# IMS Course Management Service Information Model

## Version 1.0

## Final Release
## Version 1.0

Date Issued: 30 June 2011

Latest version: http://www.imsglobal.org/lis/

# Table of Contents

# List of Figures

# List of Tables

# 1 Introduction

## 1.1 Course Management Service Overview

The Course Management Service (CMS) specification is the definition of how systems manage the exchange of information that describes courses. The Course Management Service specification is constructed following the recommendations documented in the IMS GLC Abstract Framework (IAF) [IAF, 03a], [IAF, 03b], [IAF, 03c]. This means that this specification is based upon the concepts of:

- Interoperability – Course Management Service focuses on the exchange of information about Courses between systems. There are no definitions in the specification on how the data is managed within the systems;
- Service-oriented – Course Management Service defines the exchange of information in terms of the services being supplied by the collaboration of the systems;
- Component-based – for example, the Course Management Service is combined with the Group Management Service, Membership Management Service, Person Management Service, Outcomes Management Service and Bulk Data Exchange Management Service to provide the Learning Information Services [LIS, 11a];
- Layering – the Course Management Service is a part of the Application Services layer but it interacts with the services available in the Common Services layer e.g., authentication;
- Behaviors and Data Models – the Course Management Service is defined in terms of its behaviors and data models. The behaviors cause changes in the state of the data model and the state of the data model will only be altered as a result of a clearly defined behavior;
- Multiple Bindings – the Course Management Service information model is defined using the Unified Modeling Language (UML). This enables reliable mapping of the information model into a range of different bindings. The binding of immediate importance is to the Web Services Description Language (WSDL);
- Adoption – whenever appropriate, the Course Management Service specification makes use of other IMS GLC and non-IMS GLC standards and specifications.

The logical structure of a Course is defined as composed of Templates, Offerings, Sections and Associations. A Template may have one or more Offering and each Offering may have one of more Sections. Templates, Offerings and Sections define the logical structure of a Course. Associations are collections of Sections that have some educational collaborative relevance. Participation in the activity defined in a Template, Offering and Section is described using a mapping to the equivalent Membership. The structure of a course can be extended using Groups i.e., either as sub-structures below the Section or as organizational structures above a Template. A Group is as defined in the Group Management Service specification [GMS, 11]. The standard create, read, update, replace, delete and simple discovery operations are available for managing Templates, Offerings and Sections. Other operations are defined to support specific business requirements.

## 1.2 Scope and Context

This document is the IMS Course Management Service Information Model v1.0 and as such it is used as the basis for the development of the following documents:

a) IMS Course Management Service WSDL Binding v1.0 [CMS, 11] – the description of the WSDL binding of the Information Model.

The core uses-cases for the Course Management Service are described as a subset of the Learning Information Services Specification [LIS, 11b].

This information model defines the Course Management Service Abstract Application Programming Interface (a-API). The Learning Information Services specification, of which the Course Management Service is a component, is a series of behavioral models that define how the data models are to be manipulated. These behavioral models are described using the Unified Modeling Language (UML) [SDN07, 07].

## 1.3 Structure of this Document

The structure of this document is:

| | |
|---|---|
| 2. Course Management Service Description | The description of the overall structure and operation of the Course Management Service. This includes the description of the architectural model and the domain object model; |
| 3. Behavioral Model | The definition of the operations of Course Management Service application service. This focuses on the description of the behaviors supported by the service; |
| 4. Interface Data Model | The definition of the data models exchanged between the Course Management Service End Systems. These are the parameters exchanged across the interoperability interface; |
| 5. End System Data Model | The definition of the data models in the Course Management Service End Systems. This addresses the persistence of the data with respect to interoperability; |
| 6. Extending and Profiling the Service | Identification of the ways in which the Course Management Service can be extended both in terms of the addition of new constituent services and proprietary extensions to a service; |
| Appendix A Service Status Codes | A summary list of the status codes, and their causes, that can be returned by each of the operations forming the Course Management Service; |
| Appendix B Vocabularies | A summary of the set of vocabularies that are used within the specification; |
| Appendix C File-based Data Exchange | The out-of-band file exchange used in response to receiving a URL for an external data file that contains the request data. |

## 1.4 Compatibility Commitment

The Course Management Service was not part of the IMS GLC Enterprise Services v1.0 specification [ES, 06]. Instead, this functionality was supported using the IMS GLC Group Management Service v1.0 [GMS, 06] specification in a variety of different ways. This created interoperability problems hence the creation of this service specification. The Course Management Service v1.0 is closely linked to the Group Management Service (GMS) v2.0 [GMS, 11] and Membership Management Service (MMS) v2.0 [MMS, 11]. The MMS is used to define the participants in a Course defined by the CMS and Courses are extended using the GMS. Therefore the GMS and MMS must be implemented to obtain the full functionality of the CMS.

In general, there is NO backwards compatibility between the usage of the CMSv1.0 and the ways in which GMS v1.0 has been implemented to support course management. Vendors may define compatibility bridges for their own implementations but these are outside the scope of this specification.

## 1.5 Nomenclature

| | |
|---|---|
| a-API | Abstract Application Programming Interface |
| API | Application Programming Interface |
| CMS | Course Management Service |
| GMS | Group Management Service |
| IAF | IMS GLC Abstract Framework |
| LIS | Learning Information Services |
| MMS | Membership Management Service |
| PIM | Platform Independent Model |

| | |
|---|---|
| PSM | Platform Specific Model |
| RFC | Request For Comment |
| SDN | Specification Development Note |
| UML | Unified Modeling Language |
| URL | Uniform Resource Locator |
| WSDL | Web Services Description Language |

## 1.6 References

[APG, 05a] *IMS GLC Application Profile Guidelines Overview: Part 1 – Management Overview v1.0*, IMS Global Learning Consortium, K.Riley, October 2005. http://www.imsglobal.org/ap/index.html.

[APG, 05b] *IMS GLC Application Profile Guidelines White Paper: Part 2 Technical Manual*, S.Wilson and K.Riley, Version 1.0, IMS Global Learning Consortium, October 2005. http://www.imsglobal.org/ap/index.html.

[BDEMS, 11] *IMS GLC Bulk Data Exchange Management Service v1.0 Information Model v1.0 Final Release*, L.Feng, W.Lee and C.Smythe, IMS Global Learning Consortium, June 2011.

[CMS, 11] *IMS GLC Course Management Service v1.0 WSDL Binding v1.0 Final Release*, L.Feng, W.Lee and C.Smythe, IMS Global Learning Consortium, June 2011.

[ES, 06] *IMS GLC Enterprise Services Overview v1.0 Final Release*, C.Vento and C.Smythe, IMS Global Learning Consortium, June 2006.

[GMS, 06] *IMS/GLC Group Management Services Information Model v1.0 Final Release*, C.Vento and C.Smythe, IMS Global Learning Consortium, June 2006.

[GMS, 11] *IMS GLC Group Management Service v2.0 Information Model v1.0 Final Release*, L.Feng, W.Lee and C.Smythe, IMS Global Learning Consortium, June 2011.

[GWS, 05] *IMS GLC General Web Services WSDL Binding Guidelines v1.0 Final Specification*, C.Schroeder, J.Simon and C.Smythe, IMS Global Learning Consortium, December 2005.

[IAF, 03a] *IMS Abstract Framework: Applications, Services & Components v1.0*, Ed. C.Smythe, IMS Global Learning Consortium, July 2003.

[IAF, 03b] *IMS GLC Abstract Framework: Glossary v1.0*, Ed. C.Smythe, IMS Global Learning Consortium, July 2003.

[IAF, 03c] *IMS GLC Abstract Framework: White Paper v1.0*, Ed. C.Smythe, IMS Global Learning Consortium, July 2003.

[LIS, 11a] *IMS GLC Learning Information Services v2.0 Overview v1.0 Final Release*, L.Feng, W.Lee and C.Smythe, IMS Global Learning Consortium, June 2011.

[LIS, 11b] *IMS GLC Learning Information Services v2.0 Specification v1.0 Final Release*, L.Feng, W.Lee and C.Smythe, IMS Global Learning Consortium, June 2011.

[LIS, 11c] *IMS GLC Learning Information Services v2.0 Best Practices & Implementation Guide v1.0 Final Release*, L.Feng, W.Lee and C.Smythe, IMS Global Learning Consortium, June 2011.

[MMS, 11] *IMS GLC Membership Management Service v2.0 Information Model v1.0 Final Release*, L.Feng, W.Lee and C.Smythe, IMS Global Learning Consortium, June 2011.

[SDN07, 06] *IMS GLC Specification Note: UML Profile for Platform Independent Model Descriptions of Specifications for Data Models v1.0*, C.Smythe, IMS Global Learning Consortium, October 2006.

[SDN11, 06] *IMS GLC Specification Note 11: Vocabulary Definition, Registration & Maintenance Procedures*, C.Smythe, IMS Global Learning Consortium, October 2006.

[VDEX, 04] *IMS Vocabulary Definition Exchange Best Practice and Implementation Guide, Version 1.0 Final Specification*, A. Cooper, IMS Global Learning Consortium, 2005. Online version: http://www.imsglobal.org/vdex/vdexv1p0/imsvdex_bestv1p0.html

# 2 Course Management Service Description

## 2.1 An Abstract Representation

It is important to remember that this document contains a description of the underlying information model in terms of the abstract API. The manner in which this abstract representation is visualized is not intended to dictate the implementation form of a Course Management System. The breakdown of the service into its interface classes is a convenient way to document the set of behaviors. The objective for producing these interfaces is to identify and define the messages that are exchanged between the end-systems to realize the system behaviors required of the service.

The internal organization of an implementation of the full abstract API is beyond the scope of this specification. The only constraint is that the external behavior of the abstract API complies with this specification. This means that a .NET, J2EE, etc. physical implementation of this abstract API does not have to represent the functionality using the same breakdown of operations/methods. This physical implementation is not subject to the conformance specification.

It is important to note that the UML representation of the interfaces is used to help develop and document the Course Management Service Information Model. It is not a requirement for an implementation to implement this interface as defined i.e., to use the same parameters, etc. Conformance against this specification will be confirmed by inspecting the appropriate binding of the information model and ensuring that the relevant information is present and that different sequences of activity result in the predicted and mandated behavior. It is essential that the behaviors described by each of the operations are fully supported and that the behaviors described by different sequences be also maintained.

## 2.2 Course Management Service Architecture & Specification Model

The basic architectural model for the Course Management Service specification is shown in Figure 2.1. In this architecture the scope of the Course Management Service specification is shown as the dotted line. The scope of the interoperability is the data and behavioral models of the objects being exchanged.

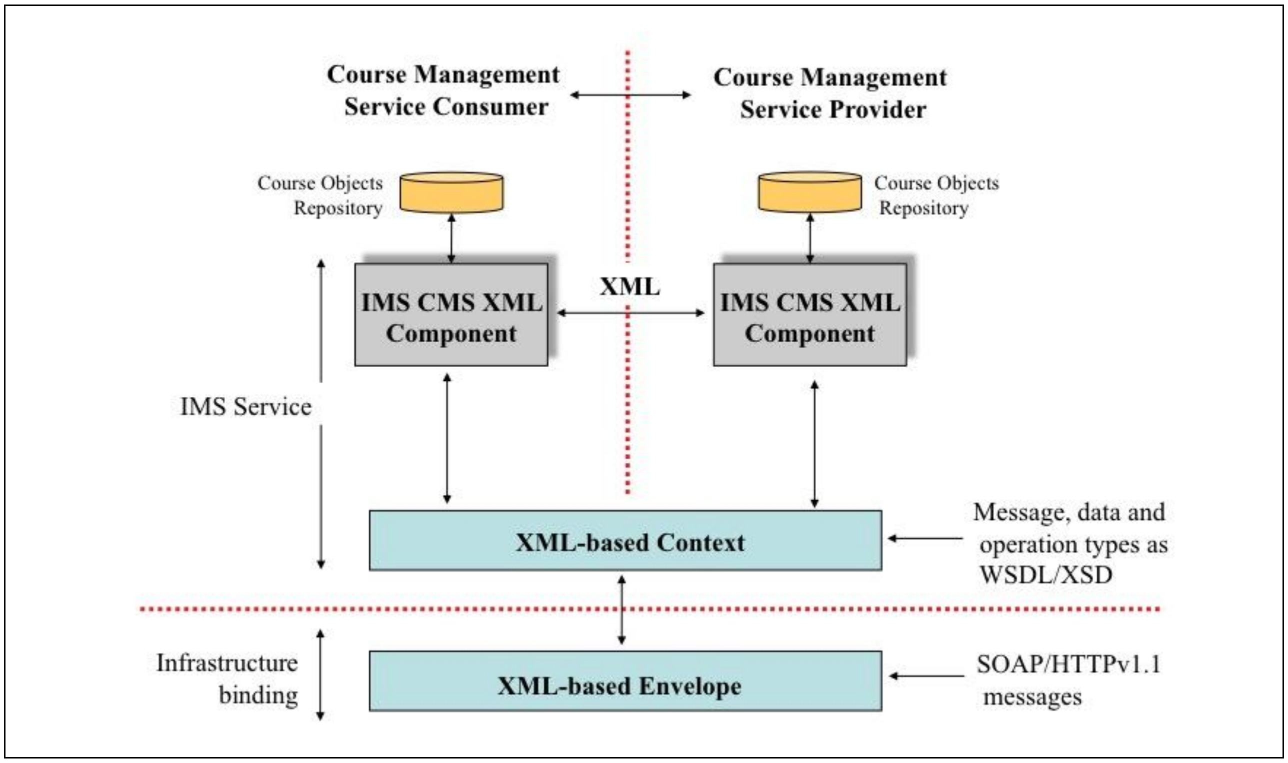

Course Management Service Consumer
Course Management Service Provider
Course Objects Repository
Course Objects Repository
IMS CMS XML Component
XML
IMS CMS XML Component
IMS Service
XML-based Context
Message, data and operation types as WSDL/XSD
Infrastructure binding
XML-based Envelope
SOAP/HTTPv1.1 messages

**Figure 2.1 Course Management service architecture model.**

It is important to remember that the structure of the exchanged information has NO bearing on how the same information is contained within the 'source' and 'target' Learning Information Services systems (the Course object repositories in the two end-systems). It is simply a representation of the data used to facilitate exchange between the end-systems. The only constraint on the end-system repositories is that they provide data persistence consistent with the required behavior.

## 2.3 Course Composite Objects

It is important to note that this is an **interoperability** specification and as such it makes no statements about how information is stored within the exchanging end systems. The objects in the end-systems **must** be persistent otherwise sequences of operation on the same object will not be possible. Reference to these objects in the interface is through a SourcedId however this identifier does not have to be the key stored within the end-systems. If different keys are used in the end-systems then it is the responsibility of the end-systems to maintain the mapping between that key and the SourcedId, i.e., the interface must never be exposed to the keys of the end-systems.

There is no such thing as a Course object. Instead, Courses are reflected in four types of object each of which has its own SourcedId. The structure of a Course is shown schematically in Figure 2.2.

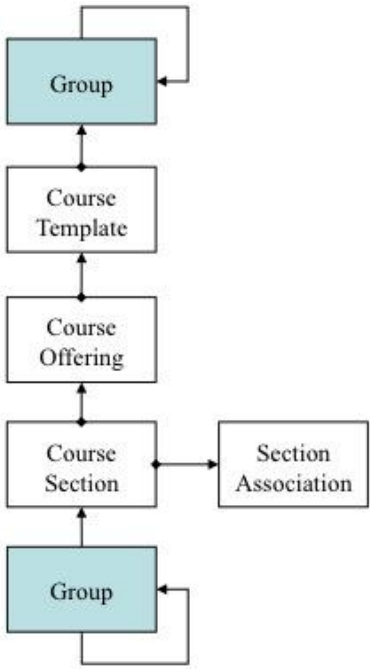


**Figure 2.2 Structure of a Course.**

The components of a Course are:

- Course Template (or just Template) – the identification of the basic and definition of the course e.g., Biology 101. A CourseTemplate will, in general, have one or more CourseOfferings associated with it;
- Course Offering (or just Offering) – the allocation of the course to an academic session e.g., Maths 101 Semester 1. A CourseOffering will, in general, have one or more CourseSections associated with it;
- Course Section (or just Section) – the assignment of teaching resources to the scheduled activities that constitute the course e.g., English 101 Semester 2 Seminars;
- Section Association (or just Association) – the association of two or more Course Sections for some educational purpose.

Groups, as defined in the IMS Group Management Service specification v2.0 [GMS, 11], can be used to extend the basic structure of a Course (See Sub-section 5.2).

## 2.4 Synchronous & Asynchronous Services

Within the context of the Course Management Service the definition of synchronous and asynchronous services is:

- Synchronous – the source service is blocked until the final response from the target service is received. A schematic representation of the information flow for a synchronous service is shown in Figure 2.3;
- Asynchronous – the source service is not blocked and so more than one request can be outstanding at any moment in time. A schematic representation of the information flow for an asynchronous service is shown in Figure 2.4.

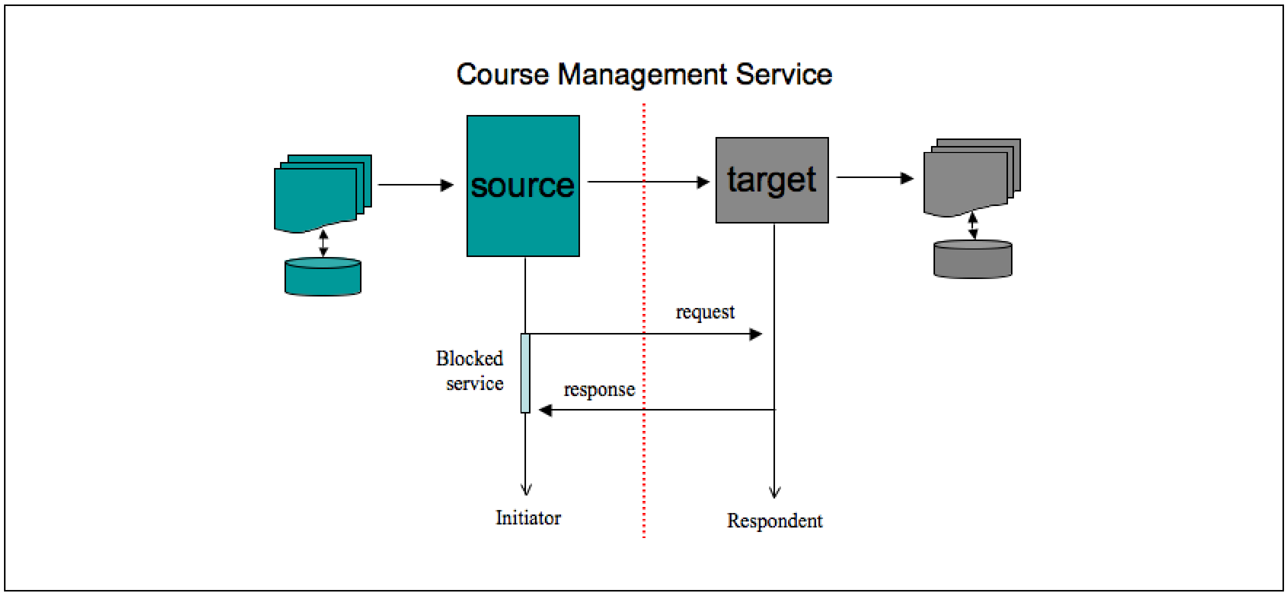


**Figure 2.3 Synchronous service.**

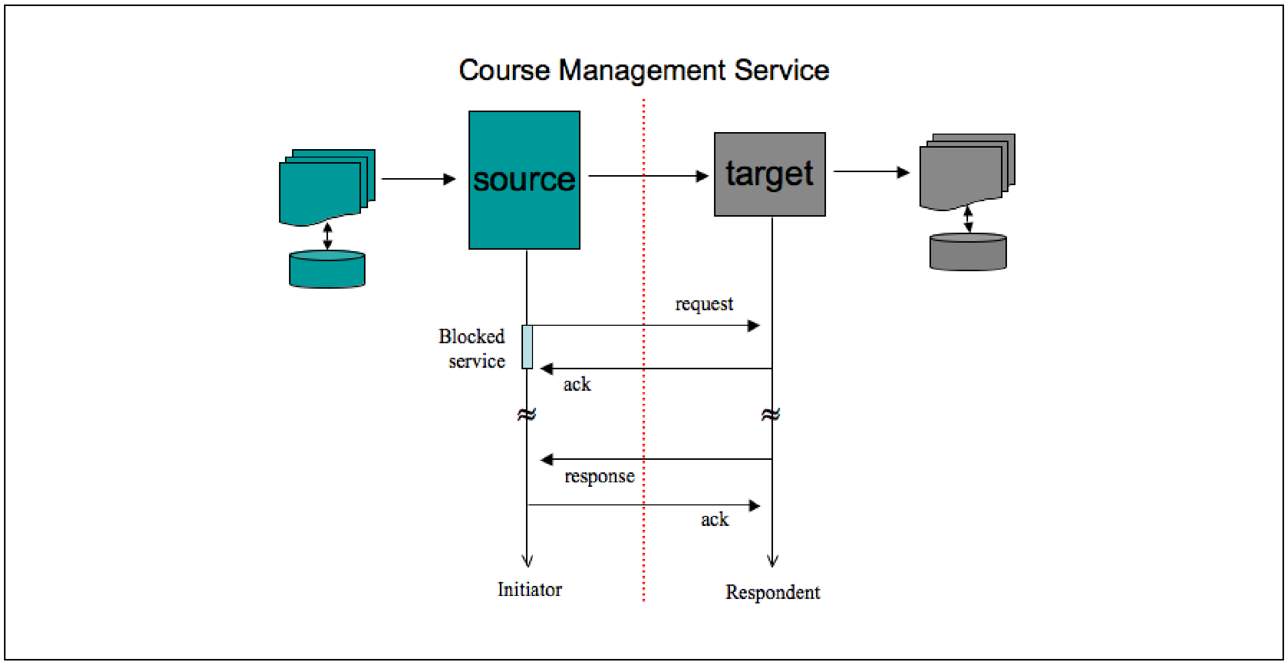


**Figure 2.4 Asynchronous service.**

It is stressed that the abstract-API dos not differentiate between synchronous and asynchronous services[1]. The support for these two approaches is at the binding level only.

The key difference is that for an asynchronous service more than one request can be issued at any one time (it should be noted that an asynchronous service can be supported using synchronous messaging). In both cases the service assumes a perfect messaging system i.e., request, response and acknowledgement messages have a guaranteed delivery grade of service.

## 2.5 Handling the Service Status Codes

Each operation in a service is mapped to an appropriate message exchange pattern. Any response/acknowledgement message will contain status information. This status information provides contextual information about the completed success or otherwise of the operation. There are two types of status information that are available to the end-systems:

- Business transaction – these are the status reports that reflect the business logic of the transactions being exchanged by the end-systems. This status information will be contained within the message header under a specially defined data structure. The status information contained herein is also used to contain any error codes i.e., error reporting is handled as a subset of status information reporting;
- Messaging fault– these are fault codes that are reported by the messaging infrastructure and which are carried in the messages.

It is important to note that messaging errors may indicate that the original request never reached the service provider end-system. In this case the service consumer implementation that handles the status information is responsible for mapping the message infrastructure failure codes to the equivalent business transaction status code. The message infrastructure failure codes have no meaning with respect to an IMS GLC specification. The IMS GLC specifications do not describe how the status information is to be handled within an end-system i.e., this will depend on how the abstract API is physically realized within an implementation. Therefore, it is important that an implementation can:

- Combine the transaction status information and any message fault error codes in a single integrated status reporting mechanism. Any other system failure information that is made available by an implementation should also use the same mechanism;
- Examine the status information reported after the completion of the appropriate phase of an operation and especially once the operation has been completed. This may require an explicit status information call or it may be reported as part of the API call;
- Differentiate the status information reports for each transaction within an operation. Remember that some specifications provide operations that can contain more than one transaction request and that a different status report may be given for each of those transactions.

Exception handling is the system’s response to known or unknown error conditions. Exception handling is outside the scope of an IMS GLC specification. However, an error condition should not cause the end-systems to fail in an uncontrolled manner. The requirement for every operation to return status information is to enable an implementation to terminate in a controlled fashion.

[1] In many implementations of the abstract-API the synchronous and asynchronous services would require different operation calls. This is just one example where an implementation does not match the definition of the abstract-API.

# 3 Behavioral Model

## 3.1 Service Definition

The CourseManagementService is used to model the service responsible for manipulating information about course structures. The CourseManagementService interfaces are shown in Figures 3.1, 3.2, 3.3 and 3.4. The CourseManagementService is defined as a four interfaces: CourseTemplateManager that supports the manipulation of CoureTemplates; CourseOfferingManager that supports the manipulation of CourseOfferings; CourseSectionManager that supports the manipulation of CourseSections; and SectionAssociation Manager that supports the manipulation of SectionAssociations.

## 3.2 CourseTemplateManager Interface Description

The CourseTemplateManager interface class, as shown in Figure 3.1, describes the operations on a CourseTemplate. The interface stereotype indicates that there are no attributes for this class. The set of operations are summarized in Table 3.1.

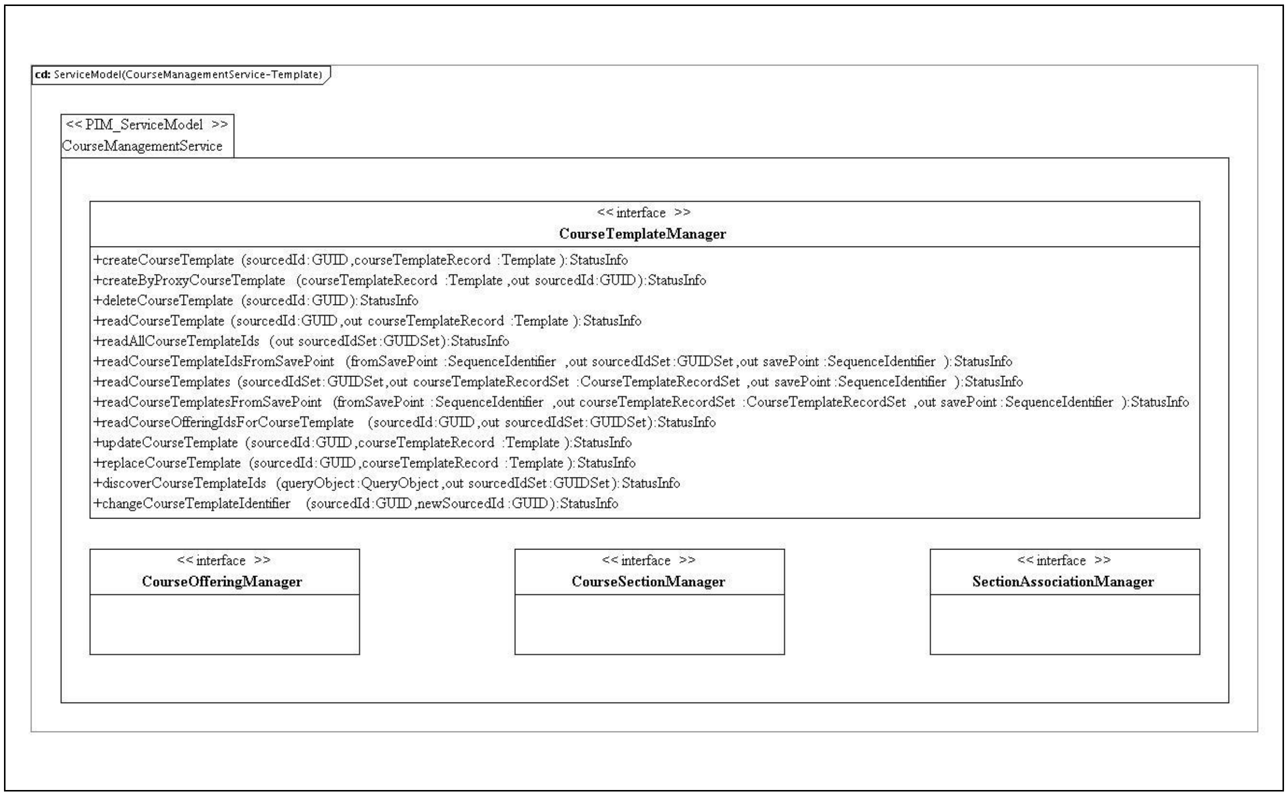

cd: ServiceModel(CourseManagementService-Template)
<< PIM_ServiceModel >>
CourseManagementService
<< interface >>
CourseTemplateManager
+createCourseTemplate (sourcedId:GUID,courseTemplateRecord :Template ):StatusInfo
+createByProxyCourseTemplate (courseTemplateRecord :Template ,out sourcedId:GUID):StatusInfo
+deleteCourseTemplate (sourcedId:GUID):StatusInfo
+readCourseTemplate (sourcedId:GUID,out courseTemplateRecord :Template ):StatusInfo
+readAllCourseTemplateIds (out sourcedIdSet:GUIDSet):StatusInfo
+readCourseTemplateIdsFromSavePoint (fromSavePoint :SequenceIdentifier ,out sourcedIdSet:GUIDSet,out savePoint :SequenceIdentifier ):StatusInfo
+readCourseTemplates (sourcedIdSet:GUIDSet,out courseTemplateRecordSet :CourseTemplateRecordSet ,out savePoint :SequenceIdentifier ):StatusInfo
+readCourseTemplatesFromSavePoint (fromSavePoint :SequenceIdentifier ,out courseTemplateRecordSet :CourseTemplateRecordSet ,out savePoint :SequenceIdentifier ):StatusInfo
+readCourseOfferingIdsForCourseTemplate (sourcedId:GUID,out sourcedIdSet:GUIDSet):StatusInfo
+updateCourseTemplate (sourcedId:GUID,courseTemplateRecord :Template ):StatusInfo
+replaceCourseTemplate (sourcedId:GUID,courseTemplateRecord :Template ):StatusInfo
+discoverCourseTemplateIds (queryObject:QueryObject,out sourcedIdSet:GUIDSet):StatusInfo
+changeCourseTemplateIdentifier (sourcedId:GUID,newSourcedId:GUID):StatusInfo
<< interface >>
CourseOfferingManager
<< interface >>
CourseSectionManager
<< interface >>
SectionAssociationManager

**Figure 3.1 CourseManagementService CourseTemplateManager interface definition.**

**Table 3.1 Summary of operations for CourseTemplateManager.**

| Operation | Description |
|---|---|
| createCourseTemplate | To request the creation of a populated CourseTemplate object on the target system where the source is responsible for the allocation of the unique identifier. |
| createByProxyCourseTemplate | To request the creation of a populated CourseTemplate object on the target system where the target is responsible for the allocation of the unique identifier. |
| deleteCourseTemplate | To request the deletion of a CourseTemplate object. The CourseTemplate object is deleted and all of its associated relationships. |
| readCourseTemplate | To read the full contents of the identified CourseTemplate object. The target must return all of the data it has for the CourseTemplate object. |
| readAllCourseTemplateIds | To obtain the set of SourcedIds which have been assigned to CourseTemplate objects. |
| readCourseOfferingIdsForCourseTemplate | To obtain the set of SourcedIds of the CourseOfferings associated with the CourseTemplate. |
| readCourseTemplateIdsFromSavePoint | To obtain the set of SourcedIds for CourseTemplates objects which have been altered since the requested reference point. The reference point is set as zero at creation and incremented after every write operation. |
| readCourseTemplates | To obtain the CourseTemplate objects for a defined set of SourcedIds. This results in a single transaction that may require the exchange of a large volume of data in the response message. |
| readCourseTemplatesFromSavePoint | To obtain the set of CourseTemplate objects which have been altered since the requested reference point. The reference point is set as ‘zero’ at creation and incremented after every write operation. This results in a single transaction that may require the exchange of a large volume of data in the response message. |
| updateCourseTemplate | To write new content into the identified CourseTemplate object. The target must write the new data into the CourseTemplate object. This is an additive operation. |
| replaceCourseTemplate | To replace the content of the identified CourseTemplate object. The target must write the new data into the CourseTemplate object. This is a destructive write-over of all of the original information. In the case of the object not existing, this operation acts as an implied ‘createCourseTemplate’. |
| discoverCourseTemplateIds | To obtain the set of SourcedIds for CourseTemplate objects whose properties agree with those defined in the query/filter. |
| changeCourseTemplateIdentifier | To change the SourcedId of the CourseTemplate record. The completion of this operation will result in later actions using the original SourcedId reporting an unknown identifier status. |

Note: In most cases the above operations act on a single instance of a CourseTemplate object i.e., ‘createCourseTemplate’, ‘createByProxyCourseTemplate’, ‘deleteCourseTemplate’, ‘readCourseTemplate’, ‘replaceCourseTemplate’ and ‘updateCourseTemplate’.

### 3.2.1 CreateCourseTemplate() Operation

| | |
|---|---|
| **Name:** | createCourseTemplate |
| **Return Function Parameter:** | *StatusInfo* – the status of the creation request. The permitted status codes are defined in Tables 3.2 and A.2. |
| **Supplied (in) Parameters:** | *sourcedId:GUID* – the sourcedId allocated by the source system. This is the identifier that must also be assigned within the target system.<br>*courseTemplateRecord:CourseTemplateRecord* – the CourseTemplate data that is to be stored in the new object. |
| **Returned (out) Parameters:** | None. |
| **Behavior:** | When the source issues the 'createCourseTemplate' request the target is instructed to create the populated CourseTemplate object and to allocate that structure the 'sourcedId' passed by the source. If the supplied 'sourcedId' has already been allocated to another object then the request is rejected and the appropriate failure code is returned. The reference point identifier is set to zero for the CourseTemplate object in both the source and target. |
| **Notes:** | This request contains the initial content for the CourseTemplate object. More content can be added/replaced using the 'updateCourseTemplate' and/or 'replaceCourseTemplate' requests. |

**Table 3.2 Status codes for the 'createCourseTemplate' operation.**

| Status Code | Explanation of the Cause of the Code |
|---|---|
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor=fullsuccess' | The creation request has been fully and successfully implemented by the target system and the CourseTemplate object has been created with a unique identifier. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=idallocinusefail' | The target could not allocate the required unique SourcedId to the CourseTemplate object as it is already in use. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=overflowfail' | The target could not create the CourseTemplate object due to lack of target allocation memory. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=invaliddata' | Part or all of the supplied data was detected as invalid by the target system. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=incompletedata' | Some mandatory part of the data has been detected as missing by the target system. |
| 'CodeMajor=Success'<br>'Severity=Warning'<br>'CodeMinor=partialdatastorage' | The target has stored a subset of the sent data record i.e., some of the optional data has not been stored (all mandatory data has been supplied and stored). |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=unknownmdvocabulary' | The target system could not identify the defined metadata vocabulary term. This may be due to an incorrect term or a missing vocabulary file. |
| 'CodeMajor=Failure' | The target system could not identify the defined vocabulary term. This |

| Status Code | Explanation of the Cause of the Code |
|---|---|
| ‘Severity=Status’<br>‘CodeMinor=unknownvocabulary’ | may be due to an incorrect term or a missing vocabulary file. |
| ‘CodeMajor=Failure’<br>‘Severity=Status’<br>‘CodeMinor=unknownextension’ | The target cannot process and store the proprietary data model extensions used in the object. |

### 3.2.2 CreateByProxyCourseTemplate() Operation

| | |
|---|---|
| **Name:** | createByProxyCourseTemplate |
| **Return Function Parameter:** | *StatusInfo* – the status of the creation request. The permitted status codes are defined in Tables 3.3 and A.2. |
| **Supplied (in) Parameters:** | *courseTemplateRecord:CourseTemplateRecord* – the CourseTemplate data that is to be stored in the new object. |
| **Returned (out) Parameters:** | *sourcedId:GUID* – the identifier allocated by the target to the newly created CourseTemplate object. |
| **Behavior:** | When the source issues the 'createByProxyCourseTemplate' request the target is instructed to create the populated CourseTemplate object and to allocate that record a unique 'sourcedId'. The reference point identifier is set to zero for the 'courseTemplate' object in both the source and target. |
| **Notes:** | This request contains the initial content for the CourseTemplate object. More content can be added/replaced using the 'updateCourseTemplate' and/or 'replaceCourseTemplate' requests. |

**Table 3.3 Status codes for the 'createByProxyCourseTemplate' operation.**

| Status Code | Explanation of the Cause of the Code |
|---|---|
| 'CodeMajor=Success' 'Severity=Status' 'CodeMinor=fullsuccess' | The creation request has been fully and successfully implemented by the target system and the CourseTemplate object has been created with the identifier supplied by the source. |
| 'CodeMajor=Failure' 'Severity=Status' 'CodeMinor=idallocfail' | The target could not allocate a unique 'identifier' to the CourseTemplate object because there are no more spare identifiers available. |
| 'CodeMajor=Failure' 'Severity=Status' 'CodeMinor=overflowfail' | The target could not create the CourseTemplate object due to lack of target allocation memory. |
| 'CodeMajor=Failure' 'Severity=Status' 'CodeMinor=invaliddata' | Part or all of the supplied data was detected as invalid by the source system. |
| 'CodeMajor=Failure' 'Severity=Status' 'CodeMinor=incompletedata' | Some mandatory part of the data has been detected as missing by the target system. |
| 'CodeMajor=Success' 'Severity=Warning' 'CodeMinor=partialdatastorage' | The target has stored a subset of the sent data record i.e., some of the optional data has not been stored (all mandatory data has been supplied and stored). |
| 'CodeMajor=Failure' 'Severity=Status' 'CodeMinor=unknownmdvocabulary' | The target system could not identify the defined metadata vocabulary term. This may be due to an incorrect term or a missing vocabulary file. |
| 'CodeMajor=Failure' 'Severity=Status' 'CodeMinor=unknownvocabulary' | The target system could not identify the defined vocabulary term. This may be due to an incorrect term or a missing vocabulary file. |
| 'CodeMajor=Failure' | The target cannot process and store the proprietary data model extensions |

| Status Code | Explanation of the Cause of the Code |
|---|---|
| ‘Severity=Status’ ‘CodeMinor=unknownextension’ | used in the object. |

### 3.2.3 DeleteCourseTemplate() Operation

| | |
|---|---|
| **Name:** | deleteCourseTemplate |
| **Return Function Parameter:** | *StatusInfo* – the status of the delete request. The permitted status codes are defined in Tables 3.4 and A.2. |
| **Supplied (in) Parameters:** | *sourcedId:GUID* – the identifier to be used by the target to identify the CourseTemplate object. |
| **Returned (out) Parameters:** | None. |
| **Behavior:** | When the source issues the 'deleteCourseTemplate' request the target is instructed to delete the identified CourseTemplate object and to remove the reference to the CourseTemplate from any of the related CourseOffering objects. All Membership associations are also deleted. If the object identified by the supplied SourcedId cannot be located then the request is rejected and the appropriate failure code is returned. |
| **Notes:** | Deletion of the CourseTemplate object does not necessarily result in the destruction of the data within the server. The true state of the data in the target is unknown. |

**Table 3.4 Status codes for the 'deleteCourseTemplate' operation.**

| Status Code | Explanation of the Cause of the Code |
|---|---|
| 'CodeMajor=Success' 'Severity=Status' 'CodeMinor=fullsuccess' | The deletion request has been fully and successfully implemented by the target system and the CourseTemplate object has been deleted. The corresponding membership records have also been deleted. |
| 'CodeMajor=Failure' 'Severity=Status' 'CodeMinor=unknownobject' | The CourseTemplate object identifier is unknown in the target system and so the object could not be deleted. |
| 'CodeMajor=Failure' 'Severity=Status' 'CodeMinor= deletefailure' | The target system has not been able to delete the identified CourseTemplate object. |

### 3.2.4 ReadCourseTemplate() Operation

| | |
|---|---|
| **Name:** | readCourseTemplate |
| **Return Function Parameter:** | *StatusInfo* – the status of the read request. The permitted status codes are given in Tables 3.5 and A.2. |
| **Supplied (in) Parameters:** | *sourcedId:GUID* – the identifier of the courseTemplate object to be read. |
| **Returned (out) Parameters:** | *courseTemplateRecord:CourseTemplateRecord* – the CourseTemplate data that is read from the object. |
| **Behavior:** | When the source issues the 'readCourseTemplate' request the target is charged with retrieving the identified record from its database and returning this data to the source. The target is responsible for ensuring that the object contains valid data. If the object identified by the supplied SourcedId cannot be located then the request is rejected and the appropriate failure code is returned. |
| **Notes:** | The returned CourseTemplate object can only be trusted if the corresponding status code is 'success'. |

**Table 3.5 Status codes for the 'readCourseTemplate' operation.**

| Status Code | Explanation of the Cause of the Code |
|---|---|
| 'CodeMajor=Success' 'Severity=Status' 'CodeMinor=fullsuccess' | The read request has been fully and successfully implemented by the target system and the identified courseTemplate object has been read from the target system. |
| 'CodeMajor=Failure' 'Severity=Status' 'CodeMinor=unknownobject' | The CourseTemplate object identifier is unknown in the target system and so the object could not be read. |
| 'CodeMajor=Failure' 'Severity=Status' 'CodeMinor=targetreadfailure' | The target system has detected an error in the stored CourseTemplate object and so cannot return the data. |
| 'CodeMajor=Failure' 'Severity=Status' 'CodeMinor=invaliddata' | Part or all of the returned data was detected as invalid by the source system. |
| 'CodeMajor=Failure' 'Severity=Status' 'CodeMinor=incompletedata' | Some mandatory part of the data has been detected as missing by the source system. |
| 'CodeMajor=Success' 'Severity=Warning' 'CodeMinor=partialdatastorage' | The target has only returned a subset of the data expected by the source e.g., only the mandatory parts. |
| 'CodeMajor=Failure' 'Severity=Status' 'CodeMinor=unknownvocabulary' | The source system could not identify the defined vocabulary term. This may be due to an incorrect term or a missing vocabulary file. |
| 'CodeMajor=Failure' 'Severity=Status' 'CodeMinor=unknownextension' | The source cannot process and store the proprietary data model extensions used in the object. |

### 3.2.5 ReadAllCourseTemplateIds() Operation

| | |
|---|---|
| **Name:** | readAllCourseTemplateIds |
| **Return Function Parameter:** | *statusInfo:StatusInfo* – the status of the read request. The permitted status codes are given in Tables 3.6 and A.2. |
| **Supplied (in) Parameters:** | None. |
| **Returned (out) Parameters:** | *sourcedIdSet:GUIDSet* – the set of SourcedIds for all of the CourseTemplate objects in the target's course template database. |
| **Behavior:** | When the source issues the 'readAllCourseTemplateIds' request the target is charged with returning the sourcedIds of all of the CourseTemplates in its course template database. |
| **Notes:** | The returned set of sourcedIds may or may not already be identified as allocated in the CourseTemplates database in the source system. If no SourcedIds have been allocated then the returned data set is empty and the success status code returned. |

**Table 3.6 Status codes for the 'readAllCourseTemplateIds' operation.**

| Status Code | Explanation of the Cause of the Code |
|---|---|
| 'CodeMajor=Success' 'Severity=Status' 'CodeMinor=fullsuccess' | The read request has been fully and successfully implemented by the target system and the identified CourseTemplate object has been read from the target system. |
| 'CodeMajor=Success' 'Severity=Status' 'CodeMinor= nosourcedids' | The read request has been fully and successfully implemented by the target system and no CourseTemplate object identifiers were found. |
| 'CodeMajor=Failure' 'Severity=Status' 'CodeMinor=toomuchdata' | The requested data cannot be returned because it would exceed some physical system constraint e.g., permitted size of SOAP message. |
| 'CodeMajor=Failure' 'Severity=Status' 'CodeMinor=invaliddata' | Part or all of the returned data was detected as invalid by the source system. |

### 3.2.6 ReadCourseTemplateIdsFromSavePoint() Operation

| | |
|---|---|
| **Name:** | readCourseTemplateIdsFromSavePoint |
| **Return Function Parameter:** | *statusInfo:StatusInfo* – the status of the read request. The permitted status codes are given in Tables 3.7 and A.2. |
| **Supplied (in) Parameters:** | *fromSavePoint:SequenceIdentifier* – the reference point from which all of the changed identifier actions are to be read. This is the value in the source system. |
| **Returned (out) Parameters:** | *sourcedIdSet:GUIDSet* – the set of sourcedIds of the CourseTemplate objects stored on the target.<br><br>*savePoint:SequenceIdentifier* – the value of the reference point counter in the target system. |
| **Behavior:** | When the source issues the 'readCourseTemplateIdsFromSavePoint' the target returns the set of SourcedIds that have been altered from the defined reference point.<br><br>If the reference counter in the source is greater than that in the target then an error code and the target value for the reference point are returned. |
| **Notes:** | If no SourcedIds have been allocated then the returned data set is empty and the success status code returned. |

**Table 3.7 Status codes for the 'readCourseTemplateIdsFromSavePoint' operation.**

| Status Code | Explanation of the Cause of the Code |
|---|---|
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor=fullsuccess' | The read request has been fully and successfully implemented by the target system and the identified CourseTemplate object has been read from the target system. |
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor= nosourcedids' | The read request has been fully and successfully implemented by the target system and no CourseTemplate object identifiers were found. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=toomuchdata' | The data cannot be returned because it would exceed some physical system constraint e.g., permitted size of SOAP message. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=invaliddata' | Part or all of the returned data was detected as invalid by the source system. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=savepointerror' | An error has occurred in the processing of the save-point identifier information making it impossible to read the correct objects from the database. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=savepointsyncerror' | The value of the save point reference from the source was later than that of the target system. No identifiers have been returned. The target system savepoint value is returned to the source system for information. |

### 3.2.7 ReadCourseOfferingIdsForCourseTemplate() Operation

| | |
|---|---|
| **Name:** | readCourseOfferingIdsForCourseTemplate |
| **Return Function Parameter:** | *statusInfo:StatusInfo* – the status of the update request. The permitted status codes are given in Tables 3.8 and A.2. |
| **Supplied (in) Parameters:** | *sourcedId:GUID* – the sourcedId of the CourseTemplate whose associated CourseOfferings are to be identified. |
| **Returned (out) Parameters:** | *sourcedIdSet:GUIDSet* – the set of sourcedIds for all of the associated CourseOffering objects in the target's database. |
| **Behavior:** | When the source issues the 'readCourseOfferingIdsForCourseTemplate' request the target is charged with using the supplied SourcedId of the reference CourseTemplate to return the SourcedIds of all the associated course offerings. These relationships are identified by searching the set of CourseOfferings for the appropriate parent CourseTemplate.<br><br>An error code is returned if the target does not recognize the SourcedId as a valid CourseTemplate. |
| **Notes:** | The returned set of SourcedIds may or may not already be allocated in the CourseOffering database in the source system. If no associated course offerings are found then the returned data set is empty and the success status code returned. |

**Table 3.8 Status codes for the 'readCourseOfferingIdsForCourseTemplate' operation.**

| Status Code | Explanation of the Cause of the Code |
|---|---|
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor=fullsuccess' | The read request has been fully and successfully implemented by the target system and the set of associated CourseOffering identifiers have been read from the target system. |
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor= nosourcedids' | The read request has been fully and successfully implemented by the target system and no CourseOffering object identifiers were found. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=unknownobject' | The CourseTemplate object identifier is unknown in the target system and so the request cannot be processed. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=targetreadfailure' | The target system has detected an error in the LineItem database and so cannot return the data. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=toomuchdata' | The requested data cannot be returned because it would exceed some physical system constraint e.g., permitted size of SOAP message. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=invaliddata' | Part or all of the returned data was detected as invalid by the source system. |

### 3.2.8 ReadCourseTemplates() Operation

| | |
|---|---|
| **Name:** | readCourseTemplates |
| **Return Function Parameter:** | *statusInfo:StatusInfo* – the status of the read request. The permitted status codes are given in Tables 3.9 and A.2. |
| **Supplied (in) Parameters:** | *sourcedIdSet:GUIDSet* – the set of identifiers of the CourseTemplate objects to be read. |
| **Returned (out) Parameters:** | *courseTemplateRecordSet:CourseTemplateRecordSet* – the set of course template records.<br><br>*savePoint:SequenceIdentifier* – the value of the reference point counter in the target system. |
| **Behavior:** | When the source issues the 'readCourseTemplates' request the target is charged with retrieving the identified set of objects from its database. The associated read savePoint reference is updated and returned.<br><br>If the object identified by the supplied SourcedId cannot be located then the request is rejected and a partial success code is returned for the operation. The target is responsible for ensuring that the records contain valid data. The target should attempt to successfully complete as much of the request as possible. |
| **Notes:** | A returned CourseTemplate record is only present if the object has been located in the target system and the full data set returned.<br><br>The enclosed data may result in a long response message. |

**Table 3.9 Status codes for the 'readCourseTemplates' operation.**

| Status Code | Explanation of the Cause of the Code |
|---|---|
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor=fullsuccess' | The read request has been fully and successfully implemented by the target system and the identified CourseTemplate object has been read from the target system. |
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor=partialreadfail' | Some of the CourseTemplate object identifiers are unknown in the target system and so those objects could not be read. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=invaliddata' | Part or all of the returned data was detected as invalid by the source system. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=unknownvocabulary' | The source system could not identify the defined vocabulary term. This may be due to an incorrect term or a missing vocabulary file. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=toomuchdata' | The data cannot be returned because it would exceed some physical system constraint e.g., permitted size of SOAP message. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=targetreadfailure' | The target system has detected an error in the stored CourseTemplate objects and so cannot return the data. |

### 3.2.9 ReadCourseTemplatesFromSavePoint() Operation

| | |
|---|---|
| **Name:** | readCourseTemplatesFromSavePoint |
| **Return Function Parameter:** | *statusInfo:StatusInfo* – the status of the read request. The permitted status codes are given in Tables 3.10 and A.2. |
| **Supplied (in) Parameters:** | *fromSavePoint:SequenceIdentifier* – the reference point from which all of the changed identifier actions are to be read. This is the value in the source system. |
| **Returned (out) Parameters:** | *coureTemplateRecordSet: coureTemplateRecordSet* – the set of course template records.<br><br>*savePoint:SequenceIdentifier* – the value of the reference point counter in the target system. |
| **Behavior:** | When the source issues the 'readCourseTemplatesFromSavePoint' request the target is charged with reading the objects that have been altered from the defined reference.<br><br>If the reference counter in the source is greater than that in the target then an error code and the target value for the reference point are returned. |
| **Notes:** | If no objects have been allocated then the data file will be empty (the file must be created and its URL returned) and the success status code returned.<br><br>The enclosed data may result in a long response message. |

**Table 3.10 Status codes for the 'readCourseTemplatesFromSavePoint' operation.**

| Status Code | Explanation of the Cause of the Code |
|---|---|
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor=fullsuccess' | The read request has been fully and successfully implemented by the target system and the identified CourseTemplate objects have been read from the target system. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=savepointerror' | An error has occurred in the processing of the save-point identifier information making it impossible to read the correct objects from the database. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=savepointsyncerror' | The value of the save point reference from the source was later than that of the target system. No identifiers have been returned. The target system savepoint value has been updated to that supplied by the source system. |
| 'CodeMajor=Success'<br>'Severity=Warning'<br>'CodeMinor=partialdatareturn' | The target has only returned a subset of the data object e.g., only the mandatory parts. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=toomuchdata' | The data cannot be returned because it would exceed some physical system constraint e.g., permitted size of SOAP message. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=targetreadfailure' | The target system has detected an error in the stored LineItem object and so cannot return the data. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=invaliddata' | Part or all of the returned data was detected as invalid by the target system. |

### 3.2.10 UpdateCourseTemplate() Operation

| | |
|---|---|
| **Name:** | updateCourseTemplate |
| **Return Function Parameter:** | *StatusInfo* – the status of the update request. The permitted status codes are given in Tables 3.11 and A.2. |
| **Supplied (in) Parameters:** | *sourcedId:GUID* – the identifier of the CourseTemplate object to be updated.<br>*courseTemplateRecord:CourseTemplateRecord* – the CourseTemplate data that is to be stored in the object. |
| **Returned (out) Parameters:** | None. |
| **Behavior:** | When the source issues the 'updateCourseTemplate' request the target is charged with writing the supplied information into the identified record. If any part of the write fails e.g., due to partial invalid data then the whole request is rejected and the record is left in its original state. This is an additive write operation of all the data fields supplied in the update request and fields not supplied remain unchanged. If a field is constrained with a multiplicity of one then the 'updateCourseTemplate' request acts as a 'replaceCourseTemplate' request for that field.<br>If the object identified by the supplied SourcedId cannot be located then the request is rejected and the appropriate failure code is returned.<br>The reference counter for the object is incremented in the target system. |
| **Notes:** | The source is responsible for determining the reason of the failure. |

**Table 3.11 Status codes for the 'updateCourseTemplate' operation.**

| Status Code | Explanation of the Cause of the Code |
|---|---|
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor=fullsuccess' | The update request has been fully and successfully implemented by the target system and the identified CourseTemplate object has been changed on the target system. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=unknownobject' | The CourseTemplate object identifier is unknown in the target system and so the object could not be updated. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=invaliddata' | Part or all of the received data was detected as invalid by the target system. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=incompletedata' | Some mandatory part of the data has been detected as missing by the target system. |
| 'CodeMajor=Success'<br>'Severity=Warning'<br>'CodeMinor=partialdatastorage' | The target has stored a subset of the sent data record i.e., some of the optional data has not been stored (all mandatory data has been supplied and stored). |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=unknownvocabulary' | The target system could not identify the defined vocabulary term. This may be due to an incorrect term or a missing vocabulary file. |
| 'CodeMajor=Failure'<br>'Severity=Status' | The target system could not identify the defined metadata vocabulary term. This may be due to an incorrect term or a missing vocabulary file. |

| | |
|---|---|
| ‘CodeMinor=unknownmdvocabulary’ | |
| ‘CodeMajor=Failure’<br>‘Severity=Status’<br>‘CodeMinor=unknownextension’ | The target cannot process the proprietary data model extensions used in the object. |

### 3.2.11 ReplaceCourseTemplate() Operation

| | |
|---|---|
| **Name:** | replaceCourseTemplate |
| **Return Function Parameter:** | *statusInfo:StatusInfo* – the status of the replace request. The permitted status codes are given in Tables 3.12 and A.2. |
| **Supplied (in) Parameters:** | *sourcedId:GUID* – the identifier of the CourseTemplate object to be replaced.<br><br>*courseTemplateRecord:TemplateRecord* – the CourseTemplate data that is to be stored in the object. |
| **Returned (out) Parameters:** | None. |
| **Behavior:** | When the source issues the 'replaceCourseTemplate' request the target is charged with writing the supplied information into the identified object. If any part of the write fails e.g., due to partial invalid data then the whole request is rejected and the record is left in its original state. This is a destructive write-over operation of the entire CourseTemplate object. This is equivalent to a 'createCourseTemplate' but for an object that already exists.<br><br>If the object identified by the supplied SourcedId cannot be located then the request is interpreted as a 'createCourseTemplate' invocation.<br><br>The reference counter for the object is incremented by one in the target system. |
| **Notes:** | The source is responsible for determining the reason of the failure. |

**Table 3.12 Status codes for the 'replaceCourseTemplate' operation.**

| Status Code | Explanation of the Cause of the Code |
|---|---|
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor=fullsuccess' | The replace request has been fully and successfully implemented by the target system and the identified CourseTemplate object has been changed on the target system. |
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor=createsuccess' | The CourseTemplate object identifier is unknown in the target system and so a new object has been successfully created instead. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=invaliddata' | Part or all of the returned data was detected as invalid by the target system. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=incompletedata' | Some mandatory part of the data has been detected as missing by the target system. |
| 'CodeMajor=Success'<br>'Severity=Warning'<br>'CodeMinor=partialdatastorage' | The target has stored a subset of the sent data record i.e., some of the optional data has not been stored (all mandatory data has been supplied and stored). |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=unknownvocabulary' | The target system could not identify the defined vocabulary term. This may be due to an incorrect term or a missing vocabulary file. |
| 'CodeMajor=Failure'<br>'Severity=Status' | The target system could not identify the defined metadata vocabulary term. This may be due to an incorrect term or a missing vocabulary file. |

| Status Code | Explanation of the Cause of the Code |
|---|---|
| ‘CodeMinor=unknownmdvocabulary’ | |
| ‘CodeMajor=Failure’<br>‘Severity=Status’<br>‘CodeMinor=unknownextension’ | The target cannot process the proprietary data model extensions used in the object. |

### 3.2.12 DiscoverCourseTemplateIds() Operation

| | |
|---|---|
| **Name:** | discoverCourseTemplateIds |
| **Return Function Parameter:** | *statusInfo:StatusInfo* – the status of the discover request. The permitted status codes are given in Tables 3.13 and A.2. |
| **Supplied (in) Parameters:** | *queryObject:QueryObject* – this is the query/filter instruction that is to be applied by the target to discover the corresponding CourseTemplate objects. |
| **Returned (out) Parameters:** | *sourcedIdSet:GUIDSet* – the set of identifiers of the CourseTemplate objects whose content conform to the query/filter conditions. |
| **Behavior:** | When the source issues the 'discoverCourseTemplate Ids' the target applies the query/filter instructions to the set of CourseTemplate objects and returns the set of SourcedIds that uphold the query/filter.<br><br>If no CourseTemplate objects have the required properties the returned data set is empty and the success status code returned.<br><br>If the target does not understand or cannot apply the requested query/filter then an error status is returned. |
| **Notes:** | The internal structure of this QueryObject is undefined (it is should be treated as a String encoded 'blob). Later versions of this specification will look at the established best practices for clarification on the use of this operation. |

**Table 3.13 Status codes for the 'discoverCourseTemplateIds' operation.**

| Status Code | Explanation of the Cause of the Code |
|---|---|
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor=fullsuccess' | The query/filter request has been fully and successfully implemented by the target system and the appropriate CourseTemplate identifiers have been read from the target system. |
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor= nosourcedids' | The discover request has been fully and successfully implemented by the target system and no CourseTemplate object identifiers were found. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=unknownquery' | The target system cannot understand the query request that has been received i.e., the query/filter language is unknown. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=toomuchdata' | The data cannot be returned because it would exceed some physical system constraint e.g., permitted size of SOAP message. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=invaliddata' | Part or all of the returned data was detected as invalid by the source system. |

### 3.2.13 ChangeCourseTemplateIdentifier() Operation

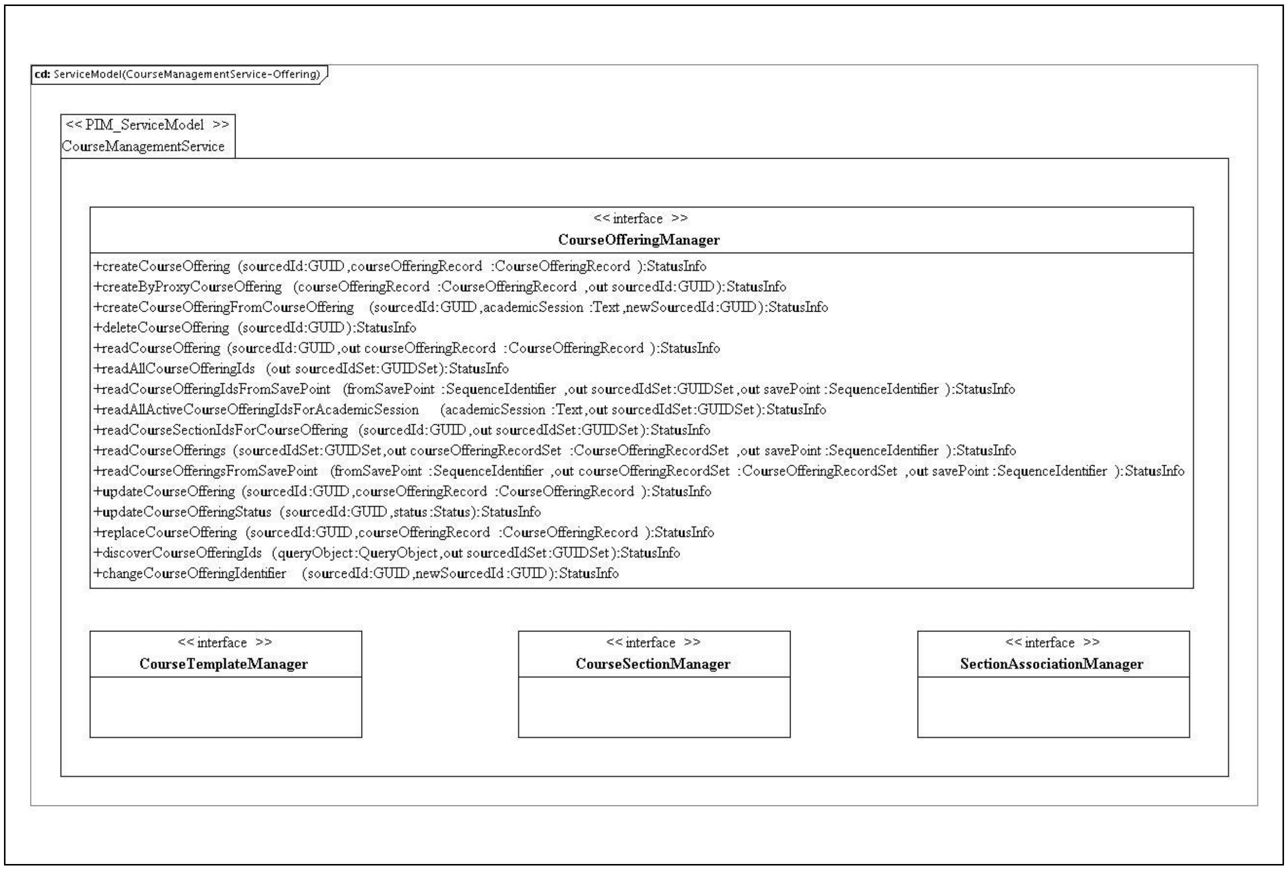
| | |
|---|---|
| **Name:** | changeCourseTemplateIdentifier |
| **Return Function Parameter:** | *StatusInfo* – the status of the change request. The permitted status codes are given in Tables 3.14 and A.2. |
| **Supplied (in) Parameters:** | *sourcedId:GUID* – the identifier of the CourseTemplate object to be changed.<br>*newSourcedId:GUID* – the new identifier to be allocated to the CourseTemplate object. |
| **Returned (out) Parameters:** | None. |
| **Behavior:** | When the source issues the 'changeCourseTemplateIdentifier' request the target is charged with replacing the original SourcedId with the new supplied SourcedId. All membership entries must be similarly changed. All further references to the object must use the new SourcedId otherwise an 'unknown' object failure status code is returned.<br>If the object identified by the supplied SourcedId cannot be located then the request is rejected and the appropriate failure code is returned. |
| **Notes:** | The reference pointer value remains unchanged. |

**Table 3.14 Status codes for the 'changeCourseTemplateIdentifier' operation.**

| Status Code | Explanation of the Cause of the Code |
|---|---|
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor=fullsuccess' | The change identifier request has been fully and successfully implemented by the target system and the CourseTemplate object SourcedId has been changed on the target system. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=idallocinusefail' | The target could not allocate the new unique 'identifier' to the CourseTemplate object as the identifier is already in use. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=unknownobject' | The current CourseTemplate identifier is unknown in the target system and so the object identifier could not be changed. |

## 3.3 CourseOfferingManager Interface Description

The CourseOfferingManager interface class, as shown in Figure 3.2, describes the operations that are permitted on a CourseOfferings. The interface stereotype indicates that there are no attributes for this class. The set of operations are summarized in Table 3.15.



```
+createCourseOffering (sourcedId:GUID ,courseOfferingRecord :CourseOfferingRecord ):StatusInfo
+createByProxyCourseOffering (courseOfferingRecord :CourseOfferingRecord ,out sourcedId:GUID ):StatusInfo
+createCourseOfferingFromCourseOffering (sourcedId:GUID ,academicSession :Text ,newSourcedId:GUID ):StatusInfo
+deleteCourseOffering (sourcedId:GUID ):StatusInfo
+readCourseOffering (sourcedId:GUID ,out courseOfferingRecord :CourseOfferingRecord ):StatusInfo
+readAllCourseOfferingIds (out sourcedIdSet:GUIDSet):StatusInfo
+readCourseOfferingIdsFromSavePoint (fromSavePoint :SequenceIdentifier ,out sourcedIdSet:GUIDSet ,out savePoint :SequenceIdentifier ):StatusInfo
+readAllActiveCourseOfferingIdsForAcademicSession (academicSession :Text ,out sourcedIdSet:GUIDSet ):StatusInfo
+readCourseSectionIdsForCourseOffering (sourcedId:GUID ,out sourcedIdSet:GUIDSet ):StatusInfo
+readCourseOfferings (sourcedIdSet:GUIDSet ,out courseOfferingRecordSet :CourseOfferingRecordSet ,out savePoint :SequenceIdentifier ):StatusInfo
+readCourseOfferingsFromSavePoint (fromSavePoint :SequenceIdentifier ,out courseOfferingRecordSet :CourseOfferingRecordSet ,out savePoint :SequenceIdentifier ):StatusInfo
+updateCourseOffering (sourcedId:GUID ,courseOfferingRecord :CourseOfferingRecord ):StatusInfo
+updateCourseOfferingStatus (sourcedId:GUID ,status :Status ):StatusInfo
+replaceCourseOffering (sourcedId:GUID ,courseOfferingRecord :CourseOfferingRecord ):StatusInfo
+discoverCourseOfferingIds (queryObject:QueryObject ,out sourcedIdSet:GUIDSet ):StatusInfo
+changeCourseOfferingIdentifier (sourcedId:GUID ,newSourcedId :GUID ):StatusInfo
```

**Figure 3.2 CourseManagementService CourseOfferingManager interface definition.**

**Table 3.15 Summary of operations for CourseOfferingManager.**

| Operation | Description |
|---|---|
| createCourseOffering | To request the creation of a populated CourseOffering object on the target system where the source is responsible for the allocation of the unique identifier. |
| createByProxyCourseOffering | To request the creation of a populated CourseOffering object on the target system where the target is responsible for the allocation of the unique identifier. |
| createCourseOfferingFromCourseOffering | To create a new CourseOffering from the supplied CourseOffering for a particular academic session. |
| deleteCourseOffering | To request the deletion of a CourseOffering object. The CourseOffering object is deleted and all of its associated relationships. |

| Operation | Description |
|---|---|
| readCourseOffering | To read the full contents of the identified CourseOffering object. The target must return all of the data it has for the identified CourseOffering object. |
| readAllCourseOfferingIds | To obtain the set of SourcedIds which have been assigned to CourseOffering objects. |
| readAllActiveCourseOfferingIdsForAcademicSession | To obtain the set of SourcedIds for all of the active CourseOfferings for the identified academic session. |
| readCourseSectionIdsForCourseOffering | To obtain the set of SourcedIds of the CourseSections associated with the CourseOffering. |
| readCourseOfferingIdsFromSavePoint | To obtain the set of SourcedIds for CourseOfferings objects which have been altered since the requested reference point. The reference point is set as zero at creation and incremented after every write operation. |
| readCourseOfferings | To obtain the CourseOffering objects for a defined set of identifiers. This results in a single transaction that may require the exchange of a large volume of data in the response message. |
| readCourseOfferingsFromSavePoint | To obtain the set of CourseOffering objects which have been altered since the requested reference point. The reference point is set as ‘zero’ at creation and incremented after every write operation. This results in a single transaction that may require the exchange of a large volume of data in the response message. |
| replaceCourseOffering | To replace the content of the identified CourseOffering object. The target must write the new data into the CourseOffering object. This is a destructive write-over of all of the original information. In the case of the object not existing, this operation acts as an implied ‘createCourseOffering’. |
| updateCourseOffering | To write new content into the identified CourseOffering object. The target must write the new data into the CourseOffering object. This is an additive operation. |
| updateCourseOfferingStatus | To change the status of the identified CourseOffering to the supplied value. |
| discoverCourseOfferingIds | To obtain the set of SourcedIds for CourseOffering objects whose properties agree with those defined in the query/filter. |
| changeCourseOfferingIdentifier | To change the SourcedId of the CourseOffering record. The completion of this operation will result in later actions using the original SourcedId reporting an unknown identifier status. |

Note: In most cases the above operations act on a single instance of a CourseOffering object i.e., ‘createCourseOffering’, ‘createByProxyCourseOffering’, ‘deleteCourseOffering’, ‘readCourseOffering’, ‘replaceCourseOffering’ and ‘updateCourseOffering’.

### 3.3.1 CreateCourseOffering() Operation

| | |
|---|---|
| **Name:** | createCourseOffering |
| **Return Function Parameter:** | *StatusInfo* – the status of the creation request. The permitted status codes are defined in Tables 3.16 and A.2. |
| **Supplied (in) Parameters:** | *sourcedId:GUID* – the SourcedId allocated by the source system. This is the identifier that must also be assigned within the target system.<br>*courseOfferingRecord:CourseOfferingRecord* – the CourseOffering data that is to be stored in the new object. |
| **Returned (out) Parameters:** | None. |
| **Behavior:** | When the source issues the 'createCourseOffering' request the target is instructed to create the populated CourseOffering object and to allocate that structure the SourcedId passed by the source. If the supplied SourcedId has already been allocated to another object then the request is rejected and the appropriate failure code is returned. The reference point identifier is set to zero for the CourseOffering object in both the source and target. |
| **Notes:** | This request contains the initial content for the CourseOffering record. More content can be added/replaced using the 'updateCourseOffering' and/or 'replaceCourseOffering' requests. |

**Table 3.16 Status codes for the 'createCourseOffering' operation.**

| Status Code | Explanation of the Cause of the Code |
|---|---|
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor=fullsuccess' | The creation request has been fully and successfully implemented by the target system and the CourseOffering object has been created with a unique identifier. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=idallocinusefail' | The target could not allocate the required unique SourcedId to the CourseOffering object as it is already in use. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=overflowfail' | The target could not create the CourseOffering object due to lack of target allocation memory. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=invaliddata' | Part or all of the supplied data was detected as invalid by the target system. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=incompletedata' | Some mandatory part of the data has been detected as missing by the target system. |
| 'CodeMajor=Success'<br>'Severity=Warning'<br>'CodeMinor=partialdatastorage' | The target has stored a subset of the sent data record i.e., some of the optional data has not been stored (all mandatory data has been supplied and stored). |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=unknownmdvocabulary' | The target system could not identify the defined metadata vocabulary term. This may be due to an incorrect term or a missing vocabulary file. |
| 'CodeMajor=Failure' | The target system could not identify the defined vocabulary term. This |

| Status Code | Explanation of the Cause of the Code |
|---|---|
| 'Severity=Status'<br>'CodeMinor=unknownvocabulary' | may be due to an incorrect term or a missing vocabulary file. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=unknownextension' | The target cannot process and store the proprietary data model extensions used in the object. |

### 3.3.2 CreateByProxyCourseOffering() Operation

| | |
|---|---|
| **Name:** | createByProxyCourseOffering |
| **Return Function Parameter:** | *StatusInfo* – the status of the creation request. The permitted status codes are defined in Tables 3.17 and A.2. |
| **Supplied (in) Parameters:** | *courseOfferingRecord:CourseOfferingRecord* – the CourseOffering data that is to be stored in the new object. |
| **Returned (out) Parameters:** | *sourcedId:GUID* – the identifier allocated by the target to the newly created CourseOffering object. |
| **Behavior:** | When the source issues the 'createByProxyCourseOffering' request the target is instructed to create the populated CourseOffering object and to allocate that record a unique SourcedId. The reference point identifier is set to zero for the CourseOffering object in both the source and target. |
| **Notes:** | This request contains the initial content for the CourseOffering object. More content can be added/replaced using the 'updateCourseOffering' and/or 'replaceCourseOffering' requests. |

**Table 3.17 Status codes for the 'createByProxyCourseOffering' operation.**

| Status Code | Explanation of the Cause of the Code |
|---|---|
| 'CodeMajor=Success' 'Severity=Status' 'CodeMinor=fullsuccess' | The creation request has been fully and successfully implemented by the target system and the CourseOffering object has been created with the identifier supplied by the source. |
| 'CodeMajor=Failure' 'Severity=Status' 'CodeMinor=idallocfail' | The target could not allocate a unique SourcedId to the CourseOffering object because there are no more spare identifiers available. |
| 'CodeMajor=Failure' 'Severity=Status' 'CodeMinor=overflowfail' | The target could not create the CourseOffering object due to lack of target allocation memory. |
| 'CodeMajor=Failure' 'Severity=Status' 'CodeMinor=invaliddata' | Part or all of the supplied data was detected as invalid by the source system. |
| 'CodeMajor=Failure' 'Severity=Status' 'CodeMinor=incompletedata' | Some mandatory part of the data has been detected as missing by the target system. |
| 'CodeMajor=Success' 'Severity=Warning' 'CodeMinor=partialdatastorage' | The target has stored a subset of the sent data record i.e., some of the optional data has not been stored (all mandatory data has been supplied and stored). |
| 'CodeMajor=Failure' 'Severity=Status' 'CodeMinor=unknownmdvocabulary' | The target system could not identify the defined metadata vocabulary term. This may be due to an incorrect term or a missing vocabulary file. |
| 'CodeMajor=Failure' 'Severity=Status' 'CodeMinor=unknownvocabulary' | The target system could not identify the defined vocabulary term. This may be due to an incorrect term or a missing vocabulary file. |
| 'CodeMajor=Failure' | The target cannot process and store the proprietary data model extensions |

| Status Code | Explanation of the Cause of the Code |
|---|---|
| ‘Severity=Status’ ‘CodeMinor=unknownextension’ | used in the object. |

### 3.3.3 CreateCourseOfferingFromCourseOffering() Operation

| | |
|---|---|
| **Name:** | createCourseOfferingFromCourseOffering |
| **Return Function Parameter:** | *StatusInfo* – the status of the creation request. The permitted status codes are defined in Tables 3.18 and A.2. |
| **Supplied (in) Parameters:** | *sourcedId:GUID* – the SourcedId for the CourseOffering to be cloned.<br>*academicSession:AcademicSession* – the academic session (from a pre-defined vocabulary) of the new CourseOffering object.<br>*newSourcedId:GUID* – the SourcedId to be allocated to the newly created 'courseOffering' object. |
| **Returned (out) Parameters:** | None. |
| **Behavior:** | When the source issues the 'createCourseOfferingFromCourseOffering' request the target is instructed to find the source CourseOffering object and to create an equivalent CourseOffering but with the new SourcedId and academic session. The reference point identifier is set to zero for the new CourseOffering object in both the source and target. |
| **Notes:** | An invalid status code will be returned if the new SourcedId has already been allocated or the academic session is incorrect. |

**Table 3.18 Status codes for the 'createCourseOfferingFromCourseOffering' operation.**

| Status Code | Explanation of the Cause of the Code |
|---|---|
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor=fullsuccess' | The creation request has been fully and successfully implemented by the target system and the new CourseOffering object has been created with a unique identifier. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=unknownobject' | The CourseOffering object identifier is unknown in the target system. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=idallocinusefail' | The target could not allocate the required unique SourcedId to the new CourseOffering object as it is already in use. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=overflowfail' | The target could not create the CourseOffering object due to lack of target allocation memory. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=invaliddata' | An invalid academic session value has been received by the target system. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=incompletedata' | Some mandatory part of the data has been detected as missing by the target system. |

### 3.3.4 DeleteCourseOffering() Operation

| | |
|---|---|
| **Name:** | deleteCourseOffering |
| **Return Function Parameter:** | *StatusInfo* – the status of the delete request. The permitted status codes are defined in Tables 3.19 and A.2. |
| **Supplied (in) Parameters:** | *sourcedId:GUID* – the identifier to be used by the target to identify the 'courseOffering' object. |
| **Returned (out) Parameters:** | None. |
| **Behavior:** | When the source issues the 'deleteCourseOffering' request the target is instructed to delete the identified CourseOffering object and to remove the reference to the CourseOffering from any of the related CourseTemplate, CourseSection and Membership objects. If the object identified by the supplied SourcedId cannot be located then the request is rejected and the appropriate failure code is returned. |
| **Notes:** | Deletion of the CourseOffering record does not necessarily result in the destruction of the data within the server. The true state of the data in the target is unknown. |

**Table 3.19 Status codes for the 'deleteCourseOffering' operation.**

| Status Code | Explanation of the Cause of the Code |
|---|---|
| 'CodeMajor=Success' 'Severity=Status' 'CodeMinor=fullsuccess' | The deletion request has been fully and successfully implemented by the target system and the CourseOffering object has been deleted. The corresponding Membership objects and other relationships have also been deleted. |
| 'CodeMajor=Failure' 'Severity=Status' 'CodeMinor=unknownobject' | The CourseOffering object identifier is unknown in the target system and so the object could not be deleted. |
| 'CodeMajor=Failure' 'Severity=Status' 'CodeMinor= deletefailure' | The target system has not been able to delete the identified CourseOffering object. |

### 3.3.5 ReadCourseOffering() Operation

| | |
|---|---|
| **Name:** | readCourseOffering |
| **Return Function Parameter:** | *StatusInfo* – the status of the read request. The permitted status codes are defined in Tables 3.20 and A.2. |
| **Supplied (in) Parameters:** | *sourcedId:GUID* – the identifier of the CourseOffering object to be read. |
| **Returned (out) Parameters:** | *courseOfferingRecord:CourseOfferingRecord* – the CourseOffering data that is read from the object. |
| **Behavior:** | When the source issues the 'readCourseOffering' request the target is charged with retrieving the identified record from its database and returning this data to the source. The target is responsible for ensuring that the object contains valid data. If the object identified by the SourcedId cannot be located then the request is rejected and the appropriate failure code is returned. |
| **Notes:** | The returned CourseOffering object can only be trusted if the corresponding status code is 'success'. |

**Table 3.20 Status codes for the 'readCourseOffering' operation.**

| Status Code | Explanation of the Cause of the Code |
|---|---|
| 'CodeMajor=Success' 'Severity=Status' 'CodeMinor=fullsuccess' | The read request has been fully and successfully implemented by the target system and the identified CourseOffering object has been read from the target system. |
| 'CodeMajor=Failure' 'Severity=Status' 'CodeMinor=unknownobject' | The CourseOffering object identifier is unknown in the target system and so the object could not be read. |
| 'CodeMajor=Failure' 'Severity=Status' 'CodeMinor=targetreadfailure' | The target system has detected an error in the stored CourseOffering object and so cannot return the data. |
| 'CodeMajor=Failure' 'Severity=Status' 'CodeMinor=invaliddata' | Part or all of the returned data was detected as invalid by the source system. |
| 'CodeMajor=Failure' 'Severity=Status' 'CodeMinor=incompletedata' | Some mandatory part of the data has been detected as missing by the source system. |
| 'CodeMajor=Success' 'Severity=Warning' 'CodeMinor=partialdatastorage' | The target has only returned a subset of the data expected by the source e.g., only the mandatory parts. |
| 'CodeMajor=Failure' 'Severity=Status' 'CodeMinor=unknownvocabulary' | The source system could not identify the defined vocabulary term. This may be due to an incorrect term or a missing vocabulary file. |
| 'CodeMajor=Failure' 'Severity=Status' 'CodeMinor=unknownextension' | The source cannot process and store the proprietary data model extensions used in the object. |

### 3.3.6 ReadAllCourseOfferingIds() Operation

| | |
|---|---|
| **Name:** | readAllCourseOfferingIds |
| **Return Function Parameter:** | *StatusInfo* – the status of the read request. The permitted status codes are defined in Tables 3.21 and A.2. |
| **Supplied (in) Parameters:** | None. |
| **Returned (out) Parameters:** | *sourcedIdSet:GUIDSet* – the set of SourcedIds for all of the CourseOffering objects in the target's database. |
| **Behavior:** | When the source issues the 'readAllCourseOfferingIds' request the target is charged with returning the SourcedIds of all of the CourseOfferings in its course offering database. |
| **Notes:** | If no SourcedIds have been allocated then the returned data set is empty and the success status code returned. |

**Table 3.21 Status codes for the 'readAllCourseOfferingIds' operation.**

| Status Code | Explanation of the Cause of the Code |
|---|---|
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor=fullsuccess' | The read request has been fully and successfully implemented by the target system and all the CourseOffering object identifiers have been read from the target system. |
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor= nosourcedids' | The read request has been fully and successfully implemented by the target system and no CourseOffering object identifiers were found. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=toomuchdata' | The requested data cannot be returned because it would exceed some physical system constraint e.g., permitted size of SOAP message. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=invaliddata' | Part or all of the returned data was detected as invalid by the source system. |

### 3.3.7 ReadAllActiveCourseOfferingIdsForAcademicSession() Operation

| | |
|---|---|
| **Name:** | readAllActiveCourseOfferingIdsForAcademicSession |
| **Return Function Parameter:** | *StatusInfo* – the status of the read request. The permitted status codes are defined in Tables 3.22 and A.2. |
| **Supplied (in) Parameters:** | *academicSession:AcademicSession* – the academic session (from a pre-defined vocabulary). |
| **Returned (out) Parameters:** | *sourcedIdSet:GUIDSet* – the set of identifiers of the active CourseOffering objects. |
| **Behavior:** | When the source issues the 'readAllActiveCourseOfferingIdsForAcademicSession' request the target is charged with returning the identifiers of all active CourseOfferings for the defined academic session. The state of a CourseOffering is held within its 'status' field in the data model. |
| **Notes:** | If there are no active CourseOfferings in the academic session, an empty set of SourcedIds is returned and the success status code returned. |

**Table 3.22 Status codes for the 'readAllActiveCourseOfferingIdsForAcademicSession' operation.**

| Status Code | Explanation of the Cause of the Code |
|---|---|
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor=fullsuccess' | The read request has been fully and successfully implemented by the target system and the set of associated CourseOffering identifiers have been read from the target system. |
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor= nosourcedids' | The read request has been fully and successfully implemented by the target system and no CourseOffering object identifiers were found. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=toomuchdata' | The requested data cannot be returned because it would exceed some physical system constraint e.g., permitted size of SOAP message. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=invaliddata' | An invalid academic session value has been received by the target system. |

### 3.3.8 ReadCourseSectionIdsForCourseOffering() Operation

| | |
|---|---|
| **Name:** | readCourseOfferingSectionIdList |
| **Return Function Parameter:** | *StatusInfo* – the status of the read request. The permitted status codes are defined in Tables 3.23 and A.2. |
| **Supplied (in) Parameters:** | *sourcedId:GUID* – the sourcedId of the CourseOffering whose associated CourseSections are to be identified. |
| **Returned (out) Parameters:** | *sourcedIdSet:GUIDSet* – the set of SourcedIds for all of the associated CourseSection objects in the target's CourseSection database. |
| **Behavior:** | When the source issues the 'readCourseOfferingSectionIdList' request the target is charged with using the supplied sourcedId of the reference CourseOffering to return the SourcedIds of all the associated course sections. These relationships are identified by searching the CourseSection database for the appropriate parent CourseOffering.<br><br>An error code is returned if the target does not recognize the sourcedId as a valid CourseOffering. |
| **Notes:** | The returned set of SourcedIds may or may not already be identified as allocated in the CourseSection database in the source system. If no associated CourseSections are found then the returned data set is empty and the success status code returned. |

**Table 3.23 Status codes for the 'readCourseSectionIdsForCourseOffering' operation.**

| Status Code | Explanation of the Cause of the Code |
|---|---|
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor=fullsuccess' | The read request has been fully and successfully implemented by the target system and the set of associated CourseSection identifiers have been read from the target system. |
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor= nosourcedids' | The read request has been fully and successfully implemented by the target system and no CourseSection object identifiers were found. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=unknownobject' | The CourseOffering object identifier is unknown in the target system and so the object could not be read. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=toomuchdata' | The requested data cannot be returned because it would exceed some physical system constraint e.g., permitted size of SOAP message. |

### 3.3.9 ReadCourseOfferingIdsFromSavePoint() Operation

| | |
|---|---|
| **Name:** | readCourseOfferingIdsFromSavePoint |
| **Return Function Parameter:** | *StatusInfo* – the status of the read request. The permitted status codes are defined in Tables 3.24 and A.2. |
| **Supplied (in) Parameters:** | *fromSavePoint:SequenceIdentifier* – the reference point from which all of the changed identifier actions are to be read. This is the value in the source system. |
| **Returned (out) Parameters:** | *sourcedIdSet:GUIDSet* – the set of SourcedIds of the CourseOffering objects stored on the target.<br>*savePoint:SequenceIdentifier* – the value of the reference point counter in the target system. |
| **Behavior:** | When the source issues the 'readCourseOfferingIdsFromSavePoint' the target returns the set of SourcedIds that have been altered from the defined reference point.<br>If the reference counter in the source is greater than that in the target then an error code and the target value for the reference point are returned. |
| **Notes:** | If no SourcedIds have been allocated then the returned data set is empty and the success status code returned. |

**Table 3.24 Status codes for the 'readCourseOfferingIdsFromSavePoint' operation.**

| Status Code | Explanation of the Cause of the Code |
|---|---|
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor=fullsuccess' | The read request has been fully and successfully implemented by the target system and the identified CourseOffering object has been read from the target system. |
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor= nosourcedids' | The read request has been fully and successfully implemented by the target system and no CourseOffering object identifiers were found. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=toomuchdata' | The data cannot be returned because it would exceed some physical system constraint e.g., permitted size of SOAP message. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=invaliddata' | Part or all of the returned data was detected as invalid by the source system. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=savepointerror' | An error has occurred in the processing of the save-point identifier information making it impossible to read the correct objects from the database. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=savepointsyncerror' | The value of the save point reference from the source was later than that of the target system. No identifiers have been returned. The target system savepoint value has been updated to that supplied by the source system. |

### 3.3.10 ReadCourseOfferings() Operation

| | |
|---|---|
| **Name:** | readCourseOfferings |
| **Return Function Parameter:** | *StatusInfo* – the status of the read request. The permitted status codes are defined in Tables 3.25 and A.2. |
| **Supplied (in) Parameters:** | *sourcedIdSet:GUIDSet* – the set of identifiers of the CourseOffering objects to be read. |
| **Returned (out) Parameters:** | *courseOfferingRecordSet:CourseOfferingRecordSet* – the set of course offering records.<br>*savePoint:SequenceIdentifier* – the value of the reference point counter in the target system. |
| **Behavior:** | When the source issues the 'readCourseOfferings' request the target is charged with retrieving the identified set of objects from its database and placing this information in a persistent data file. The associated read savePoint reference is updated and returned.<br>If the object identified by the supplied 'SourcedId' cannot be located then the request is rejected and a partial success code is returned for the operation. The target is responsible for ensuring that the records contain valid data. The target should attempt to successfully complete as much of the request as possible. |
| **Notes:** | A returned CourseOffering object is only present in the data file if the object has been located in the target system and the full data set returned.<br>The enclosed data may result in a long response message. |

**Table 3.25 Status codes for the 'readCourseOfferings' operation.**

| Status Code | Explanation of the Cause of the Code |
|---|---|
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor=fullsuccess' | The read request has been fully and successfully implemented by the target system and the identified CourseOffering objects have been read from the target system. |
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor=partialreadfail' | Some of the CourseOffering object identifiers are unknown in the target system and so those objects could not be read. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=invaliddata' | Part or all of the returned data was detected as invalid by the source system. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=unknownvocabulary' | The source system could not identify the defined vocabulary term. This may be due to an incorrect term or a missing vocabulary file. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=toomuchdata' | The data cannot be returned because it would exceed some physical system constraint e.g., permitted size of SOAP message. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=targetreadfailure' | The target system has detected an error in the stored CourseTemplate objects and so cannot return the data. |

### 3.3.11 ReadCourseOfferingsFromSavePoint() Operation

| | |
|---|---|
| **Name:** | readCourseOfferingsFromSavePoint |
| **Return Function Parameter:** | *StatusInfo* – the status of the creation request. The permitted status codes are defined in Tables 3.26 and A.2. |
| **Supplied (in) Parameters:** | *fromSavePoint:SequenceIdentifier* – the reference point from which all of the changed identifier actions are to be read. This is the value in the source system. |
| **Returned (out) Parameters:** | *courseOfferingRecordSet:CourseOfferingRecordSet* – the set of course offering records.<br><br>*savePoint:SequenceIdentifier* – the value of the reference point counter in the target system. |
| **Behavior:** | When the source issues the 'readCourseOfferingsFromSavePoint' request the target is charged with reading the objects that have been altered from the defined reference point.<br><br>If the reference counter in the source is greater than that in the target then an error code and the target value for the reference point are returned. |
| **Notes:** | If no objects have been allocated then the return message will be empty.<br><br>The enclosed data may result in a long response message. |

**Table 3.26 Status codes for the 'readCourseOfferingsFromSavePoint' operation.**

| Status Code | Explanation of the Cause of the Code |
|---|---|
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor=fullsuccess' | The read request has been fully and successfully implemented by the target system and the identified CourseOffering objects have been read from the target system. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=savepointerror' | An error has occurred in the processing of the save-point identifier information making it impossible to read the correct objects from the database. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=savepointsyncerror' | The value of the save point reference from the source was later than that of the target system. No identifiers have been returned. The target system savepoint value is returned to the source system for information. |
| 'CodeMajor=Success'<br>'Severity=Warning'<br>'CodeMinor=partialdatareturn' | The target has only returned a subset of the data object e.g., only the mandatory parts. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=toomuchdata' | The data cannot be returned because it would exceed some physical system constraint e.g., permitted size of SOAP message. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=targetreadfailure' | The target system has detected an error in the stored LineItem object and so cannot return the data. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=invaliddata' | Part or all of the returned data was detected as invalid by the target system. |

### 3.3.12 ReplaceCourseOffering() Operation

| | |
|---|---|
| **Name:** | replaceCourseOffering |
| **Return Function Parameter:** | *StatusInfo* – the status of the replace request. The permitted status codes are defined in Tables 3.27 and A.2. |
| **Supplied (in) Parameters:** | *sourcedId:GUID* – the identifier of the CourseOffering object to be replaced.<br>*courseOfferingRecord:OfferingRecord* – the CourseOffering data that is to be stored in the object. |
| **Returned (out) Parameters:** | None. |
| **Behavior:** | When the source issues the 'replaceCourseOffering' request the target is charged with writing the supplied information into the identified object. If any part of the write fails e.g., due to partial invalid data then the whole request is rejected and the record is left in its original state. This is a destructive write-over operation of the entire CourseOffering object. This is equivalent to a 'createCourseOffering' but for an object that already exists.<br>If the object identified by the supplied SourcedId cannot be located then the request is interpreted as a 'createCourseOffering' invocation.<br>The reference counter for the object is incremented by one in the target system. |
| **Notes:** | The source is responsible for determining the reason of the failure. |

**Table 3.27 Status codes for the 'replaceCourseOffering' operation.**

| Status Code | Explanation of the Cause of the Code |
|---|---|
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor=fullsuccess' | The replace request has been fully and successfully implemented by the target system and the identified CourseOffering object has been changed on the target system. |
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor=createsuccess' | The CourseOffering object identifier is unknown in the target system and so a new object has been successfully created instead. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=invaliddata' | Part or all of the returned data was detected as invalid by the target system. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=incompletedata' | Some mandatory part of the data has been detected as missing by the target system. |
| 'CodeMajor=Success'<br>'Severity=Warning'<br>'CodeMinor=partialdatastorage' | The target has stored a subset of the sent data record i.e., some of the optional data has not been stored (all mandatory data has been supplied and stored). |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=unknownvocabulary' | The target system could not identify the defined vocabulary term. This may be due to an incorrect term or a missing vocabulary file. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=unknownmdvocabulary' | The target system could not identify the defined metadata vocabulary term. This may be due to an incorrect term or a missing vocabulary file. |

| Status Code | Explanation of the Cause of the Code |
|---|---|
| ‘CodeMajor=Failure’<br>‘Severity=Status’<br>‘CodeMinor=unknownextension’ | The target cannot process the proprietary data model extensions used in the object. |

### 3.3.13 UpdateCourseOffering() Operation

| | |
|---|---|
| **Name:** | updateCourseOffering |
| **Return Function Parameter:** | *StatusInfo* – the status of the update request. The permitted status codes are defined in Tables 3.28 and A.2. |
| **Supplied (in) Parameters:** | *sourcedId:GUID* – the identifier of the CourseOffering object to be updated.<br>*courseOfferingRecord:CourseOfferingRecord* – the CourseOffering data that is to be stored in the object. |
| **Returned (out) Parameters:** | None. |
| **Behavior:** | When the source issues the 'updateCourseOffering' request the target is charged with writing the supplied information into the identified record. If any part of the write fails e.g., due to partial invalid data then the whole request is rejected and the record is left in its original state. This is an additive write operation of all the data fields supplied in the update request and fields not supplied remain unchanged. If a field is constrained with a multiplicity of one then the 'updateCourseOffering' request acts as a replaceCourseOffering' request for that field.<br>If the object identified by the supplied SourcedId cannot be located then the request is rejected and the appropriate failure code is returned.<br>The reference counter for the object is incremented in the target system. |
| **Notes:** | The source is responsible for determining the reason of the failure. |

**Table 3.28 Status codes for the 'updateCourseOffering' operation.**

| Status Code | Explanation of the Cause of the Code |
|---|---|
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor=fullsuccess' | The update request has been fully and successfully implemented by the target system and the identified CourseOffering object has been changed on the target system. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=unknownobject' | The CourseOffering object identifier is unknown in the target system and so the object could not be updated. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=invaliddata' | Part or all of the returned data was detected as invalid by the target system. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=incompletedata' | Some mandatory part of the data has been detected as missing by the target system. |
| 'CodeMajor=Success'<br>'Severity=Warning'<br>'CodeMinor=partialdatastorage' | The target has stored a subset of the sent data record i.e., some of the optional data has not been stored (all mandatory data has been supplied and stored). |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=unknownvocabulary' | The target system could not identify the defined vocabulary term. This may be due to an incorrect term or a missing vocabulary file. |
| 'CodeMajor=Failure' | The target system could not identify the defined metadata vocabulary |

| | |
|---|---|
| ‘Severity=Status’<br>‘CodeMinor=unknownmdvocabulary’ | term. This may be due to an incorrect term or a missing vocabulary file. |
| ‘CodeMajor=Failure’<br>‘Severity=Status’<br>‘CodeMinor=unknownextension’ | The target cannot process the proprietary data model extensions used in the object. |

### 3.3.14 UpdateCourseOfferingStatus() Operation

| | |
|---|---|
| **Name:** | updateCourseOfferingStatus |
| **Return Function Parameter:** | *StatusInfo* – the status of the update request. The permitted status codes are defined in Tables 3.29 and A.2. |
| **Supplied (in) Parameters:** | *sourcedId:GUID* – the identifier of the CourseOffering object whose status is to be updated.<br>*status:Status* – the new Status (form an agreed vocabulary) that is to be assigned to the CourseOffering object. |
| **Returned (out) Parameters:** | None. |
| **Behavior:** | When the source issues the 'updateCourseOfferingStatus' request the target is charged with changing the status of the associated object. This is a destructive write i.e., the previous status is replaced.<br>If the object identified by the supplied SourcedId cannot be located then the request is rejected and the appropriate failure code is returned.<br>The reference counter for the object is incremented by one in the target system. |
| **Notes:** | None. |

**Table 3.29 Status codes for the 'updateCourseOfferingStatus' operation.**

| Status Code | Explanation of the Cause of the Code |
|---|---|
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor=fullsuccess' | The update status request has been fully and successfully implemented by the target system and the status of the identified CourseOffering object has been changed on the target system. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=unknownobject' | The CourseOffering object identifier is unknown in the target system and so the object could not be updated. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=invaliddata' | The target system has received and invalid status value. |

### 3.3.15 DiscoverCourseOfferingIds() Operation

| | |
|---|---|
| **Name:** | discoverCourseOfferingIds |
| **Return Function Parameter:** | *StatusInfo* – the status of the discover request. The permitted status codes are defined in Tables 3.30 and A.2. |
| **Supplied (in) Parameters:** | *queryObject:QueryObject* – this is the query/filter instruction that is to be applied by the target to discover the corresponding CourseOffering objects. |
| **Returned (out) Parameters:** | *sourcedIdSet:GUIDSet* – the set of identifiers of the CourseOffering objects whose content conform to the query/filter conditions. |
| **Behavior:** | When the source issues the 'discoverCourseOfferingIds' the target applies the query/filter instructions to the set of CourseOffering objects and returns the set of SourcedIds that uphold the query/filter.<br>If no CourseOffering objects have the required properties the returned data set is empty and the success status code returned.<br>If the target does not understand or cannot apply the requested query/filter then an error status is returned. |
| **Notes:** | The internal structure of this QueryObject is undefined (it is should be treated as a String encoded 'blob). Later versions of this specification will look at the established best practices for clarification on the use of this operation. |

**Table 3.30 Status codes for the 'discoverCourseOfferingIds' operation.**

| Status Code | Explanation of the Cause of the Code |
|---|---|
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor=fullsuccess' | The discover request has been fully and successfully implemented by the target system and the appropriate CourseOffering identifiers have been retrieved from the target system. |
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor= nosourcedids' | The discover request has been fully and successfully implemented by the target system and no CourseOffering object identifiers were found. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=toomuchdata' | The data cannot be returned because it would exceed some physical system constraint e.g., permitted size of SOAP message. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=unknownquery' | The target system cannot understand the query request that has been received i.e., the query/filter language is unknown. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=invaliddata' | Part or all of the returned data was detected as invalid by the source system. |

### 3.3.16 ChangeCourseOfferingIdentifier() Operation

| | |
|---|---|
| **Name:** | changeCourseOfferingIdentifier |
| **Return Function Parameter:** | *StatusInfo* – the status of the change identifier request. The permitted status codes are given in Tables 3.31 and A.2. |
| **Supplied (in) Parameters:** | *sourcedId:GUID* – the identifier of the CourseOffering object to be changed.<br><br>*newSourcedId:GUID* – the new identifier to be allocated to the CourseOffering object. |
| **Returned (out) Parameters:** | None. |
| **Behavior:** | When the source issues the 'changeCourseOfferingIdentifier' request the target is charged with replacing the original SourcedId with the new supplied SourcedId. All membership entries must be similarly changed. All further references to the object must use the new SourcedId otherwise an 'unknown' object failure status code is returned.<br><br>If the object identified by the supplied SourcedId cannot be located then the request is rejected and the appropriate failure code is returned. |
| **Notes:** | The reference pointer value remains unchanged. |

**Table 3.31 Status codes for the 'changeCourseOfferingIdentifier' operation.**

| Status Code | Explanation of the Cause of the Code |
|---|---|
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor=fullsuccess' | The change identifier request has been fully and successfully implemented by the target system and the CourseOffering object SourcedId has been changed on the target system. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=idallocinusefail' | The target could not allocate the new unique 'identifier' to the CourseOffering object as the identifier is already in use. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=unknownobject' | The current CourseOffering SourcedId identifier is unknown in the target system and so the object identifier could not be changed. |

## 3.4 CourseSectionManager Interface Description

The CourseSectionManager interface class, as shown in Figure 3.3, describes the operations that are permitted on a CourseSections. The interface stereotype indicates that there are no attributes for this class. The set of operations are summarized in Table 3.32.

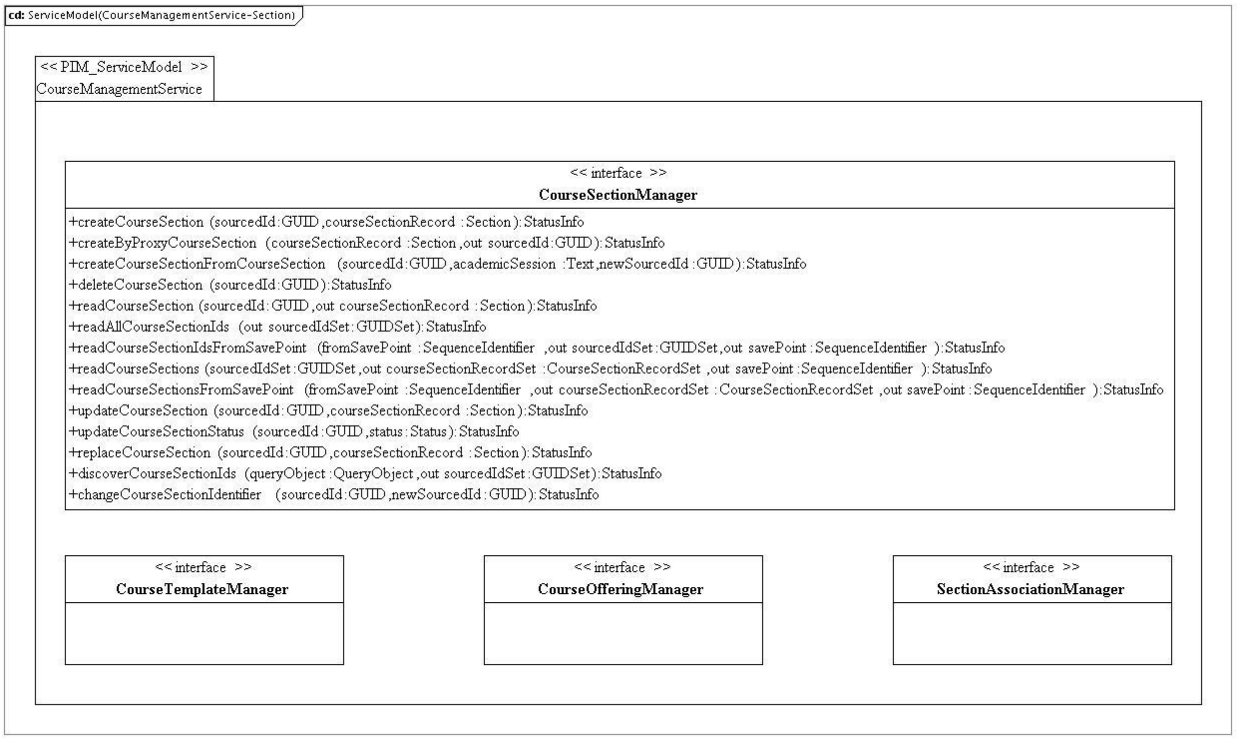

cd: ServiceModel(CourseManagementService-Section)
<< PIM_ServiceModel >>
CourseManagementService
<< interface >>
CourseSectionManager
+createCourseSection (sourcedId : GUID ,courseSectionRecord : Section ): StatusInfo
+createByProxyCourseSection (courseSectionRecord : Section ,out sourcedId : GUID ): StatusInfo
+createCourseSectionFromCourseSection (sourcedId : GUID ,academicSession : Text ,newSourcedId : GUID ): StatusInfo
+deleteCourseSection (sourcedId : GUID ): StatusInfo
+readCourseSection (sourcedId : GUID ,out courseSectionRecord : Section ): StatusInfo
+readAllCourseSectionIds (out sourcedIdSet : GUIDSet ): StatusInfo
+readCourseSectionIdsFromSavePoint (fromSavePoint : SequenceIdentifier ,out sourcedIdSet : GUIDSet ,out savePoint : SequenceIdentifier ): StatusInfo
+readCourseSections (sourcedIdSet : GUIDSet ,out courseSectionRecordSet : CourseSectionRecordSet ,out savePoint : SequenceIdentifier ): StatusInfo
+readCourseSectionsFromSavePoint (fromSavePoint : SequenceIdentifier ,out courseSectionRecordSet : CourseSectionRecordSet ,out savePoint : SequenceIdentifier ): StatusInfo
+updateCourseSection (sourcedId : GUID ,courseSectionRecord : Section ): StatusInfo
+updateCourseSectionStatus (sourcedId : GUID ,status : Status ): StatusInfo
+replaceCourseSection (sourcedId : GUID ,courseSectionRecord : Section ): StatusInfo
+discoverCourseSectionIds (queryObject : QueryObject ,out sourcedIdSet : GUIDSet ): StatusInfo
+changeCourseSectionIdentifier (sourcedId : GUID ,newSourcedId : GUID ): StatusInfo
<< interface >>
CourseTemplateManager
<< interface >>
CourseOfferingManager
<< interface >>
SectionAssociationManager

**Figure 3.3 CourseManagementService interface definition.**

**Table 3.32 Summary of operations for CourseSectionManager.**

| Operation | Description |
|---|---|
| createCourseSection | To request the creation of a populated CourseSection object on the target system where the source is responsible for the allocation of the unique identifier. |
| createByProxyCourseSection | To request the creation of a populated CourseSection object on the target system where the target is responsible for the allocation of the unique identifier. |
| createCourseSectionFromCourseSection | To create a new CourseSection from the supplied CourseSection for a particular academic session. |
| deleteCourseSection | To request the deletion of a CourseSection object. The CourseSection object is deleted along with all of its associated Memberships. |
| readCourseSection | To read the full contents of the identified CourseSection object. The target must return all of the data it has for the identified CourseSection object. |
| readAllCourseSectionIds | To obtain the set of sourcedIds which have been assigned to CourseSection objects. |
| readCourseSectionIdsFromSavePoint | To obtain the set of sourcedIds for CourseSection objects which have been altered since the supplied reference point. The reference point is set as ‘zero’ at creation and incremented after every write operation. |
| readCourseSections | To obtain the CourseSection objects for a defined set of identifiers. This results in a single transaction that may require the exchange of a large volume of data in the response message. |
| readCourseSectionsFromSavePoint | To obtain the set of CourseSection objects which have been altered since the requested reference point. The reference point is set as ‘zero’ at creation and incremented after every write operation. This results in a single transaction that may require the exchange of a large volume of data in the response message. |
| replaceCourseSection | To replace the content of the identified CourseSection object. The target must write the new data into the CourseSection object. This is a destructive write-over of all of the original information. In the case of the object not existing, this operation acts as an implied ‘createCourseSection’. |
| updateCourseSection | To write new content into the identified CourseSection object. The target must write the new data into the CourseSection object. This is an additive operation. |
| updateCourseSectionStatus | To change the status of the identified CourseSection. |
| discoverCourseSectionIds | To obtain the set of SourcedIds for CourseSection objects whose properties agree with those defined in the query/filter. |
| changeCourseSectionIdentifier | To change the SourcedId of the CourseSection record. The completion of this operation will result in later actions using the original SourcedId reporting an unknown identifier status. |

Note: In most cases the above operations act on a single instance of a CourseSection object i.e., ‘createCourseSection’, ‘createByProxyCourseSection’, ‘deleteCourseSection’, ‘readCourseSection’, ‘replaceCourseSection’ and ‘updateCourseSection’.

### 3.4.1 CreateCourseSection() Operation

| | |
|---|---|
| **Name:** | createCourseSection |
| **Return Function Parameter:** | *StatusInfo* – the status of the creation request. The permitted status codes are defined in Tables 3.33 and A.2. |
| **Supplied (in) Parameters:** | *sourcedId:GUID* – the sourcedId allocated by the source system. This is the identifier that must also be assigned within the target system.<br>*courseSectionRecord:CourseSectionRecord* – the CourseSection data that is to be stored in the new object. |
| **Returned (out) Parameters:** | None. |
| **Behavior:** | When the source issues the 'createCourseSection' request the target is instructed to create the populated CourseSection object and to allocate that structure the SourcedId passed by the source. If the supplied SourcedId has already been allocated to another object then the request is rejected and the appropriate failure code is returned. The reference point identifier is set to zero for the 'person' object in both the source and target. |
| **Notes:** | This request contains the initial content for the CourseSection record. More content can be added/replaced using the 'updateCourseSection' and/or 'replaceCourseSection' requests. |

**Table 3.33 Status codes for the 'createCourseSection' operation.**

| Status Code | Explanation of the Cause of the Code |
|---|---|
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor=fullsuccess' | The creation request has been fully and successfully implemented by the target system and the CourseSection object has been created with the identifier supplied by the source. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=idallocinusefail' | The target could not allocate the required unique SourcedId to the CourseSection object as it is already in use. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=overflowfail' | The target could not create the CourseSection object due to lack of target allocation memory. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=invaliddata' | Part or all of the supplied data was detected as invalid by the target system. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=incompletedata' | Some mandatory part of the data has been detected as missing by the target system. |
| 'CodeMajor=Success'<br>'Severity=Warning'<br>'CodeMinor=partialdatastorage' | The target has stored a subset of the sent data record i.e., some of the optional data has not been stored (all mandatory data has been supplied and stored). |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=unknownmdvocabulary' | The target system could not identify the defined metadata vocabulary term. This may be due to an incorrect term or a missing vocabulary file. |
| 'CodeMajor=Failure' | The target system could not identify the defined vocabulary term. This |

| Status Code | Explanation of the Cause of the Code |
|---|---|
| ‘Severity=Status’ ‘CodeMinor=unknownvocabulary’ | may be due to an incorrect term or a missing vocabulary file. |
| ‘CodeMajor=Failure’ ‘Severity=Status’ ‘CodeMinor=unknownextension’ | The target cannot process and store the proprietary data model extensions used in the object. |

### 3.4.2 CreateByProxyCourseSection() Operation

| | |
|---|---|
| **Name:** | createByProxyCourseSection |
| **Return Function Parameter:** | *StatusInfo* – the status of the creation request. The permitted status codes are defined in Tables 3.34 and A.2. |
| **Supplied (in) Parameters:** | *courseSectionRecord:CourseSectionRecord* – the CourseSection data that is to be stored in the new object. |
| **Returned (out) Parameters:** | *sourcedId:GUID* – the identifier allocated by the target to the newly created CourseSection object. |
| **Behavior:** | When the source issues the 'createByProxyCourseSection' request the target is instructed to create the populated CourseSection object and to allocate that record a unique SourcedId. The reference point identifier is set to zero for the CourseSection object in both the source and target. |
| **Notes:** | This request contains the initial content for the CourseSection object. More content can be added/replaced using the 'updateCourseSection' and/or 'replaceCourseSection' requests. |

**Table 3.34 Status codes for the 'createByProxyCourseSection' operation.**

| Status Code | Explanation of the Cause of the Code |
|---|---|
| 'CodeMajor=Success' 'Severity=Status' 'CodeMinor=fullsuccess' | The creation request has been fully and successfully implemented by the target system and the CourseSection object has been created with an identifier supplied by the target. |
| 'CodeMajor=Failure' 'Severity=Status' 'CodeMinor=idallocfail' | The target could not allocate a unique 'identifier' to the CourseSection object because there are no more spare identifiers available. |
| 'CodeMajor=Failure' 'Severity=Status' 'CodeMinor=overflowfail' | The target could not create the CourseSection object due to lack of target allocation memory. |
| 'CodeMajor=Failure' 'Severity=Status' 'CodeMinor=invaliddata' | Part or all of the supplied data was detected as invalid by the source system. |
| 'CodeMajor=Failure' 'Severity=Status' 'CodeMinor=incompletedata' | Some mandatory part of the data has been detected as missing by the target system. |
| 'CodeMajor=Success' 'Severity=Warning' 'CodeMinor=partialdatastorage' | The target has stored a subset of the sent data record i.e., some of the optional data has not been stored (all mandatory data has been supplied and stored). |
| 'CodeMajor=Failure' 'Severity=Status' 'CodeMinor=unknownmdvocabulary' | The target system could not identify the defined metadata vocabulary term. This may be due to an incorrect term or a missing vocabulary file. |
| 'CodeMajor=Failure' 'Severity=Status' 'CodeMinor=unknownvocabulary' | The target system could not identify the defined vocabulary term. This may be due to an incorrect term or a missing vocabulary file. |
| 'CodeMajor=Failure' | The target cannot process and store the proprietary data model extensions |

| Status Code | Explanation of the Cause of the Code |
|---|---|
| ‘Severity=Status’ ‘CodeMinor=unknownextension’ | used in the object. |

### 3.4.3 CreateCourseSectionFromCourseSection() Operation

| | |
|---|---|
| **Name:** | createCourseSectionFromCourseSection |
| **Return Function Parameter:** | *StatusInfo* – the status of the creation request. The permitted status codes are defined in Tables 3.35 and A.2. |
| **Supplied (in) Parameters:** | *sourcedId:GUID* – the SourcedId for the CourseSection to be cloned.<br>*academicSession:AcademicSession* – the academic session (from a pre-defined vocabulary) of the new CourseSection object.<br>*newSourcedId:GUID* – the sourcedId to be allocated to the newly created CourseSection object. |
| **Returned (out) Parameters:** | None. |
| **Behavior:** | When the source issues the 'createCourseSectionFromCourseSection' request the target is instructed to find the source CourseSection object and to create an equivalent CourseSection but with the new SourcedId and academic session. The reference point identifier is set to zero for the CourseSection object in both the source and target. |
| **Notes:** | An invalid status code will be returned if the new SourcedId has already been allocated or the academic session is incorrect. |

**Table 3.35 Status codes for the 'createCourseSectionFromCourseSection' operation.**

| Status Code | Explanation of the Cause of the Code |
|---|---|
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor=fullsuccess' | The creation request has been fully and successfully implemented by the target system and the new CourseSection object has been created with a unique identifier. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=unknownobject' | The CourseSection object identifier is unknown in the target system. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=idallocinusefail' | The target could not allocate the required unique 'identifier' to the new CourseSection object as it is already in use. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=overflowfail' | The target could not create the CourseSection object due to lack of target allocation memory. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=invaliddata' | An invalid academic session value has been received by the target system. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=incompletedata' | Some mandatory part of the data has been detected as missing by the target system. |

### 3.4.4 DeleteCourseSection() Operation

| | |
|---|---|
| **Name:** | deleteCourseSection |
| **Return Function Parameter:** | *StatusInfo* – the status of the delete request. The permitted status codes are defined in Tables 3.36 and A.2. |
| **Supplied (in) Parameters:** | *sourcedId:GUID* – the identifier to be used by the target to identify the CourseSection object. |
| **Returned (out) Parameters:** | None. |
| **Behavior:** | When the source issues the 'deleteCourseSection' request the target is instructed to delete the identified CourseSection object and to remove the reference to the CourseSection from any of the related CourseOffering and Membership objects. If the object identified by the supplied SourcedId cannot be located then the request is rejected and the appropriate failure code is returned. |
| **Notes:** | Deletion of the CourseSection object does not necessarily result in the destruction of the data within the server. The true state of the data in the target is unknown. |

**Table 3.36 Status codes for the 'deleteCourseSection' operation.**

| Status Code | Explanation of the Cause of the Code |
|---|---|
| 'CodeMajor=Success' 'Severity=Status' 'CodeMinor=fullsuccess' | The deletion request has been fully and successfully implemented by the target system and the CourseSection object has been deleted. The corresponding Membership records have also been deleted. |
| 'CodeMajor=Failure' 'Severity=Status' 'CodeMinor=unknownobject' | The CourseSection object identifier is unknown in the target system and so the object could not be deleted. |
| 'CodeMajor=Failure' 'Severity=Status' 'CodeMinor= deletefailure' | The target system has not been able to delete the identified CourseSection object. |

### 3.4.5 ReadCourseSection() Operation

| | |
|---|---|
| **Name:** | readCourseSection |
| **Return Function Parameter:** | *StatusInfo* – the status of the read request. The permitted status codes are defined in Tables 3.37 and A.2. |
| **Supplied (in) Parameters:** | *sourcedId:GUID* – the identifier of the CourseSection object to be read. |
| **Returned (out) Parameters:** | *courseSectionRecord:CourseSectionRecord* – the CourseSection data that is read from the object. |
| **Behavior:** | When the source issues the 'readCourseSection' request the target is charged with retrieving the identified object from its database and returning this data to the source. The target is responsible for ensuring that the object contains valid data. If the object identified by the supplied SourcedId cannot be located then the request is rejected and the appropriate failure code is returned. |
| **Notes:** | The returned CourseSection object can only be trusted if the corresponding status code is 'success'. |

**Table 3.37 Status codes for the 'readCourseSection' operation.**

| Status Code | Explanation of the Cause of the Code |
|---|---|
| 'CodeMajor=Success' 'Severity=Status' 'CodeMinor=fullsuccess' | The read request has been fully and successfully implemented by the target system and the identified CourseSection object has been read from the target system. |
| 'CodeMajor=Failure' 'Severity=Status' 'CodeMinor=unknownobject' | The CourseSection object identifier is unknown in the target system and so the object could not be read. |
| 'CodeMajor=Failure' 'Severity=Status' 'CodeMinor=targetreadfailure' | The target system has detected an error in the stored CourseSection object and so cannot return the data. |
| 'CodeMajor=Failure' 'Severity=Status' 'CodeMinor=invaliddata' | Part or all of the returned data was detected as invalid by the source system. |
| 'CodeMajor=Failure' 'Severity=Status' 'CodeMinor=incompletedata' | Some mandatory part of the data has been detected as missing by the source system. |
| 'CodeMajor=Success' 'Severity=Warning' 'CodeMinor=partialdatastorage' | The target has only returned a subset of the data expected by the source e.g., only the mandatory parts. |
| 'CodeMajor=Failure' 'Severity=Status' 'CodeMinor=unknownvocabulary' | The source system could not identify the defined vocabulary term. This may be due to an incorrect term or a missing vocabulary file. |
| 'CodeMajor=Failure' 'Severity=Status' 'CodeMinor=unknownextension' | The source cannot process and store the proprietary data model extensions used in the object. |

### 3.4.6 ReadAllCourseSectionIds() Operation

| | |
|---|---|
| **Name:** | readAllCourseSectionIds |
| **Return Function Parameter:** | *StatusInfo* – the status of the read request. The permitted status codes are defined in Tables 3.38 and A.2. |
| **Supplied (in) Parameters:** | None. |
| **Returned (out) Parameters:** | *sourcedIdSet:GUIDSet* – the set of SourcedIds for all of the CourseSection objects in the target's CourseSection database. |
| **Behavior:** | When the source issues the 'readAllCourseSectionIds' request the target is charged with returning the SourcedIds of all of the CourseSection objects in the database. |
| **Notes:** | The returned set of SourcedIds may or may not already be identified as allocated in the CourseSections database in the source system. If no SourcedIds have been allocated then the returned data set is empty and the success status code returned. |

**Table 3.38 Status codes for the 'readAllCourseSectionIds' operation.**

| Status Code | Explanation of the Cause of the Code |
|---|---|
| 'CodeMajor=Success' 'Severity=Status' 'CodeMinor=fullsuccess' | The read request has been fully and successfully implemented by the target system and the identified CourseSection object identifiers have been read from the target system. |
| 'CodeMajor=Success' 'Severity=Status' 'CodeMinor= nosourcedids' | The read request has been fully and successfully implemented by the target system and no CourseSection object identifiers were found. |
| 'CodeMajor=Failure' 'Severity=Status' 'CodeMinor=toomuchdata' | The requested data cannot be returned because it would exceed some physical system constraint e.g., permitted size of SOAP message. |
| 'CodeMajor=Failure' 'Severity=Status' 'CodeMinor=invaliddata' | Part or all of the returned data was detected as invalid by the source system. |

### 3.4.7 ReadCourseSectionIdsFromSavePoint() Operation

| | |
|---|---|
| **Name:** | readCourseSectionIdsFromSavePoint |
| **Return Function Parameter:** | *StatusInfo* – the status of the read request. The permitted status codes are defined in Tables 3.39 and A.2. |
| **Supplied (in) Parameters:** | *fromSavePoint:SequenceIdentifier* – the reference point from which all of the changed identifier actions are to be read. This is the value in the source system. |
| **Returned (out) Parameters:** | *sourcedIdSet:GUIDSet* – the set of SourcedIds of the CourseSection objects stored on the target.<br>*savePoint:SequenceIdentifier* – the value of the reference point counter in the target system. |
| **Behavior:** | When the source issues the 'readCourseSectionIdsFromSavePoint' the target returns the set of SourcedIds that have been altered from the defined reference point.<br>If the reference counter in the source is greater than that in the target then an empty set is returned for the SourcedIds and the target value for the reference point is returned. |
| **Notes:** | If no SourcedIds have been allocated then the returned data set is empty and the success status code returned. |

**Table 3.39 Status codes for the 'readCourseSectionIdsFromSavePoint' operation.**

| Status Code | Explanation of the Cause of the Code |
|---|---|
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor=fullsuccess' | The read request has been fully and successfully implemented by the target system and the identified CourseSection objects have been read from the target system. |
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor= nosourcedids' | The read request has been fully and successfully implemented by the target system and no CourseSection object identifiers were found. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=toomuchdata' | The requested data cannot be returned because it would exceed some physical system constraint e.g., permitted size of SOAP message. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=invaliddata' | Part or all of the returned data was detected as invalid by the source system. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=savepointerror' | An error has occurred in the processing of the save-point identifier information making it impossible to read the correct objects from the database. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=savepointsyncerror' | The value of the save point reference from the source was later than that of the target system. No identifiers have been returned. The target system savepoint value is returned to the source system for information. |

### 3.4.8 ReadCourseSections() Operation

| | |
|---|---|
| **Name:** | readCourseSections |
| **Return Function Parameter:** | *StatusInfo* – the status of the read request. The permitted status codes are defined in Tables 3.40 and A.2. |
| **Supplied (in) Parameters:** | *sourcedIdSet:GUIDSet* – the set of identifiers of the CourseSection objects to be read. |
| **Returned (out) Parameters:** | *courseSectionRecordSet:CourseSectionRecordSet* – the set of course section records.<br><br>*savePoint:SequenceIdentifier* – the value of the reference point counter in the target system. |
| **Behavior:** | When the source issues the 'readCourseSections' request the target is charged with retrieving the identified set of objects from its database and placing this information in a persistent data file. The associated read savePoint reference is updated and returned.<br><br>If the object identified by the supplied SourcedId cannot be located then the request is rejected and a partial success code is returned for the operation. The target is responsible for ensuring that the records contain valid data. The target should attempt to successfully complete as much of the request as possible. |
| **Notes:** | A returned CourseSection object is only present if the object has been located in the target system and the full data set returned.<br><br>The enclosed data may result in a long response message. |

**Table 3.40 Status codes for the 'readCourseSections' operation.**

| Status Code | Explanation of the Cause of the Code |
|---|---|
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor=fullsuccess' | The read request has been fully and successfully implemented by the target system and the identified CourseSection objects have been read from the target system. |
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor=partialreadfail' | Some of the CourseSection object identifiers are unknown in the target system and so those objects could not be read. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=invaliddata' | Part or all of the returned data was detected as invalid by the source system. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=unknownvocabulary' | The source system could not identify the defined vocabulary term. This may be due to an incorrect term or a missing vocabulary file. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=toomuchdata' | The data cannot be returned because it would exceed some physical system constraint e.g., permitted size of SOAP message. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=targetreadfailure' | The target system has detected an error in the stored CourseTemplate objects and so cannot return the data. |

### 3.4.9 ReadCourseSectionsFromSavePoint() Operation

| | |
|---|---|
| **Name:** | readCourseSectionsFromSavePoint |
| **Return Function Parameter:** | *StatusInfo* – the status of the read request. The permitted status codes are defined in Tables 3.41 and A.2. |
| **Supplied (in) Parameters:** | *fromSavePoint:SequenceIdentifier* – the reference point from which all of the changed identifier actions are to be read. This is the value in the source system. |
| **Returned (out) Parameters:** | *courseSectionRecordSet:CourseSectionRecordSet* – the set of course section records.<br>*savePoint:SequenceIdentifier* – the value of the reference point counter in the target system. |
| **Behavior:** | When the source issues the 'readCourseSectionsFromSavePoint' request the target is charged with reading the objects that have been altered from the defined reference point.<br>If the reference counter in the source is greater than that in the target then an empty data file is returned and the target value for the reference point is returned. |
| **Notes:** | If no objects have been allocated then the return message will be empty.<br>The enclosed data may result in a long response message. |

**Table 3.41 Status codes for the 'readCourseSectionsFromSavePoint' operation.**

| Status Code | Explanation of the Cause of the Code |
|---|---|
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor=fullsuccess' | The read request has been fully and successfully implemented by the target system and copies of the identified CourseSection objects have been placed in the data file. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=savepointerror' | An error has occurred in the processing of the save-point identifier information making it impossible to read the correct objects from the database. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=savepointsyncerror' | The value of the save point reference from the source was later than that of the target system. No identifiers have been returned. The target system savepoint value is returned to the source system for information. |
| 'CodeMajor=Success'<br>'Severity=Warning'<br>'CodeMinor=partialdatareturn' | The target has only returned a subset of the data object e.g., only the mandatory parts. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=toomuchdata' | The data cannot be returned because it would exceed some physical system constraint e.g., permitted size of SOAP message. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=targetreadfailure' | The target system has detected an error in the stored LineItem object and so cannot return the data. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=invaliddata' | Part or all of the returned data was detected as invalid by the target system. |

### 3.4.10 ReplaceCourseSection() Operation

| | |
|---|---|
| **Name:** | replaceCourseSection |
| **Return Function Parameter:** | *StatusInfo* – the status of the replace request. The permitted status codes are defined in Tables 3.42 and A.2. |
| **Supplied (in) Parameters:** | *sourcedId:GUID* – the identifier of the CourseSection object to be replaced.<br>*courseSectionRecord:CourseSectionRecord* – the CourseSection data that is to be stored in the object. |
| **Returned (out) Parameters:** | None. |
| **Behavior:** | When the source issues the 'replaceCourseSection' request the target is charged with writing the supplied information into the identified object. If any part of the write fails e.g., due to partial invalid data then the whole request is rejected and the object is left in its original state. This is a destructive write-over operation of the entire CourseSection object. This is equivalent to a 'createCourseSection' but for an object that already exists.<br>If the object identified by the supplied SourcedId cannot be located then the request is interpreted as a 'createCourseSection' invocation.<br>The reference counter for the object is incremented by one in the target system. |
| **Notes:** | The source is responsible for determining the reason of the failure. |

**Table 3.42 Status codes for the 'replaceCourseSection' operation.**

| Status Code | Explanation of the Cause of the Code |
|---|---|
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor=fullsuccess' | The replace request has been fully and successfully implemented by the target system and the identified CourseSection object has been changed on the target system. |
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor=createsuccess' | The CourseTemplate object identifier is unknown in the target system and so a new object has been successfully created instead. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=invaliddata' | Part or all of the returned data was detected as invalid by the target system. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=incompletedata' | Some mandatory part of the data has been detected as missing by the target system. |
| 'CodeMajor=Success'<br>'Severity=Warning'<br>'CodeMinor=partialdatastorage' | The target has stored a subset of the sent data record i.e., some of the optional data has not been stored (all mandatory data has been supplied and stored). |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=unknownvocabulary' | The target system could not identify the defined vocabulary term. This may be due to an incorrect term or a missing vocabulary file. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=unknownmdvocabulary' | The target system could not identify the defined metadata vocabulary term. This may be due to an incorrect term or a missing vocabulary file. |

| Status Code | Explanation of the Cause of the Code |
|---|---|
| ‘CodeMajor=Failure’<br>‘Severity=Status’<br>‘CodeMinor=unknownextension’ | The target cannot process the proprietary data model extensions used in the object. |

### 3.4.11 UpdateCourseSection() Operation

| | |
|---|---|
| **Name:** | updateCourseSection |
| **Return Function Parameter:** | *StatusInfo* – the status of the update request. The permitted status codes are defined in Tables 3.43 and A.2. |
| **Supplied (in) Parameters:** | *sourcedId:GUID* – the identifier of the CourseSection object to be updated.<br>*courseSectionRecord:CourseSectionRecord* – the CourseSection data that is to be stored in the object. |
| **Returned (out) Parameters:** | None. |
| **Behavior:** | When the source issues the 'updateCourseSection' request the target is charged with writing the supplied information into the identified object. If any part of the write fails e.g., due to partial invalid data then the whole request is rejected and the object is left in its original state. This is an additive write operation of all the data fields supplied in the update request and fields not supplied remain unchanged. If a field is constrained with a multiplicity of one then the 'updateCourseSection' request acts as a 'replaceCourseSection' request for that field.<br>If the object identified by the supplied SourcedId cannot be located then the request is rejected and the appropriate failure code is returned.<br>The reference counter for the object is incremented in the target system. |
| **Notes:** | The source is responsible for determining the reason of the failure. |

**Table 3.43 Status codes for the 'updateCourseSection' operation.**

| Status Code | Explanation of the Cause of the Code |
|---|---|
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor=fullsuccess' | The update request has been fully and successfully implemented by the target system and the identified CourseSection object has been changed on the target system. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=unknownobject' | The CourseSection object identifier is unknown in the target system and so the object could not be updated. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=invaliddata' | Part or all of the returned data was detected as invalid by the target system. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=incompletedata' | Some mandatory part of the data has been detected as missing by the target system. |
| 'CodeMajor=Success'<br>'Severity=Warning'<br>'CodeMinor=partialdatastorage' | The target has stored a subset of the sent data record i.e., some of the optional data has not been stored (all mandatory data has been supplied and stored). |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=unknownvocabulary' | The target system could not identify the defined vocabulary term. This may be due to an incorrect term or a missing vocabulary file. |
| 'CodeMajor=Failure' | The target system could not identify the defined metadata vocabulary |

| | |
|---|---|
| ‘Severity=Status’ ‘CodeMinor=unknownmdvocabulary’ | term.  This may be due to an incorrect term or a missing vocabulary file. |
| ‘CodeMajor=Failure’ ‘Severity=Status’ ‘CodeMinor=unknownextension’ | The target cannot process the proprietary data model extensions used in the object. |

### 3.4.12 UpdateCourseSectionStatus() Operation

| | |
|---|---|
| **Name:** | updateCourseSectionStatus |
| **Return Function Parameter:** | *StatusInfo* – the status of the update request. The permitted status codes are defined in Tables 3.44 and A.2. |
| **Supplied (in) Parameters:** | *sourcedId:GUID* – the identifier of the CourseSection object whose status is to be updated.<br>*status:Status* – the new Status (from an agreed vocabulary) that is to be assigned to the CourseSection object. |
| **Returned (out) Parameters:** | None. |
| **Behavior:** | When the source issues the 'updateCourseSectionStatus' request the target is charged with changing the status of the associated object. This is a destructive write i.e., the previous status is replaced.<br>If the object identified by the supplied SourcedId cannot be located then the request is rejected and the appropriate failure code is returned.<br>The reference counter for the object is incremented by one in the target system. |
| **Notes:** | None. |

**Table 3.44 Status codes for the 'updateCourseSectionStatus' operation.**

| Status Code | Explanation of the Cause of the Code |
|---|---|
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor=fullsuccess' | The update status request has been fully and successfully implemented by the target system and the status of the identified CourseSection object has been changed on the target system. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=unknownobject' | The CourseSection object identifier is unknown in the target system and so the object status could not be updated. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=invaliddata' | The target system has received and invalid status value. |

### 3.4.13 DiscoverCourseSectionIds() Operation

| | |
|---|---|
| **Name:** | discoverCourseSectionIds |
| **Return Function Parameter:** | *StatusInfo* – the status of the discover request. The permitted status codes are defined in Tables 3.45 and A.2. |
| **Supplied (in) Parameters:** | *queryObject:QueryObject* – this is the query/filter instruction that is to be applied by the target to discover the corresponding CourseSection objects. |
| **Returned (out) Parameters:** | *sourcedIdSet:GUIDSet* – the set of identifiers of the CourseSection objects whose content conform to the query/filter conditions. |
| **Behavior:** | When the source issues the 'discoverCourseSectionIds' the target applies the query/filter instructions to the set of CourseSection objects and returns the set of SourcedIds that uphold the query/filter.<br>If no CourseSection objects have the required properties the returned data set is empty and the success status code returned.<br>If the target does not understand or cannot apply the requested query/filter then an error status is returned. |
| **Notes:** | The internal structure of this QueryObject is undefined (it is should be treated as a String encoded 'blob). Later versions of this specification will look at the established best practices for clarification on the use of this operation. |

**Table 3.45 Status codes for the 'discoverCourseSectionIds' operation.**

| Status Code | Explanation of the Cause of the Code |
|---|---|
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor=fullsuccess' | The discover request has been fully and successfully implemented by the target system and the appropriate CourseSection identifiers have been read from the target system. |
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor= nosourcedids' | The discover request has been fully and successfully implemented by the target system and no CourseSection object identifiers were found. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=unknownquery' | The target system cannot understand the query request that has been received i.e., the query/filter language is unknown. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=toomuchdata' | The data cannot be returned because it would exceed some physical system constraint e.g., permitted size of SOAP message. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=invaliddata' | Part or all of the returned data was detected as invalid by the source system. |

### 3.4.14 ChangeCourseSectionIdentifier() Operation

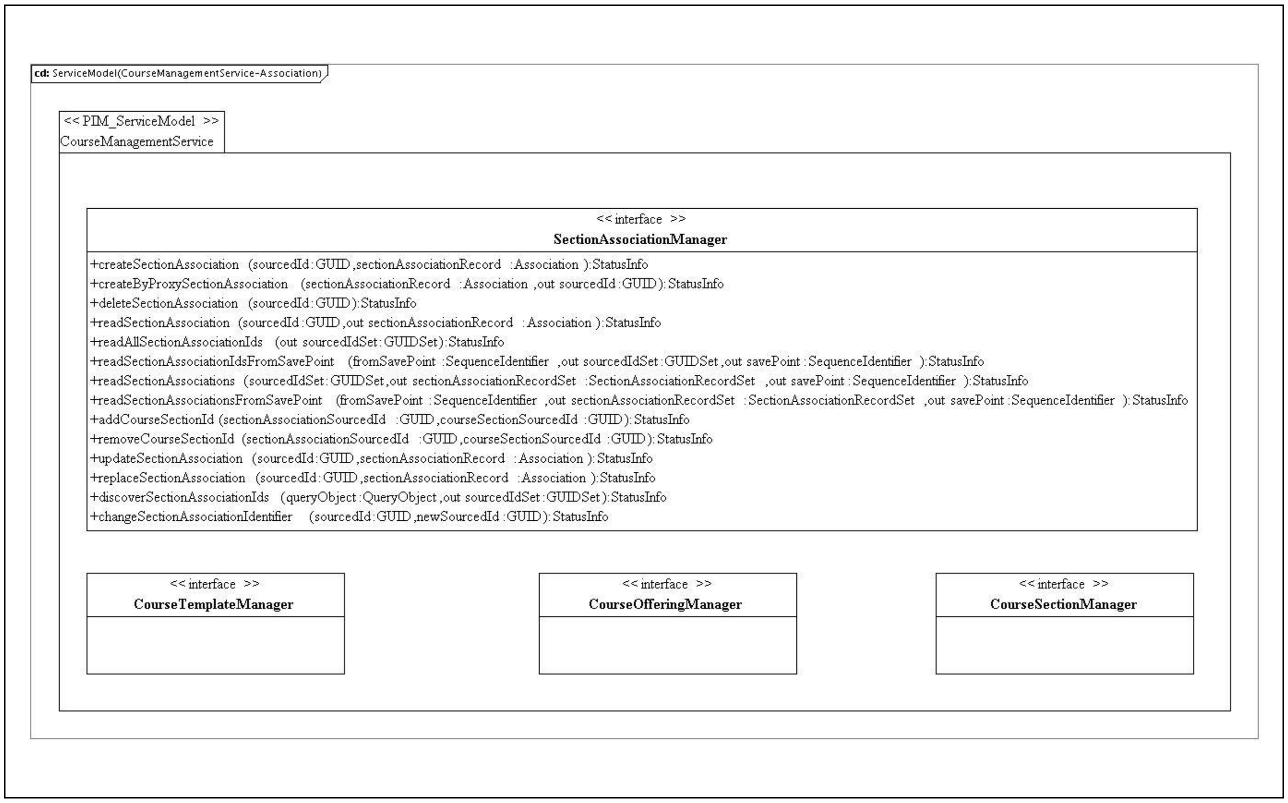
| | |
|---|---|
| **Name:** | changeCourseSectionIdentifier |
| **Return Function Parameter:** | *StatusInfo* – the status of the change identifier request. The permitted status codes are given in Tables 3.46 and A.2. |
| **Supplied (in) Parameters:** | *sourcedId:GUID* – the identifier of the CourseSection object to be changed.<br>*newSourcedId:GUID* – the new identifier to be allocated to the CourseSection object. |
| **Returned (out) Parameters:** | None. |
| **Behavior:** | When the source issues the 'changeCourseSectionIdentifier' request the target is charged with replacing the original SourcedId with the new supplied SourcedId. All membership entries must be similarly changed. All further references to the object must use the new SourcedId otherwise an 'unknown' object failure status code is returned.<br>If the object identified by the supplied SourcedId cannot be located then the request is rejected and the appropriate failure code is returned. |
| **Notes:** | The reference pointer value remains unchanged. |

**Table 3.46 Status codes for the 'changeCourseSectionIdentifier' operation.**

| Status Code | Explanation of the Cause of the Code |
|---|---|
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor=fullsuccess' | The change identifier request has been fully and successfully implemented by the target system and the CourseSection object SourcedId has been changed on the target system. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=idallocinusefail' | The target could not allocate the new unique 'identifier' to the CourseSection object as the identifier is already in use. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=unknownobject' | The current CourseSection SourcedId identifier is unknown in the target system and so the object identifier could not be changed. |

## 3.5 SectionAssociationManager Interface Description

The SectionAssociationManager interface class, as shown in Figure 3.4, describes the operations that are permitted on a SectionAssociation. The interface stereotype indicates that there are no attributes for this class. The set of operations are summarized in Table 3.47.



```
+createSectionAssociation (sourcedId:GUID,sectionAssociationRecord :Association ):StatusInfo
+createByProxySectionAssociation (sectionAssociationRecord :Association ,out sourcedId:GUID):StatusInfo
+deleteSectionAssociation (sourcedId:GUID):StatusInfo
+readSectionAssociation (sourcedId:GUID,out sectionAssociationRecord :Association ):StatusInfo
+readAllSectionAssociationIds (out sourcedIdSet:GUIDSet):StatusInfo
+readSectionAssociationIdsFromSavePoint (fromSavePoint :SequenceIdentifier ,out sourcedIdSet:GUIDSet,out savePoint:SequenceIdentifier ):StatusInfo
+readSectionAssociations (sourcedIdSet:GUIDSet,out sectionAssociationRecordSet :SectionAssociationRecordSet ,out savePoint:SequenceIdentifier ):StatusInfo
+readSectionAssociationsFromSavePoint (fromSavePoint :SequenceIdentifier ,out sectionAssociationRecordSet :SectionAssociationRecordSet ,out savePoint:SequenceIdentifier ):StatusInfo
+addCourseSectionId (sectionAssociationSourcedId :GUID,courseSectionSourcedId :GUID):StatusInfo
+removeCourseSectionId (sectionAssociationSourcedId :GUID,courseSectionSourcedId :GUID):StatusInfo
+updateSectionAssociation (sourcedId:GUID,sectionAssociationRecord :Association ):StatusInfo
+replaceSectionAssociation (sourcedId:GUID,sectionAssociationRecord :Association ):StatusInfo
+discoverSectionAssociationIds (queryObject:QueryObject,out sourcedIdSet:GUIDSet):StatusInfo
+changeSectionAssociationIdentifier (sourcedId:GUID,newSourcedId:GUID):StatusInfo
```

**Figure 3.4 CourseManagementService SectionAssociationManager interface definition.**

**Table 3.47 Summary of operations for SectionAssociationManager.**

| Operation | Description |
|---|---|
| createSectionAssociation | To request the creation of a populated SectionAssociation object on the target system where the source is responsible for the allocation of the unique identifier. |
| createByProxySectionAssociation | To request the creation of a populated SectionAssociation object on the target system where the target is responsible for the allocation of the unique identifier. |
| deleteSectionAssociation | To request the deletion of a SectionAssociation object. The SectionAssociation object is deleted and all of its associated relationships. |
| readSectionAssociation | To read the full contents of the identified SectionAssociation object. The target must return all of the data it has for the identified SectionAssociation object. |
| readAllSectionsAssociationIds | To obtain the set of SourcedIds which have been assigned to SectionAssociation objects. |
| readSectionAssociationIdsFromSavePoint | To obtain the set of SourcedIds for SectionAssociations objects which have been altered since the requested reference point. The reference point is set as ‘zero’ at creation and incremented after every write operation. |
| readCourseSectionAssociations | To obtain the SectionAssociation objects for a defined set of identifiers. This results in a single transaction that may require the exchange of a large volume of data in the response message. |
| readSectionAssociationsFromSavePoint | To obtain the set of SectionAssociation objects which have been altered since the requested reference point. The reference point is set as ‘zero’ at creation and incremented after every write operation. This results in a single transaction that may require the exchange of a large volume of data in the response message. |
| addCourseSectionId | To add a new CourseSection identifier to the SectionAssociation. |
| removeCourseSectionId | To remove a CourseSection identifier from the SectionAssociation. |
| replaceSectionAssociation | To replace the content of the identified SectionAssociation object. The target must write the new data into the SectionAssociation object. This is a destructive write-over of the original information. In the case of the object not existing, this operation acts as an implied ‘createSectionAssociation’. |
| updateSectionAssociation | To write new content into the identified SectionAssociation object. The target must write the new data into the SectionAssociation object. This is an additive operation. |
| discoverSectionAssociationIds | To obtain the set of sourcedIds for SectionAssociation objects whose properties agree with those defined in the query/filter. |
| changeSectionAssociationIdentifier | To change the sourcedId of the SectionAssociation record. The completion of this operation will result in later actions using the original sourcedId reporting an unknown identifier status. |

Note: In most cases the above operations act on a single instance of a SectionAssociation object i.e., ‘createSectionAssociation’, ‘createByProxySectionAssociation’, ‘deleteSectionAssociation’, ‘readSectionAssociation’, ‘replaceSectionAssociation’ and ‘updateSectionAssociation’.

### 3.5.1 CreateSectionAssociation() Operation

| | |
|---|---|
| **Name:** | createSectionAssociation |
| **Return Function Parameter:** | *StatusInfo* – the status of the creation request. The permitted status codes are defined in Tables 3.48 and A.2. |
| **Supplied (in) Parameters:** | *sourcedId:GUID* – the SourcedId allocated by the source system. This is the identifier that must also be assigned within the target system.<br>*sectionAssociationRecord:SectionAssociationRecord* – the SectionAssociation data that is to be stored in the new object. |
| **Returned (out) Parameters:** | None. |
| **Behavior:** | When the source issues the 'createSectionAssociation' request the target is instructed to create the populated SectionAssociation object and to allocate that structure the SourcedId passed by the source. If the supplied SourcedId has already been allocated to another object then the request is rejected and the appropriate failure code is returned. The reference point identifier is set to 'zero' for the SectionAssociation object in both the source and target. |
| **Notes:** | This request contains the initial content for the SectionAssociationRecord. More content can be added/replaced using the 'updateSectionAssociation' and/or 'replaceSectionAssociation' requests. |

**Table 3.48 Status codes for the 'createSectionAssociation' operation.**

| Status Code | Explanation of the Cause of the Code |
|---|---|
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor=fullsuccess' | The creation request has been fully and successfully implemented by the target system and the SectionAssociation object has been created with a unique identifier. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=idallocinusefail' | The target could not allocate the required unique 'identifier' to the SectionAssociation object as it is already in use. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=overflowfail' | The target could not create the SectionAssociation object due to lack of target allocation memory. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=invaliddata' | Part or all of the supplied data was detected as invalid by the target system. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=incompletedata' | Some mandatory part of the data has been detected as missing by the target system. |
| 'CodeMajor=Success'<br>'Severity=Warning'<br>'CodeMinor=partialdatastorage' | The target has stored a subset of the sent data record i.e., some of the optional data has not been stored (all mandatory data has been supplied and stored). |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=unknownmdvocabulary' | The target system could not identify the defined metadata vocabulary term. This may be due to an incorrect term or a missing vocabulary file. |
| 'CodeMajor=Failure' | The target system could not identify the defined vocabulary term. This |

| Status Code | Explanation of the Cause of the Code |
|---|---|
| 'Severity=Status'<br>'CodeMinor=unknownvocabulary' | may be due to an incorrect term or a missing vocabulary file. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=unknownextension' | The target cannot process and store the proprietary data model extensions used in the object. |

### 3.5.2 CreateByProxySectionAssociation() Operation

| | |
|---|---|
| **Name:** | createByProxySectionAssociation |
| **Return Function Parameter:** | *StatusInfo* – the status of the creation request. The permitted status codes are defined in Tables 3.49 and A.2. |
| **Supplied (in) Parameters:** | *sectionAssociationRecord:SectionAssociationRecord* – the SectionAssociation data that is to be stored in the new object. |
| **Returned (out) Parameters:** | *sourcedId:GUID* – the identifier allocated by the target to the newly created SectionAssociation object. |
| **Behavior:** | When the source issues the 'createByProxySectionAssociation' request the target is instructed to create the populated SectionAssociation object and to allocate that record a unique SourcedId. The reference point identifier is set to 'zero' for the SectionAssociation object in both the source and target. |
| **Notes:** | This request contains the initial content for the SectionAssociation object. More content can be added/replaced using the 'updateSectionAssociation' and/or 'replaceSectionAssociation' requests. |

**Table 3.49 Status codes for the 'createByProxySectionAssociation' operation.**

| Status Code | Explanation of the Cause of the Code |
|---|---|
| 'CodeMajor=Success' 'Severity=Status' 'CodeMinor=fullsuccess' | The creation request has been fully and successfully implemented by the target system and the SectionAssociation object has been created with the identifier supplied by the target. |
| 'CodeMajor=Failure' 'Severity=Status' 'CodeMinor=idallocfail' | The target could not allocate a unique SourcedId to the SectionAssociation object because there are no unused identifiers available. |
| 'CodeMajor=Failure' 'Severity=Status' 'CodeMinor=overflowfail' | The target could not create the SectionAssociation object due to lack of target allocation memory. |
| 'CodeMajor=Failure' 'Severity=Status' 'CodeMinor=invaliddata' | Part or all of the supplied data was detected as invalid by the source system. |
| 'CodeMajor=Failure' 'Severity=Status' 'CodeMinor=incompletedata' | Some mandatory part of the data has been detected as missing by the target system. |
| 'CodeMajor=Success' 'Severity=Warning' 'CodeMinor=partialdatastorage' | The target has stored a subset of the sent data record i.e., some of the optional data has not been stored (all mandatory data has been supplied and stored). |
| 'CodeMajor=Failure' 'Severity=Status' 'CodeMinor=unknownmdvocabulary' | The target system could not identify the defined metadata vocabulary term. This may be due to an incorrect term or a missing vocabulary file. |
| 'CodeMajor=Failure' 'Severity=Status' 'CodeMinor=unknownvocabulary' | The target system could not identify the defined vocabulary term. This may be due to an incorrect term or a missing vocabulary file. |
| 'CodeMajor=Failure' | The target cannot process and store the proprietary data model extensions |

| Status Code | Explanation of the Cause of the Code |
|---|---|
| ‘Severity=Status’ ‘CodeMinor=unknownextension’ | used in the object. |

### 3.5.3 DeleteSectionAssociation() Operation

| | |
|---|---|
| **Name:** | deleteSectionAssociation |
| **Return Function Parameter:** | *StatusInfo* – the status of the delete request. The permitted status codes are defined in Tables 3.50 and A.2. |
| **Supplied (in) Parameters:** | *sourcedId:GUID* – the identifier to be used by the target to identify the SectionAssociation object. |
| **Returned (out) Parameters:** | None. |
| **Behavior:** | When the source issues the 'deleteSectionAssociation' request the target is instructed to delete the identified SectionAssociation object and to remove the reference to the SectionAssociation from any of the related Membership objects. If the object identified by the supplied SourcedId cannot be located then the request is rejected and the appropriate failure code is returned. |
| **Notes:** | Deletion of the SectionAssociation object does not necessarily result in the destruction of the data within the server. The true state of the data in the target is unknown. |

**Table 3.50 Status codes for the 'deleteSectionAssociation' operation.**

| Status Code | Explanation of the Cause of the Code |
|---|---|
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor=fullsuccess' | The deletion request has been fully and successfully implemented by the target system and the SectionAssociation object has been deleted. The corresponding Membership records have also been deleted. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=unknownobject' | The SectionAssociation object identifier is unknown in the target system and so the object could not be deleted. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor= deletefailure' | The target system has not been able to delete the identified SectionAssociation object. |

### 3.5.4 ReadSectionAssociation() Operation

| | |
|---|---|
| **Name:** | readSectionAssociation |
| **Return Function Parameter:** | *StatusInfo* – the status of the read request. The permitted status codes are defined in Tables 3.51 and A.2. |
| **Supplied (in) Parameters:** | *sourcedId:GUID* – the identifier of the SectionAssociation object to be read. |
| **Returned (out) Parameters:** | *sectionAssociationRecord:SectionAssociationRecord* – the SectionAssociation data that is read from the object. |
| **Behavior:** | When the source issues the 'readSectionAssociation' request the target is charged with retrieving the identified record from its database and returning this data to the source. The target is responsible for ensuring that the object contains valid data. If the object identified by the supplied SourcedId cannot be located then the request is rejected and the appropriate failure code is returned. |
| **Notes:** | The returned SectionAssociation object can only be trusted if the corresponding status code is 'success'. |

**Table 3.51 Status codes for the 'readSectionAssociation' operation.**

| Status Code | Explanation of the Cause of the Code |
|---|---|
| 'CodeMajor=Success' 'Severity=Status' 'CodeMinor=fullsuccess' | The read request has been fully and successfully implemented by the target system and the identified SectionAssociation object has been read from the target system. |
| 'CodeMajor=Failure' 'Severity=Status' 'CodeMinor=unknownobject' | The SectionAssociation object identifier is unknown in the target system and so the object could not be read. |
| 'CodeMajor=Failure' 'Severity=Status' 'CodeMinor=targetreadfailure' | The target system has detected an error in the stored SectionAssociation object and so cannot return the data. |
| 'CodeMajor=Failure' 'Severity=Status' 'CodeMinor=invaliddata' | Part or all of the returned data was detected as invalid by the source system. |
| 'CodeMajor=Failure' 'Severity=Status' 'CodeMinor=incompletedata' | Some mandatory part of the data has been detected as missing by the source system. |
| 'CodeMajor=Success' 'Severity=Warning' 'CodeMinor=partialdatastorage' | The target has only returned a subset of the data expected by the source e.g., only the mandatory parts. |
| 'CodeMajor=Failure' 'Severity=Status' 'CodeMinor=unknownvocabulary' | The source system could not identify the defined vocabulary term. This may be due to an incorrect term or a missing vocabulary file. |
| 'CodeMajor=Failure' 'Severity=Status' 'CodeMinor=unknownextension' | The source cannot process and store the proprietary data model extensions used in the object. |

### 3.5.5 ReadAllSectionAssociationIds() Operation

| | |
|---|---|
| **Name:** | readAllSectionAssociationIds |
| **Return Function Parameter:** | *StatusInfo* – the status of the read request. The permitted status codes are defined in Tables 3.52 and A.2. |
| **Supplied (in) Parameters:** | None. |
| **Returned (out) Parameters:** | *sourcedIdSet:GUIDSet* – the set of SourcedIds for all of the SectionAssociation objects in the target's database. |
| **Behavior:** | When the source issues the 'readAllSectionAssociationIds' request the target is charged with returning the SourcedIds of all of the SectionAssociations in its database. |
| **Notes:** | The returned set of SourcedIds may or may not already be identified as allocated in the SectionAssociations database in the source system. If no SourcedIds have been allocated then the returned data set is empty and the success status code returned. |

**Table 3.52 Status codes for the 'readAllSectionAssociationIds' operation.**

| Status Code | Explanation of the Cause of the Code |
|---|---|
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor=fullsuccess' | The read request has been fully and successfully implemented by the target system and the corresponding SectionAssociation identifiers have been read from the target system. |
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor= nosourcedids' | The read request has been fully and successfully implemented by the target system and no SectionAssociation object identifiers were found. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=toomuchdata' | The requested data cannot be returned because it would exceed some physical system constraint e.g., permitted size of SOAP message. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=invaliddata' | Part or all of the returned data was detected as invalid by the source system. |

### 3.5.6 ReadSectionAssociationIdsFromSavePoint() Operation

| | |
|---|---|
| **Name:** | readSectionAssociationIdsFromSavePoint |
| **Return Function Parameter:** | *StatusInfo* – the status of the creation request. The permitted status codes are defined in Tables 3.53 and A.2. |
| **Supplied (in) Parameters:** | *fromSavePoint:SequenceIdentifier* – the reference point from which all of the changed identifier actions are to be read. This is the value in the source system. |
| **Returned (out) Parameters:** | *sourcedIdSet:GUIDSet* – the set of SourcedIds of the SectionAssociation objects stored on the target.<br>*savePoint:SequenceIdentifier* – the value of the reference point counter in the target system. |
| **Behavior:** | When the source issues the 'readSectionAssociationIdsFromSavePoint' the target returns the set of SourcedIds that have been altered after the defined reference point.<br>If the reference counter in the source is greater than that in the target then an error code and the target value for the reference point are returned. |
| **Notes:** | If no SourcedIds have been allocated then the returned data set is empty and the success status code returned. |

**Table 3.53 Status codes for the 'readSectionAssociationIdsFromSavePoint' operation.**

| Status Code | Explanation of the Cause of the Code |
|---|---|
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor=fullsuccess' | The read request has been fully and successfully implemented by the target system and the identified SectionAssociation object has been read from the target system. |
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor= nosourcedids' | The read request has been fully and successfully implemented by the target system and no SectionAssociation object identifiers were found. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=toomuchdata' | The data cannot be returned because it would exceed some physical system constraint e.g., permitted size of SOAP message. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=invaliddata' | Part or all of the returned data was detected as invalid by the source system. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=savepointerror' | An error has occurred in the processing of the save-point identifier information making it impossible to read the correct objects from the database. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=savepointsyncerror' | The value of the save point reference from the source was later than that of the target system. No identifiers have been returned. The target system savepoint value is returned to the source system for information. |

### 3.5.7 ReadSectionAssociations() Operation

| | |
|---|---|
| **Name:** | readSectionAssociations |
| **Return Function Parameter:** | *StatusInfo* – the status of the read request. The permitted status codes are defined in Tables 3.54 and A.2. |
| **Supplied (in) Parameters:** | *sourcedIdSet:GUIDSet* – the set of identifiers of the SectionAssociation objects to be read. |
| **Returned (out) Parameters:** | *sectionAssociationRecordSet:SectionAssociationRecordSet* – the set of section association records.<br><br>*savePoint:SequenceIdentifier* – the value of the reference point counter in the target system. |
| **Behavior:** | When the source issues the 'readSectionAssociations' request the target is charged with retrieving the identified set of objects from its database and placing this information in a persistent data file. The associated read savePoint reference is updated and returned.<br><br>If the object identified by the supplied SourcedId cannot be located then the request is rejected and a partial success code is returned for the operation. The target is responsible for ensuring that the records contain valid data. The target should attempt to successfully complete as much of the request as possible. |
| **Notes:** | A returned SectionAssociation object is only present if the object has been located in the target system and the full data set returned.<br><br>The enclosed data may result in a long response message. |

**Table 3.54 Status codes for the 'readSectionAssociations' operation.**

| Status Code | Explanation of the Cause of the Code |
|---|---|
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor=fullsuccess' | The read request has been fully and successfully implemented by the target system and copies of the identified SectionAssociation objects have been placed in the data file. |
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor=partialreadfail' | Some of the SectionAssociation object identifiers are unknown in the target system and so those objects could not be read. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=invaliddata' | Part or all of the returned data was detected as invalid by the source system. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=unknownvocabulary' | The source system could not identify the defined vocabulary term. This may be due to an incorrect term or a missing vocabulary file. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=toomuchdata' | The data cannot be returned because it would exceed some physical system constraint e.g., permitted size of SOAP message. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=targetreadfailure' | The target system has detected an error in the stored CourseTemplate objects and so cannot return the data. |

### 3.5.8 ReadSectionAssociationsFromSavePoint() Operation

| | |
|---|---|
| **Name:** | readSectionAssociationsFromSavePoint |
| **Return Function Parameter:** | *StatusInfo* – the status of the read request. The permitted status codes are defined in Tables 3.55 and A.2. |
| **Supplied (in) Parameters:** | *fromSavePoint:SequenceIdentifier* – the reference point from which all of the changed identifier actions are to be read. This is the value in the source system. |
| **Returned (out) Parameters:** | *sectionAssociationRecordSet:SectionAssociationRecordSet* – the set of section association records.<br>*savePoint:SequenceIdentifier* – the value of the reference point counter in the target system. |
| **Behavior:** | When the source issues the 'readSectionAssociationsFromSavePoint' request the target is charged with reading the objects that have been altered from the defined reference point.<br>If the reference counter in the source is greater than that in the target then an error code and the target value for the reference point is returned. |
| **Notes:** | If no objects have been allocated then the return message will be empty.<br>The enclosed data may result in a long response message. |

**Table 3.55 Status codes for the 'readSectionAssociationsFromSavePoint' operation.**

| Status Code | Explanation of the Cause of the Code |
|---|---|
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor=fullsuccess' | The read request has been fully and successfully implemented by the target system and copies of the identified SectionAssociation objects have been stored in the data file. |
| 'CodeMajor=Success'<br>'Severity=Warning'<br>'CodeMinor=partialdatareturn' | The target has only returned a subset of the data object e.g., only the mandatory parts. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=toomuchdata' | The data cannot be returned because it would exceed some physical system constraint e.g., permitted size of SOAP message. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=targetreadfailure' | The target system has detected an error in the stored LineItem object and so cannot return the data. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=invaliddata' | Part or all of the returned data was detected as invalid by the target system. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=savepointerror' | An error has occurred in the processing of the save-point identifier information making it impossible to read the correct objects from the database. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=savepointsyncerror' | The value of the save point reference from the source was later than that of the target system. No identifiers have been returned. The target system savepoint value is returned to the source system for information. |

### 3.5.9 AddCourseSection() Operation

| | |
|---|---|
| **Name:** | addCourseSection |
| **Return Function Parameter:** | *StatusInfo* – the status of the add request. The permitted status codes are defined in Tables 3.56 and A.2. |
| **Supplied (in) Parameters:** | *sectionAssociationSourcedId:GUID* – the identifier of the SectionAssociation object to be updated.<br>*courseSectionSourcedId:GUID* – the identifier of the CourseSection object to be added to the SectionAssociation. |
| **Returned (out) Parameters:** | None. |
| **Behavior:** | When the source issues the 'addCourseSection' request the target is charged with adding the new Course Section identifier to the Section Association. If the object identified by the supplied SourcedId cannot be located then the request is rejected and the appropriate failure code is returned.<br>The reference counter for the object is incremented by one in the target system. |
| **Notes:** | If the SourcedId is already contained in the SectionAssociation then the 'fullsuccess' status is returned. |

**Table 3.56 Status codes for the 'addCourseSection' operation.**

| Status Code | Explanation of the Cause of the Code |
|---|---|
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor=fullsuccess' | The add request has been fully and successfully implemented by the target system and the identified CourseSection identifier has been added to the SectionAssociation. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=unknownobject' | The SectionAssociation object identifier is unknown in the target system and so the object could not be updated. |

### 3.5.10 RemoveCourseSection() Operation

| | |
|---|---|
| **Name:** | removeCourseSection |
| **Return Function Parameter:** | *StatusInfo* – the status of the remove request. The permitted status codes are defined in Tables 3.57 and A.2. |
| **Supplied (in) Parameters:** | *sectionAssociationSourcedId:GUID* – the identifier of the SectionAssociation object to be updated.<br>*courseSectionSourcedId:GUID* – the identifier of the CourseSection object to be removed from the SectionAssociation. |
| **Returned (out) Parameters:** | None. |
| **Behavior:** | When the source issues the 'removeCourseSection' request the target is charged with removing the given CourseSection identifier from the SectionAssociation. If the object identified by the supplied SourcedId cannot be located then the request is rejected and the appropriate failure code is returned.<br>The reference counter for the object is incremented by one in the target system. |
| **Notes:** | None. |

**Table 3.57 Status codes for the 'removeCourseSection' operation.**

| Status Code | Explanation of the Cause of the Code |
|---|---|
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor=fullsuccess' | The remove request has been fully and successfully implemented by the target system and the identified CourseSection identifier has been removed from the SectionAssociation. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=unknownobject' | The SectionAssociation object identifier is unknown in the target system and so the object could not be changed. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=invaliddata' | The CourseSection object identifier is unknown in the target system and so it could not be added to the SectionAssociation. |

### 3.5.11 UpdateSectionAssociation() Operation

| | |
|---|---|
| **Name:** | updateSectionAssociation |
| **Return Function Parameter:** | *StatusInfo* – the status of the update request. The permitted status codes are defined in Tables 3.58 and A.2. |
| **Supplied (in) Parameters:** | *sourcedId:GUID* – the identifier of the SectionAssociation object to be updated.<br>*sectionAssociationRecord:SectionAssociationRecord* – the SectionAssociation data that is to be stored in the object. |
| **Returned (out) Parameters:** | None. |
| **Behavior:** | When the source issues the 'updateSectionAssociation' request the target is charged with writing the supplied information into the identified object. If any part of the write fails e.g., due to partial invalid data then the whole request is rejected and the record is left in its original state. This is an additive write operation of all the data fields supplied in the update request and fields not supplied remain unchanged. If a field is constrained with a multiplicity of one then the 'updateSectionAssociation' request acts as a 'replaceSectionAssociation' request for that field.<br>If the object identified by the supplied SourcedId cannot be located then the request is rejected and the appropriate failure code is returned.<br>The reference counter for the object is incremented in the target system. |
| **Notes:** | The source is responsible for determining the reason of the failure. |

**Table 3.58 Status codes for the 'updateSectionAssociation' operation.**

| Status Code | Explanation of the Cause of the Code |
|---|---|
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor=fullsuccess' | The update request has been fully and successfully implemented by the target system and the identified SectionAssociation object has been changed on the target system. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=unknownobject' | The SectionAssociation object identifier is unknown in the target system and so the object could not be updated. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=invaliddata' | Part or all of the returned data was detected as invalid by the target system. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=incompletedata' | Some mandatory part of the data has been detected as missing by the target system. |
| 'CodeMajor=Success'<br>'Severity=Warning'<br>'CodeMinor=partialdatastorage' | The target has stored a subset of the sent data record i.e., some of the optional data has not been stored (all mandatory data has been supplied and stored). |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=unknownvocabulary' | The target system could not identify the defined vocabulary term. This may be due to an incorrect term or a missing vocabulary file. |

| | |
|---|---|
| ‘CodeMajor=Failure’<br>‘Severity=Status’<br>‘CodeMinor=unknownmdvocabulary’ | The target system could not identify the defined metadata vocabulary term. This may be due to an incorrect term or a missing vocabulary file. |
| ‘CodeMajor=Failure’<br>‘Severity=Status’<br>‘CodeMinor=unknownextension’ | The target cannot process the proprietary data model extensions used in the object. |

### 3.5.12 ReplaceSectionAssociaton() Operation

| | |
|---|---|
| **Name:** | replaceSectionAssociation |
| **Return Function Parameter:** | *StatusInfo* – the status of the replace request. The permitted status codes are defined in Tables 3.59 and A.2. |
| **Supplied (in) Parameters:** | *sourcedId:GUID* – the identifier of the SectionAssociation object to be updated.<br>*sectionAssociationRecord:SectionAssociationRecord* – the SectionAssociation data that is to be stored in the object. |
| **Returned (out) Parameters:** | None. |
| **Behavior:** | When the source issues the 'replaceSectionAssociation' request the target is charged with writing the supplied information into the identified object. If any part of the write fails e.g., due to partial invalid data then the whole request is rejected and the record is left in its original state. This is a destructive write-over operation of the entire SectionAssociation object. This is equivalent to a 'createSectionAssociation' but for an object that already exists.<br>If the object identified by the supplied SourcedId cannot be located then the request is interpreted as a 'createSectionAssociation' invocation.<br>The reference counter for the object is incremented by one in the target system. |
| **Notes:** | The source is responsible for determining the reason of the failure. |

**Table 3.59 Status codes for the 'replaceSectionAssociation' operation.**

| Status Code | Explanation of the Cause of the Code |
|---|---|
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor=fullsuccess' | The replace request has been fully and successfully implemented by the target system and the identified SectionAssociation object has been changed on the target system. |
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor=createsuccess' | The SectionAssociation object identifier is unknown in the target system and so a new object has been successfully created instead. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=invaliddata' | Part or all of the supplied data was detected as invalid by the target system. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=incompletedata' | Some mandatory part of the data has been detected as missing by the target system. |
| 'CodeMajor=Success'<br>'Severity=Warning'<br>'CodeMinor=partialdatastorage' | The target has stored a subset of the sent data record i.e., some of the optional data has not been stored (all mandatory data has been supplied and stored). |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=unknownvocabulary' | The target system could not identify the defined vocabulary term. This may be due to an incorrect term or a missing vocabulary file. |

| Status Code | Explanation of the Cause of the Code |
|---|---|
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=unknownmdvocabulary' | The target system could not identify the defined metadata vocabulary term. This may be due to an incorrect term or a missing vocabulary file. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=unknownextension' | The target cannot process the proprietary data model extensions used in the object. |

### 3.5.13 DiscoverSectionAssociationIds() Operation

| | |
|---|---|
| **Name:** | discoverSectionAssociationIds |
| **Return Function Parameter:** | *StatusInfo* – the status of the discover request. The permitted status codes are defined in Tables 3.60 and A.2. |
| **Supplied (in) Parameters:** | *queryObject:QueryObject* – this is the query/filter instruction that is to be applied by the target to discover the corresponding SectionAssociation objects. |
| **Returned (out) Parameters:** | *sourcedIdSet:GUIDSet* – the set of identifiers of the SectionAssociation objects whose content conform to the query conditions. |
| **Behavior:** | When the source issues the 'discoverSectionAssociationIds' the target applies the query/filter instructions to the set of SectionAssociation objects and returns the set of SourcedIds that uphold the query/filter.<br><br>If no SectionAssociation objects have the required properties the returned data set is empty and the success status code returned.<br><br>If the target does not understand or cannot apply the requested query/filter then an error status is returned. |
| **Notes:** | The internal structure of this QueryObject is undefined (it is should be treated as a String encoded 'blob). Later versions of this specification will look at the established best practices for clarification on the use of this operation. |

**Table 3.60 Status codes for the 'discoverSectionAssociationIds' operation.**

| Status Code | Explanation of the Cause of the Code |
|---|---|
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor=fullsuccess' | The discover request has been fully and successfully implemented by the target system and the appropriate SectionAssociation identifiers have been read from the target system. |
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor= nosourcedids' | The discover request has been fully and successfully implemented by the target system and no CourseSection object identifiers were found. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=unknownquery' | The target system cannot understand the query request that has been received i.e., the query/filter language is unknown. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=toomuchdata' | The data cannot be returned because it would exceed some physical system constraint e.g., permitted size of SOAP message. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=invaliddata' | Part or all of the returned data was detected as invalid by the source system. |

### 3.5.14 ChangeSectionAssociationIdentifier() Operation

| | |
|---|---|
| **Name:** | changeSectionAssociationIdentifier |
| **Return Function Parameter:** | *StatusInfo* – the status of the change identifier request. The permitted status codes are given in Tables 3.61 and A.2. |
| **Supplied (in) Parameters:** | *sourcedId:GUID* – the identifier of the SectionAssociation object to be changed.<br>*newSourcedId:GUID* – the new identifier to be allocated to the SectionAssociation object. |
| **Returned (out) Parameters:** | None. |
| **Behavior:** | When the source issues the 'changeSectionAssociationIdentifier' request the target is charged with replacing the original SourcedId with the new supplied SourcedId. All membership entries must be similarly changed. All further references to the object must use the new SourcedId otherwise an 'unknown' object failure status code is returned.<br>If the object identified by the supplied SourcedId cannot be located then the request is rejected and the appropriate failure code is returned. |
| **Notes:** | The reference pointer value remains unchanged. |

**Table 3.61 Status codes for the 'changeSectionAssociationIdentifier' operation.**

| Status Code | Explanation of the Cause of the Code |
|---|---|
| 'CodeMajor=Success'<br>'Severity=Status'<br>'CodeMinor=fullsuccess' | The change identifier request has been fully and successfully implemented by the target system and the SectionAssociation object SourcedId has been changed on the target system. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=idallocinusefail' | The target could not allocate the new unique SourcedId to the SectionAssociation object as the identifier is already in use. |
| 'CodeMajor=Failure'<br>'Severity=Status'<br>'CodeMinor=unknownobject' | The current SectionAssociation SourcedId identifier is unknown in the target system and so the object identifier could not be changed. |

## 3.6 Course Object State Machines

The permitted state activity on a CourseTemplate, CourseOffering, CourseSection and SectionAssociation objects are shown in Figures 3.5, 3.6, 3.7 and 3.8 respectively. Each state diagram has three states (the arcs are annotated with the operations that are associated with the change of state):

- ‘No Object’ state – no object exists with a particular sourcedId;
- ‘Object with Provider assigned sourcedId’ – an object exists with the sourcedId allocated by the Provider system;
- ‘Object with Consumer assigned sourcedId’ – an object exists with the sourcedId allocated by the Consumer system.

The state machine for a CourseTemplate object is shown in Figure 3.5.

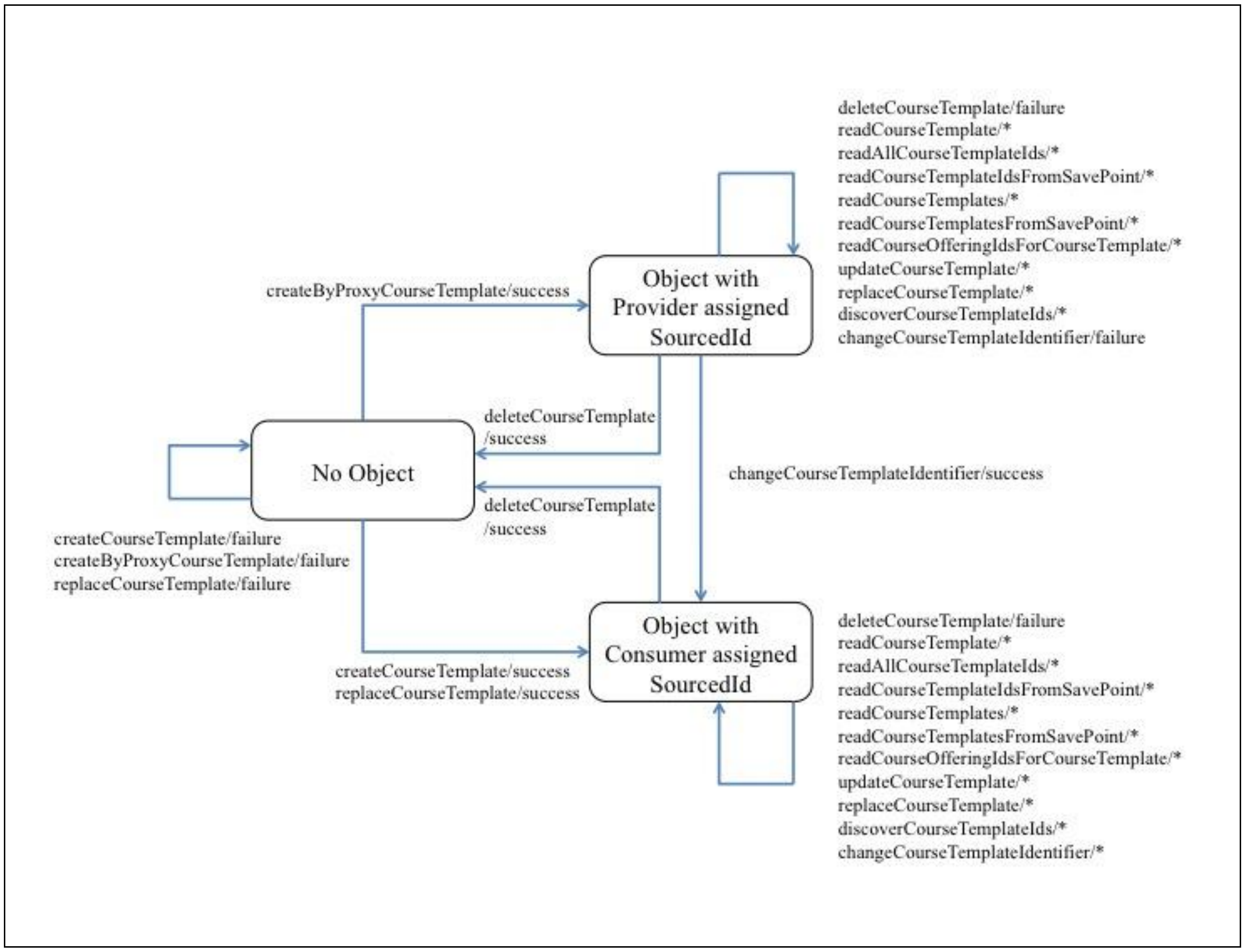

deleteCourseTemplate/failure
readCourseTemplate/*
readAllCourseTemplateIds/*
readCourseTemplateIdsFromSavePoint/*
readCourseTemplates/*
readCourseTemplatesFromSavePoint/*
readCourseOfferingIdsForCourseTemplate/*
updateCourseTemplate/*
replaceCourseTemplate/*
discoverCourseTemplateIds/*
changeCourseTemplateIdentifier/failure
createByProxyCourseTemplate/success
Object with Provider assigned SourcedId
deleteCourseTemplate /success
No Object
changeCourseTemplateIdentifier/success
deleteCourseTemplate /success
createCourseTemplate/failure
createByProxyCourseTemplate/failure
replaceCourseTemplate/failure
Object with Consumer assigned SourcedId
createCourseTemplate/success
replaceCourseTemplate/success
deleteCourseTemplate/failure
readCourseTemplate/*
readAllCourseTemplateIds/*
readCourseTemplateIdsFromSavePoint/*
readCourseTemplates/*
readCourseTemplatesFromSavePoint/*
readCourseOfferingIdsForCourseTemplate/*
updateCourseTemplate/*
replaceCourseTemplate/*
discoverCourseTemplateIds/*
changeCourseTemplateIdentifier/*

**Figure 3.5 State machine for a ‘courseTemplate’ object.**

The start state is ‘No Object’ i.e., the CourseTemplate object has not yet been created. Only the ‘createCourseTemplate()’, ‘replaceCourseTemplate()’ and ‘createByProxyCourseTemplate()’ operations are possible. Once the CourseTemplate object has been created it then persists until a successful ‘deleteCourseTemplate()’ operation is completed. The ‘createCourseTemplate()’ and ‘createCourseTemplate()’ operations take the system into the ‘Object with Consumer assigned sourcedId’ state whereas the ‘createByProxyCourseTemplate()’ takes the system into the ‘Object with Provider assigned sourcedId’ state.

The system can be moved from the ‘Object with Provider assigned sourcedId’ state into the ‘Object with Consumer assigned sourcedId’ state by the successful completion of the ‘changeCourseTemplateIdentifier()’ operation.

Once the system is in the ‘Object with Consumer assigned sourcedId’ or the ‘Object with Provider assigned sourcedId’ states then the ‘readCourseTemplate()’, ‘readAllCourseTemplateIds()’, ‘readCourseTemplateIdsFromSavePoint()’, ‘readCourseTemplates()’, ‘readCourseTemplatesFromSavePoint()’, ‘readCourseOfferingIdsForCourseTemplate()’, ‘updateCourseTemplate()’, ‘replaceCourseTemplate()’ and ‘discoverCourseTemplateIds()’ operations are now possible.

This is the state machine for each CourseTemplate object in the Service Consumer and the Service Provider. The binding of the Information Model must guarantee that these two state machines remain synchronized for each CourseTemplate object.

The state machine for a CourseOffering object is shown in Figure 3.6.

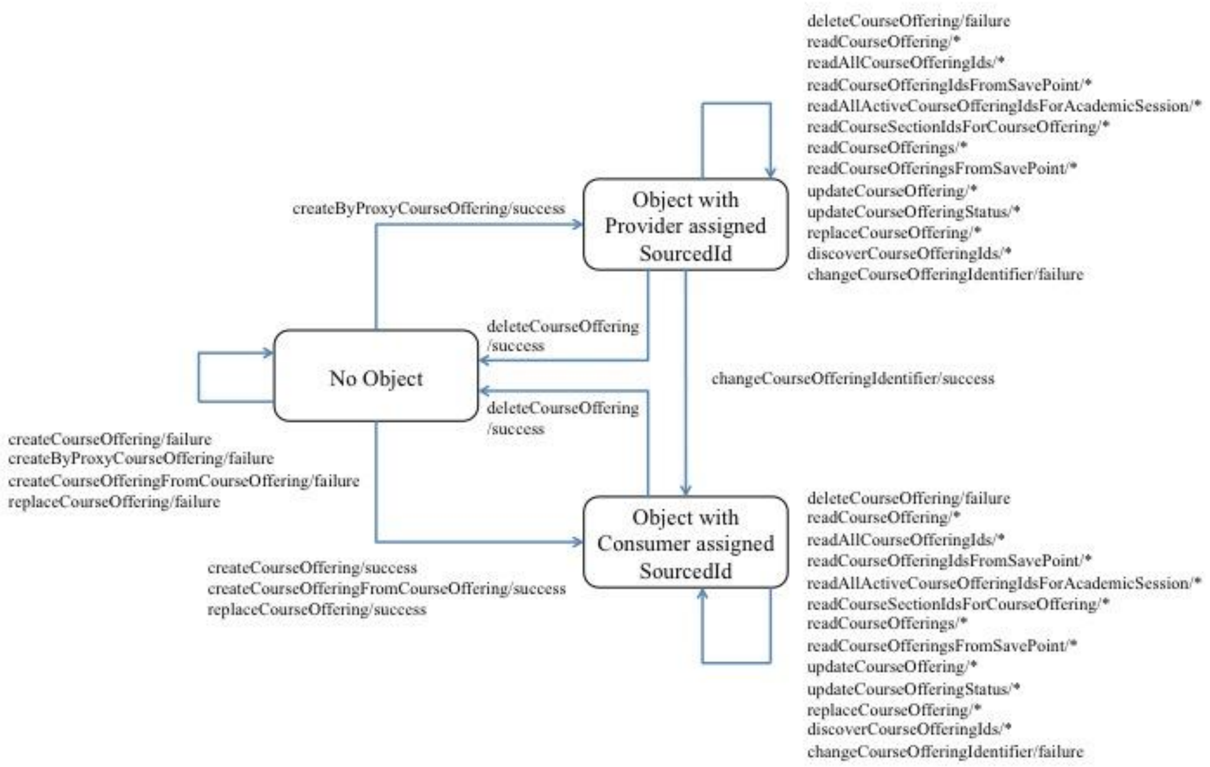

deleteCourseOffering/failure
readCourseOffering/*
readAllCourseOfferingIds/*
readCourseOfferingIdsFromSavePoint/*
readAllActiveCourseOfferingIdsForAcademicSession/*
readCourseSectionIdsForCourseOffering/*
readCourseOfferings/*
readCourseOfferingsFromSavePoint/*
updateCourseOffering/*
updateCourseOfferingStatus/*
replaceCourseOffering/*
discoverCourseOfferingIds/*
changeCourseOfferingIdentifier/failure
createByProxyCourseOffering/success
Object with Provider assigned SourcedId
deleteCourseOffering /success
No Object
changeCourseOfferingIdentifier/success
deleteCourseOffering /success
createCourseOffering/failure
createByProxyCourseOffering/failure
createCourseOfferingFromCourseOffering/failure
replaceCourseOffering/failure
Object with Consumer assigned SourcedId
createCourseOffering/success
createCourseOfferingFromCourseOffering/success
replaceCourseOffering/success
deleteCourseOffering/failure
readCourseOffering/*
readAllCourseOfferingIds/*
readCourseOfferingIdsFromSavePoint/*
readAllActiveCourseOfferingIdsForAcademicSession/*
readCourseSectionIdsForCourseOffering/*
readCourseOfferings/*
readCourseOfferingsFromSavePoint/*
updateCourseOffering/*
updateCourseOfferingStatus/*
replaceCourseOffering/*
discoverCourseOfferingIds/*
changeCourseOfferingIdentifier/failure

**Figure 3.6 State machine for a ‘courseOffering’ object.**

The start state is ‘No Object’ i.e., the CourseOffering object has not yet been created. Only the ‘createCourseOffering()’, ‘createByProxyCourseOffering ()’, ‘replaceCourseOffering()’ and ‘createCourseOfferingFromCourseOffering()’ operations are possible. Once the CourseOffering object has been created it then persists until a successful ‘deleteCourseOffering()’ operation is completed. The ‘createCourseOffering()’, ‘replaceCourseOffering()’ and ‘createCourseOfferingFromCourseOffering()’ operations take the system into the ‘Object with Consumer assigned sourcedId’ state whereas the ‘createByProxyCourseOffering()’ takes the system into the ‘Object with Provider assigned sourcedId’ state.

The system can be moved from the ‘Object with Provider assigned sourcedId’ state into the ‘Object with Consumer assigned sourcedId’ state by the successful completion of the ‘changeCourseOfferingIdentifier()’ operation.

Once the system is in the ‘Object with Consumer assigned sourcedId’ or the ‘Object with Provider assigned sourcedId’ states then the ‘readCourseOffering()’, ‘readAllCourseOfferingIds()’, ‘readCourseOfferingIdsFromSavePoint()’, ‘readAllActiveCourseOfferingIdsForAcademicSession()’, ‘readCourseSectionIdsForCourseOffering()’, ‘readCourseOfferings()’, ‘readCourseOfferingsFromSavePoint()’, ‘updateCourseOffering()’, ‘updateCourseOfferingStatus’, ‘replaceCourseOffering()’ and ‘discoverCourseOfferingIds()’ operations are now possible.

This is the state machine for each CourseOffering object in the Service Consumer and the Service Provider. The binding of the Information Model must guarantee that these two state machines remain synchronized for each CourseOffering object.

The state machine for a CourseSection object is shown in Figure 3.7.

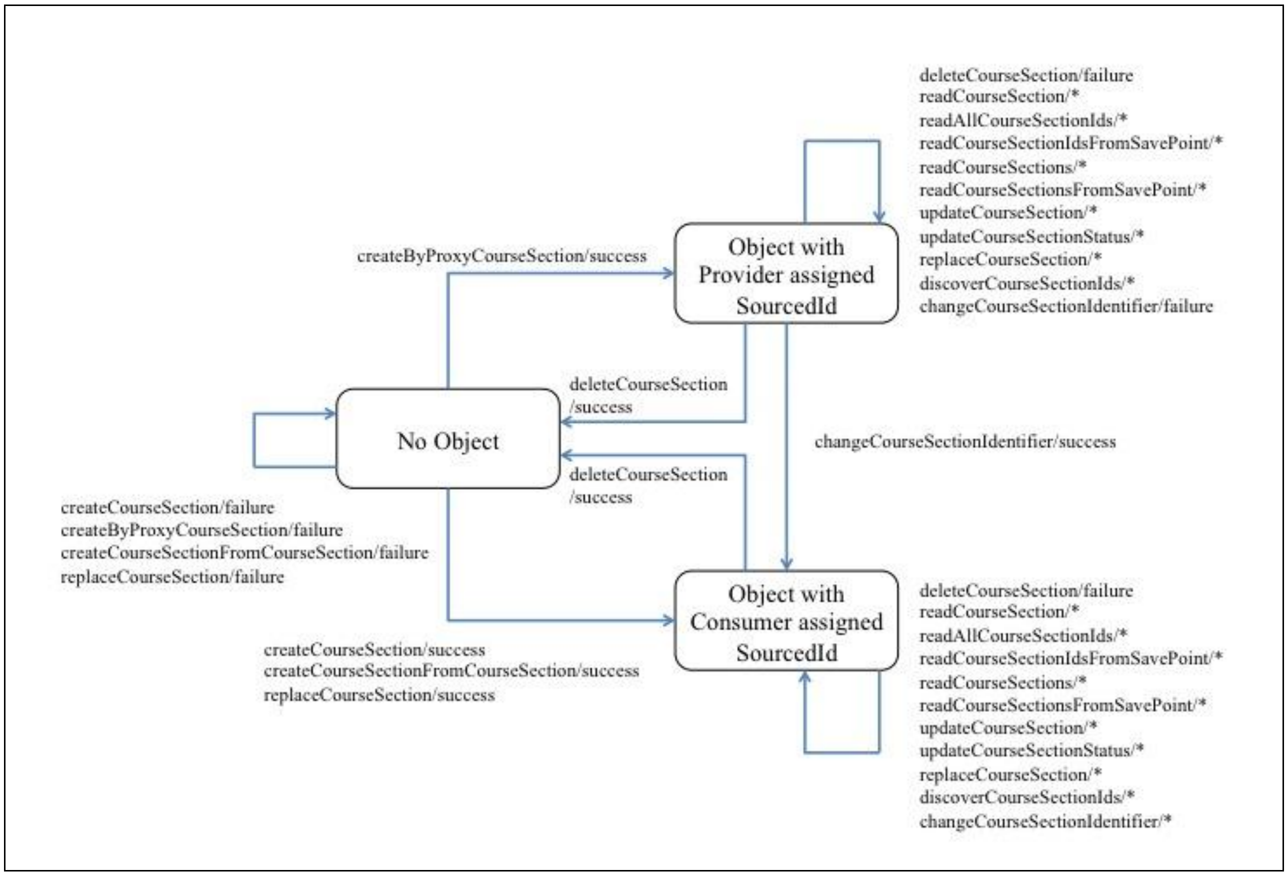

deleteCourseSection/failure
readCourseSection/*
readAllCourseSectionIds/*
readCourseSectionIdsFromSavePoint/*
readCourseSections/*
readCourseSectionsFromSavePoint/*
updateCourseSection/*
updateCourseSectionStatus/*
replaceCourseSection/*
discoverCourseSectionIds/*
changeCourseSectionIdentifier/failure
createByProxyCourseSection/success
Object with Provider assigned SourcedId
deleteCourseSection /success
No Object
changeCourseSectionIdentifier/success
deleteCourseSection /success
createCourseSection/failure
createByProxyCourseSection/failure
createCourseSectionFromCourseSection/failure
replaceCourseSection/failure
Object with Consumer assigned SourcedId
createCourseSection/success
createCourseSectionFromCourseSection/success
replaceCourseSection/success
deleteCourseSection/failure
readCourseSection/*
readAllCourseSectionIds/*
readCourseSectionIdsFromSavePoint/*
readCourseSections/*
readCourseSectionsFromSavePoint/*
updateCourseSection/*
updateCourseSectionStatus/*
replaceCourseSection/*
discoverCourseSectionIds/*
changeCourseSectionIdentifier/*

**Figure 3.7 State machine for a CourseSection’ object.**

The start state is ‘No Object’ i.e., the CourseSection object has not yet been created. Only the ‘createCourseSection()’, ‘createByProxyCourseSection()’, ‘replaceCourseSection()’ and ‘createCourseSectionFromCourseSection()’ operations are possible. Once the CourseSection object has been created it then persists until a successful ‘deleteCourseSection()’ operation is completed. The ‘createCourseSection()’, ‘replaceCourseSection()’ and ‘createCourseSectionFromCourseSection’ operations take the system into the ‘Object with Consumer assigned sourcedId’ state whereas the ‘createByProxyCourseSection()’ takes the system into the ‘Object with Provider assigned sourcedId’ state.

The system can be moved from the ‘Object with Provider assigned sourcedId’ state into the ‘Object with Consumer assigned sourcedId’ state by the successful completion of the ‘changeCourseSectionIdentifier()’ operation.

Once the system is in the ‘Object with Consumer assigned sourcedId’ or the ‘Object with Provider assigned sourcedId’ states then the ‘readCourseSection()’, ‘readAllCourseSectionIds()’, ‘readCourseSectionIdsFromSavePoint()’, ‘readCourseSections()’, ‘readCourseSectionsFromSavePoint()’, ‘updateCourseSection()’, ‘updateCourseSectionStatus()’, ‘replaceCourseSection()’ and ‘discoverCourseSectionIds()’ operations are now possible.

This is the state machine for each CourseSection object in the Service Consumer and the Service Provider. The binding of the Information Model must guarantee that these two state machines remain synchronized for each CourseSection object.

The state machine for a SectionAssociation object is shown in Figure 3.8.

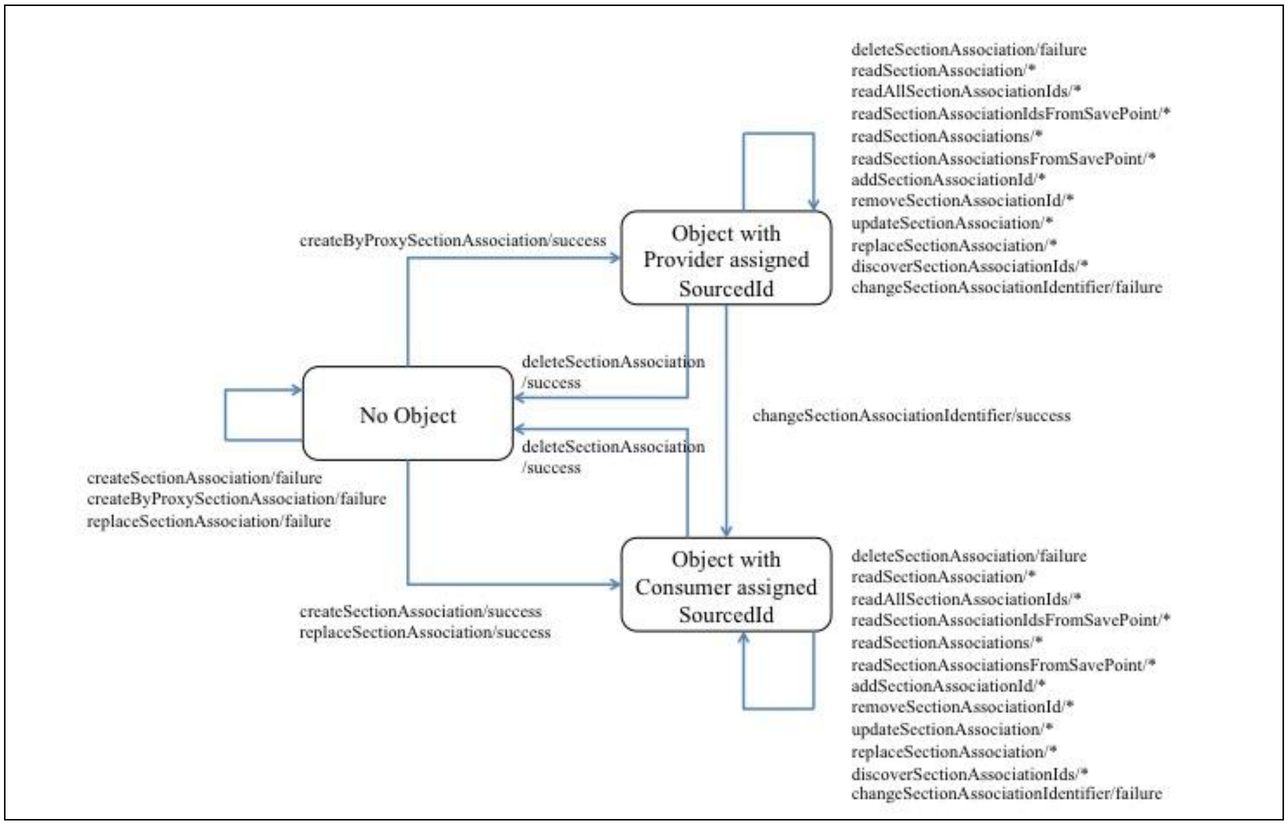

deleteSectionAssociation/failure
readSectionAssociation/*
readAllSectionAssociationIds/*
readSectionAssociationIdsFromSavePoint/*
readSectionAssociations/*
readSectionAssociationsFromSavePoint/*
addSectionAssociationId/*
removeSectionAssociationId/*
updateSectionAssociation/*
replaceSectionAssociation/*
discoverSectionAssociationIds/*
changeSectionAssociationIdentifier/failure
createByProxySectionAssociation/success
Object with Provider assigned SourcedId
deleteSectionAssociation /success
No Object
changeSectionAssociationIdentifier/success
deleteSectionAssociation /success
createSectionAssociation/failure
createByProxySectionAssociation/failure
replaceSectionAssociation/failure
Object with Consumer assigned SourcedId
createSectionAssociation/success
replaceSectionAssociation/success
deleteSectionAssociation/failure
readSectionAssociation/*
readAllSectionAssociationIds/*
readSectionAssociationIdsFromSavePoint/*
readSectionAssociations/*
readSectionAssociationsFromSavePoint/*
addSectionAssociationId/*
removeSectionAssociationId/*
updateSectionAssociation/*
replaceSectionAssociation/*
discoverSectionAssociationIds/*
changeSectionAssociationIdentifier/failure

**Figure 3.8 State machine for a 'sectionAssociation' object.**

The start state is 'No Object' i.e., the SectionAssociation object has not yet been created. Only the 'createSectionAssociation()', 'replaceSectionAssociation()' and 'createByProxySectionAssociation()' operations are possible. Once the SectionAssociation object has been created it then persists until a successful 'deleteSectionAssociation()' operation is completed. The 'createSectionAssociation()' and 'replaceSectionAssociation()' operations take the system into the 'Object with Consumer assigned sourcedId' state whereas the 'createByProxySectionAssociation()' takes the system into the 'Object with Provider assigned sourcedId' state.

The system can be moved from the 'Object with Provider assigned sourcedId' state into the 'Object with Consumer assigned sourcedId' state by the successful completion of the 'changeSectionAssociationIdentifier()' operation.

Once the system is in the 'Object with Consumer assigned sourcedId' or the 'Object with Provider assigned sourcedId' states then the 'readSectionAssociation()', 'readAllSectionAssociationIds()', 'readSectionAssociationIdsFromSavePoint()', 'readSectionAssociations()', 'addSectionAssociationId()', 'removeSectionAssociationId()', 'readSectionAssociationsFromSavePoint()', 'updateSectionAssociation()', 'replaceSectionAssociation()' and 'discoverSectionAssociationIds()' operations are now possible.

This is the state machine for each CourseSection object in the Service Consumer and the Service Provider. The binding of the Information Model must guarantee that these two state machines remain synchronized for each SectionAssociation object.

# 4 Interface Data Model

The set of operations described within the behavior model (Section 3) are based upon class descriptions specific to the parameters of the operations. It is these classes that are now described.

## 4.1 AcademicSession Class Description

This is the container for the states used to define the academic session for a CourseOffering and CourseSection. This is an implementation dependent vocabulary based enumeration.

## 4.2 CourseOfferingRecord Class Description

This is the data-type for CourseOfferingRecords. The data model for a Course OfferingRecord is described in Section 5. A key difference for an object passed in the interface, as opposed to the requirement for an end-system, is that the content is dependent on the type of operation. A CourseOfferingRecord object must consist of the SourcedId of the CourseOffering object and the CourseOffering object itself.

## 4.3 CourseOfferingRecordSet Class Description

This is the data-type for a set of CourseOfferingRecords (zero or more). Any implementation of the CourseOfferingRecordSet must be able to contain at least 100,000 CourseOfferingRecords i.e., the smallest permitted maximum number.

## 4.4 CourseSectionRecord Class Description

This is the data-type for CourseSections. The data model for a Course Section is described in Section 5. A key difference for an object passed in the interface, as opposed to the requirement for an end-system, is that the content is dependent on the type of operation. A CourseSectionRecord object must consist of the SourcedId of the CourseSection object and the CourseSection object itself.

## 4.5 CourseSectionRecordSet Class Description

This is the data-type for a set of CourseSectionRecords (zero or more). Any implementation of the CourseSectionRecordSet must be able to contain at least 250,000 CourseSectionRecords i.e., the smallest permitted maximum number.

## 4.6 CourseTemplateRecord Class Description

This is the data-type for CourseTemplates. The data model for a Course Template is described in Section 5. A key difference for an object passed in the interface, as opposed to the requirement for an end-system, is that the content is dependent on the type of operation. A CourseTemplateRecord object must consist of the SourcedId of the CourseTemplate object and the CourseTemplate object itself.

## 4.7 CourseTemplateRecordSet Class Description

This is the data-type for a set of CourseTemplateRecords (zero or more). Any implementation of the CourseTemplateRecordSet must be able to contain at least 100,000 CourseTemplateRecords i.e., the smallest permitted maximum number.

## 4.8 GUID Class Description

This is the data type for the globally unique sourcedIds. These GUIDs must be unique across the set of communicating end-systems within the Learning Information Services system. The internal format of the GUID is outside the scope of this specification but they must all be valid XML strings. Any implementation of the GUID class must be able to support GUIDs of at least 1024 octets in length i.e., the shortest permitted maximum length.

## 4.9 GUIDSet Class Description

This is the data-type for a set of GUIDs (zero or more). Any implementation of the GUIDSet must be able to contain at least 250,000 GUIDs i.e., the smallest permitted maximum number.

## 4.10 QueryObject Class Description

This is the data-type for the query instruction. This is a String 'blob' with the smallest permitted maximum length of 4096 octets. The internal structure of this string is undefined. Later versions of this specification will look at the established best practices for clarification on the use of this string.

## 4.11 SectionAssociationRecord Class Description

This is the data-type for SectionAssociationRecords. The data model for a SectionAssociationRecord is described in Section 5. A key difference for an object passed in the interface, as opposed to the requirement for an end-system, is that the content is dependent on the type of operation. A SectionAssociationRecord object must consist of the SourcedId of the SectionAssociation object and the SectionAssociation object itself.

## 4.12 SectionAssociationRecordSet Class Description

This is the data-type for a set of SectionAssociationRecords (zero or more). Any implementation of the SectionAssociationRecordSet must be able to contain at least 10,000 SectionAssociationRecords i.e., the smallest permitted maximum number.

## 4.13 SequenceIdentifier Class Description

This is the data-type for the sequence identifier used to denote identify the synchronization reference point between the two communicating systems. The sequence is denoted by the date-time string YYYY-MM-DDTHH:MM:SS.NNN where 'YYYY' denotes the year, the first 'MM' string the month (01-12), 'DD' the day (01-31), 'HH' the hour (00-23), the second 'MM' string the minute (00-59), 'SS' the second (00-59) and 'NN' the millisecond value (000-999).

At initialization the value is set to '1000-01-01T00:00:00.000'. The value is changed to the current time for every operation that results in a change of the value of the data stored in the 'group' object.

All values are to be rounded down at the level of greatest resolution.

## 4.14 Status Class Description

This is the data-type for the enumerated list of states used to define the status of a CourseTemplate, CourseOffering, CourseSection and SectionAssociation. An external vocabulary is used to store the enumerated list of vales. The list of permitted values is: { Active, Inactive}.

## 4.15 StatusInfo Class Description

This is the data-type for the status information returned by the target to the source. The structure of this class is described in the IMS GLC General Web Services specification v1.0 (Appendix A) [GWS, 05].

# 5 End System Data Model

The end system data model defines the persistence model that must be maintained by an end system to ensure the correct system behavior.

An informative overview of the entire Persistence Data Model is provided as a Platform Independent Model (PIM) expressed in UML constructs. All UML diagrams expressed as “Platform Independent Model” are non-normative. Normative tables defining the classes in this Information Model follow the informative UML diagrams. A full definition of the UML Profile and the terms used in the normative tabular descriptions in this document to describe the PIM can be found in [SDN07, 06].

In the tables in this section the character sequence “n/a” is used to mark a field “not applicable.” Any field so marked is not relevant to the class being defined. Features so marked shall be ignored when binding a class defined by this Information Model.

## 5.1 Key Terms and Concepts

Classes in this information model are classified into four abstract class types. These abstractions are bound to specific data structures for machine processing in the IMS Course Management Services WSDL Binding [CMS, 011]. The four abstract class types are:

- **container:** A container class may be a parent of one or more child classes;
- **value:** A value class shall not be a parent. That is, it shall not be a composite of characteristic, container, value, or unspecified class types. A value class shall always be a child of a container class and shall have semantic value within the scope of its parent class’s semantic value;
- **characteristic:** A characteristic class shall not be a parent. A characteristic class shall declare a trait or value that is an intrinsic feature or part of a container class. A characteristic class is tightly coupled to the **container** class it modifies or one of which facets it describes;
- **unspecified: An unspecified class may be a parent.** An unspecified class serves as an extension point for this Information Model.

Table 5.1 lists the class descriptors used to describe the abstract classes and definitions of the descriptors.

**Table 5.1 Class descriptors**

| Descriptor | Definition |
|---|---|
| Class name | The name given to the class being described. |
| Class type | The abstract class type of this class. |
| Data type | For value and characteristic classes, the allowed structure for valid values for the class. Valid data types are:<br><br>**AcademicSession:** Used to denote the string that contains the value for the academic session.<br><br>**Boolean:** The primitive, two-valued data type that uses the keywords “true” and “false” to indicate the logical state of an object.<br><br>**Date:** The date represents a date in the format of ISO 8601 i.e., ‘YYYY-MM-DD’.<br><br>**DateTime:** The DateTime represents a combined date and time in the format of ISO 8601 i.e., ‘YYYY-MM-DDThh:mm:ssTZD’. The time is denoted in Coordinated Universal Time (UTC) with TZD denoting the time zone offset in hours and minutes with respect to UTC.<br><br>**GUID**: An identifier that is globally unique within the Learning Information Services |

| Descriptor | Definition |
|---|---|
| | System. This will be based upon the Normalized String data-type that has a constrained value-space. This has a length [1..4095] characters.<br>**Integer:** An integer.<br>**Language:** This data-type is used to denote that the attribute is used to identify the language of the associated entry. The language values are defined as per RFC4646.<br>**NormalizedString:** A sequence of printable characters that does not contain carriage returns or tabs.<br>**Status:** The status of the course component. This is an enumerated vocabulary.<br>**String:** A sequence of printable characters.<br>**Text:** A language annotated string. The string is accompanied by a language identifier that demotes the language for the string.<br>**Time:** The time, including timezone, represents a date in the format of ISO 8601 i.e., 'HH:MM:SSTZD'. The time is denoted in Coordinated Universal Time (UTC) with TZD denoting the time zone offset in hours and minutes with respect to UTC.<br>**URI**: Any syntactically valid instance of a URI as defined in RFC3986. Note: Many of the foundational Specifications, Standards, and Recommendations referred to by this Information Model use RFC2396 and RFC2732 as the definitions of URI. These are made obsolete by RFC3986, but many of the foundational documents have not been updated to reference RFC3986.<br>**URL:** A normalized string that is used to contain a Universal Resource Location.<br>**unspecified:** The data type is not known or is not important. |
| Value space | The range of valid values for this class. If the value space is unspecified, it is not known or is not important. |
| Multiplicity | A property of a class indicating the number of times it may be used or appear in a given parent context. The values of this property are expressed as a range or shorthand for a range using this notation:<br>• **'0..1'** [optional; restricted]<br>• **'0..unbounded'** [optional; unrestricted]<br>• **'1..1'** [mandatory; restricted]<br>• **'1..unbounded'** [mandatory; unrestricted]<br>Multiplicities may also appear in short-hand notation in the UML models. The short-hand equivalents shall be (exclusive of bracketed comments):<br>• **'*'** [optional; unrestricted]<br>• **'1'** [mandatory; restricted]<br>• **'1..*'** [mandatory; unrestricted]<br>Where multiplicity is greater than one, the importance of the ordering of siblings is also indicated by appending either **","ordered** or **"," unordered.**<br>**ordered** specifies a sequence of siblings as listed, **unordered** specifies a collection or bag of siblings for which the order is not important. |
| Parents | Lists classes that may be parents of this class. |
| Children | Lists the possible child classes of this class in the form **"[" child *"," child "]".** One or more child classes may be expressed within square brackets. Each child class shall be separated by a comma. |

| Descriptor | Definition |
|---|---|
| | Where more than one child is listed, the importance of the ordering of siblings is also indicated by appending either **“,”ordered** or **“,” unordered.** **ordered** specifies a sequence of siblings as listed. **unordered** specifies a collection or bag of sibling for which the order is not important. |
| Description | Contains descriptions relating to the class and its values space. |

In general, this specification does not define the ways in which an end system must be realized. However, the required interoperability behavior requires that an end system have certain characteristics. The static properties of these characteristics are defined in this Section, including:

- When an attribute has a multiplicity of ‘1..1’ then an end system must be capable of supporting one instance;
- When an attribute has a multiplicity of ‘1..*’ then an end system must be capable of supporting at least one instance. The specification will also define the smallest permitted maximum number of instances that must also be supported by the end system;
- When an attribute has a multiplicity of ‘0..1’ then an end system should support a single instance;
- When an attribute has a multiplicity of ‘0..*’ then the specification will define the smallest permitted maximum number of instances that must also be supported by the end system.

When the object is passed as part of a service call then attributes that have a ‘1..1’ or ‘1..*’ multiplicity may or may not be exchanged. This is because the specification of an end system defines capability; an operational system may or may not exchange the associated information.

## 5.2 The Structure of a Course

The conceptual data model for the structure of a course is shown in Figure 5.1. There are three facets of a course structure:

- The components of the core data model of the course denoted in Figure 5.1 by the CourseTemplate, CourseOffering, CourseSection and SectionAssociation classes;
- Identification of the participants in an instance of a part of the course structure is through a Membership object;
- The reference structures that are used to identify specific instances of each part of the course denoted in Figure 5.1 by the classes Template, Offering, Section and Association.

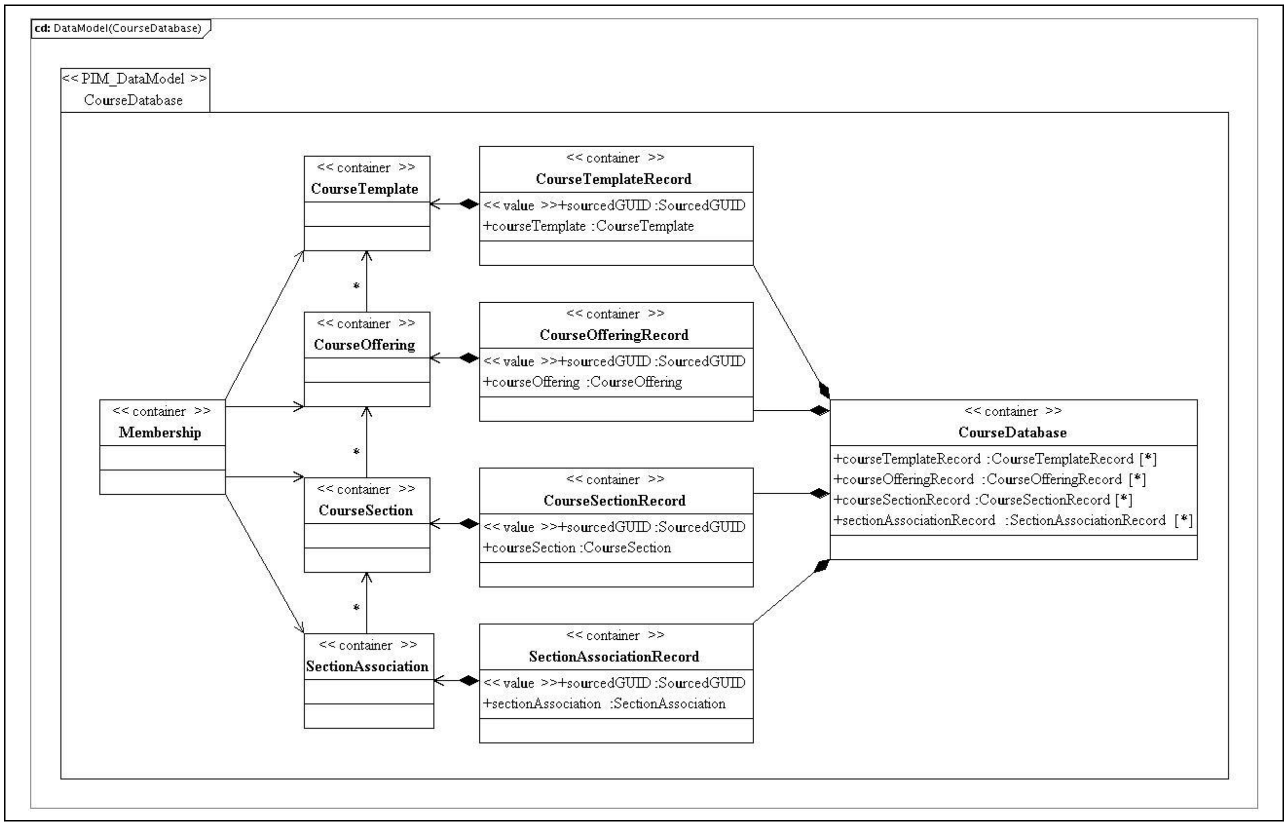

cd: DataModel(CourseDatabase)
<< PIM_DataModel >> CourseDatabase
<< container >> CourseTemplate
<< container >> CourseOffering
<< container >> CourseSection
<< container >> SectionAssociation
<< container >> Membership
<< container >> CourseTemplateRecord
<< value >>+sourcedGUID :SourcedGUID
+courseTemplate :CourseTemplate
<< container >> CourseOfferingRecord
<< value >>+sourcedGUID :SourcedGUID
+courseOffering :CourseOffering
<< container >> CourseSectionRecord
<< value >>+sourcedGUID :SourcedGUID
+courseSection :CourseSection
<< container >> SectionAssociationRecord
<< value >>+sourcedGUID :SourcedGUID
+sectionAssociation :SectionAssociation
<< container >> CourseDatabase
+courseTemplateRecord :CourseTemplateRecord [*]
+courseOfferingRecord :CourseOfferingRecord [*]
+courseSectionRecord :CourseSectionRecord [*]
+sectionAssociationRecord :SectionAssociationRecord [*]
*
*
*

**Figure 5.1 CourseDatabase class diagram.**

The Course structure can be extended by using the GMS [GMS, 11] to create Groups that are appended to the relevant Course components. Figure 5.2 shows how Course structures can be extended. The two extensions are:

- Sub-structures under CourseSection used to represent arbitrarily small course component objects e.g., part of a lecture, etc.
- Parent structures of CourseTemplates used to represent organization structures that have hierarchical ownership of the ‘Course’ e.g., the Department, the Faculty, the University, etc.

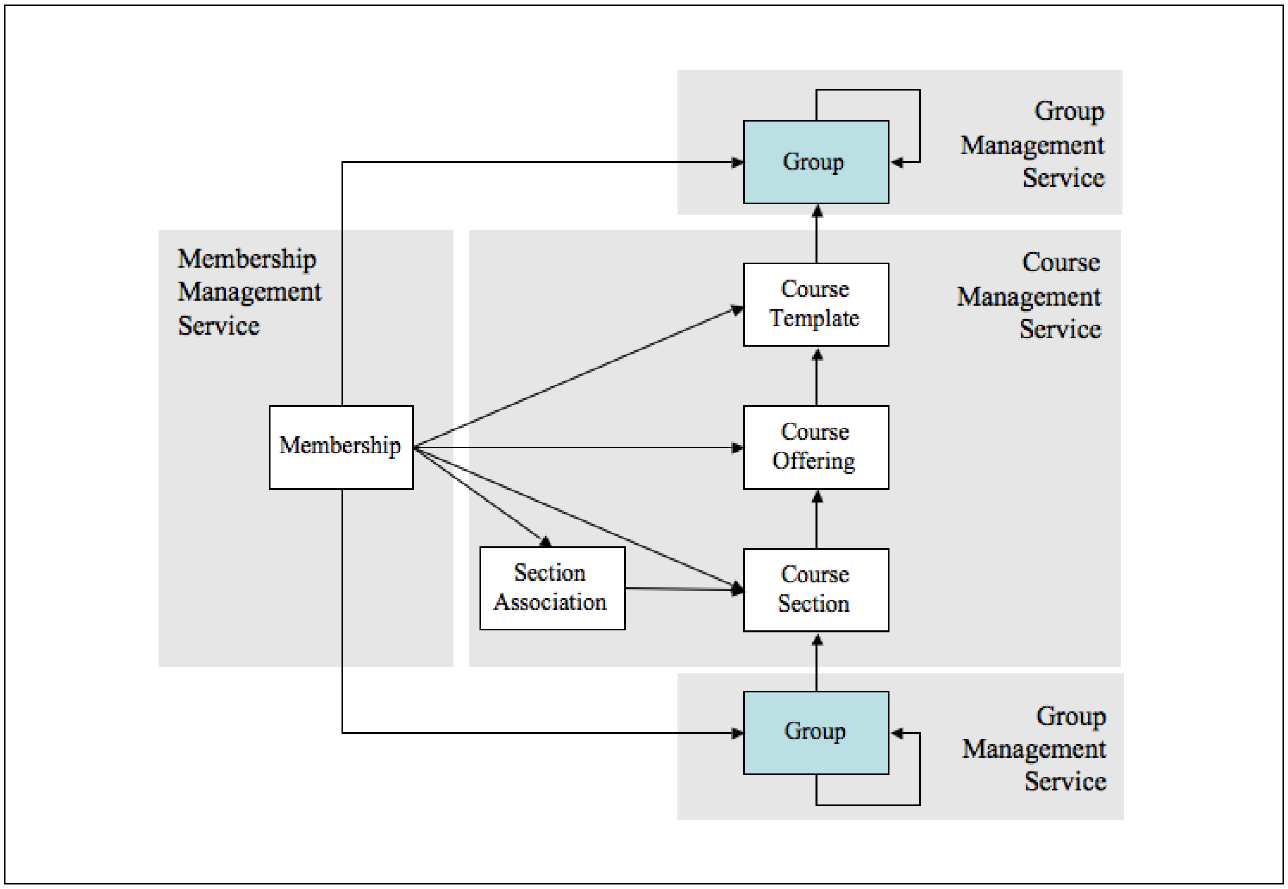


**Figure 5.2 Extending a course.**

The Groups are linked to the relevant Course component using the relationship structures within the Group definition. Multiple linked Group objects can be used to create arbitrarily large structures. The MMS [MMS, 11] can be used to define memberships of the course structures including those defined as Groups

## 5.3 CourseDatabase Class Description

**Table 5.1 Description of the 'CourseDatabase' class.**

| Descriptor | Definition |
|---|---|
| Class name | CourseDatabase |
| Class type | container |
| Multiplicity | 1 |
| Parents | Root |
| Children | [ courseTemplateRecord, courseOfferingRecord, courseSectionRecord, sectionAssociationRecord ], ordered |
| Description | This is the database within the end-system that contains all of the course objects. There are four types of course object, namely: CourseTemplateRecord, CourseOfferingRecord, CourseSectionRecord and SectionAssociationRecord. Each course object consists of a globally unique identifier, its SourcedId, and either the CourseTemplate, CourseOffering, CourseSection or SectionAssociation data itself. The database consists of the set of course objects, the set of GUIDs and the relationship mapping between them. The manner in which this information is physically stored is outside of the scope of this specification.<br>An implementation must be capable of supporting at least: 2000 CourseTemplates; 10,000 CourseOfferings; 100,000 CourseSections; and 1,000 SectionAssociations. |

## 5.4 CourseTemplateRecord Class Description

**Table 5.2 Description of the 'CourseTemplateRecord' class.**

| Descriptor | Definition |
|---|---|
| Class name | CourseTemplateRecord |
| Class type | container |
| Multiplicity | 0..unbounded, unordered |
| Parents | CourseDatabase |
| Children | [sourcedGUID, courseTemplate ], unordered |
| Description | The CourseTemplateRecord object represents the association between the unique identifier (SourcedGUID) for the CourseTemplate object with the CourseTemplate object itself. The 'identifier' object is not a part of the CourseTemplate object but both are managed within the Course Database. There is an isomorphic association between each pair of SourcedGUID and CourseTemplate objects. |

### 5.4.1 CourseTemplateId Attribute Description

**Table 5.3 Description of the 'sourcedGUID' attribute for the 'Template' class.**

| Descriptor | Definition |
|---|---|
| Attribute name | sourcedGUID |
| Data type | SourcedGUID |
| Value space | container |
| Multiplicity | 1 |
| Description | This is the globally unique identifier that has been assigned to the associated CourseTemplate object. Each course template object must have only one SourcedGUID but this may be changed, any number of times, during the object's lifetime. |

## 5.5 CourseOfferingRecord Class Description

**Table 5.4 Description of the 'CourseOfferingRecord' class.**

| Descriptor | Definition |
|---|---|
| Class name | CourseOfferingRecord |
| Class type | container |
| Multiplicity | 0..unbounded, ordered |
| Parents | CourseDatabase |
| Children | [ sourcedGUID, courseOffering ], unordered |
| Description | The CourseOfferingRecord object represents the association between the unique identifier (SourcedGUID) for the CourseOffering object with the CourseOffering object itself. The identifier object is not a part of the CourseOffering object but both are managed within the Course Database. There is an isomorphic association between each pair of SourcedGUID and CourseOffering objects. |

### 5.5.1 CourseOfferingId Attribute Description

**Table 5.5 Description of the 'sourcedGUID' attribute for the 'Offering' class.**

| Descriptor | Definition |
|---|---|
| Attribute name | sourcedGUID |
| Data type | SourcedGUID |
| Value space | container |
| Multiplicity | 1 |
| Description | This is the globally unique identifier that has been assigned to the associated CourseOffering object. Each CourseOffering object must have only one SourcedGUID but this may be changed, any number of times, during the object's lifetime. |

## 5.6 CourseSectionRecord Class Description

**Table 5.6 Description of the 'CourseSectionRecord' class.**

| Descriptor | Definition |
|---|---|
| Class name | CourseSectionRecord |
| Class type | container |
| Multiplicity | 0..unbounded, unordered |
| Parents | CourseDatabase |
| Children | [ sourcedGUID, courseSection ], unordered |
| Description | The CourseSectionRecord object represents the association between the unique identifier (SourcedGUID) for the CourseSection object with the CourseSection object itself. The identifier object is not a part of the CourseSection object but both are managed within the Course Database. There is an isomorphic association between each pair of SourcedGUID and CourseSection objects. |

### 5.6.1 CourseSectionId Attribute Description

**Table 5.7 Description of the 'sourcedGUID' attribute for the 'Section' class.**

| Descriptor | Definition |
|---|---|
| Attribute name | sourcedGUID |
| Data type | SourcedGUID |
| Value space | container |
| Multiplicity | 1 |
| Description | This is the globally unique identifier that has been assigned to the associated CourseSection object. Each course section object must have only one SourcedGUID but this may be changed, any number of times, during the object's lifetime. |

## 5.7 SectionAssociationRecord Class Description

**Table 5.8 Description of the 'SectionAssociationRecord' class.**

| Descriptor | Definition |
|---|---|
| Class name | SectionAssociationRecord |
| Class type | container |
| Multiplicity | 0..unbounded, ordered |
| Parents | CourseDatabase |
| Children | [ sourcedGUID, sectionAssociation ], unordered |
| Description | The SectionAssociationRecord object represents the association between the unique identifier (SourcedGUID) for the SectionAssociation object with the SectionAssociation object itself. The identifier object is not a part of the SectionAssociation object but both are managed within the Course Database. There is an isomorphic association between each pair of SourcedGUID and SectionAssociation objects. |

### 5.7.1 SectionAssociationId Attribute Description

**Table 5.9 Description of the 'sourcedGUID' attribute for the 'Association' class.**

| Descriptor | Definition |
|---|---|
| Attribute name | sourcedGUID |
| Data type | SourcedGUID |
| Value space | container |
| Multiplicity | 1 |
| Description | This is the globally unique identifier that has been assigned to the associated SectionAssociation object. Each course section object must have only one SourcedGUID but this may be changed, any number of times, during the object's lifetime. |

## 5.8 CourseTemplate Class Description

The PIM for the CourseTemplate is shown in Figure 5.3.

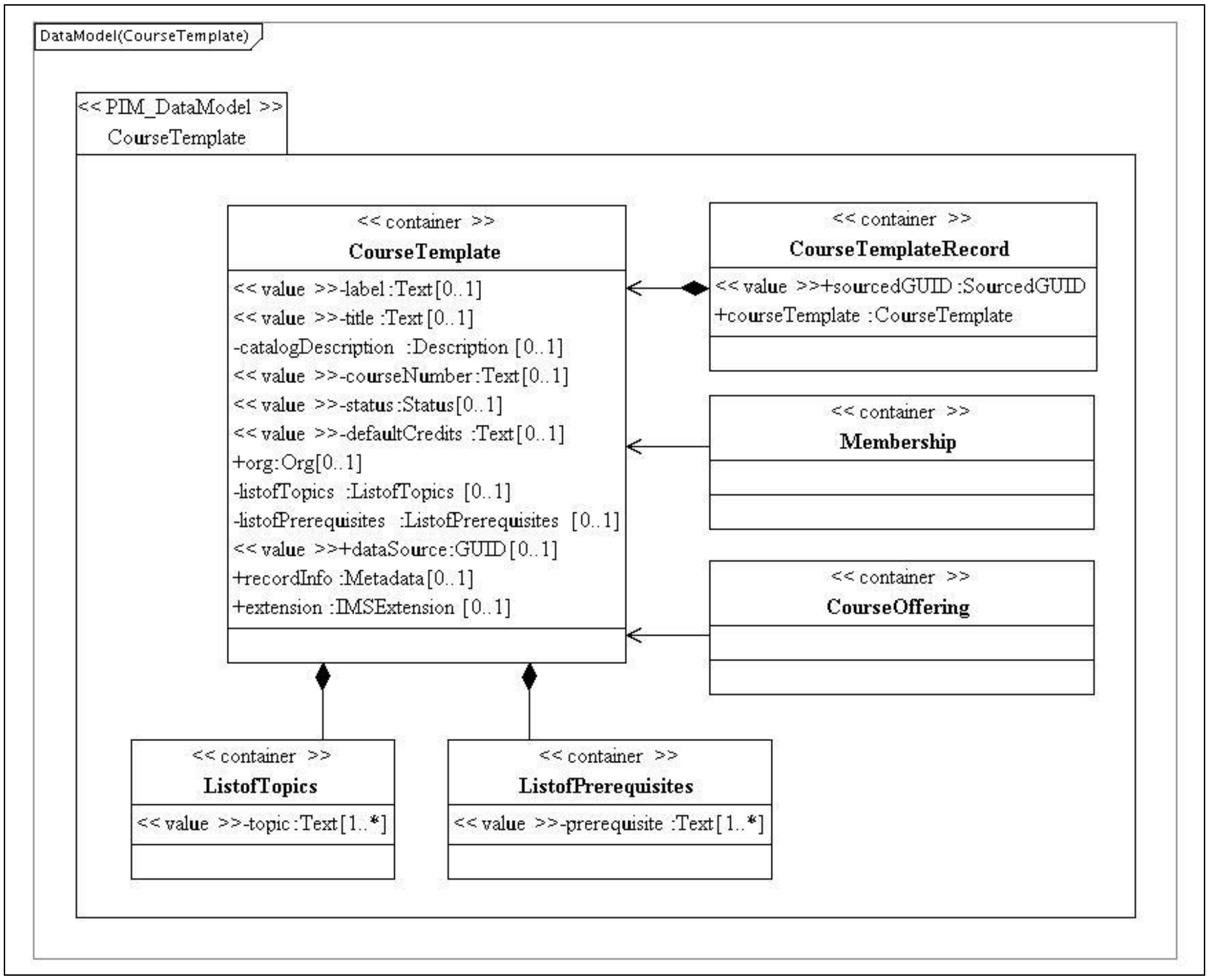

DataModel(CourseTemplate)
<< PIM_DataModel >>
CourseTemplate
<< container >>
CourseTemplate
<< value >>-label :Text [0..1]
<< value >>-title :Text [0..1]
-catalogDescription :Description [0..1]
<< value >>-courseNumber :Text [0..1]
<< value >>-status :Status [0..1]
<< value >>-defaultCredits :Text [0..1]
+org:Org[0..1]
-listofTopics :ListofTopics [0..1]
-listofPrerequisites :ListofPrerequisites [0..1]
<< value >>+dataSource:GUID [0..1]
+recordInfo :Metadata [0..1]
+extension :IMSExtension [0..1]
<< container >>
CourseTemplateRecord
<< value >>+sourcedGUID :SourcedGUID
+courseTemplate :CourseTemplate
<< container >>
Membership
<< container >>
CourseOffering
<< container >>
ListofTopics
<< value >>-topic :Text [1..*]
<< container >>
ListofPrerequisites
<< value >>-prerequisite :Text [1..*]

**Figure 5.3 CourseTemplate class diagram.**

**Table 5.10 Description of the 'CourseTemplate' class.**

| Descriptor | Definition |
|---|---|
| Class name | CourseTemplate |
| Class type | container |
| Multiplicity | 1 |
| Parents | CourseTemplate |
| Children | [ label, title, catalogDescription, courseNumber, status, defaultCredits, org, listofTopics, listofPrerequisites, dataSource, recordInfo, extension ], ordered |
| Description | A CourseTemplate is a general course that exists across terms, semesters, etc. It is an abstract course representation. Examples of instances of CourseTemplates are Biology 101, Mathematics Module 2, etc. |

### 5.8.1 Label Attribute Description

**Table 5.11 Description of the 'label' attribute for the 'CourseTemplate' class.**

| Descriptor | Definition |
|---|---|
| Attribute name | label |
| Data type | Text |
| Value space | Language dependent String [1-255 characters]. The default language is 'en-US'. |
| Multiplicity | 1 |
| Description | A human readable label used to help identify the CourseTemplate. |

### 5.8.2 Title Attribute Description

**Table 5.12 Description of the 'title' attribute for the 'CourseTemplate' class.**

| Descriptor | Definition |
|---|---|
| Attribute name | title |
| Data type | Text |
| Value space | Language dependent String [1-255 characters]. The default language is 'en-US'. |
| Multiplicity | 1 |
| Description | The title of the CourseTemplate. |

### 5.8.3 CatalogDescription Attribute Description

**Table 5.13 Description of the 'catalogDescription' attribute for the 'CourseTemplate' class.**

| Descriptor | Definition |
|---|---|
| Attribute name | catalogDescription |
| Data type | Description |
| Value space | container |
| Multiplicity | 1 |
| Description | This is a description of the CourseTemplate. Several forms of description can be made i.e., short, long or full. The full description can include multimedia materials. |

### 5.8.4 CourseNumber Attribute Description

**Table 5.14 Description of the 'courseNumber' attribute for the 'CourseTemplate' class.**

| Descriptor | Definition |
|---|---|
| Attribute name | courseNumber |
| Data type | Text |
| Value space | Language dependent String [1-255 characters]. The default language is 'en-US'. |
| Multiplicity | 1 |
| Description | The course number e.g., Biology 101. In general this number is not just a numeric value. |

### 5.8.5 Status Attribute Description

**Table 5.15 Description of the 'status' attribute for the 'CourseTemplate' class.**

| Descriptor | Definition |
|---|---|
| Attribute name | status |
| Data type | Enumerated vocabulary. |
| Value space | Vocabulary-based. The core vocabulary is given in Appendix B. |
| Multiplicity | 1 |
| Description | The data that is used to describe the status of the instance of the CourseTemplate. The status of a CourseTemplate will change during its operational lifetime. The value space for this vocabulary is approved by IMS GLC and made available to the public as defined in [SDN11, 06]. The syntax and semantics of the approved list of terms shall be supported by all software components implementing this Information Model.<br>The value space for the vocabulary may be extended. Such extending expressions may be created and used only when no approved IMS GLC value satisfies the expressive need of an implementing community to define the shape of a collection. |

### 5.8.6 DefaultCredits Attribute Description

**Table 5.16 Description of the 'defaultCredits' attribute for the 'CourseTemplate' class.**

| Descriptor | Definition |
|---|---|
| Attribute name | defaultCredits |
| Data type | Text |
| Value space | Language dependent String [1-2047 characters]. The default language is 'en-US'. |
| Multiplicity | 0..1 |
| Description | The default credits set for this CourseTemplate. The unit of credits depends upon the instance. |

### 5.8.7 Org Attribute Description

**Table 5.17 Description of the 'org' attribute for the 'CourseTemplate' class.**

| Descriptor | Definition |
|---|---|
| Attribute name | org |
| Data type | Org |
| Value space | container |
| Multiplicity | 0..1 |
| Description | The organization that has 'ownership' of the CourseTemplate in terms of administering or sponsoring it. |

### 5.8.8 ListofTopics Attribute Description

**Table 5.18 Description of the 'listofTopics' attribute for the 'CourseTemplate' class.**

| Descriptor | Definition |
|---|---|
| Attribute name | listofTopics |
| Data type | ListofTopics |
| Value space | container |
| Multiplicity | 0..1 |
| Description | The list of topics that are covered in this Course.  The coverage of a topic is implementation dependent. |

### 5.8.9 ListofPrerequisities Attribute Description

**Table 5.19 Description of the 'listofPrerequisities' attribute for the 'CourseTemplate' class.**

| Descriptor | Definition |
|---|---|
| Attribute name | listofPrerequisites |
| Data type | ListofPrerequisites |
| Value space | container |
| Multiplicity | 0..1 |
| Description | The list of pre-requisites for entry on courses derived from this Course Template. |

### 5.8.10 DataSource Attribute Description

**Table 5.20 Description of the ‘dataSource’ attribute for the ‘CourseTemplate’ class.**

| Descriptor | Definition |
|---|---|
| Attribute name | dataSource |
| Data type | GUID |
| Value space | See Table 5.1. |
| Multiplicity | 0..1 |
| Description | An identifier of the original source system of the Membership object. |

### 5.8.11 RecordInfo Attribute Description

**Table 5.21 Description of the ‘recordInfo’ attribute for the ‘CourseTemplate’ class.**

| Descriptor | Definition |
|---|---|
| Attribute name | recordInfo |
| Data type | Metadata |
| Value space | container |
| Multiplicity | 0..1 |
| Description | The container for metadata about the CourseTemplate object. No particular form of metadata is mandated. |

### 5.8.12 Extension Attribute Description

**Table 5.22 Description of the ‘extension’ attribute for the ‘CourseTemplate’ class.**

| Descriptor | Definition |
|---|---|
| Attribute name | extension |
| Data type | IMSExtension |
| Value space | container |
| Multiplicity | 0..1 |
| Description | The extension mechanism for the CourseTemplate data model. |

## 5.9 ListofPrerequisites Class Description

**Table 5.23 Description of the ‘ListofPrerequisites’ class.**

| Descriptor | Definition |
|---|---|
| Class name | ListofPrerequisites |
| Class type | container |
| Multiplicity | 0..1 |
| Parents | CourseTemplate |
| Children | [ prerequisite ], ordered |
| Description | The set of pre-requisites that are assigned to a CourseTemplate. |

**Table 5.24 Description of the ‘prerequisite’ attribute for the ‘ListofPrerequisites’ class.**

| Descriptor | Definition |
|---|---|
| Attribute name | prerequisite |
| Data type | Text |
| Value space | Language dependent String [1-255 characters]. The default language is ‘en-US’. |
| Multiplicity | 1..unbounded, unordered |
| Description | The pre-requisite expressed in human readable form. An implementation must be capable of supporting 63 pre-requisites. |

## 5.10 ListofTopics Class Description

**Table 5.25 Description of the 'ListofTopics' class.**

| Descriptor | Definition |
|---|---|
| Class name | ListofTopics |
| Class type | container |
| Multiplicity | 0..1 |
| Parents | CourseTemplate |
| Children | [ topic ], unordered |
| Description | The list of topics that are covered in a Course derived from the CourseTemplate. There is no significance in the order of the topics. |

**Table 5.26 Description of the 'topic' attribute 'ListofTopics' class.**

| Descriptor | Definition |
|---|---|
| Attribute name | topic |
| Data type | Text |
| Value space | Language dependent String [1-255 characters]. The default language is 'en-US'. |
| Multiplicity | 1..unbounded, unordered |
| Description | A topic covered in a Course expressed in human readable form. An implementation must be capable of supporting 63 topics. |

## 5.11 CourseOffering Class Description

The PIM for the CourseOffering is shown in Figure 5.4.

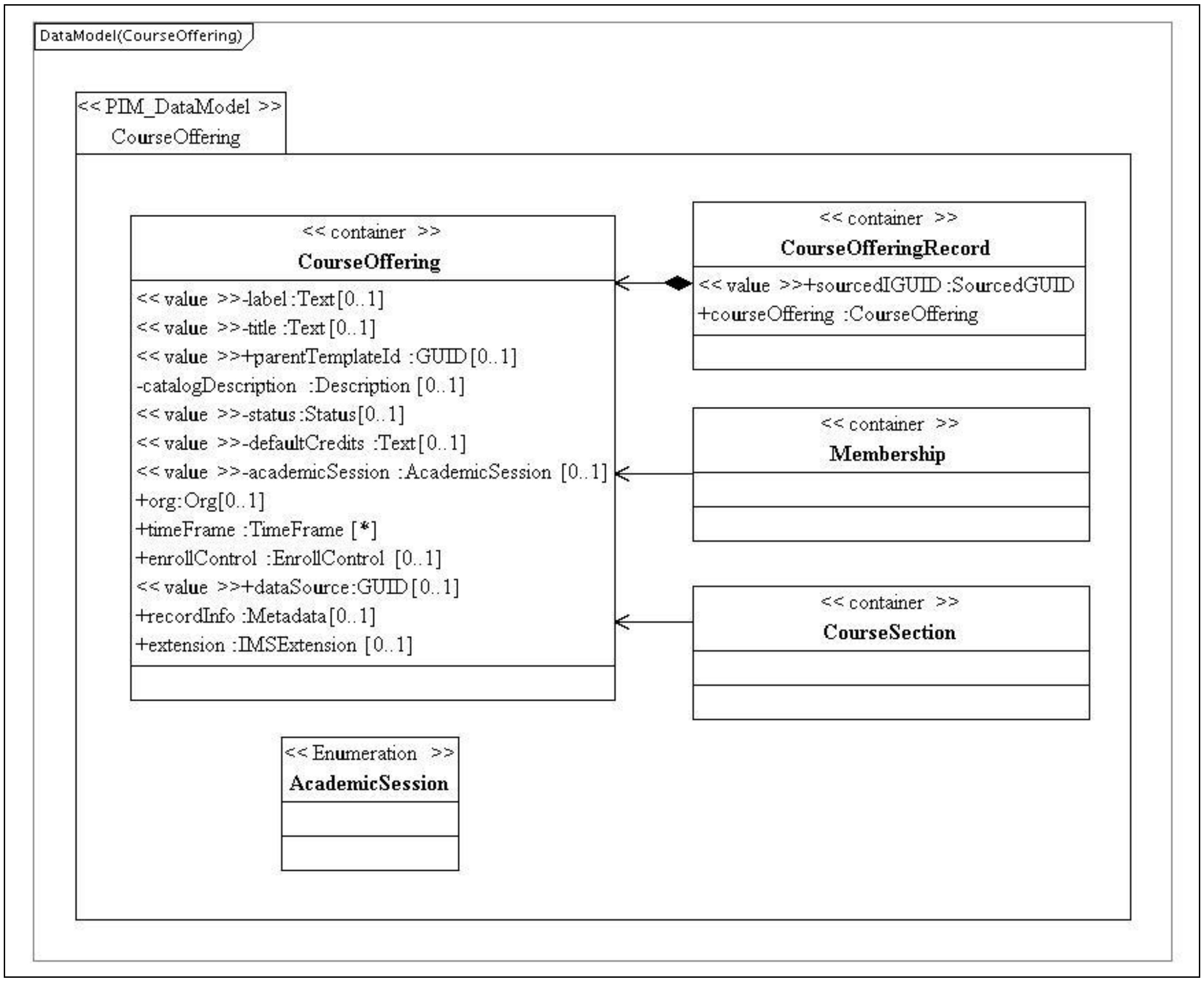

DataModel(CourseOffering)
<< PIM_DataModel >>
CourseOffering
<< container >>
CourseOffering
<< value >>-label :Text [0..1]
<< value >>-title :Text [0..1]
<< value >>+parentTemplateId :GUID [0..1]
-catalogDescription :Description [0..1]
<< value >>-status :Status [0..1]
<< value >>-defaultCredits :Text [0..1]
<< value >>-academicSession :AcademicSession [0..1]
+org:Org [0..1]
+timeFrame :TimeFrame [*]
+enrollControl :EnrollControl [0..1]
<< value >>+dataSource:GUID [0..1]
+recordInfo :Metadata [0..1]
+extension :IMSExtension [0..1]
<< container >>
CourseOfferingRecord
<< value >>+sourcedIGUID :SourcedGUID
+courseOffering :CourseOffering
<< container >>
Membership
<< container >>
CourseSection
<< Enumeration >>
AcademicSession

**Figure 5.4 CourseOffering class diagram.**

**Table 5.27 Description of the CourseOffering class.**

| Descriptor | Definition |
|---|---|
| Class name | CourseOffering |
| Class type | container |
| Multiplicity | 1 |
| Parents | CourseOfferingRecord |
| Children | [ label, title, parentTemplateId, catalogDescription, status, defaultCredits, academicSession, org, timeFrame, enrollControl, dataSource, recordInfo, extension ], ordered |
| Description | A CourseOffering is the occurrence of a course in a specific term, semester, etc. A CourseTemplate can have several CourseOfferings and each CourseOffering can have several CourseSections. If the CourseTemplate instance is English 101 then the CourseOfferings could be English 101 (Semester 1) and English 101 (Semester 2). |

### 5.11.1 Label Attribute Description

**Table 5.28 Description of the 'label' attribute for the 'CourseOffering' class.**

| Descriptor | Definition |
|---|---|
| Attribute name | label |
| Data type | Text |
| Value space | Language dependent String [1-255 characters]. The default language is 'en-US'. |
| Multiplicity | 1 |
| Description | A human readable label used to help identify the CourseOffering. |

### 5.11.2 Title Attribute Description

**Table 5.29 Description of the 'title' attribute for the 'CourseOffering' class.**

| Descriptor | Definition |
|---|---|
| Attribute name | title |
| Data type | Text |
| Value space | Language dependent String [1-255 characters]. The default language is 'en-US'. |
| Multiplicity | 1 |
| Description | The title of the CourseOffering. |

### 5.11.3 ParentTemplateId Attribute Description

**Table 5.30 Description of the 'parentTemplateId' attribute for the 'CourseOffering' class.**

| Descriptor | Definition |
|---|---|
| Attribute name | parentTemplateId |
| Data type | GUID |
| Value space | See Table 5.1. |
| Multiplicity | 0..1 |
| Description | The GUID for the parent CourseTemplate. It should be noted that a CourseOffering may not have a parent CourseTemplate. |

### 5.11.4 CatalogDescription Attribute Description

**Table 5.31 Description of the 'catalogDescription' attribute for the 'CourseOffering' class.**

| Descriptor | Definition |
|---|---|
| Attribute name | catalogDescription |
| Data type | Description |
| Value space | Container |
| Multiplicity | 1 |
| Description | This is a description of the CourseOffering. Several forms of description can be made i.e., short, long or full. The full description can include multimedia materials. |

### 5.11.5 DefaultCredits Attribute Description

**Table 5.32 Description of the 'defaultCredits' attribute for the 'CourseOffering' class.**

| Descriptor | Definition |
|---|---|
| Attribute name | defaultCredits |
| Data type | Text |
| Value space | Language dependent String [1-2047 characters]. The default language is 'en-US'. |
| Multiplicity | 0..1 |
| Description | The default credits set for this CourseOffering. The unit of credits depends upon the implementation. |

### 5.11.6 Status Attribute Description

**Table 5.33 Description of the ‘status’ attribute for the ‘CourseOffering’ class.**

| Descriptor | Definition |
|---|---|
| Attribute name | status |
| Data type | Enumerated vocabulary. |
| Value space | Vocabulary-based. The core vocabulary is given in Appendix B. |
| Multiplicity | 1 |
| Description | The data that is used to describe the status of the instance of the course offering. The status of a course template will change during its operational lifetime. The value space for this vocabulary is approved by IMS GLC and made available to the public as defined in [SDN11, 06]. The syntax and semantics of the approved list of terms shall be supported by all software components implementing this Information Model.<br>The value space for the vocabulary may be extended. Such extending expressions may be created and used only when no approved IMS GLC value satisfies the expressive need of an implementing community to define the shape of a collection. |

### 5.11.7 AcademicSession Attribute Description

**Table 5.34 Description of the ‘academicSession’ attribute for the ‘CourseOffering’ class.**

| Descriptor | Definition |
|---|---|
| Attribute name | academicSession |
| Data type | Enumerated vocabulary. |
| Value space | Implementation dependent. |
| Multiplicity | 1 |
| Description | The data that is used to describe the academic session for the course offering. The value space for this vocabulary is approved by IMS GLC and made available to the public as defined in [SDN11, 06]. The syntax and semantics of the approved list of terms shall be supported by all software components implementing this Information Model.<br>The value space for the vocabulary may be extended. Such extending expressions may be created and used only when no approved IMS GLC value satisfies the expressive need of an implementing community to define the shape of a collection. |

### 5.11.8 Org Attribute Description

**Table 5.35 Description of the 'org' attribute for the 'CourseOffering' class.**

| Descriptor | Definition |
|---|---|
| Attribute name | org |
| Data type | Org |
| Value space | Container |
| Multiplicity | 0..1 |
| Description | The organization that has 'ownership' of the CourseOffering in terms of administering or sponsoring it. |

### 5.11.9 TimeFrame Attribute Description

**Table 5.36 Description of the 'timeFrame' attribute for the 'CourseOffering' class.**

| Descriptor | Definition |
|---|---|
| Attribute name | timeFrame |
| Data type | TimeFrame |
| Value space | container |
| Multiplicity | 0..unbounded, unordered |
| Description | The period when the CourseOffering is available. |

### 5.11.10 EnrollControl Attribute Description

**Table 5.37 Description of the 'enrollControl' attribute for the 'CourseOffering' class.**

| Descriptor | Definition |
|---|---|
| Attribute name | enrollControl |
| Data type | EnrollControl |
| Value space | container |
| Multiplicity | 0..1 |
| Description | Indicates if the enrolment on the CourseOffering is available. |

### 5.11.11 DataSource Attribute Description

**Table 5.38 Description of the ‘dataSource’ attribute for the ‘CourseOffering’ class.**

| Descriptor | Definition |
|---|---|
| Attribute name | dataSource |
| Data type | GUID |
| Value space | See Table 5.1. |
| Multiplicity | 0..1 |
| Description | An identifier of the original source system of the CourseOffering object. |

### 5.11.12 RecordInfo Attribute Description

**Table 5.39 Description of the ‘recordInfo’ attribute for the ‘CourseOffering’ class.**

| Descriptor | Definition |
|---|---|
| Attribute name | recordInfo |
| Data type | Metadata |
| Value space | container |
| Multiplicity | 0..1 |
| Description | The container for metadata about the CourseOffering object. No particular form of metadata is mandated. |

### 5.11.13 Extension Attribute Description

**Table 5.40 Description of the ‘extension’ attribute for the ‘CourseOffering’ class.**

| Descriptor | Definition |
|---|---|
| Attribute name | extension |
| Data type | IMSExtension |
| Value space | container |
| Multiplicity | 0..1 |
| Description | The extension mechanism for the CourseOffering data model. |

## 5.12 CourseSection Class Description

The PIM for the CourseSection is shown in Figure 5.5.

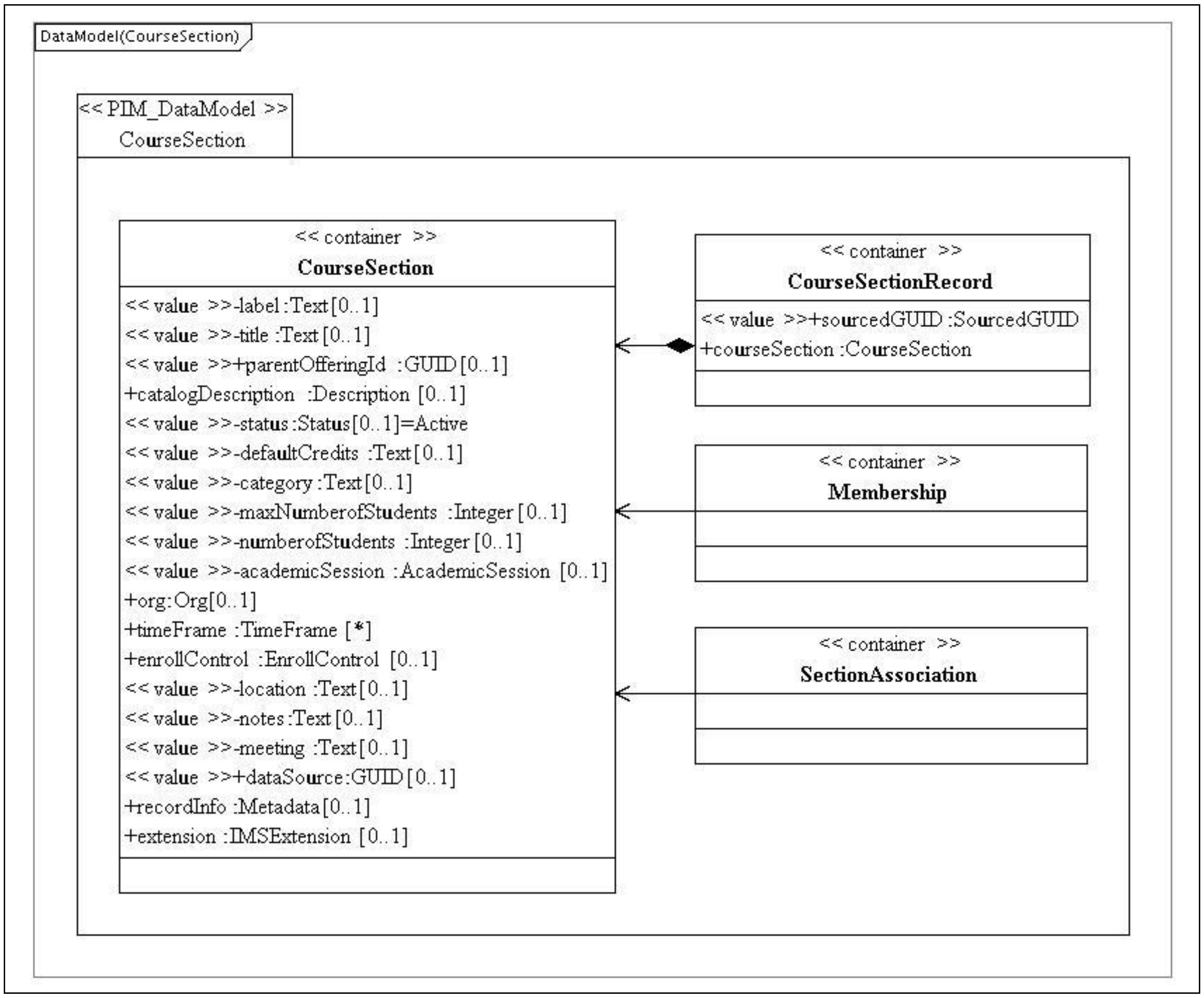

DataModel(CourseSection)
<< PIM_DataModel >>
CourseSection
<< container >>
CourseSection
<< value >>-label :Text [0..1]
<< value >>-title :Text [0..1]
<< value >>+parentOfferingId :GUID [0..1]
+catalogDescription :Description [0..1]
<< value >>-status :Status [0..1]=Active
<< value >>-defaultCredits :Text [0..1]
<< value >>-category :Text [0..1]
<< value >>-maxNumberofStudents :Integer [0..1]
<< value >>-numberofStudents :Integer [0..1]
<< value >>-academicSession :AcademicSession [0..1]
+org:Org [0..1]
+timeFrame :TimeFrame [*]
+enrollControl :EnrollControl [0..1]
<< value >>-location :Text [0..1]
<< value >>-notes :Text [0..1]
<< value >>-meeting :Text [0..1]
<< value >>+dataSource :GUID [0..1]
+recordInfo :Metadata [0..1]
+extension :IMSExtension [0..1]
<< container >>
CourseSectionRecord
<< value >>+sourcedGUID :SourcedGUID
+courseSection :CourseSection
<< container >>
Membership
<< container >>
SectionAssociation

**Figure 5.5 CourseSection class diagram.**

**Table 5.41 Description of the CourseSection class.**

| Descriptor | Definition |
|---|---|
| Class name | CourseSection |
| Class type | container |
| Multiplicity | 1 |
| Parents | CourseSectionRecord |
| Children | [ label, title, parentOfferingId, catalogDescription, status, defaultCredits, category, maxNumberofStudents, numberofStudents, academicSession, org, timeFrame, enrollControl, location, notes, meeting, dataSource, recordInfo, extension ], ordered |
| Description | A CourseSection is a way to represent a group of people associated with a course or class. These groups may include everyone in the class or course, or may be subsets of that whole group. CourseSections may have sub-sections (these are created as separate Group objects linked using the relationship). Examples of a CourseSection are Lecture, Laboratory, Studio, Seminar, etc. There may be several instances of a type of CourseSection e.g., multiple lectures.<br>Several CourseSections can be associated using a SectionAssociation. |

### 5.12.1 Label Attribute Description

**Table 5.42 Description of the ‘label’ attribute for the ‘CourseSection’ class.**

| Descriptor | Definition |
|---|---|
| Attribute name | label |
| Data type | Text |
| Value space | Language dependent String [1-255 characters]. The default language is ‘en-US’. |
| Multiplicity | 1 |
| Description | A human readable label used to help identify the CourseSection. |

### 5.12.2 Title Attribute Description

**Table 5.43 Description of the ‘title’ attribute for the ‘CourseSection’ class.**

| Descriptor | Definition |
|---|---|
| Attribute name | title |
| Data type | Text |
| Value space | Language dependent String [1-255 characters]. The default language is ‘en-US’. |
| Multiplicity | 1 |
| Description | The title of the CourseSection. |

### 5.12.3 ParentOfferingId Attribute Description

**Table 5.44 Description of the ‘parentOfferingId’ attribute for the ‘CourseSection’ class.**

| Descriptor | Definition |
|---|---|
| Attribute name | parentOfferingId |
| Data type | GUID |
| Value space | See table 5.1. |
| Multiplicity | 0..1 |
| Description | The GUID for the parent CourseOffering. It should be noted that a CourseSection may not have a parent CourseOffering. |

### 5.12.4 CatalogDescription Attribute Description

**Table 5.45 Description of the ‘catalogDescription’ attribute for the ‘CourseSection’ class.**

| Descriptor | Definition |
|---|---|
| Attribute name | catalogDescription |
| Data type | Description |
| Value space | container |
| Multiplicity | 1 |
| Description | This is a description of the CourseSection. Several forms of description can be made i.e., short, long or full. The full description can include multimedia materials. |

### 5.12.5 Status Attribute Description

**Table 5.46 Description of the ‘status’ attribute. for the ‘CourseSection’ class**

| Descriptor | Definition |
|---|---|
| Attribute name | status |
| Data type | Enumerated vocabulary. |
| Value space | Vocabulary-based. The core vocabulary is given in Appendix B. |
| Multiplicity | 0..1 |
| Description | The data that is used to describe the status of the instance of the CourseSection. The status of a CourseSection will change during its operational lifetime. The value space for this vocabulary is approved by IMS GLC and made available to the public as defined in [SDN11, 06]. The syntax and semantics of the approved list of terms shall be supported by all software components implementing this Information Model. The value space for the vocabulary may be extended. Such extending expressions may be created and used only when no approved IMS GLC value satisfies the expressive need of an implementing community to define the shape of a collection. |

### 5.12.6 DefaultCredits Attribute Description

**Table 5.47 Description of the ‘defaultCredits’ attribute for the ‘CourseSection’ class.**

| Descriptor | Definition |
|---|---|
| Attribute name | defaultCredits |
| Data type | Text |
| Value space | Language dependent String [1-2047 characters]. The default language is ‘en-US’. |
| Multiplicity | 0..1 |
| Description | The default credits set for this CourseSection. The unit of credits depends upon the instance. |

### 5.12.7 Category Attribute Description

**Table 5.48 Description of the ‘category’ attribute for the ‘CourseSection’ class.**

| Descriptor | Definition |
|---|---|
| Attribute name | category |
| Data type | Enumerated vocabulary. |
| Value space | Implementation dependent. |
| Multiplicity | 0..1 |
| Description | The data that is used to describe the type of CourseSection e.g., Lecture, laboratory, etc. The value space for this vocabulary is approved by IMS GLC and made available to the public as defined in [SDN11, 06]. The syntax and semantics of the approved list of terms shall be supported by all software components implementing this Information Model.<br>The value space for the vocabulary may be extended. Such extending expressions may be created and used only when no approved IMS GLC value satisfies the expressive need of an implementing community to define the shape of a collection. |

### 5.12.8 MaxNumberofStudents Attribute Description

**Table 5.49 Description of the ‘maxNumberofStudents’ attribute. for the ‘CourseSection’ class**

| Descriptor | Definition |
|---|---|
| Attribute name | maxNumberofStudents |
| Data type | Integer |
| Value space | Integer [1-999]. |
| Multiplicity | 1 |
| Description | This defines the maximum number of students that can be enrolled on this CourseSection. |

### 5.12.9 NumberofStudents Attribute Description

**Table 5.50 Description of the ‘numberofStudents’ attribute for the ‘CourseSection’ class.**

| Descriptor | Definition |
|---|---|
| Attribute name | numberofStudents |
| Data type | Integer |
| Value space | Integer: 1-999 |
| Multiplicity | 0..1 |
| Description | The number of students who are enrolled on this instance of the CourseSection. |

### 5.12.10 AcademicSession Attribute Description

**Table 5.51 Description of the ‘academicSession’ attribute for the ‘CourseSection’ class.**

| Descriptor | Definition |
|---|---|
| Attribute name | academicSession |
| Data type | Enumerated vocabulary. |
| Value space | Implementation dependent. |
| Multiplicity | 1 |
| Description | The data that is used to describe the academic session for the course section. The value space for this vocabulary is approved by IMS GLC and made available to the public as defined in [SDN11, 06]. The syntax and semantics of the approved list of terms shall be supported by all software components implementing this Information Model.<br>The value space for the vocabulary may be extended. Such extending expressions may be created and used only when no approved IMS GLC value satisfies the expressive need of an implementing community to define the shape of a collection. |

### 5.12.11 Org Attribute Description

**Table 5.52 Description of the 'org' attribute for the 'CourseSection' class.**

| Descriptor | Definition |
|---|---|
| Attribute name | org |
| Data type | Org |
| Value space | container |
| Multiplicity | 0..1 |
| Description | The organization that has 'ownership' of the CourseSection in terms of administering or sponsoring it. |

### 5.12.12 TimeFrame Attribute Description

**Table 5.53 Description of the 'timeFrame' attribute for the 'CourseSection' class.**

| Descriptor | Definition |
|---|---|
| Attribute name | timeFrame |
| Data type | TimeFrame |
| Value space | container |
| Multiplicity | 0..unbounded, unordered |
| Description | The period when the CourseSection is available. |

### 5.12.13 EnrollControl Attribute Description

**Table 5.54 Description of the 'enrollControl' attribute for the 'CourseSection' class.**

| Descriptor | Definition |
|---|---|
| Attribute name | enrollControl |
| Data type | EnrollControl |
| Value space | container |
| Multiplicity | 0..1 |
| Description | Indicates if the enrolment on the CourseSection is available. |

### 5.12.14 Location Attribute Description

**Table 5.55 Description of the ‘Location’ attribute for the ‘CourseSection’ class.**

| Descriptor | Definition |
|---|---|
| Attribute name | location |
| Data type | Text |
| Value space | Language dependent String [1-255 characters]. The default language is ‘en-US’. |
| Multiplicity | 1 |
| Description | The location of the CourseSection activity e.g., the building and room number of a laboratory. |

### 5.12.15 Notes Attribute Description

**Table 5.56 Description of the ‘notes’ attribute for the ‘CourseSection’ class.**

| Descriptor | Definition |
|---|---|
| Attribute name | notes |
| Data type | Text |
| Value space | Language dependent String [1-255 characters]. The default language is ‘en-US’. |
| Multiplicity | 0..1 |
| Description | Special notes about the CourseSection. |

### 5.12.16 Meeting Attribute Description

**Table 5.57 Description of the ‘meeting’ attribute for the ‘CourseSection’ class.**

| Descriptor | Definition |
|---|---|
| Attribute name | meeting |
| Data type | Text |
| Value space | Language dependent String [1-255 characters]. The default language is ‘en-US’. |
| Multiplicity | 0..1 |
| Description | Human readable information about the CourseSection. |

### 5.12.17 DataSource Attribute Description

**Table 5.58 Description of the ‘dataSource’ attribute for the Membership class for the ‘CourseSection’ class.**

| Descriptor | Definition |
|---|---|
| Attribute name | dataSource |
| Data type | GUID |
| Value space | See Table 5.1. |
| Multiplicity | 0..1 |
| Description | An identifier of the original source system of the CourseSection object. |

### 5.12.18 RecordInfo Attribute Description

**Table 5.59 Description of the ‘recordInfo’ attribute for the ‘CourseSection’ class.**

| Descriptor | Definition |
|---|---|
| Attribute name | recordInfo |
| Data type | Metadata |
| Value space | container |
| Multiplicity | 0..1 |
| Description | The container for metadata about the CourseSection object. No particular form of metadata is mandated. |

### 5.12.19 Extension Attribute Description

**Table 5.60 Description of the ‘extension’ attribute for the ‘CourseSection’ class.**

| Descriptor | Definition |
|---|---|
| Attribute name | extension |
| Data type | IMSExtension |
| Value space | container |
| Multiplicity | 0..1 |
| Description | The extension mechanism for the CourseSection data model. |

## 5.13 SectionAssociation Class Description

The PIM for the SectionAssociation is shown in Figure 5.6.

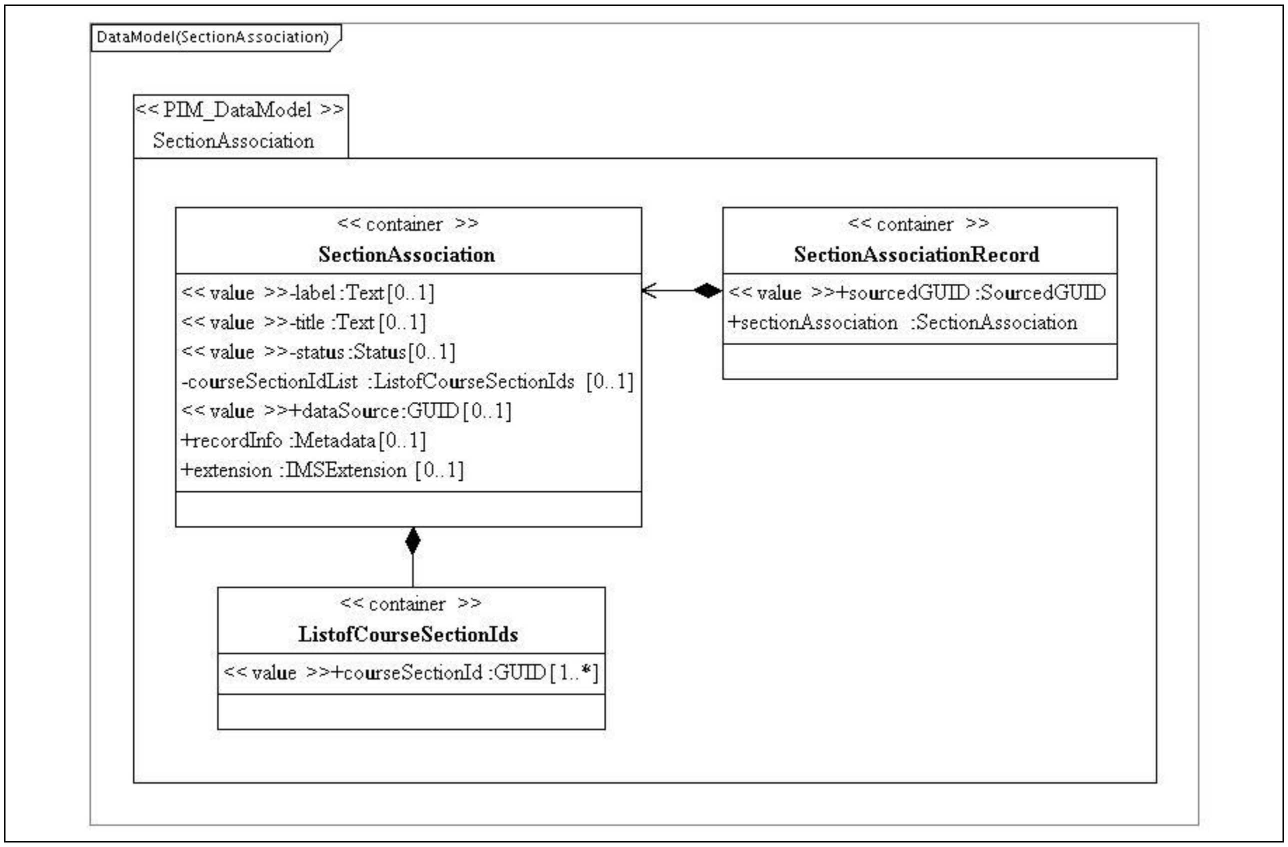

DataModel(SectionAssociation)
<< PIM_DataModel >>
SectionAssociation
<< container >>
SectionAssociation
<< value >>-label :Text [0..1]
<< value >>-title :Text [0..1]
<< value >>-status :Status [0..1]
-courseSectionIdList :ListofCourseSectionIds [0..1]
<< value >>+dataSource :GUID [0..1]
+recordInfo :Metadata [0..1]
+extension :IMSExtension [0..1]
<< container >>
SectionAssociationRecord
<< value >>+sourcedGUID :SourcedGUID
+sectionAssociation :SectionAssociation
<< container >>
ListofCourseSectionIds
<< value >>+courseSectionId :GUID [1..*]

**Figure 5.6 SectionAssociation class diagram.**

**Table 5.61 Description of the SectionAssociation class.**

| Descriptor | Definition |
|---|---|
| Class name | SectionAssociation |
| Class type | container |
| Multiplicity | 1 |
| Parents | SectionAssociationRecord |
| Children | [ label, title, status, courseSectionIdList, Metadata, dataSource, recordInfo, extension ], ordered |
| Description | Defines which CourseSections are associated. |

### 5.13.1 Label Attribute Description

**Table 5.62 Description of the 'label' attribute for the 'SectionAssociation' class.**

| Descriptor | Definition |
|---|---|
| Attribute name | label |
| Data type | Text |
| Value space | Language dependent String [1-255 characters]. The default language is 'en-US'. |
| Multiplicity | 1 |
| Description | A human readable label used to help identify the SectionAssociation. |

### 5.13.2 Title Attribute Description

**Table 5.63 Description of the 'title' attribute for the 'SectionAssociation' class.**

| Descriptor | Definition |
|---|---|
| Attribute name | title |
| Data type | Text |
| Value space | Language dependent String [1-255 characters]. The default language is 'en-US'. |
| Multiplicity | 1 |
| Description | The title of the SectionAssociation. |

### 5.13.3 CourseSectionIdList Attribute Description

**Table 5.64 Description of the 'courseSectionIdList' attribute for the 'SectionAssociation' class.**

| Descriptor | Definition |
|---|---|
| Attribute name | courseSectionIdList |
| Data type | ListofCourseSectionIds |
| Value space | container |
| Multiplicity | 0..1 |
| Description | The container for the list of CourseSection identifiers that are to be associated. |

### 5.13.4 Status Attribute Description

**Table 5.65 Description of the 'status' attribute for the 'SectionAssociation' class.**

| Descriptor | Definition |
|---|---|
| Attribute name | status |
| Data type | Enumerated vocabulary. |
| Value space | Vocabulary-based. The core vocabulary is given in Appendix B. |
| Multiplicity | 0..1 |
| Description | The data that is used to describe the status of the instance of the SectionAssociation. The status of a SectionAssociation will change during its operational lifetime. The value space for this vocabulary is approved by IMS GLC and made available to the public as defined in [SDN11, 06]. The syntax and semantics of the approved list of terms shall be supported by all software components implementing this Information Model.<br>The value space for the vocabulary may be extended. Such extending expressions may be created and used only when no approved IMS GLC value satisfies the expressive need of an implementing community to define the shape of a collection. |

### 5.13.5 DataSource Attribute Description

**Table 5.66 Description of the 'dataSource' attribute for the 'SectionAssociation' class.**

| Descriptor | Definition |
|---|---|
| Attribute name | dataSource |
| Data type | GUID |
| Value space | See Table 5.1. |
| Multiplicity | 0..1 |
| Description | An identifier of the original source system of the SectionAssociation object. |

### 5.13.6 RecordInfo Attribute Description

**Table 5.67 Description of the 'recordInfo' attribute for the 'SectionAssociation' class.**

| Descriptor | Definition |
|---|---|
| Attribute name | recordInfo |
| Data type | Metadata |
| Value space | container |
| Multiplicity | 0..1 |
| Description | The container for metadata about the SectionAssociation object. No particular form of metadata is mandated. |

### 5.13.7 Extension Attribute Description

**Table 5.68 Description of the 'extension' attribute for the 'SectionAssociation' class.**

| Descriptor | Definition |
|---|---|
| Attribute name | extension |
| Data type | IMSExtension |
| Value space | container |
| Multiplicity | 0..1 |
| Description | The extension mechanism for the SectionAssociation data model. |

## 5.14 ListofCourseSectionIds Class Description

**Table 5.69 Description of the ListofCourseSectionIds class.**

| Descriptor | Definition |
|---|---|
| Class name | SectionAssociation |
| Class type | container |
| Multiplicity | 0..1 |
| Parents | SectionAssociation |
| Children | None. |
| Description | The set of CourseSection identifiers that form the SectionAssociation. |

### 5.14.1 CourseSectionId Attribute Description

**Table 5.70 Description of the ‘courseSectionId’ attribute for the ‘ListofCourseSectionIds’ class.**

| Descriptor | Definition |
|---|---|
| Attribute name | courseSectionId |
| Data type | GUID |
| Value space | See Table 5.1. |
| Multiplicity | 1..unbounded, unordered |
| Description | The GUID for the CourseSection that is to be a part of the SectionAssociation. An implementation must support at least 100 instances of this attribute. |

## 5.15 Common Classes Descriptions

The PIM for the common classes is shown in Figure 5.7.

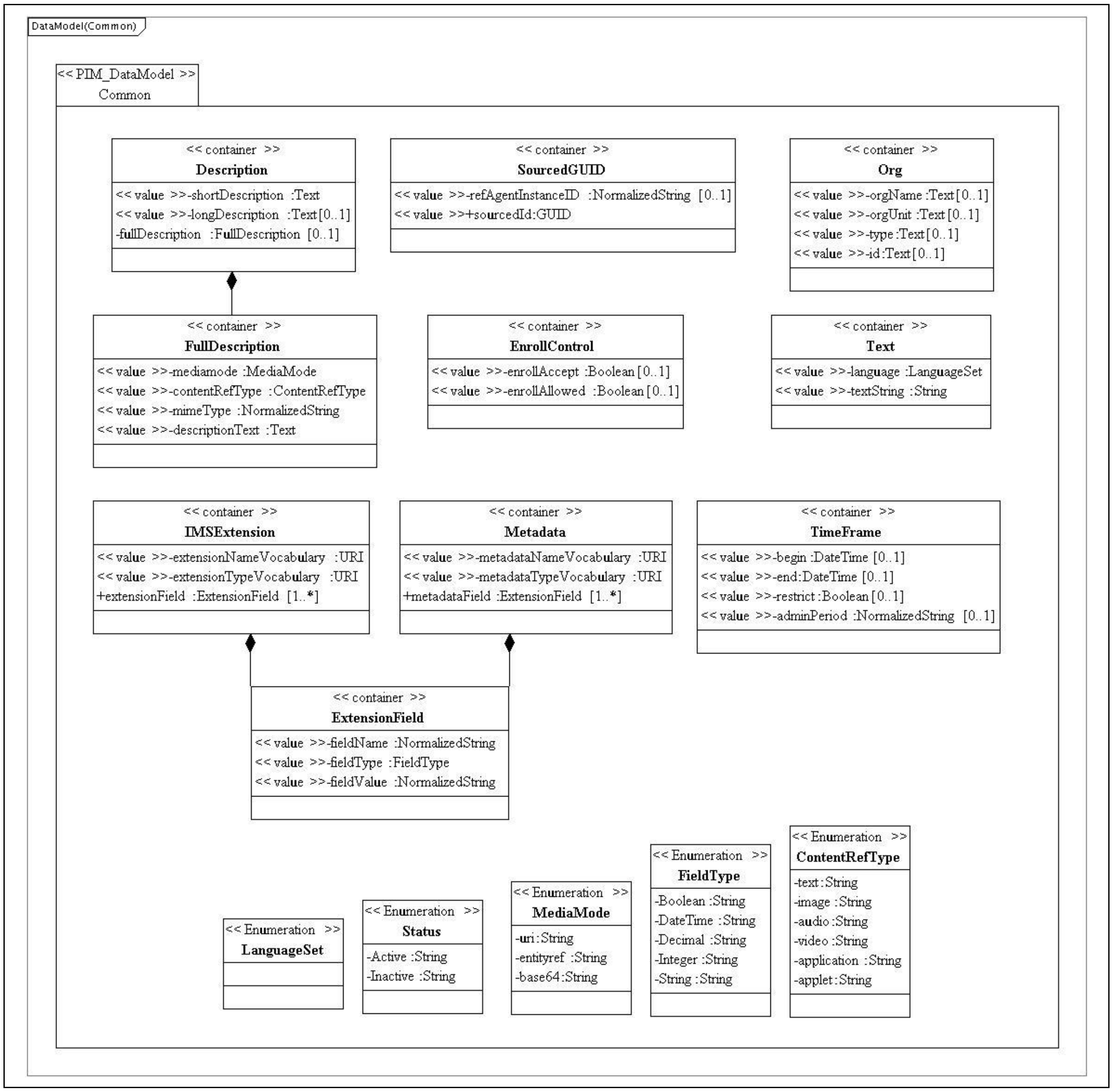

DataModel(Common)
<< PIM_DataModel >> Common
<< container >> Description
<< value >>-shortDescription :Text
<< value >>-longDescription :Text [0..1]
-fullDescription :FullDescription [0..1]
<< container >> SourcedGUID
<< value >>-refAgentInstanceID :NormalizedString [0..1]
<< value >>+sourcedId:GUID
<< container >> Org
<< value >>-orgName :Text [0..1]
<< value >>-orgUnit :Text [0..1]
<< value >>-type :Text [0..1]
<< value >>-id :Text [0..1]
<< container >> FullDescription
<< value >>-mediamode :MediaMode
<< value >>-contentRefType :ContentRefType
<< value >>-mimeType :NormalizedString
<< value >>-descriptionText :Text
<< container >> EnrollControl
<< value >>-enrollAccept :Boolean [0..1]
<< value >>-enrollAllowed :Boolean [0..1]
<< container >> Text
<< value >>-language :LanguageSet
<< value >>-textString :String
<< container >> IMSExtension
<< value >>-extensionNameVocabulary :URI
<< value >>-extensionTypeVocabulary :URI
+extensionField :ExtensionField [1..*]
<< container >> Metadata
<< value >>-metadataNameVocabulary :URI
<< value >>-metadataTypeVocabulary :URI
+metadataField :ExtensionField [1..*]
<< container >> TimeFrame
<< value >>-begin :DateTime [0..1]
<< value >>-end:DateTime [0..1]
<< value >>-restrict :Boolean [0..1]
<< value >>-adminPeriod :NormalizedString [0..1]
<< container >> ExtensionField
<< value >>-fieldName :NormalizedString
<< value >>-fieldType :FieldType
<< value >>-fieldValue :NormalizedString
<< Enumeration >> LanguageSet
<< Enumeration >> Status
-Active :String
-Inactive :String
<< Enumeration >> MediaMode
-uri :String
-entityref :String
-base64 :String
<< Enumeration >> FieldType
-Boolean :String
-DateTime :String
-Decimal :String
-Integer :String
-String :String
<< Enumeration >> ContentRefType
-text :String
-image :String
-audio :String
-video :String
-application :String
-applet :String

**Figure 5.7 Common class diagram.**

### 5.15.1 Org Class Description

**Table 5.71 Description of the Org class.**

| Descriptor | Definition |
|---|---|
| Class name | org |
| Class type | container |
| Children | [ orgName, orgUnit, type, id ], ordered |
| Description | Information about an organization that has ‘ownership’ of a Course or some component of a Course. |

**Table 5.72 Description of the ‘orgName’ attribute for the ‘Org’ class.**

| Descriptor | Definition |
|---|---|
| Attribute name | orgName |
| Data type | Text |
| Value space | Language dependent String [1-255 characters]. The default language is ‘en-US’. |
| Multiplicity | 0..1 |
| Description | The name of the organization. |

**Table 5.73 Description of the ‘orgUnit’ attribute for the ‘Org’ class.**

| Descriptor | Definition |
|---|---|
| Attribute name | orgUnit |
| Data type | Text |
| Value space | Language dependent String [1-255 characters]. The default language is ‘en-US’. |
| Multiplicity | 0..1 |
| Description | Name of the sponsoring or administering unit within the organization. One or more departments or units can sponsor the Group e.g., ‘0-158 – Math Department’. |

**Table 5.74 Description of the ‘type’ attribute for the ‘Org’ class.**

| Descriptor | Definition |
|---|---|
| Attribute name | type |
| Data type | Text |
| Value space | Language dependent String [1-255 characters]. The default language is ‘en-US’. |
| Multiplicity | 0..1 |
| Description | Used to distinguish general categories of the organization e.g., ‘Academic Unit’, ‘HR Department’, etc. |

**Table 5.75 Description of the ‘id’ attribute for the ‘Org’ class.**

| Descriptor | Definition |
|---|---|
| Attribute name | id |
| Data type | Text |
| Value space | Language dependent String [1-255 characters]. The default language is ‘en-US’. |
| Multiplicity | 0..1 |
| Description | Identifier of the organization. If there is a code for the organization, it can be specified separately in this field. |

### 5.15.2 EnrollControl Class Description

**Table 5.76 Description of the EnrollControl class.**

| Descriptor | Definition |
|---|---|
| Class name | EnrollControl |
| Class type | container |
| Children | [ enrollAccept, enrollAllowed ], ordered |
| Description | To control enrolment on the Course component. This control is on the Course component plus the target system. |

**Table 5.77 Description of the ‘enrollAccept’ attribute for the ‘EnrollControl’ class.**

| Descriptor | Definition |
|---|---|
| Attribute name | enrollAccept |
| Data type | Boolean |
| Value space | Enumerated: {true=accept enrolment; false=do not accept enrolment} |
| Multiplicity | 0..1 |
| Description | Indicates if the Course is accepting enrolments. There can be different reasons for a Course being closed e.g., it may be full, it may be cancelled, etc |

**Table 5.78 Description of the ‘enrollAllowed’ attribute for the ‘EnrollControl’ class.**

| Descriptor | Definition |
|---|---|
| Attribute name | enrollAllowed |
| Data type | Boolean |
| Value space | Enumerated: {true=accept enrolment; false=do not accept enrolment} |
| Multiplicity | 0..1 |
| Description | Determines if the target system can enroll people. If ‘false’, then only the source system can enroll people. |

### 5.15.3 TimeFrame Class Description

**Table 5.79 Description of the TimeFrame class for the ‘EnrollControl’ class.**

| Descriptor | Definition |
|---|---|
| Class name | TimeFrame |
| Class type | container |
| Children | [ begin, end, restrict, adminPeriod ], ordered |
| Description | Defines the period for which a particular activity is permitted. |

**Table 5.80 Description of the ‘begin’ attribute for the ‘EnrollControl’ class.**

| Descriptor | Definition |
|---|---|
| Attribute name | begin |
| Data type | DateTime |
| Value space | See Table 5.1. |
| Multiplicity | 0..1 |
| Description | The start date/time of the activity. |

**Table 5.81 Description of the ‘end’ attribute for the ‘EnrollControl’ class.**

| Descriptor | Definition |
|---|---|
| Attribute name | end |
| Data type | DateTime |
| Value space | See Table 5.1. |
| Multiplicity | 0..1 |
| Description | The end date/time of the activity. |

**Table 5.82 Description of the ‘restrict’ attribute for the ‘EnrollControl’ class.**

| Descriptor | Definition |
|---|---|
| Attribute name | restrict |
| Data type | Boolean |
| Value space | Enumerated: {true=restriction is active; false=restriction is not active} |
| Multiplicity | 0..1 |
| Description | Define if the restriction is active or not. |

**Table 5.83 Description of the 'adminPeriod' attribute for the 'EnrollControl' class.**

| Descriptor | Definition |
|---|---|
| Attribute name | adminPeriod |
| Data type | Text |
| Value space | A language dependent String [1-127 characters]. The default language is 'en-US'. |
| Multiplicity | 0..1 |
| Description | A short descriptive name of the period being defined. This should be human readable. |

### 5.15.4 Description Class Description

**Table 5.84 Description of the Description class.**

| Descriptor | Definition |
|---|---|
| Class name | Description |
| Class type | container |
| Children | [ shortDescription, longDescription, fullDescription ], ordered |
| Description | The container for descriptive material about the associated object. The description can take the form of text, image, video, audio, etc. |

**Table 5.85 Description of the 'shortDescription' attribute for the 'Description' class.**

| Descriptor | Definition |
|---|---|
| Attribute name | shortDescription |
| Data type | Text |
| Value space | A language dependent String [1-127 characters]. The default language is 'en-US'. |
| Multiplicity | 1 |
| Description | A short textual description. |

**Table 5.86 Description of the ‘longDescription’ attribute for the ‘Description’ class.**

| Descriptor | Definition |
|---|---|
| Attribute name | longDescription |
| Data type | Text |
| Value space | A language dependent String [1-2095 characters]. The default language is ‘en-US’. |
| Multiplicity | 0..1 |
| Description | A long textual description. |

**Table 5.87 Description of the ‘fullDescription’ attribute for the ‘Description’ class.**

| Descriptor | Definition |
|---|---|
| Attribute name | fullDescription |
| Data type | FullDescription |
| Value space | container |
| Multiplicity | 0..1 |
| Description | The full description consists of the required material types. |

### 5.15.5 FullDescription Class Description

**Table 5.88 Description of the FullDescription class.**

| Descriptor | Definition |
|---|---|
| Class name | FullDescription |
| Class type | container |
| Multiplicity | Description |
| Children | [ mediamode, contentRefType, mimeType, descriptionText ], ordered |
| Description | A full description of the activity, etc. using text, images, etc. |

**Table 5.89 Description of the 'mediaMode' attribute for the 'FullDescription' class.**

| Descriptor | Definition |
|---|---|
| Attribute name | mediaMode |
| Data type | Enumerated |
| Value space | Enumerated as:{ uri, entityref, base64 } |
| Multiplicity | 1 |
| Description | The reference form to the material. |

**Table 5.90 Description of the 'contentRefType' attribute for the 'FullDescription' class.**

| Descriptor | Definition |
|---|---|
| Attribute name | contentRefType |
| Data type | Enumerated |
| Value space | { text, image, audio, video, application, applet } |
| Multiplicity | 1 |
| Description | The type of the material. |

**Table 5.91 Description of the 'mimeType' attribute for the 'FullDescription' class.**

| Descriptor | Definition |
|---|---|
| Attribute name | mimeType |
| Data type | NormalizedString |
| Value space | Normalized String [1..63] characters. |
| Multiplicity | 1 |
| Description | The mime-type for the material. |

**Table 5.92 Description of the ‘descriptionText’ attribute for the ‘FullDescription’ class.**

| Descriptor | Definition |
|---|---|
| Attribute name | descriptionText |
| Data type | Text |
| Value space | A language dependent String [1-1027 characters]. The default language is ‘en-US’. |
| Multiplicity | 1 |
| Description | A textual description of the material. |

### 5.15.6 Text Class Description

**Table 5.93 Description of the ‘Text’ class.**

| Descriptor | Definition |
|---|---|
| Class name | Text |
| Class type | container |
| Children | [ language, textString ] |
| Description | Text to be stored. This is a language/string tuple. |

**Table 5.94 Description of the ‘language’ attribute for the ‘Text’ class.**

| Descriptor | Definition |
|---|---|
| Attribute name | language |
| Data type | Enumerated. |
| Value space | RFC4646 language codes. The default value is: ‘en-US’. |
| Multiplicity | 1 |
| Description | The language for the associated text string. |

**Table 5.95 Description of the ‘text’ attribute for the ‘Text’ class.**

| Descriptor | Definition |
|---|---|
| Attribute name | textString |
| Data type | Text |
| Value space | String [1-4095 characters]. |
| Multiplicity | 1 |
| Description | The container for the string. |

### 5.15.7 Metadata Class Description

The PIM for the Metadata class is shown in Figure 5.7.

**Table 5.96 Description of the Metadata class.**

| Descriptor | Definition |
|---|---|
| Class name | Metadata |
| Class type | container |
| Children | [ metadataNameVocabulary, metadataTypeVocabulary, metadataField ], ordered |
| Description | This is the container for meta-data about the corresponding object. The meta-data entries are supplied using a name/type/value triple based upon external vocabularies. |

**Table 5.97 Description of the ‘metadataNameVocabulary’ attribute for the Metadata class.**

| Descriptor | Definition |
|---|---|
| Attribute name | metadataNameVocabulary |
| Data type | URI |
| Value space | See Table 5.1. |
| Multiplicity | 1 |
| Description | The URI for the vocabulary that is used to define the set of permitted fieldName values. If this is a reference to a VDEX file it is the ‘vocabIdentifier’ of the vocabulary. |

**Table 5.98 Description of the 'metadataTypeVocabulary' attribute for the Metadata class.**

| Descriptor | Definition |
|---|---|
| Attribute name | metadataTypeVocabulary |
| Data type | URI |
| Value space | See Table 5.1. |
| Multiplicity | 1 |
| Description | The URI for the vocabulary that is used to define the set of permitted fieldType values. If this is a reference to a VDEX file it is the 'vocabIdentifier' of the vocabulary. |

**Table 5.99 Description of the 'metadataField' attribute for the Metadata class.**

| Descriptor | Definition |
|---|---|
| Attribute name | metadataField |
| Data type | ExtensionField |
| Value space | container |
| Multiplicity | 1..unbounded, unordered |
| Description | The container for the triples that are used to define each extension data element. |

### 5.15.8 IMSExtension Class Description

The PIM for the IMSExtension class is shown in Figure 5.7.

**Table 5.100 Description of the IMSExtension class.**

| Descriptor | Definition |
|---|---|
| Class name | IMSExtension |
| Class type | container |
| Children | [ extensionNameVocabulary, extensionTypeVocabulary, extensionField ], ordered |
| Description | The container for the extension of the Group data model. |

**Table 5.101 Description of the ‘extensionNameVocabulary’ attribute for the IMSExtension class.**

| Descriptor | Definition |
|---|---|
| Attribute name | extensionNameVocabulary |
| Data type | URI |
| Value space | See Table 5.1. |
| Multiplicity | 1 |
| Description | The URI for the vocabulary that is used to define the set of permitted fieldName values. If this is a reference to a VDEX file it is the ‘vocabIdentifier’ of the vocabulary. |

**Table 5.102 Description of the ‘extensionTypeVocabulary’ attribute for the IMSExtension class.**

| Descriptor | Definition |
|---|---|
| Attribute name | extensionTypeVocabulary |
| Data type | URI |
| Value space | See Table 5.1. |
| Multiplicity | 1 |
| Description | The URI for the vocabulary that is used to define the set of permitted fieldType values. If this is a reference to a VDEX file it is the ‘vocabIdentifier’ of the vocabulary. |

### 5.15.9 ExtensionField Class Description

The PIM for the ExtensionField class is shown in Figure 5.7.

**Table 5.103 Description of the ExtensionField class.**

| Descriptor | Definition |
|---|---|
| Class name | ExtensionField |
| Class type | container |
| Children | None. |
| Description | The container for each triple that describes an extension field. Each triple consists of field name, field type and field value. |

**Table 5.104 Description of the ‘fieldName’ attribute for the ‘ExtensionField’ class.**

| Descriptor | Definition |
|---|---|
| Attribute name | fieldName |
| Data type | NormalizedString |
| Value space | A language dependent String [1-127 characters]. The default language is ‘en-US’. |
| Multiplicity | 1 |
| Description | The container for the name of the extension field. This is used to identify the full triple. |

**Table 5.105 Description of the ‘fieldType’ attribute for the ‘ExtensionField’ class.**

| Descriptor | Definition |
|---|---|
| Attribute name | fieldType |
| Data type | Enumerated vocabulary. |
| Value space | Vocabulary-based. The core vocabulary is given in Appendix B. |
| Multiplicity | 1 |
| Description | This defines the data-type for the extension triple. The value space for this vocabulary is approved by IMS GLC and made available to the public as defined in [SDN11, 06]. The syntax and semantics of the approved list of terms shall be supported by all software components implementing this Information Model.<br>The value space for the vocabulary may be extended. Such extending expressions may be created and used only when no approved IMS GLC value satisfies the expressive need of an implementing community to define the shape of a collection. |

**Table 5.106 Description of the ‘fieldValue’ attribute for the ‘ExtensionField’ class.**

| Descriptor | Definition |
|---|---|
| Attribute name | fieldValue |
| Data type | NormalizedString |
| Value space | A language dependent String [1-127 characters]. The default language is ‘en-US’. |
| Multiplicity | 1 |
| Description | The container for the data value itself. This is stored as a string but should be interpreted as per the data-type defined in the ‘fieldType’ part of the triple. |

### 5.15.10 SourcedGUID Class Description

**Table 5.107 Description of the SourcedGUID class.**

| Descriptor | Definition |
|---|---|
| Class name | SourcedGUID |
| Class type | container |
| Children | [ refAgentInstanceID, sourcedId ], ordered |
| Description | This is a structured GUID that consists of an instance identifier and a sourcedId. |

**Table 5.108 Description of the 'refAgentInstanceID' attribute for the SourcedGUID class.**

| Descriptor | Definition |
|---|---|
| Attribute name | refAgentInstanceID |
| Data type | NormalizedString |
| Value space | Normalized string [1..31 characters]. |
| Multiplicity | 0..1 |
| Description | This is an instance identifier used to differentiate, if necessary, between multiple end system reference agents. |

**Table 5.109 Description of the 'sourcedId' attribute for the SourcedGUID class.**

| Descriptor | Definition |
|---|---|
| Attribute name | sourcedId |
| Data type | GUID |
| Value space | See Table 5.1. |
| Multiplicity | 1 |
| Description | The sourcedId for the object. This should be a GUID. |

# 6 Extending and Profiling the Service

## 6.1 Proprietary Extensions

Proprietary extensions of the service are based upon two approaches:

a) The extension of the data models being manipulated by the current set of operations;

b) The inclusion of new operations to support new proprietary functionality.

It is NOT permitted to change the behavior of the current set of operations. Such changes MUST be supported by the creation of new operations.

### 6.1.1 Proprietary Operations

The definition of new operations should follow the same format as adopted herein. The new operations should be defined using a new interface type. Every operation must result in the return of a status code that describes the final state of the request on the target end system.

An example of creating such an extension is given in the accompanying Best Practices document [LIS, 11c].

### 6.1.2 Proprietary Data Elements

Extensions to the data model are only permitted where the *IMSExtension* class is available. Within the Course data model only the ‘CourseTemplate’, ‘CourseOffering’, ‘CourseSection’ and ‘SectionAssociation’ classes can be extended. The extension takes the form of a Name/Type/Value triple. Many extension fields can be added but hierarchical structures must be emulated using the appropriate delimited notation in the ‘Name’ field. This triple consists of:

- Name – the name assigned to the extension field (this is a string that can support any naming convention);
- Type – the data-type that is to be used for the value (this is used for interpreting the associated value);
- Value – the data value for the extension (the value is supplied as a string).

## 6.2 Profiling the Service

This Service can be profiled. In general, Profiling is used to:

a) Refine which Interfaces are used and which operations are supported for each Interface;

b) Refine the data models (see the IMS GLC Application Profiling guidelines for more details on how data models can be profiled [APG, 05a][APG, 05b]).

Valid Profiles must be restrictive i.e., optional features can be removed or constraints increased but new features must not be added. A Profile of this service is made by annotating the UML supplied with the documentation for the specification.

# Appendix A – Service Status Codes

The summary list of status codes that can be returned by the different operations through the StatusInfo object is given in Table A.1. The key to the entries is: 'Y' denotes the code may be returned by that operation. A blank entry means that the code cannot be returned by that operation.

**Table A.1 Status codes for the service operations.**

| CodeMinor Status Code | create[1] | createByProxy[2] | delete[3] | read[4] | update[5] | replace[6] | discovery[7] | changeIdentifier[8] |
|---|---|---|---|---|---|---|---|---|
| 'fullsuccess' | Y | Y | Y | Y | Y | Y | Y | Y |
| 'createsuccess' | | | | | | Y | | |
| 'nosourcedids' | | | | Y | | | Y | |
| 'idallocfail' | | Y | | | | | | |
| 'overflowfail' | Y | Y | | | | | | |
| 'idallocinusefail' | Y | | | | | | | Y |
| 'invaliddata' | Y | Y | | | Y | Y | Y | |
| 'incompletedata' | Y | Y | | Y | Y | Y | | |
| 'toomuchdata' | | | | Y | | | Y | |
| 'partialdatastorage' | Y | Y | | Y | Y | Y | | |
| 'unknownobject' | | | Y | Y | Y | | | Y |
| 'unknownmdvocabulary' | Y | Y | | | Y | Y | | |
| 'unknownvocabulary' | Y | Y | | | Y | Y | | |
| 'unknownquery' | | | | | | | Y | |
| 'deletefailure' | | | Y | | | | | |
| 'targetreadfailure' | | | | Y | | | | |
| 'savepointsyncerror' | | | | Y | | | | |
| 'savepointerror' | | | | Y | | | | |
| 'unknownextension' | Y | Y | | Y | Y | Y | | |
| 'targetisbusy' | Y | Y | Y | Y | Y | Y | Y | Y |
| 'linkfailure' | Y | Y | Y | Y | Y | Y | Y | Y |
| 'unauthorizedrequest' | Y | Y | Y | Y | Y | Y | Y | Y |
| 'unsupported' | Y | Y | Y | Y | Y | Y | Y | Y |

Note:

1. Denotes used by the ‘createCourseTemplate’, ‘createCourseOffering’, ‘createCourseSection’, ‘createSectionAssociation’, ‘createCourseOfferingFromCourseOffering’ and ‘createCourseSectionFromCourseSection’ operations;
2. Denotes used by the ‘createByProxyCourseTemplate’, ‘createByProxyCourseOffering’, ‘createByProxyCourseSection’ and ‘createByProxySectionAssociation’ operations;
3. Denotes used by the ‘deleteCourseTemplate’, ‘deleteCourseOffering’, ‘deleteCourseSection’ and ‘deleteSectionAssociation’ operations;
4. Denotes used by the ‘readCourseTemplate’, ‘readCourseOffering’, ‘readCourseSection’, ‘readSectionAssociation’, ‘readCourseTemplates’, ‘readCourseOfferings’, ‘readCourseSections’, ‘readSectionAssociations’, ‘readCourseTemplatesFromSavePoint’, ‘readCourseOfferingsFromSavePoint’, ‘readCourseSectionsFromSavePoint’, ‘readSectionAssociationsFromSavePoint’, ‘readAllCourseTemplateIds’, ‘readCourseOfferingIdsForCourseTemplate’, ‘readCourseTemplateIdsFromSavepoint’, ‘readAllCourseOfferingIds’, ‘readAllActiveCourseOfferingIdsForAcademicSession’, ‘readCourseOfferingIdsFromSavePoint’ ‘readCourseSectionIdsForCourseOffering’, ‘readAllCourseSectionIds’, ‘readCourseSectionIdsFromSavePoint’ and ‘readSectionAssociationIdsFromSavePoint’ operations;
5. Denotes used by the ‘updateCourseTemplate’, ‘updateCourseOffering’, ‘updateCourseSection’ and ‘updateSectionAssociation’ operations;
6. Denotes used by the ‘replaceCourseTemplate’, ‘replaceCourseOffering’, ‘replaceCourseSection’ and ‘replaceSectionAssociation’ operations;
7. Denotes used by the ‘discoverCourseTemplateIds’, ‘discoverCourseOfferingIds’, ‘discoverCourseSectionIds’ and ‘discoverSectionAssociationIds’ operations;
8. Denotes used by the ‘changeCourseTemplateIdentifier’, ‘changeCourseOfferingIdentifier’, ‘changeCourseSectionIdentifier’ and ‘changeSectionAssociationIdentifier’ operations.

There is a set of status codes that must be supported by each of the Course Management Service operations. These codes are described in Table A.2.

**Table A.2 Common status codes for the service operations.**

| Status Code | Explanation of the Cause of the Code |
|---|---|
| ‘CodeMajor=Failure’<br>‘Severity=Status’<br>‘CodeMinor=unauthorizedrequest’ | The source system is not authorized to make this request of the target. The reason for the refusal can be one of several causes. |
| ‘CodeMajor=Failure’<br>‘Severity=Status’<br>‘CodeMinor=targetisbusy’ | The target end-system received the request but is busy and cannot process the request. The request should be resubmitted. |
| ‘CodeMajor=Failure’<br>‘Severity=Error’<br>‘CodeMinor=linkfailure’ | There has been a failure in the end-to-end system communications mechanism and so the request has not been delivered. |
| ‘CodeMajor=UnsupportedLIS’<br>‘Severity=Status’<br>‘CodeMinor=*’ | This service in LIS is not supported by the target system. Every system that implements any part of the LIS specification **must** return this status code for a service component in LIS that is not supported. |
| ‘CodeMajor=UnsupportedLISoperation’<br>‘Severity=Status’<br>‘CodeMinor=*’ | This operation is not supported by the target system. Every system that implements any part of the LIS specification **must** return this status code for an operation that is not supported in a supported service. |

# Appendix B Vocabularies

## B1 Set of Defined Vocabularies

The vocabularies listed in Table B1.1, B1.2 and B1.3 are the default set maintained under the IMS GLC Vocabulary Registry [SDN11, 06]. It is the responsibility of an implementation to ensure that it is using the correct and latest versions of the vocabulary files. Changes to the default vocabularies are permitted; this results in the creation of a new vocabulary that should be registered with IMS GLC. As part of a profiling process entirely new vocabularies may be defined to replace the default set.

### B1.1 Implementation-dependent Vocabularies

The implementation-dependent vocabularies are identified in Table B1.1.

**Table B1.1 Set of implementation dependent vocabularies.**

| Vocabulary | Description |
|---|---|
| CourseOffering Class<br>‘academicSession’ attribute | The permitted values for the identification of an academic session.<br>This is an implementation dependent vocabulary. |
| CourseSection Class<br>‘category’ attribute | The permitted values for the type of CourseSection.<br>This is an implementation dependent vocabulary. |

### B1.2 LineItemType Vocabulary

The vocabulary for the ‘status’ attribute is listed in Table B1.2.

**Table B1.2 The status attribute external vocabulary.**

| Vocabulary | Description |
|---|---|
| CourseTemplate Class<br>‘status’ attribute<br>CourseOffering Class<br>‘status’ attribute<br>CourseSection Class<br>‘status’ attribute<br>SectionAssociation Class<br>‘status’ attribute | The permitted status values for each component of a Course. The core vocabulary is:<br>• Active (default)<br>• Inactive |

Rule B.2-01: Active – the associated entity is in an active state;

Rule B.2-02: Inactive – the associated entity is in an inactive state.

### B1.3 FieldType Vocabulary

The vocabulary for type of field in ‘fieldType’ is listed in Table B1.3.

**Table B1.3 The fieldType external vocabulary.**

| Vocabulary | Description |
|---|---|
| ExtensionField Class<br>‘fieldType’ attribute | Data types that are permitted for the extension fields. These data-types reflect the permitted types for XML. Enumerated as:<br>• Boolean<br>• DateTime<br>• Integer<br>• Decimal<br>• String |

Rule B.3-01: Boolean – the data-type is equivalent to the definition of a ‘Boolean’ in XML;

Rule B.3-02: DateTime – the data-type is equivalent to the definition of a ‘DateTime’ in XML;

Rule B.3-03: Integer – the data-type is equivalent to the definition of an ‘Integer’ in XML;

Rule B.3-04: Decimal – the data-type is equivalent to the definition of a ‘Decimal’ in XML;

Rule B.3-05: String – the data-type is equivalent to the definition of a ‘String’ in XML.

### B1.4 Language Vocabulary

The language code/country code combination used to identify the language for a piece of text is an enumerated external IMS GLC vocabulary that captures the full set of entries from RFC4646.

## B2 Using Vocabularies for the Metadata Class

The Metadata class consists of attributes:

- metadataNameVocabulary – identifies the vocabulary that contains the reference set of fieldName values for the meta-data[2];
- metadataTypeVocabulary – identifies the vocabulary that contains the reference set of fieldType values for the meta-data. The value for this attribute is the same as the ‘vocabIdentifier’ of the VDEX instance for this vocabulary;
- metadataField – contains the set of triples (fieldName/fieldType/fieldValue) for each meta-data entry.

The value in the ‘fieldName’ must be from the vocabulary (identified using the metadataNameVocabulary attribute). The value in the ‘fieldType’ must be from the external vocabulary containing the permitted set of external field types (as listed in the metadataTypeVocabulary attribute). The value in the ‘fieldValue’ is the metadata value itself. Nested values are possible using a dot notation in the ‘fieldName’ e.g., for LOM this could be ‘general.keyword.string’, etc.

[2] The corresponding vocabulary must be defined. It is recommended that the vocabulary registered with IMS GLC made available as a VDEX file. If the vocabulary is defined as a VDEX file then the value for ‘metadataNameVocabulary’ should be the ‘vocabIdentifier’ of the VDEX instance.

## B3 Using Vocabularies for the IMSExtension Class

The IMSExtension class consists of attributes:

- extensionNameVocabulary[3] – identifies the vocabulary that contains the reference set of fieldName values for the extension;
- extensionTypeVocabulary – identifies the vocabulary that contains the reference set of fieldType values for the extension. The value for this attribute is the same as the 'vocabIdentifier' of the VDEX instance for this vocabulary;
- extensionField – contains the set of triples (fieldName/fieldType/fieldValue) for each extension.

The value in the 'fieldName' must be from the vocabulary (identified using the extensionNameVocabulary attribute). The value in the 'typeType' must be from the external vocabulary containing the permitted set of external field types (as listed in the extensionTypeVocabulary attribute). The value in the 'fieldValue' is the extension value itself. Nested values are possible using a dot notation in the 'fieldName' cf. for meta-data.

[3] The corresponding vocabulary must be defined. It is recommended that the vocabulary registered with IMS GLC made available as a VDEX file. If the vocabulary is defined as a VDEX file then the value for 'extensionNameVocabulary' should be the 'vocabIdentifier' of the VDEX instance.

# Appendix C – File-based Data Exchange

The IMS GLC Bulk Data Exchange Management Service [BDEMS, 11] is used to exchange bulk Outcomes and related information. The Course information is exchanged by placing multiple *CourseTemplateRecord*, *CourseOfferingRecord*, *CourseSectionRecord* and *SectionAssociationRecord* structures within a bulk data container. Figure C1.1 shows the key class structure and the associated definitions are provided in Section 5 of this document.

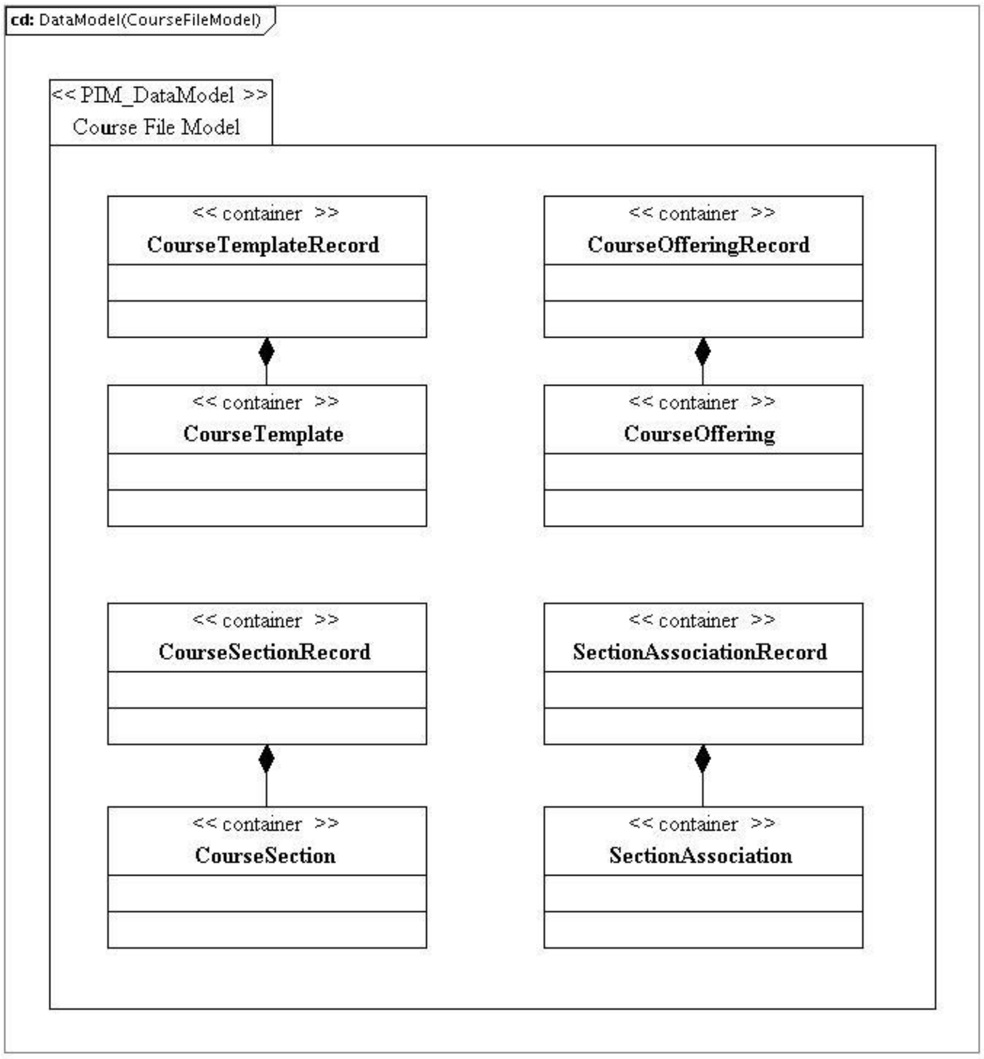


**Figure C.1 Course class diagram for file-based data exchange.**

Note that separate binding instance validation files will be used for containing the description of CourseTemplateRecords, CourseOfferingRecords, CourseSectionRecords and SectionAssociationRecords within a file. This ensures that only the required data structures are contained in the corresponding binding validation files.

# About This Document

| | |
|---|---|
| **Title:** | IMS GLC Course Management Service Information Model |
| **Editor:** | Colin Smythe (IMS) |
| **Co-chairs:** | Linda Feng (Oracle) and Bill Lee (Desire2learn) |
| **Version:** | 1.0 |
| **Version Date**: | 30 June 2010 |
| **Release**: | Final 1.0 |
| **Status:** | **Final Release** |
| **Summary:** | This document contains the IMS GLC Course Management Service v1.0 Information Model. This service is used to exchange information about courses. The conceptual data model for a course consists of templates, offerings, sections and associations. The business transactions include create, delete, read, update, replace and simple discovery for each of the four course components. This document contains the definition of the abstract application-programming interface for the Course Management Service. |
| **Revision Information:** | Original Release. |
| **Purpose:** | This document is made available for adoption by the public community at large. |
| **Document Location:** | http://www.imsglobal.org/lis/ |

## List of Contributors

The following individuals contributed to the development of this document:

| | | | |
|---|---|---|---|
| Kerry Blinco | DEEWR (Australia) | Zack Leavitt | Pearson (USA) |
| Kirk Bunte | SungardHE (USA) | Bill Lee | Desire2Learn (Canada) |
| Angus Chan | Desire2Learn (Canada) | Richard Moon | SungardHE (USA) |
| Adam Cooper | JISC/JISC-CETIS (UK) | Mike Parkhill | Desire2Learn (Canada) |
| Michael Feldstein | Oracle (USA) | Colin Smythe | IMS GLC (UK) |
| Linda Feng | Oracle (USA) | Reinhold Staudinger | Blackboard (USA) |
| Chris Hatton | Pearson (USA) | Nick Terrible | University of Wisconsin (USA) |
| Jon Fontaine | Blackboard (USA) | Jason Zhong | SungardHE (USA) |
| Karen Kuffner | University of Michigan (USA) | | |

# Revision History

| Version No. | Release Date | Comments |
|---|---|---|
| Final Release v1.0 | 30 June 2011 | The first formal release of the Final Release version of this document. |
| | | |
| | | |

# Index