Open Service Interface Definition

Grading

Document Release: 2.0
OSID 2.0

Summary

The Grading Open Service Interface Definition (OSID) supports characterizing, storing and retrieving Grades. A Grade is specified with four elements: a GradeValue, GradeType, GradeScale, and ScoringDefinition. These four elements provide a general and flexible way to characterize a Grade. The Service also provides for managing GradeRecords, which join information about the Grade, the Agent[1] (Student) whose Grade it is, and the object that was Graded. This Service also includes methods for iterating through the GradeTypes, GradeScales, and ScoringDefinitions supported by a particular implementation. One can also iterate through the GradableObjects included among the GradeRecords; and through the GradeRecords themselves.

A note about the relationship between the Grading OSID and the Assessment OSID: The Assessment OSID concerns the definition, management, and evaluation of Assessments. The evaluation component of the Assessment OSID is distinct and most likely more complex than a Grade, however it may be the case that an Assessment element is the gradable object referred to in Grade.

[1] Refer to Agent OSID for more information on Agents.

Service Definition

An examination of an Open Service Interface Definition (OSID) begins with the Manager. All Managers[2] provide the way to create the objects that implement the principal interfaces in the Service.

org.osid.grading.GradingManager

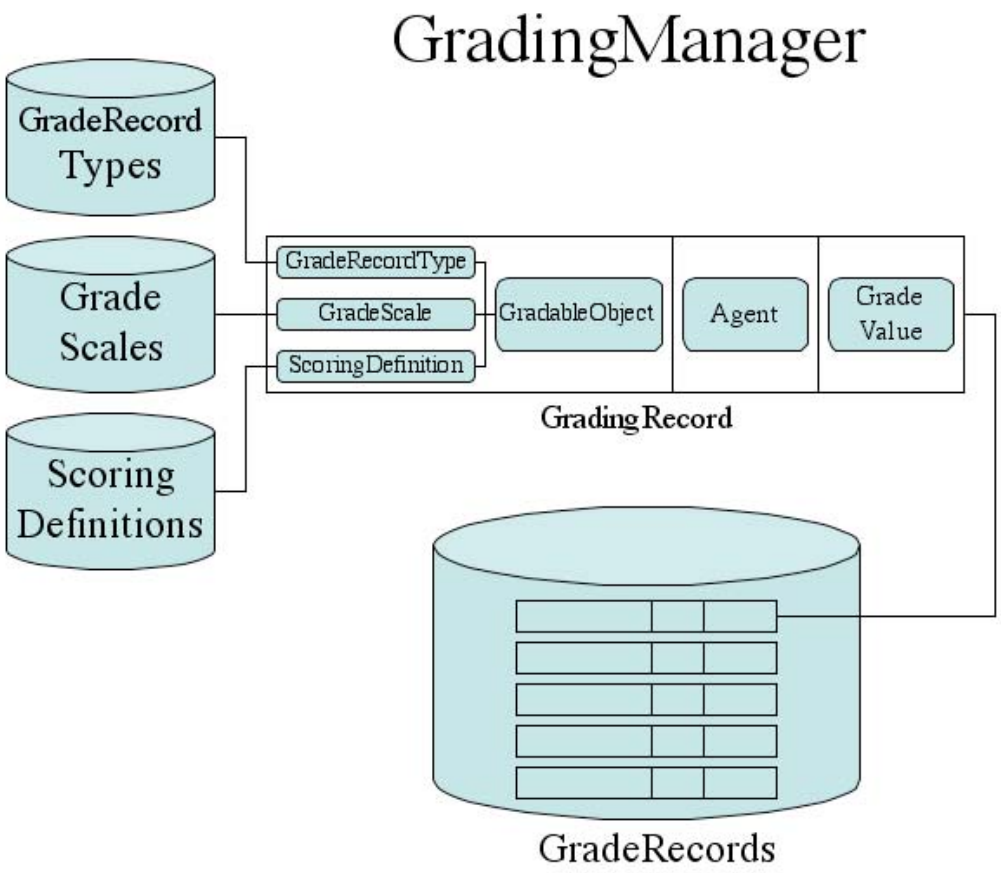

Figure 1: Grading Manager

The GradingManager is primarily responsible for the creation and persistence of GradeRecords. The database of GradeRecords can be thought of as a grade book that captures students' assignment grades. The **createGradeRecord**() method accepts an GradableObject, Agent, GradeValue and GradeRecordType[3] and returns a new GradeRecord object that implements the GradeRecord interface.[4] Note that the GradeRecordType describes the kind of GradeRecord, e.g. mid-term or final, and is distinct from the GradeType, e.g. letter or numeric. The created object is given a unique identifier[5] by the Manager and stored. Any GradeRecord created in this fashion can also be removed using the Manager's **deleteGradeRecord**() method. The **getGradeRecords**() method returns the set of GradeRecords being managed.[6] The **getGradeRecords**() method accepts arguments, specifically a CourseSectionId, GradableObjectId,

[2] org.osid.OsidManager defines the interface extended by the Managers in each OSID. Here we will be discussing org.osid.grading.GradingManager. Refer to OsidManager for more information.
[3] Refer to Shared OSID for more information on Types in OSID.
[4] Refer to the discussion of OsidManager for information on create methods and their use in implementation substitution.
[5] Refer to the discussion of Unique Identifiers in the Shared OSID and the Id OSID.
[6] Objects are returned through an iterator, in this particular case a GradeRecordIterator. Iterators return objects in sequence. Refer to the general discussion of Iterators for more information.

Agent, and GradeType as a means of filtering which GradeRecords are returned. If any argument is not null, all GradeRecords returned must match it.

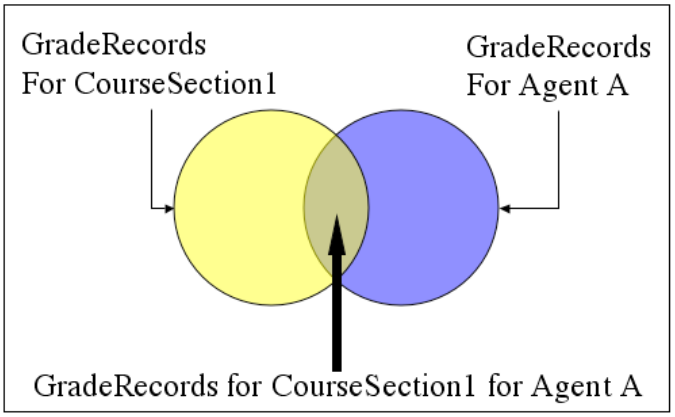

Figure 2: Returning a Subset of All GradeRecords

The Manager can create GradableObjects. An GradableObject has a name and description and holds information about a specific CourseSection, a specific gradable object such as an Assessment element, a GradeType, GradeScale, and ScoringDefinition. The **createGradableObject**() method accepts these arguments and returns a new GradableObject based on them and including a unique Id generated by the implementation. Similar to the case with GradeRecords, GradableObjects can be deleted with **deleteGradableObject**() and enumerated with **getGradableObjects**()[7] If the CourseSection or GradableObject passed to getGradableObjects is not null, only GradableObjects matching the value are returned. Unlike GradeRecords, the Manager only returns all GradableObjects and does not offer filtering.

The Manager also returns the set of GradeRecordTypes, GradeTypes, GradeScales, and ScoringDefinitions supported through **getGradeRecordTypes**(), **getGradeTypes**(), **getGradeScales**(), and **getScoringDefinitions**(), respectively.

GradingManager Method Summary

GradableObject	createGradableObject(String displayName, String description, Id courseSectionId, Id externalReferenceId, Type gradeType, Type scoringDefinition, Type gradeScale, int gradeWeight)
GradeRecord	createGradeRecord(Id gradableObjectId, Id agentId, Serializable gradeValue, Type gradeRecordType)
void	deleteGradableObject(Id gradableObjectId)
void	deleteGradeRecord(Id gradeRecordId, Id agentId, Type gradeRecordType)
GradableObject	getGradableObject(Id gradableObjectId)
GradableObjectIterator	getGradableObjects(Id courseSectionId, Id externalReferenceId)
GradeRecordIterator	getGradeRecords(Id courseSectionId, Id externalReferenceId, Id gradableObjectId, Id agentId, Type gradeRecordType)
TypeIterator	getGradeRecordTypes()
TypeIterator	getGradeScales()
TypeIterator	getGradeTypes()
TypeIterator	getScoringDefinitions()

[7] Note that there are no methods for returning all Agents or CourseSections. Agents are managed by the Agent OSID and CourseSections by the CourseManagement SID. One could write an application to examine all the GradeRecords and assemble a list of the unique Agents and CourseSections found therein.

org.osid.grading.GradableObject

A Grade exists in the context of a specific GradableObject that is graded. Each GradableObject is given a display name and description, both of which can be changed, as well as an Id that cannot. The methods for working with display names, descriptions, and Ids are the same across many interfaces. The **getDisplayName**() method returns the name while the **updateDisplayName**() method sets and persists it. The **getDescription** () method returns the description while the **updateDescription**() method sets and persists it. Both name and description are also passed into the Manager's **createGradableObject**() method and set by that implementation. The implementation also set's the Id that is the unique way to identify the GradableObject. The **getId**() method returns this Id.

The GradableObject can include a reference (an Id) to something defined external to the Grading OSID. This can be an Assessment element or anything else that is meaningful. The **getGradableObject**() method returns this information. The GradableObject can also include a reference (an Id) to a CourseSection. CourseSections are defined in the CourseManagement OSID. The **getCourseSection**() method returns this information.

The GradableObject includes all the information about the kind of grade that would be appropriate, namely the GradeType, which is returned by **getGradeType**(), the GradeScale, which is returned by **getGradeScale**(), the GradeWeight which is returned by **getGradeWeight**(), and the ScoringDefinition, which is returned by **getScoringDefinition**().

GradableObject Method Summary

Id	getCourseSection()
String	getDescription()
String	getDisplayName()
Id	getExternalReference()
Type	getGradeScale()
Type	getGradeType()
int	getGradeWeight()
Id	getId()
Id	getModifiedBy()
long	getModifiedDate()
PropertiesIterator	getProperties()
Properties	getPropertiesByType(Type propertiesType)
TypeIterator	getPropertyType()
Type	getScoringDefinition()
void	updateDescription(String description)
void	updateDisplayName(String displayName)

org.osid.grading.GradeRecord

A GradingRecord represents the intersection of:

- an Agent (the Student whose GradableObject is graded)
- a GradableObject (what is being graded)
- a GradeValue (the Student's Grade for the GradableObject)

There is a GradingRecord for each permutation of these three elements.[8] The **getAgent**() method returns the Agent; **getGradableObject**() method returns the GradableObject's identifier; and **getGradeValue**() returns the Serializable value; **updateGradeValue**() modifies it. Given a GradableObject, one can get the GradeType. **getGradeRecordType**() returns the GraderecordType.

[8] The implementation's strategy for managing and storing this data would not necessarily include a data structure with these dimensions.

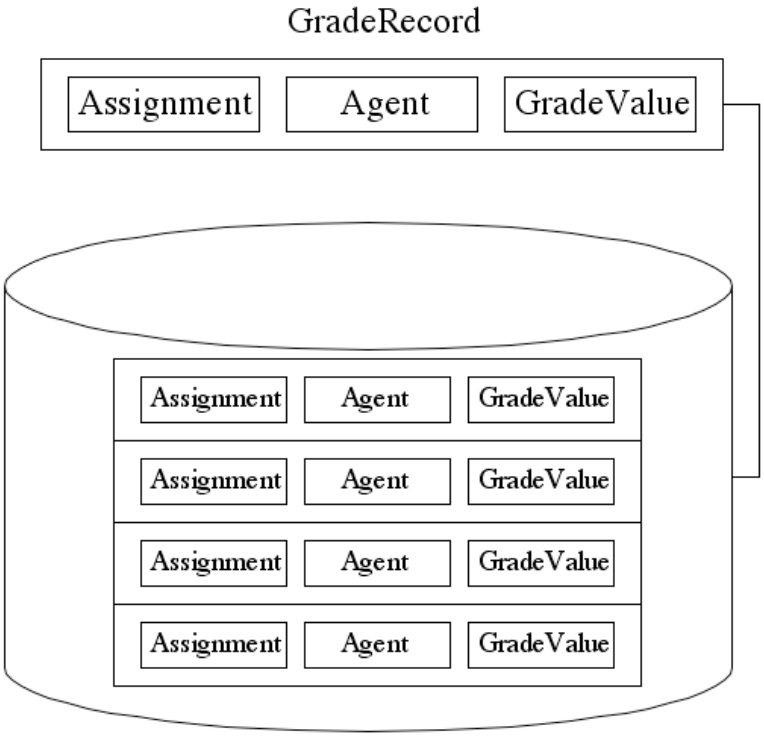

Figure 3: GradeRecord Structure

It is important to know who has entered or modified Grades and when. The **getModifiedBy**() method returns the Agent that most recently modified the GradeRecord and the **getModifiedDate**() method returns when the modification was performed.[9]

GradeRecord Method Summary

Id	getAgentId()
Id	getGradableObject ()
Type	getGradeRecordType()
Serializable	getGradeValue()
Type	getGradeType()
Id	getModifiedBy()
long	getModifiedDate()
PropertiesIterator	getProperties()
Properties	getPropertiesByType(Type propertiesType)
TypeIterator	getPropertyTypes()
void	updateGradeValue(Serializable gradeValue)

GradeTypes, GradeScales, and ScoringDefinitions

We need to know the kind of GradeValue we have in a GradeRecord. For example, is it numeric, letter, or something else such as “pass” or “good”? The GradeType captures this. We need to know the range or scale against which GradeValues are interpreted. For example, are GradeValues on a scale of 0 to 100,

[9] The Grading OSID does not specify maintaining anything but the most recent Agent and modification date. An application could want that kind of historical information. A simple solution is for the implementation to provide a log using the Logging OSID, although there is no guarantee that a log would be available from all implementations. Similarly, some implementations may choose to support some transaction processing such as commit, mark, and rollback which is not covered in the OSID.

0 to 10, or “A” to “F”? The GradeScale captures this. We also need to know how to interpret the meaning of a particular GradeValue. For example, was does an “A” mean for this GradableObject? The ScoringDefinition captures this.

While the GradeValue is specific to an Agent and GradableObject, the GradeType, GradeScale, and ScoringDefintion will likely be the same for many GradableObjects. The GradingManager returns the set of GradeTypes, GradeScales, and ScoringDefinitions supported.

GradeTypes, like all OSID Types, include four String properties: authority, domain, keyword, and description. A letter grade, such as “A” or “B” might be assigned the GradeType:

authority	“mit.edu”
domain	“undergraduate”
keyword:	“letter grade”
description	“a single letter optionally followed by + or –“

We know that a Grade of “A” is better than a Grade of “B” because we understand both Grades lie on a scale and that A is farther in the direction of being correct. A variety of scales are in use. GradeScale is an OSID Type and so has four String properties. An example of a GradeScale is:

authority	“mit.edu”
domain	“undergraduate”
keyword:	“A..F”
description	“The scale includes: A+, A, A-, B+, B, B-, C+, C, C-, D, F“

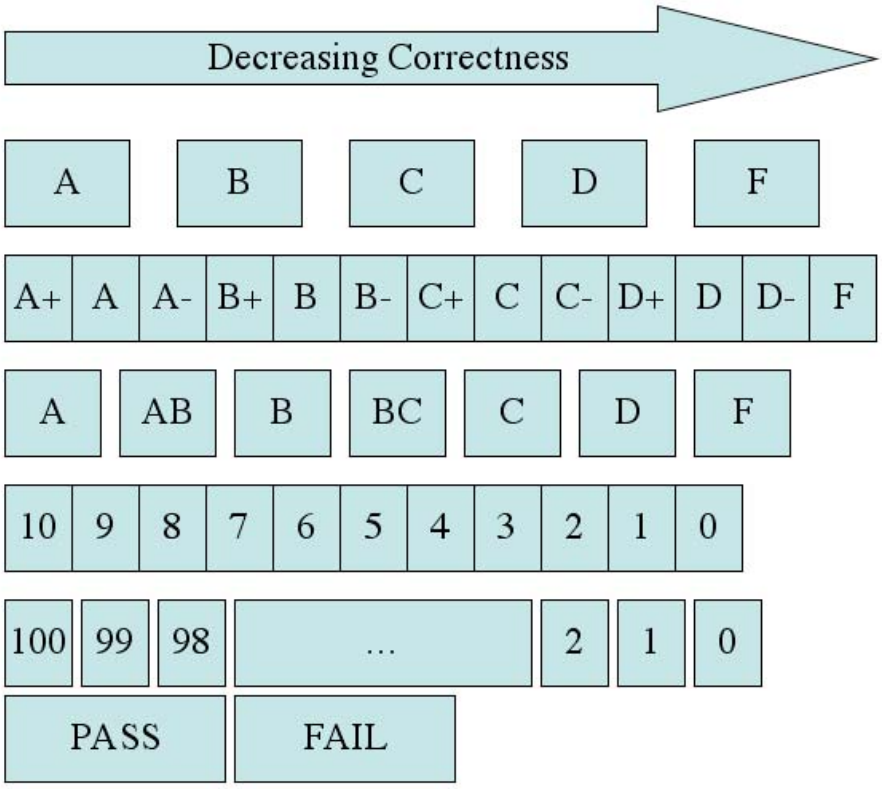

Figure 4: GradeScales

GradeValues not only have an associated Type and GradeScale, they have an associated ScoringDefintion. This definition is helpful in interpreting the GradeValues and perhaps in comparing GradeValues across organizations. An example of an underlying ScoringDefintion would be:

A = 100…90
B = 89…80
C = 79...70
D = 69…60
F = < 60

and might be translated into the Type:

domain	“undergraduate”
authority	“mit.edu”
keyword:	“ten point interval scoring”
description	“The scale assigns A=90..100, B=80...89, …“

org.osid.grading.*Object* Iterators

Iterators return a set of elements of a specific type, one at a time. The purpose of all Iterators is to offer a way for OSID methods to return multiple values of a common type and not use an array. Returning an array may not be appropriate if the number of values returned is large or is fetched remotely. Iterators do not allow access to values by index, rather you must access values in sequence. Similarly, there is no way to go backwards through the sequence unless you place the values in a data structure, such as an array, that allows for access by index.

All iterators contain two methods. The **hasNext<*Object type*>**() method returns true if there are more values of the iterator type available; false otherwise. The **next<*Object type*>** () method returns the next element in the sequence. Note that in many cases, the order of elements is not guaranteed. The following iterators are included in this OSID:

- GradableObjectIterator
- GradeRecordIterator

Type iterators, such as that used for are defined in org.osid.shared.TypeIterator.

org.osid.grading.GradingException

The service definitions make use of Exceptions as a mechanism for responding to error or unusual conditions. Exceptional circumstances are signaled by the implementation throwing an exception, which is GradingException in the case of the Grading OSID. The Exception contains a message that is a String. Note that other kinds of Exception constructors are not used as all do or can devolve to a String. All methods of all interfaces of all OSIDs throw a subclass of org.osid.OsidException. This requires the caller of any implementation method handle the Exception.

If a method performs an operation without incident, an object or primitive may be returned, but in most cases, methods do not return error codes or a success or failure boolean. For example, a method that deletes an object with a particular identifier, would throw an Exception if the identifier were unknown; the method would not return, for example, false.