and attribute *Types.* Some attribute types are primitive, like STRINGs, BOOLEANs, and DECIMALS. Other primitive types that are more complex include Id, Type, DateTime, Duration, and DisplayText which are defined further in the Primitive Objects section.

All other attribute types refer to other entities covered across the models, most of which appear in this examination, but some do not. There is a section at the end that touches on parts of the holistic model that are referenced, but that we have chosen not to examine further in this document.

In translating these models for a particular use, say serializing a learner record so it can be shared, decisions will have to be made regarding when attributes should be serialized to represent the entire entity with all of its corresponding attributes, and when they should just reference an Id. Doing the latter assumes that inspecting the data referenced by the Id will require a separate functional operation to get that object’s information.

Let’s get started!

[3] In the OSID specifications attributes are defined in the entity interfaces and expressed through programmatic methods. This allows the specifications to be truly agnostic to underlying technology. For instance, there is no inherent assumption that attributes are explicitly persisted at all. Certain attributes might be algorithmically derived.

The Chronicle (Learner Record) Model

<table>
<tr><td colspan="3">Chronicle (Learner Record) Model</td><td colspan="2">Learning Model</td></tr>
<tr><td>Program Model</td><td>Course Model</td><td>Assessment</td><td>Recognition Model</td><td>Grading</td></tr>
<tr><td colspan="3">Requisite Model</td><td colspan="2">Course Registration Model</td></tr>
</table>

A Chronicle represents a “Learner Record”. The entries in the record provide summary information for learners in formal, or “institutional” programs and courses as well as any credentials, awards, or assessment scores the learner may have received. In higher education this is commonly called an “Academic Record”.[4]

A system managing learner record information will support basic functionality to read one or more types of Learner Record entries for a particular student:

Read Functionality for a Learner’s Record

getEnrolledProgramEntiresForStudent (by id)	ProgramEntry[][5]
getCompletedProgramEntriesForStudent (by id)	ProgramEntry[]
getProgramEntriesForStudent (by id, optionally by date range or by Term)	ProgramEntry[]
getCompletedCourseEntriesForStudent (by id)	CourseEntry[]
getCourseEntriesForStudent (by id, optionally by date range or by Term)	CourseEntry[]
getCredentialEntriesForStudent (by id, optionally by date range or by Term)	CredentialEntry[]
getAssessmentEntriesForStudent (by id, optionally by date range or by Term)	AssessmentEntry[]
getAwardEntriesForStudent (by id, optionally by date range or by Term)	AwardEntry[]

Note that common functionality for fetching in bulk or querying across student populations are available through the common read functions for each kind of learner record entry as referenced in each of the Entry definitions below.

ProgramEntry

A learner record entry for a Program. The fields in the ProgramEntry may be for the entire enrollment through the academic career or specific to a Term.

Attribute	***Type***	***Definition***
id	Id	The unique id
displayName	DisplayText	The name for display
description	DisplayText	The description of this entry
genusType	Type	The type of this entry

[4] Ref. OSID Chronicle model: http://osid.org/specifications/osid/course/chronicle/package.html
[5] Note that a [] next to an object type means that a list of zero or many may be returned.

startDate	DateTime	Start date of this entry
endDate	DateTime	End date of this entry
endReason	State	The reason that this entry ended, if applicable
admissionDate	DateTime	The date in which the student was admitted
complete	BOOLEAN	Whether the program has been completed. If this entry is for summary information in a past term, this may be true
term	Term	The term, if this entry is a progression entry applying to a single term
creditScale	GradeSystem	The credit scale
creditsEarned	DECIMAL	The number of credits earned in this program or earned within the included term
gpaScale	GradeSystem	The GPA scale, if a GPA is available
gpa	DECIMAL	The GPA, if one is available
program	Program	The program associated with this entry
student	Resource	The student associated with this entry

Common Functions:

Read, Write, Notification and Cataloging Functions for ProgramEntries

CourseEntry

A learner record entry for a Course. The fields in the CourseEntry are specific to a Term

Attribute	***Type***	***Definition***
id	Id	The unique id
displayName	DisplayText	The name for display
description	DisplayText	The description of this entry
genusType	Type	The type of this entry
startDate	DateTime	Start date of this entry
endDate	DateTime	End date of this entry
endReason	State	The reason that this entry ended, if applicable
term	Term	The term associated with this entry
complete	BOOLEAN	Whether the course has been completed
creditScale	GradeSystem	The credit scale
creditsEarned	DECIMAL	The number of credits earned in this course
grade	Grade	The grade, if available
scoreScale	GradeSystem	The grade system, if a cumulative score is available
score	DECIMAL	The cumulative score, if available

registrations	Registration[]	The registrations, if available
course	Course	The course associated with this entry
student	Resource	The student associated with this entry

Common Functions:

Read, Write, Notification and Cataloging Functions for CourseEntries

CredentialEntry

A learner record entry for a Credential. Might also be related to a "badge" awarded to a learner based on completion of an educational Program.

Attribute	***Type***	***Definition***
id	Id	The unique id
displayName	DisplayText	The name for display
description	DisplayText	The description of this entry
genusType	Type	The type of this entry
startDate	DateTime	Start date of this entry
endDate	DateTime	End date of this entry
endReason	State	The reason that this entry ended, if applicable
dateAwarded	DateTime	The award date
program	Program	The program associated with this entry, if available
credential	Credential	The credential associated with this entry
student	Resource	The student associated with this entry

Common Functions:

Read, Write, Notification and Cataloging Functions for CredentialEntries

AssessmentEntry

A learner record entry for an Assessment. This is intended to record results of standardized tests and other major assessments associated with a learner's record. Tests or quizzes within a course or as part of independent learning are tracked elsewhere in the holistic model, and can be part of a more "comprehensive" learner record that might include actual evidence of learning.

Attribute	***Type***	***Definition***
id	Id	The unique id
displayName	DisplayText	The name for display
description	DisplayText	The description of this entry
genusType	Type	The type of this entry
startDate	DateTime	Start date of this entry

endDate	DateTime	End date of this entry
endReason	State	The reason that this entry ended, if applicable
dateCompleted	DateTime	The completion date
program	Program	The program associated with this entry, if available
course	Course	The course associated with this entry, if available
grade	Grade	The grade associated with this entry, if available
scoreScale	GradeSystem	The grade system for the score, if a score is available
score	DECIMAL	The score, if available
assessment	Assessment	The assessment associated with this entry
student	Resource	The student associated with this entry

Common Functions:

Read, Write, Notification and Cataloging Functions for AssessmentEntries

AwardEntry

An academic record entry for a Recognition. Might also be related to a "badge" conferred upon a learner.

Attribute	***Type***	***Definition***
id	Id	The unique id
displayName	DisplayText	The name for display
description	DisplayText	The description of this entry
genusType	Type	The type of this entry
startDate	DateTime	Start date of this entry
endDate	DateTime	End date of this entry
endReason	State	The reason that this entry ended, if applicable
dateAwarded	DateTime	The award date
program	Program	The program associated with this entry, if available
course	Course	The course associated with this entry, if available
assessment	Assessment	The assessment associated with this entry, if available
award	Award	The award associated with this entry
student	Resource	The student associated with this entry

Common Functions:

Read, Write, Notification and Cataloging Functions for AwardEntries

Regarding Comprehensive Learner Records

This examination does not currently include “evidence of learning” which may be considered part of a more comprehensive learner record model. Initiatives aimed at comprehensiveness include, in addition to traditional learner record information, actual evidence of learning or skill, whether institutionally verifiable or not. This evidence might include assignments or assessments that have been taken by the learner, along with the learner’s actual submissions or responses and instructor feedback and grades.

Much of the holistic model for including this evidence, and defining integration points with systems that hold this kind of information, would be found across entities of the Assessment domain that will not be covered in this document. These can be explored in a future version of this examination extended to include more comprehensive information as required.

Regarding Badges

A learner can be assessed on a learning Objective, resulting in a Proficiency being drawn between the learner and the Objective. In the holistic model, formal learning Credentials are not directly tied to Objectives, and can only be issued through completion of a Program. This may be a little at odds with some perspectives of “badges” which have become a popular kind of credentialing along with increasing on-line educational opportunities, and are often issued upon completion of a learning goal. There are, however, a couple of ways to model learning outcome based “badges” depending on how formal or informal the issuer of the badge wishes to be:

1. For most on-line learning activities through which badges are issued upon achieving a learning Objective, the more formal academic Program model can be used. In this model Credentials are awarded upon successful completion of Programs, and a CredentialEntry can be inserted into the learner’s record. A Program can be of any size and complexity, and could be as simple as a Program which has as its completion requirements one LearningObjectiveRequirement that defines a minimum Proficiency level, as defined in the Learning model. More typically, the minimum requirement for an on-line program might be an AssessmentRequirement that indicates achievement of a learning objective or mastery of skill, etc.

2. A badge can be awarded as part of the Recognition model. This is more informal and doesn’t carry along with it all the trappings of an educational process. In this case a badge would map to an Award in the model. While the Chronicle can include entries for Awards, The Recognition model itself also defines the conferral of Awards, where a Conferral can optionally reference another entity, like a learning outcome if required.

The problem with equating badges directly with Proficiencies within a learning outcome or competency model is that competency or LO frameworks might be shared among multiple awarding or credentialing organizations. Awards and Credentials include references to an issuing authority, which is necessary if a badge is to be taken seriously by an interested third party. The issuing authority may be an organization (like a school or other educational institution), a person, or even a machine algorithm. In any case, someone reviewing the badge will want to know who or what the issuing authority is. For this reason, the

mechanism by which such artifacts are issued resides in these other models that will include issuing information in their own ways.

Both Awards and Credentials can be exposed through the Learner Record as AwardEntries or CredentialEntries.

Regarding Stacked Credentials and Micro-Credentials

As we will see later in the Requisite Model, It is notable that a Program can have Requisites that in turn require other “sub-programs” to be successfully completed, which can result in associated ProgramEntries and CredentialEntries in a learner record. This model, and definition of the word “program” to also include these supporting programs, is not generally embraced among typical educational institutions. It does, however, lead to a very powerful and flexible structure for designing Programs that require various groupings of Courses or other activities to be completed, and eventually rolled-up into the kinds of “degree granting programs” that are more recognizable to most in higher education.

The model outlined here should feel familiar to those that are promoting and designing “micro-credentials” and/or “stacked credential programs”. Developing such programs usually entails designing a hierarchical organization of supportive programs that have as requirements the completion of yet more supportive programs or some selection of courses or other learning activities that can be completed. These supportive programs might produce a Credential intended to be meaningful to an employer, regardless of whether an individual has yet to complete the overall degree Program. This kind of Program design is well supported by the Program/Requisites model.

The Learning Model

<table>
<tr><td colspan="3">Chronicle (Learner Record) Model</td><td colspan="3">Learning Model</td></tr>
<tr><td>Program Model</td><td>Course Model</td><td>Assessment</td><td colspan="2">Recognition Model</td><td>Grading</td></tr>
<tr><td colspan="4">Requisite Model</td><td colspan="2">Course Registration Model</td></tr>
</table>

While traditional educational institutions organize learning activities into programs and courses, learning may be more accurately defined by learning Objectives and similar entities. A learning Objective describes measurable learning goals at varying levels of detail. Objectives may also be known as "competencies", "learning outcomes", "skills", etc.[6]

Objective

A statable learning objective or goal. Objectives describe measurable learning goals. A learning objective may be measured by a related Assessment. Objectives may be mapped to levels, A level is represented by a Grade which is used to indicate an educational grade level or level of difficulty.

Objectives are hierarchical. An Objective with children represents an objective that is inclusive of all its children. For example, an Objective that represents learning in arithmetic may be composed of objectives that represent learning in both addition and subtraction. Objectives may also have requisites. A requisite Objective is one that should be achieved before another Objective is attempted, which can then define learning "pathways".

Attribute	***Type***	***Definition***
id	Id	The unique id
displayName	DisplayText	The name for display
description	DisplayText	The description of this objective
genusType	Type	The type of this objective
assessment	Assessment	The assessment associated with this learning objective, if available
knowledgeCategory	Grade	The knowledge category of this learning objective, if available
cognitiveProcess	Grade	The cognitive process of this learning objective, if available

Common Functions:

Read, Write, Notification and Cataloging Functions for Objectives

Hierarchy Traversal Functions:

getChildObjectives (for Id)	Get all child Objectives for a given Objective Id
getParentObjectives (for Id)	Get all parent Objectives for a given Objective Id
getRootObjectives (for Id)	Get all root Objectives for a given Objective Id

[6] Ref. OSID Learning model: http://osid.org/specifications/osid/learning/package.html

Hierarchy Design Functions:

addRootObjective (Id)	Add a given Objective as a “root” in the hierarchy
addChildObjective (id for id)	Add an Objective as a child of another Objective
removeChildObjective (Id for Id)	Remove a child Objective from an Objective
removeChildObjectives (Id)	Remove all child Objectives from an Objective

Requisite Traversal Functions

getRequisiteObjectives (for id)	Get all requisites of a given Objective
getDependantObjectives (for Id)	Get all dependents of a given Objective
getEquivelantObjectives (for Id)	Get all equivalent Objectives given an Objective

Requisite Assignment Functions

assignObjectiveRequisite (Id for id)	Create a requirement dependency between two Objectives
unassignObjectiveRequisite (Id for id)	Remove a requisite Objective from an Objectives
assignEquivelantObjective (Id, id)	Make one Objective equivalent to another
unassignEquivelantRequisite (Id, id)	Remove equivelancy between two Objectives

Activity

Represents learning material or other learning activities to meet an Objective. An Activity may relate to a set of Assets for self learning, recommended Courses to take, or a learning Assessment. These Activity Assessments differs from the Objective Assessment in that these are used for aiding in learning while the Objective Assessment is used to test for proficiency in the Objective

Attribute	***Type***	***Definition***
id	Id	The unique id
displayName	DisplayText	The name for display
description	DisplayText	The description of this activity
genusType	Type	The type of this activity
assets	Asset[]	The assets associated with this activity, if available
courses	Course[]	The courses associated with this activity, if available
programs	Program[]	The programs associated with this activity, if available
assessments	Assessment[]	The assessments associated with this activity, if available
objective	Objective	The learning objective associated with this activity

Common Functions:

Read, Write, Notification and Cataloging Functions for Activities

Proficiency

Represents an individual's competency with a learning Objective. Proficiencies typically relate a person to an Objective. But any other kind of resource can be related to Objectives through Proficiencies. For instance, the extent to which a Course is able to teach an Objective could also be measured through Proficiency and measured through accreditation or student course evaluations.

Attribute	***Type***	***Definition***
id	Id	The unique id
displayName	DisplayText	The name for display
description	DisplayText	The description of this proficiency
genusType	Type	The type of this proficiency
startDate	DateTime	Start date of this proficiency
endDate	DateTime	End date of this proficiency
endReason	State	The reason that this entry ended, if applicable
completion	DECIMAL	The completion of this objective as a percentage 0-100.
level	Grade	The proficiency level expressed as a grade, if available
resource	Resource	The resource to whom this proficiency applies. Usualyl a person, but could be anything
objective	Objective	The learning objective to which this proficiency applies

Common Functions:

Read, Write, Notification and Cataloging Functions for Proficiencies

ObjectiveBank (Catalog, Competency Framework)

A Catalog for all "learning" related entities. The term Catalog is generic, but other terms may be used to define collections of Outcomes or other Learning related entities. The term "Framework" is often used as an alternate name to describe an organization of competencies.

Attribute	***Type***	***Definition***
id	Id	The unique id
displayName	DisplayText	The name for display
description	DisplayText	The description of this catalog
genusType	Type	The type of this catalog
provider	Resource	The resource representing the provider of this catalog, if available
branding	Asset[]	The branding, such as an image or logo, if available
license	DisplayText	The terms of usage. An empty license means the terms are unknown

Common Functions:

Read, Write, Notification Functions for ObjectiveBanks	
Hierarchy Functions:	
getChildObjectiveBanks (for Id)	Get all child banks for a given bank Id
getParentObjectiveBanks (for Id)	Get all parent banks for a given bank Id
getRootObjectiveBanks (for Id)	Get all root banks for a given bank Id
Hierarchy Design Functions:	
addRootObjectiveBank (Id)	Add a given bank as a “root” in the hierarchy
addChildObjectiveBank (id for id)	Add a given bank as a child of another bank
removeChildObjectiveBank (Id for Id)	Remove a child bank from another bank
removeChildObjectiveBanks (Id)	Remove all child banks from a bank

The Program Model

Chronicle (Learner Record) Model			Learning Model	
Program Model	Course Model	Assessment	Recognition Model	Grading
Requisite Model				Course Registration Model

The program model focuses on the Program as a canonical learning unit describing the overall content of a program, completion requirements, and earned credentials.[7] Courses may be associated with Programs, but completion requirements are managed through the Requisite Model.

Program

A canonical program instantiated for offering through the creation of a ProgramOffering

Attribute	***Type***	***Definition***
id	Id	The unique id
displayName	DisplayText	The name for display
description	DisplayText	The description of this program
genusType	Type	The type of this program
title	DisplayText	The formal title of this program. It may be the same as the display name or it may be used to more formally label the program
number	STRING	The program number which is a label generally used to index the program in a catalog, such as "16c.
sponsors	Resource[]	The sponsors, if applicable. Usually Organizations
completionRequirementsInfo	DisplayText	An informational string for the program completion
completionRequirements	Requisite[]	The requisites for the program completion, if available. Each requisite is an AND term and must be true for the requirements to be satisfied
credentials	Credential[]	The awarded credentials, if applicable
learningObjectives	Objective[]	The overall competencies or learning objectives for this program, if available.

Common Functions:

Read, Write, Notification and Cataloging Functions for Programs

[7] Ref. OSID Program model: http://osid.org/specifications/osid/course/program/package.html

ProgramOffering

A ProgramOffering represents a Program offered during a Term. Note that many of the attributes for ProgramOffering may simply reflect the attributes of the associated Program, or may be different.

Attribute	***Type***	***Definition***
id	Id	The unique id
displayName	DisplayText	The name for display
description	DisplayText	The description of this offering
genusType	Type	The type of this offering
startDate	DateTime	Start date of this offering
endDate	DateTime	End date of this offering
endReason	State	The reason that this offering ended, if applicable
title	DisplayText	The formal title of this program offering. It may be the same as the display name or it may be used to more formally label the program
number	STRING	Gets the program number which is a label generally used to index the program offering in a catalog, such as 16
sponsors	Resource[]	The sponsors, if applicable. Usually Organizations
completionRequirementsInfo	DisplayText	An informational string for the program completion
completionRequirements	Requisite[]	The requisites for the program completion, if available. Each requisite is an AND term and must be true for the requirements to be satisfied
credentials	Credential[]	The awarded credentials, if applicable
requiresRegistration	BOOLEAN	Whether this program offering requires advanced registration
minimumSeats	CARDINAL	The minimum number of students this offering can have
maximumSeats	CARDINAL	The maximum number of students this offering can have, if applicable
url	STRING	An external resource, such as a web site describing the program of study, associated with this offering.
program	Program	The canonical program associated with this offering
term	Term	The Term of this offering

Common Functions:

Read, Write, Notification and Cataloging Functions for ProgramOfferings

Credential

Something awarded at program completion. This might also be related to a “badge”.

Attribute	***Type***	***Definition***
id	Id	The unique id
displayName	DisplayText	The name for display
description	DisplayText	The description of this offering
genusType	Type	The type of this offering
lifetime	Duration	The lifetime of this credential once awarded

Common Functions:

Read, Write, Notification and Cataloging Functions for Credentials

Enrollment

An Enrollment is a relationship between a student and a ProgramOffering. Note that while Enrollment is part of the holistic model but not strictly needed as part of this Learning Record investigation. It can be added later if determined to be required.

The Course Model

<table>
<tr><td colspan="3">Chronicle (Learner Record) Model</td><td colspan="2">Learning Model</td></tr>
<tr><td>Program Model</td><td>Course Model</td><td>Assessment</td><td>Recognition Model</td><td>Grading</td></tr>
<tr><td colspan="3">Requisite Model</td><td colspan="2">Course Registration Model</td></tr>
</table>

These entities define course management services for educational offerings or other learning units related to an event or curriculum. A Course may be used to model a class, conference tutorial session, or even a meetup group.[8]

This model defines two sets of entities. Courses and ActivityUnits represent canonical curriculum. Canonical Courses and ActivityUnits includes the description of the curriculum, requirements, and learning objectives independent of any offering. CourseOfferings and Activities are offerings of a canonical unit in a Term. Activities and ActivityUnits will not be covered in this examination, as they are not typically included in a learner record. They can be added later.

Course

A canonical course that manages the curriculum of learning units.

Attribute	***Type***	***Definition***
id	Id	The unique id
displayName	DisplayText	The name for display
description	DisplayText	The description of this course
genusType	Type	The type of this course
title	STRING	The formal title of this course. It may be the same as the display name or it may be used to more formally label the course
number	STRING	The course number which is a label generally used to index the course in a catalog, such as T101 or 16.004.
sponsors	Resource[]	Sponsors of this course, if applicable. Usually Organizations
credits	Grade[]	The credits in which this course can be offered
prerequisitesInfo	STRING	An informational string for the course prerequisites
prerequisites	Requisite[]	The requisites for the course prerequisites, if available. Each requisite is an AND term such that all requisites must be true for the prerequisites to be satisfied
levels	Grade[]	The grade levels of this course. Multiple levels may exist
gradingOptions	GradeSystem[]	The various grading options available to register for, if applicable

[8] Ref. OSID Course model: http://osid.org/specifications/osid/course/package.html

learningObjectives	Objective[]	The overall learning objectives for this course, if available

Common Functions:

Read, Write, Notification and Cataloging Functions for Courses

Term

A period of time in which a course is offered. This Term contains information on several milestones that may drive the behavior of schedules and registration

Attribute	***Type***	***Definition***
id	Id	The unique id
displayName	DisplayText	The name for display
description	DisplayText	The description of this term
genusType	Type	The type of this term
displayLabel	STRING	A display label for this term which may be less formal than the display name
openDate	DateTime	The open date when published offerings are finalized, if applicable
registrationStart	DateTime	The start of the registration period, if applicable
registrationEnd	DateTime	The end of the registration period, if applicable
classesStart	DateTime	The start of classes
classesEnd	DateTime	The end of classes
addDate	DateTime	The add (add/drop) date, if applicable
dropDate	DateTime	The drop (add/drop) date, if applicable
finalExamStart	DateTime	The start of the final exam period, if applicable
finalExamEnd	DateTime	The end of the final exam period, if applicable
closeDate	DateTime	The close date when results of course offerings are finalized, if applicable

Common Functions:

Read, Write, Notification and Cataloging Functions for Terms

Hierarchy Traversal Functions:

getChildTerms (for Id)	Get all child Terms for a given Term Id
getParentTerms (for Id)	Get all parent Terms for a given Term Id
getRootTerms (for Id)	Get all root Terms for a given Term Id

Hierarchy Design Functions:

addRootTerm (Id)	Add a given Term as a “root” in the hierarchy
addChildTerm (id for id)	Add an Term as a child of another Term
removeChildTerm (Id for Id)	Remove a child Term from a Term

removeChildTerms (Id)	Remove all child Terms from a Term

CourseOffering

A learning unit offered during a Term. Note that many of the attributes for CourseOffering may simply reflect the attributes of the associated Course, or may be different.

Attribute	***Type***	***Definition***
id	Id	The unique id
displayName	DisplayText	The name for display
description	DisplayText	The description of this offering
genusType	Type	The type of this offering
startDate	DateTime	Start date of this offering
endDate	DateTime	End date of this offering
endReason	State	The reason that this offering ended, if applicable
title	STRING	The formal title of this course. It may be the same as the display name or it may be used to more formally label the course
number	STRING	The course number which is a label generally used to index the course in a catalog, such as T101 or 16.004.
instructors	Resource[]	The instructors. If each activity has its own instructor, the headlining instructors may be returned
sponsors	Resource[]	The sponsors for this course
credits	Grade[]	The credits in which this course can be registered.
gradingOptions	GradeSystem[]	The various grading options available to register in this course, if available
requiresRegistration	BOOLEAN	Whether this course offering requires advanced registration
minimumSeats	CARDINAL	The minimum number of students this offering can have
maximumSeats	CARDINAL	The maximum number of students this offering can have, if applicable
url	STRING	An external resource, such as a class web site, associated with this offering.
scheduleInfo	STRING	The an informational string for the course offering schedule
event	Event	The calendaring event with the schedule details, if available
course	Course	The canonical course associated with this course offering
term	Term	The Term of this offering

Common Functions:

Read, Write, Notification and Cataloging Functions for CourseOfferings

ActivityUnit

An ActivityUnit is a component of a canonical Course that describes in more detail a learning activity. An ActivityUnit relates to a set of learning Objectives. Learning Objectives managed at the ActivityUnit level can be rolled up to the canonical Course. Note that while ActivityUnit is part of the Course model but not strictly needed as part of this Learning Record investigation. It can be added later if determined to be required.

Activity

An Activity is an offering of an ActivityUnit. The Activity allows for some information, such as seating constraints and instructors, to be managed at this level instead of the CourseOffering level. The Activity also allows for the effort numbers to be refined from its canonical counterpart. Note that while Activity is part of the Course model but not strictly needed as part of this Learning Record investigation. It can be added later if determined to be required.

CourseCatalog

Catalog for all "course" related entities

Attribute	***Type***	***Definition***
id	Id	The unique id
displayName	DisplayText	The name for display
description	DisplayText	The description of this offering
genusType	Type	The type of this offering
provider	Resource	The resource representing the provider of this catalog, if available
branding	Asset[]	The branding, such as an image or logo, if available
license	DisplayText	The terms of usage. An empty license means the terms are unknown

Common Functions:

Read, Write, Notification Functions for CourseCatalogs

Hierarchy Functions:

getChildCourseCatalogs (for Id)	Get all child catalogs for a given catalog Id
getParentCourseCatalogs (for Id)	Get all parent catalogs for a given catalog Id
getRootCourseCatalogs (for Id)	Get all root catalogs for a given catalog Id

Hierarchy Design Functions:

addRootCourseCatalog (Id)	Add a given catalog as a “root” in the hierarchy
addChildCourseCatalog (id for id)	Add a given catalog as a child of another catalog
removeChildCourseCatalog (Id for Id)	Remove a child catalog from another catalog
removeChildCourseCatalogs (Id)	Remove all child catalogs from a catalog

The Assessment Model

<table>
<tr><td colspan="3">Chronicle (Learner Record) Model</td><td colspan="2">Learning Model</td></tr>
<tr><td>Program Model</td><td>Course Model</td><td>Assessment</td><td>Recognition Model</td><td>Grading</td></tr>
<tr><td colspan="3">Requisite Model</td><td colspan="2">Course Registration Model</td></tr>
</table>

The model for Assessment is very broad, and provides the means to create, access, and take assessments. An Assessment may represent a quiz, survey, or other evaluation that includes assessment Items. Operational functions describe the flow of control and the relationships among the objects. Assessment Items are extensible objects to capture various types of questions, such as a multiple choice, short answer, asset submission, etc.[9]

The OSID Assessment models can be broken down into several distinct functional areas:

- Assessment and Item authoring
- Offering Assessments
- Taking Assessments
- Examining Assessment responses and results
- Accessing and managing banks of assessments and items

However, for the purpose of this Learner Record investigation, only two entities of the overall Assessment model are strictly required; Assessment and Catalog (Assessment Bank). A more comprehensive learner record may also include actual evidence of learning based on Assessments. In that case more of the assessment model would need to be included to allow the actual Assessment Items and learner responses to be included as part of a learner record.

Assessment

An Assessment represents a sequence of assessment items that can be offered by an assessing authority and taken by a learner. An Assessment may have an accompanying rubric used for measuring performance. Note that Assessment Items are not included in the attributes of the Assessment entity. There are various operational functions for exposing the Items themselves, depending on the kinds of operations being done, like authoring Assessments, taking Assessments, or reviewing the results of an Assessment. This way the canonical Assessment entity becomes useful in describing assessments for the purposes of other entities in other models This is where, for instance, one would uniquely reference the “SAT Math Level 1” test in an AssessmentEntry.

Attribute	***Type***	***Definition***
id	Id	The unique id
displayName	DisplayText	The name for display

[9] Ref. OSID Assessment model: http://osid.org/specifications/osid/assessment/package.html

description	DisplayText	The description of this assessment
genusType	Type	The type of this assessment
level	Grade	The grade corresponding to the assessment difficulty
rubric	Assessment	The rubric, if available

Common Functions:

Read, Write, Notification and Cataloging Functions for Assessments

AssessmentOffered

An AssessmentOffered represents an offering of a sequence of assessment items. It encapsulates information required for describing how a conical Assessment is to be taken.

Attribute	***Type***	***Definition***
id	Id	The unique id
displayName	DisplayText	The name for display
description	DisplayText	The description of this offered assessment
genusType	Type	The type of this offered assessment
assessment	Assessment	The Assessment related to this offered assessment
level	Grade	The grade corresponding to the assessment difficulty
rubric	AssessmentOffered	The offered rubric, if available
itemsSequential	BOOLEAN	Whether items or parts in this assessment are taken sequentially
itemsShuffled	BOOLEAN	Whether items or parts in this assessment appear in a random order
startTime	DateTime	The start time for this assessment, if applicable
deadline	DateTime	The end time for this assessment, if applicable
duration	Duration	The duration for this assessment, if applicable
scoreSystem	GradeSystem	The grade system for the score, if assessment is scored
gradeSystem	GradeSystem	The grade system for the grade, if assessment is graded
rubric	AssessmentOffered	The offered rubric, if available

Common Functions:

Read, Write, Notification and Cataloging Functions for Assessments

Bank (Catalog)

Catalog for all "assessment" related entities. ACT and the College Board represent catalogs of Assessments.

Attribute	***Type***	***Definition***
id	Id	The unique id

displayName	DisplayText	The name for display
description	DisplayText	The description of this catalog
genusType	Type	The type of this catalog
provider	Resource	The resource representing the provider of this catalog, if available
branding	Asset[]	The branding, such as an image or logo, if available
license	DisplayText	The terms of usage. An empty license means the terms are unknown

Common Functions:

Read, Write, Notification Functions for Assessment Banks

Hierarchy Functions:

getChildBanks (for Id)	Get all child banks for a given bank Id
getParentBanks (for Id)	Get all parent banks for a given bank Id
getRootBanks (for Id)	Get all root banks for a given bank Id

Hierarchy Design Functions:

addRootBank (Id)	Add a given bank as a “root” in the hierarchy
addChildBank (id for id)	Add a given bank as a child of another bank
removeChildBank (Id for Id)	Remove a child bank from another bank
removeChildBanks (Id)	Remove all child banks from a bank

The Recognition Model

Chronicle (Learner Record) Model			**Learning Model**	
Program Model	**Course Model**	**Assessment**	**Recognition Model**	**Grading**
Requisite Model				**Course Registration Model**

The Recognition model defines functionality to confer Awards on Resources, typically people.[10]

Award

Represents something conferred on to a recipient. This might also be related to a badge.

Attribute	***Type***	***Definition***
id	Id	The unique id
displayName	DisplayText	The name for display
description	DisplayText	The description of this award
genusType	Type	The type of this award

Common Functions:

Read, Write, Notification and Cataloging Functions for Awards

Conferral

A Conferral represents an Award conferred to a Recipient for something (the reference). Often, an Award is conferred for a body of work. The body of work may be represented by another entity, such as an Asset, an Assessment, learning outcome Proficiency, and may be related using an Id.

Attribute	***Type***	***Definition***
id	Id	The unique id
displayName	DisplayText	The name for display
description	DisplayText	The description of this conferral
genusType	Type	The type of this conferral
startDate	DateTime	Start date of this conferral
endDate	DateTime	End date of this conferral
endReason	State	The reason that this conferral ended, if applicable
award	Award	The Award conferred
recipient	Resource	The recipient (e.g. Samuel Goldwyn)

[10] Ref. OSID Recognition model: http://osid.org/specifications/osid/recognition/package.html

reference	Resource	The reference, if available. This could be a learning Outcome for which proficiency has been determined
convocation	Convocation	The convocation (e.g. 19th Academy Awards for the time period 1946)

Common Functions:

Read, Write, Notification and Cataloging Functions for Conferrals

Convocation

A Convocation represents an issuing of Conferrals for a managed set of Awards. Convocations also serve to relate the Conferrals to a TimePeriod. Note that while Convocation is part of the Recognition model but not strictly needed as part of this Learning Record investigation. It can be added later if determined to be required.

Academy (Catalog)

Catalog for all "recognition" related entities

Attribute	***Type***	***Definition***
id	Id	The unique id
displayName	DisplayText	The name for display
description	DisplayText	The description of this catalog
genusType	Type	The type of this catalog
provider	Resource	The resource representing the provider of this catalog, if available
branding	Asset[]	The branding, such as an image or logo, if available
license	DisplayText	The terms of usage. An empty license means the terms are unknown

Common Functions:

Read, Write, Notification Functions for Academies

Hierarchy Functions:

getChildAcademies (for Id)	Get all child academies for a given academy Id
getParentAcademies (for Id)	Get all parent academies for a given academy Id
getRootAcademies (for Id)	Get all root academies for a given academy Id

Hierarchy Design Functions:

addRootAcademy (Id)	Add a given bank as a “root” in the hierarchy
addChildAcademy (id for id)	Add a given bank as a child of another academy
removeChildAcademy (Id for Id)	Remove a child bank from another academy
removeChildAcademies (Id)	Remove all child academies from a academy

The Grading Model

Chronicle (Learner Record) Model			Learning Model	
Program Model	Course Model	Assessment	Recognition Model	**Grading**
Requisite Model				Course Registration Model

The Grading model defines functionality and entities for applying grades or ratings. Grades (or marks) are usually an important part of a learner record, and two entities related to grading are included in this investigation. The Grade and GradeSystem definitions not only define grades but also begin to provide information about how to compare grades.

In addition to these two entities, other entities defined in the core OSID Grading model[11] include, Gradebook, GradeEntry, GradebookColumn, GradebookColumnSummary, but are not directly needed for this Learner Record examination. They may become useful for investigating a more comprehensive record model in the future.

Grade

Qualified performance levels defined within a grading system

Attribute	***Type***	***Definition***
id	Id	The unique id
displayName	DisplayText	The name for display
description	DisplayText	The description of this grade
genusType	Type	The type of this grade
inputScoreStartRange	DECIMAL	The low end of the input score range equivalent to this grade
inputScoreEndRange	DECIMAL	The high end of the input score range equivalent to this grade
outputScore	DECIMAL	The output score for this grade used for calculating cumulative or performing articulation
gradeSystem	GradeSystem	The GradeSystem in which this grade belongs

GradeSystem

A grading system. The system can be based on assigned Grades or based on a numeric scale

Attribute	***Type***	***Definition***
id	Id	The unique id
displayName	DisplayText	The name for display

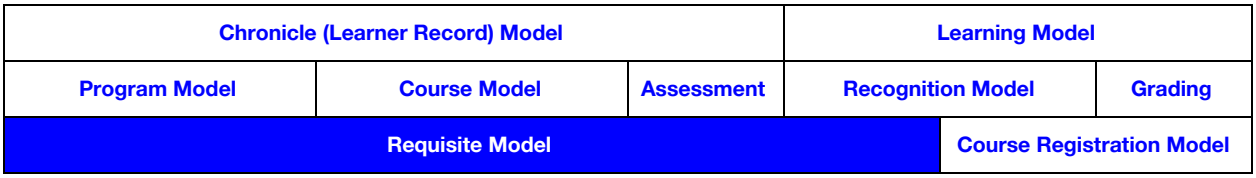
[11] Ref. OSID Grading model: http://osid.org/specifications/osid/grading/package.html

description	DisplayText	The description of this grade system
genusType	Type	The type of this grade system
basedOnGrades	BOOLEAN	Tests if the grading system is based on grades
grades	Grade[]	The grades in this system ranked from highest to lowest
lowestNumericScore	DECIMAL	The lowest number in a numeric grading system
numericScoreIncrement	DECIMAL	The incremental step in a numeric grading system
highestNumericScore	DECIMAL	The highest number in a numeric grading system

Common Functions:

Read, Write, Notification and Cataloging Functions for GradeSystems

The Requisite Model



The Requisite model defines the rules to be managed and evaluated to determine whether educational requirements have been satisfied. It encapsulates the information necessary to fully describe the various ways in which an educational goal can be achieved. When paired with a learner record it can be used to evaluate how a learner achieved the goal.[12]

This model sits three steps away from the Learner Record model (ProgramEntry -> Program -> Requisite) but is included here because some of the efforts to standardize learner records have attempted to include this kind of information. An interested party trying to understand a learner's record may also want to know something about the completion requirements for a Program for which the learner has earned a Credential. However, whether this information should be included in a more comprehensive record or inspected out of band is debatable.

A Requisite has 7 terms for evaluation that must all be true for this Requisite to be satisfied. Additional terms may be defined.

- CourseRequirements
- ProgramRequirements
- CredentialRequirements
- AssessmentRequirements
- LearningObjectiveRequirements
- An external Rule
- Requisite options for nested OR terms

Requisite

The Requisite is a rule used for a requirement component. A Requisite has 7 terms for evaluation that must all be true for this Requisite to be satisfied. Additional terms may be defined in the RequisiteRecord.

A Requisite is true if the active Requisite options term AND the CourseRequirements term AND the ProgramRequirements term AND CredentialRequirements AND AssessmentRequirements AND the LearningObjectiveRequirements AND external Rule are all true.

[12] Ref. OSID Requisite model: http://osid.org/specifications/osid/course/requisite/package.html

A Requisite may also contain options, or sub-requisites. These options are Requisite nodes in a statement tree hierarchy that represent OR terms. Requisites can be defined to have effectiveness and may apply to specific populations of students or by calendar schedules or events.

Attribute	***Type***	***Definition***
id	Id	The unique id
displayName	DisplayText	The name for display
description	DisplayText	The description
genusType	Type	The type
startDate	DateTime	Start date of this requisite
endDate	DateTime	End date of this requisite
requisiteOptions	Requisite[]	The requisite options for the Requisite options term, if available. If any active requisite options are available, meeting the requirements of any one of these requisite options evaluates this requisite options term as true. If no active requisite options apply then this term evaluates to false.
courseRequirements	Course-Requirement[]	The course requirements term, if available. If any CourseRequirements are available, meeting the requirements of any one of the CourseRequirements evaluates this course requirements term as true. If no active course requirements apply then this course requirements term is false.
Program-Requirements	Program-Requirement[]	The program requirements term, if available. If any ProgramRequirements are available, meeting the requirements of any one of the ProgramRequirements evaluates this program requirements term as true. If no program requirements apply then this program requirements term evaluates to false.
Credential-Requirements	Credential-Requirement[]	The credential requirements term, if available. If any CredentialRequirements are available, meeting the requirements of any one of the CredentialRequirements evaluates this credential requirements term as true. If no credential requirements are available, this credential requirements term evaluates to false.
learningObjective-Requirements	LearningObjective-Requirement[]	The learning objective requirements term, if available. If any LearningObjectiveRequirements are available, meeting the requirements of any one of the LearningObjectiveRequirements evaluates this learning objective requirements term as true. If no learning objective requirements apply then this learning objective requirements term evaluates to false.

Assessment-Requirements	Assessment-Requirement[]	The assessment requirements term, if available. If any AssessmentRequirements are available, meeting the requirements of any one of the AssessmentRequirements evaluates this assessment requirements term as true. If no assessment requirements are available, this assessment requirements term evaluates to false.
awardRequirements	AwardRequirement[]	The award requirements term, if available. If any AwardRequirements are available, meeting the requirements of any one of the AwardRequirements evaluates this award requirements term as true. If no award requirements are available, this award requirements term evaluates to false.
rule	Rule	An external rule, if defined
enabled	BOOLEAN	Whether this Requisite is administratively enabled or disabled.
operational	BOOLEAN	Whether this Requisite is operationally enabled or disabled based on demographics, schedules or events

Common Functions:

Read, Write, Notification and Cataloging Functions for Requisites

ProgramRequirement

A ProgramRequirement is a rule for a Program to be satisfied. The Program may be required to be taken in the past for a prerequisite, in the present for a co-requisite, or completed at some point in the future. The ProgramRequirement may also require a minimum GPA or credits earned. Additional constraints may be specified through extension of this model or an external Rule.

All specified rules in the ProgramRequirement must be true for the requirement to be satisfied. Alternative Requisites may be specified inside the ProgramRequirement to append OR terms.

Attribute	***Type***	***Definition***
id	Id	The unique id
displayName	DisplayText	The name for display
description	DisplayText	The description
genusType	Type	The type
altReqiusites	Requisite[]	Any Requisites that may be substituted in place of this ProgramRequirement. All Requisites must be satisfied to be a substitute for this program requirement. Inactive Requisites are not evaluated but if no applicable requisite exists, then the alternate requisite is not satisfied
program	Program	The Program

requiresCompletion	BOOLEAN	Whether this requirement requires completion of the program.
timeFrame	Duration	The timeframe in which the program has to be completed, if applicable. A negative duration indicates the program had to be completed within the specified amount of time in the past. A positive duration indicates the program must be completed within the specified amount of time in the future. A zero duration indicates the program must be completed in the current term.
minimumGPASystem	GradeSystem	The scoring system Id for the minimum GPA, if a minimum GPA applies.
minimumGPA	DECIMAL	The minimum GPA above passing required in the completion of the program or maintained at this level during enrollment, if applicable.
minumumEarnedCredits	DECIMAL	The minimum earned credits required, if applicable
rule	Rule	An external rule, if defined
enabled	BOOLEAN	Whether this requirement is administratively enabled or disabled

Common Functions:

Read, Write, Notification and Cataloging Functions for ProgramRequirements

CourseRequirement

A CourseRequirement is a rule for a Course to be satisfied. The Course may be required to be taken in the past for a prerequisite, in the present for a co-requisite, or completed at some point in the future. The CourseRequirement may also require a minimum grade or credits earned. Additional constraints may be specified in the CourseRequirementRecord or an external Rule .

All specified rules in the CourseRequirement must be true for the requirement to be satisfied. Alternative Requisites may be specified inside the CourseRequirement to append OR terms.

Attribute	***Type***	***Definition***
id	Id	The unique id
displayName	DisplayText	The name for display
description	DisplayText	The description
genusType	Type	The type
altRequisites	Requisite[]	Any Requisites that may be substituted in place of this CourseRequirement. All Requisites must be satisfied to be a substitute for this course requirement. Inactive Requisites are not evaluated but if no applicable requisite exists, then the alternate requisite is not satisfied
course	Course	The Course

requiresCompletion	BOOLEAN	Whether this requirement requires completion of the course with a passing grade, if applicable
timeFrame	Duration	The timeframe in which the course has to be completed, if applicable. A negative duration indicates the course had to be completed within the specified amount of time in the past. A positive duration indicates the course must be completed within the specified amount of time in the future. A zero duration indicates the course must be completed in the current term.
minimumGrade	Grade	The minimum grade above passing required in the completion of the course or maintained at this level during a co-requisite, if applicable.
minimumScoreSystem	GradeSystem	The scoring system for the minimum score, if a minimum score applies.
minimumScore	DECIMAL	The minimum score above passing required in the completion of the course or maintained at this level during a co-requisite, if applicable.
minimumEarnedCredits	DECIMAL	The minimum earned credits, if applicable.
rule	Rule	An external rule, if defined
enabled	BOOLEAN	Whether this requirement is administratively enabled or disabled

Common Functions:

Read, Write, Notification and Cataloging Functions for CourseRequirements

CredentialRequirement

A CredentialRequirement is a rule for a Credential to be satisfied. Additional constraints may be specified through extension of this model or an external Rule.

All specified rules in the CredentialRequirement must be true for the requirement to be satisfied. Alternative Requisites may be specified inside the CredentialRequirement to append OR terms.

Attribute	***Type***	***Definition***
id	Id	The unique id
displayName	DisplayText	The name for display
description	DisplayText	The description
genusType	Type	The type
altRequisites	Requisite[]	Any Requisites that may be substituted in place of this CredentialRequirement. All Requisites must be satisfied to be a substitute for this credential requirement. Inactive Requisites are not evaluated but if no applicable requisite exists, then the alternate

		requisite is not satisfied.
credential	Credential	The Credential
timeFrame	Duration	The timeframe in which the credential has to be earned, if applicable. A negative duration indicates the credential had to be earned within the specified amount of time in the past. A positive duration indicates the credential must be earned within the specified amount of time in the future. A zero duration indicates the credential must be earned in the current term.
enabled	BOOLEAN	Whether this requirement is administratively enabled or disabled

Common Functions:

Read, Write, Notification and Cataloging Functions for CredentialRequirements

LearningObjectiveRequirement

A LearningObjectiveRequirement is a rule for a learning Objective to be satisfied. The LearningObjectiveRequirement may also require a minimum proficiency expressed as a Grade. Additional constraints may be specified through extension of this model or an external Rule.

All specified rules in the LearningObjectiveRequirement must be true for the requirement to be satisfied. Alternative Requisites may be specified inside the LearningObjectiveRequirement to append OR terms.

Attribute	***Type***	***Definition***
id	Id	The unique id
displayName	DisplayText	The name for display
description	DisplayText	The description
genusType	Type	The type
altRequisites	Requisite[]	Any Requisites that may be substituted in place of this LearningObjectiveRequirement. All Requisites must be satisfied to be a substitute for this learning objective requirement. Inactive Requisites are not evaluated but if no applicable requisite exists, then the alternate requisite is not satisfied.
learningObjective	Objective	The Objective
minimumProficiency	Grade	the minimum proficiency required for this Objective expressed as a Grade, if applicable.
rule	Rule	An external rule, if defined
enabled	BOOLEAN	Whether this requirement is administratively enabled or disabled

Common Functions:

Read, Write, Notification and Cataloging Functions for LearningObjectiveRequirements

AssessmentRequirement

An AssessmentRequirement is a rule for an Assessment to be satisfied. The Assessment may be required to be taken in the past for a prerequisite, in the present for a co-requisite, or completed at some point in the future. The AssessmentRequirement may also require a minimum Grade or score. Additional constraints may be specified through extension of this model or an external Rule.

All specified rules in the AssessmentRequirement must be true for the requirement to be satisfied. Alternative Requisites may be specified inside the AssessmentRequirement to append OR terms.

Attribute	***Type***	***Definition***
id	Id	The unique id
displayName	DisplayText	The name for display
description	DisplayText	The description
genusType	Type	The type
altRequisites	Requisite[]	Any Requisites that may be substituted in place of this AssessmentRequirement. All Requisites must be satisfied to be a substitute for this assessment requirement. Inactive Requisites are not evaluated but if no applicable requisite exists, then the alternate requisite is not satisfied.
assessment	Assessment	The Assessment
		The timeframe in which the assessment has to be completed, if applicable. A negative duration indicates the assessment had to be completed within the specified amount of time in the past. A positive duration indicates the assessment must be completed within the specified amount of time in the future. A zero duration indicates the assessment must be completed in the current term.
minimumGrade	Grade	The minimum grade above passing required in the completion of the assessment, if applicable.
minimumScoreSystem	GradeSystem	The score system for the minimum score, if a minimum score applies.
minimumScore	DECIMAL	The minimum score above passing required for this Assessment, if applicable.
rule	Rule	An external rule, if defined
enabled	BOOLEAN	Whether if this requirement is administratively enabled or disabled

Common Functions:

Read, Write, Notification and Cataloging Functions for AssetRequirements

AwardRequirement

An AwardRequirement is a rule for an Award to be satisfied. Additional constraints may be specified through extension of this model or an external Rule.

All specified rules in the AwardRequirement must be true for the requirement to be satisfied. Alternative Requisites may be specified inside the AwardRequirement to append OR terms.

Attribute	***Type***	***Definition***
id	Id	The unique id
displayName	DisplayText	The name for display
description	DisplayText	The description
genusType	Type	The type
altRequisite	Requisite[]	Any Requisites that may be substituted in place of this AwardRequirement. All Requisites must be satisfied to be a substitute for this award requirement. Inactive Requisites are not evaluated but if no applicable requisite exists, then the alternate requisite is not satisfied.
award	Award	The Award
timeFrame	Duration	The timeframe in which the award has to be earned, if applicable. A negative duration indicates the award had to be completed within the specified amount of time in the past. A positive duration indicates the award must be completed within the specified amount of time in the future. A zero duration indicates the award must be completed in the current term.
rule	Rule	An external rule, if defined
enabled	BOOLEAN	Whether this requirement is administratively enabled or disabled

Common Functions:

Read, Write, Notification and Cataloging Functions for AwardRequirements

The Course Registration Model

Chronicle (Learner Record) Model			Learning Model	
Program Model	Course Model	Assessment	Recognition Model	Grading
Requisite Model				Course Registration Model

The Course Registration model defines course registration functionality. This model may feel pretty far afield of the entities required to support a Learner Record. It is included here because Registration is directly referenced in the CourseEntry itself, and of potential interest if there is interest in what grading options a student registered under .[13]

ActivityBundle

A set of Activities in which a Registration is permissible (like a lecture plus an offered lab)

Attribute	***Type***	***Definition***
id	Id	The unique id
displayName	DisplayText	The name for display
description	DisplayText	The description of this bundle
genusType	Type	The type of this bundle
credits	DECIMAL[]	The number of credits available to register for in this course, if applicable. Each array element is a distinct option
gradingOptions	GradeSystem[]	The various grading options available to register in this course
activities	Activity[]	The activities in this bundle (note that this is not the Activity in the learning objective model. It refers to the Activity in the Course model)
courseOffering	CourseOffering	The course offering associated with this activity bundle

Common Functions:

Read, Write, Notification and Cataloging Functions for ActivityBundles

Registration

Defines a relationship between a student and an ActivityBundle

Attribute	***Type***	***Definition***
id	Id	The unique id
displayName	DisplayText	The name for display
description	DisplayText	The description of this registration

[13] Ref. OSID Registration model: http://osid.org/specifications/osid/course/registration/package.html

genusType	Type	The type of this registration
startDate	DateTime	Start date of this registration
endDate	DateTime	End date of this registration
endReason	State	The reason that this registration ended, if applicable
credits	DECIMAL[]	The number of credits the student is registered to earn, if available. Multiple credit options indicates a set of credits to be determined at the completion of the course
gradingOption	GradeSystem	The grading option for this registration
activityBundle	ActivityBundle	the activity bundle associated with this registration
student	Resource	the student associated with this registration

Common Functions:

Read, Write, Notification and Cataloging Functions for Registrations

Primitives Objects

This section defines a number of the primitive objects referenced as attribute types in various other entities above.

Id

An identifier object. Ids are designated by the following elements: namespace, identifier and authority. Two Ids are equal if their namespace, identifier and authority strings are all equal.[14]

Attribute	***Type***	***Definition***
authority	STRING	The authority of this Id. The authority is a string used to ensure the uniqueness of this Id when using a non-federated identifier space. Generally, it is a service name identifying the provider of this Id.
namespace	STRING	The namespace of the identifier. The namespace reflects the domain in which the identifier is unique. When using a global identifier scheme, the namespace may indicate the name of the scheme. When using a local identification scheme, the namespace may be more specific, such as the name of a database or file in which the identifiers exist
identifier	STRING	The identifier of this Id. May use a global or local identification scheme

DisplayText

Text to be displayed to a user.[15]

Attribute	***Type***	***Definition***
languageType	Type	The language type
scriptType	Type	The script type
formatType	Type	The the format type
text	STRING	The actual text string to be displayed

Type

The Type is a form of identifier that is primarily used to identify entity specifications. The Type differs from Id in that it offers display information and Types may be arranged in hierarchies to indicate an extended interface.[16]

Attribute	***Type***	***Definition***
authority	STRING	Similar to Id authority

[14] Ref. OSID Id specification: http://osid.org/specifications/osid/id/Id.html
[15] Ref. OSID DisplayText specification: http://osid.org/specifications/osid/locale/DisplayText.html
[16] Ref. OSID Type specification: http://osid.org/specifications/osid/type/Type.html

namespace	STRING	Similar to Id namespace
identifier	STRING	Similar to Id identifier
displayName	DisplayText	The full display name of this Type
displayLabel	DisplayText	The shorter display label for this Type . Where a display name of a Type might be " Critical Logging Priority Type", the display label could be "critical".
description	DisplayText	A description of this Type
domain	DisplayText	The domain can provide an information label about ths application space of this Type

DateTime

DateTime defines a date and/or time. The OSID's provide a rich interface for describing DateTime which includes features beyond what is found in most programming language libraries. This interface provides a very broad range of dates, describes more or less precision, and/or conveys an uncertainty. Due to the large number of attributes defined, the DateTime specification will not be listed here. For details reference the OSID DateTime specification directly.[17]

Duration

The Duration is a length of time. While a DateTime represents a point on a calendar and a tick on a clock, a Duration represents a measurement. The OSID's provide a rich interface for describing Duration which includes features beyond what is found in most programming language libraries. Due to the large number of attributes defined, the Duration specification will not be listed here. For details reference the OSID Duration specification directly.[18]

Places We Did Not Go

In the opening sections of this document it was stated that this kind of examination has to stop somewhere. That in a holistic model, which defines relationships across many different submodels, there will be entities that are referenced but not fully expired. In addition to some of the entities only briefly touched on in the sections above (most notably in the Assessment Model, the following entities exist in other sub-models that we will not examine further as part of the learner record holistic model, for now

Resource

Resource is a simple entity that captures the name, description and avatar that may represent people, places or a set or arbitrary entities that are used throughout the models as references to indirect entities. For most of the entities defined in this examination Resources reference individual people or "students", but using this abstraction allows for a lot of flexibility. For instance, the learning Proficiency model can be

[17] Ref. OSID DateTime specification: http://osid.org/specifications/osid/calendaring/DateTime.html
[18] Ref. OSID Duration specification: http://osid.org/specifications/osid/calendaring/Duration.html

used to measure to what extent a digital asset or a course covers a particular learning Objective. The Resource entity is part of the OSID Resource model.[19]

Person and Organization

Referred to primarily through the abstraction of the Resource object, people (like students and instructors) and organizations (like the sponsors of courses or programs) have models of their own. These can be more fully fleshed out in a more comprehensive examination, but for now it is sufficient to assume that the models around people and organizations have a level of complexity that is not required to surface for this initial Learner Record examination. These entities are part of the OSID Personnel model.[20]

State

A State is a simple entity that captures the name and description of a state, and is part of the OSID Process model.[21]

Asset

An Asset represents a unit of content, whether it be an image, a video, an application document or some text, and is part of the broader OSID Repository model.[22]

Event

An Event is a range of time associated with a location and event sponsors and is part of the OSID Calendaring model.[23] Events may be managed singularly, or be generated through other types of events, such as recurring events, superseding events or offset events. The Calendaring model is quite complex and covers a wide range of entities and functions related to calendaring and scheduling, etc. It would make for another nice starting point for holistic examination.

Rule

A Rule represents an entity that can be executed in a rules engine. The Rules OSID model[24] defines the means to evaluate Rules and retrieve results.

[19] Ref. OSID Resource model: http://osid.org/specifications/osid/resource/package.html
[20] Ref. OSID Personnel model: http://osid.org/specifications/osid/personnel/package.html
[21] Ref. OSID Process model: http://osid.org/specifications/osid/process/package.html
[22] Ref. OSID Repository model: http://osid.org/specifications/osid/repository/package.html
[23] Ref. OSID Calendaring model: http://osid.org/specifications/osid/calendaring/package.html
[24] Ref. OSID Rules model: http://osid.org/specifications/osid/rules/package.html