IMS GLC Learning Information Services Specification

Version 2.0

Final Release
Version 1.0

Date Issued: 30 June 2011

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

IPR and Distribution Notices
Recipients of this document are requested to submit, with their comments, notification of any relevant patent claims or other intellectual property rights of which they may be aware that might be infringed by any implementation of the specification set forth in this document, and to provide supporting documentation.

IMS takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Information on IMS's procedures with respect to rights in IMS specifications can be found at the IMS Intellectual Property Rights web page: http://www.imsglobal.org/ipr/imsipr_policyFinal.pdf.

Copyright © 2011 IMS Global Learning Consortium. All Rights Reserved.

Use of this specification to develop products or services is governed by the license with IMS found on the IMS website: http://www.imsglobal.org/license.html.

Permission is granted to all parties to use excerpts from this document as needed in producing requests for proposals.

The limited permissions granted above are perpetual and will not be revoked by IMS or its successors or assigns.

THIS SPECIFICATION IS BEING OFFERED WITHOUT ANY WARRANTY WHATSOEVER, AND IN PARTICULAR, ANY WARRANTY OF NONINFRINGEMENT IS EXPRESSLY DISCLAIMED. ANY USE OF THIS SPECIFICATION SHALL BE MADE ENTIRELY AT THE IMPLEMENTER'S OWN RISK, AND NEITHER THE CONSORTIUM, NOR ANY OF ITS MEMBERS OR SUBMITTERS, SHALL HAVE ANY LIABILITY WHATSOEVER TO ANY IMPLEMENTER OR THIRD PARTY FOR ANY DAMAGES OF ANY NATURE WHATSOEVER, DIRECTLY OR INDIRECTLY, ARISING FROM THE USE OF THIS SPECIFICATION.

© 2011 IMS Global Learning Consortium, Inc.
All Rights Reserved.

The IMS Logo is a trademark of the IMS Global Learning Consortium Inc.

Executive Summary

The Learning Information Services (LIS) specification is the definition of how systems manage the exchange of information that describes people, groups, memberships, courses and outcomes within the context of learning. The LIS v2.0 specification supersedes the IMS GLC Enterprise Services v1.0 specification. The LIS specification is based upon the aggregation of the Person Management, Group Management, Membership Management, Course Management, Outcomes Management and the Bulk Data Exchange Management Services specifications. The LIS v2.0 can be implemented using a Web Services infrastructure (based upon a SOAP/http transport mechanism).

An implementation is not required to support each and every service. Neither is an implementation required to support each and every operation. The specific requirements are defined in the corresponding profile. Interoperability is supported between systems that implement the same profile. Cross-profile interoperability may occur but this is a by-product and should NOT be used as the basis for any system realization. One of the outputs of the LIS specification is the set of Web Services Description Language/XML Schema Definition (WSDL/XSD) binding files. Each service has its own set of WSDL/XSD files. It is these files that are used by code-generation tools to create the source code that handles the SOAP messages and XML data structures. Some small changes are required to the WSDL files to map the SOAP messages to the actual server-based implementation of the Web Service.

The LIS documentation set consists of:

- Information Models – these documents contain the normative description of the various service definitions, data structures and their relationships. These descriptions use the Unified Modeling Language (UML). Each of the six services has its own Information Model;
- WSDL Bindings – these documents contain the Platform Specific Model (PSM) for the service. This PSM has been transformed into the corresponding WSDL/XSD files using the IMS GLC Binding Auto-generation Tool-kit (I-BAT). The Binding document describes the underlying structure of the WSDL/XSD, the associated vocabulary files and the formats of the corresponding SOAP messages;
- Best Practice & Implementation Guide – this document is intended to provide vendors with an overall understanding of the IMS GLC LIS Specification, the relationship of the specification with other IMS GLC specifications, and a best practices guide derived from experiences of those using the specification. The guide also includes several examples that describe how vendors can make the best use of the IMS LIS Specification;
- Core Profile & Conformance Specification – the Core Profile defines the minimal subset of the functionality that must be supported by systems developed for deployment between a Student Information System and a Learning Management System. This Profile (there is a Core plus several Additions) defines the set of operations and data models that must be supported by the systems implementing the set of services within the LIS. A system can support greater functionality but there is no guarantee of interoperability for those extra features. Interoperability is only guaranteed for the functionality described in the Core Profile.

Table of Contents

EXECUTIVE SUMMARY 2

LIST OF FIGURES 5

LIST OF TABLES 5

1 INTRODUCTION 6

1.1 LEARNING INFORMATION SERVICE SYSTEMS 6

1.2 THE SCOPE AND CONTEXT 6

1.3 STRUCTURE OF THIS DOCUMENT 6

1.4 NOMENCLATURE 7

1.5 REFERENCES 8

2 THE LEARNING INFORMATION SERVICES SPECIFICATION 10

2.1 CORE USE-CASES 10

2.1.1 Management and Manipulate Information about People 10

2.1.2 Management and Manipulate Enrolment of People on Courses 10

2.1.3 Management and Manipulate Organizational Structures 10

2.1.4 Management and Manipulate Course Structure Information 11

2.1.5 Management and Manipulate of Grade Book Information 11

2.1.6 Batch Processing 11

2.2 AN ABSTRACT REPRESENTATION 12

2.3 BUILDING REAL SYSTEMS 12

3 THE HISTORIC PERSPECTIVE 14

3.1 PAST VERSIONS 14

3.2 A SUMMARY OF CHANGES FROM VERSION 1 14

3.2.1 Person Management Service 14

3.2.2 Group Management Service 14

3.2.3 Membership Management Service 15

3.2.4 Course Management Service 15

3.2.5 Outcomes Management Service 16

3.2.6 Bulk Data Exchange Management Service 16

3.3 VERSION 2 AND VERSION 1 COMPATIBILITY 16

4 USING THE NEW VERSION 17

4.1 BUILDING AN INTEROPERABLE SOLUTION 17

4.2 THE CORE PROFILES 18

4.3 CREATING A PROFILE 18

4.4 CONFORMANCE TO A PROFILE 19

5 THE DOCUMENT SET 20

5.1 THE SET OF DOCUMENTS 20

5.1.1 The LIS Specification 20

5.1.2 Information Models 20

5.1.3 WSDL Bindings 20

5.1.4 Best Practice & Implementation Guide 20

5.1.5 Core Profile & Conformance Specification 20

5.1.6 Related Documents ... 20

5.2 THE BINDING FILES ... 21

5.2.1 WSDL Files ... 21

5.2.2 XSD Files ... 21

5.2.3 Vocabulary Files ... 21

5.3 USING THE DOCUMENTS ... 21

APPENDIX A PERSON MANAGEMENT SERVICE 2.0 OVERVIEW ... 23

A1 PMS DESCRIPTION ... 23

A2 PMS OPERATIONS ... 23

APPENDIX B GROUP MANAGEMENT SERVICE 2.0 OVERVIEW ... 25

B1 GMS DESCRIPTION ... 25

B2 GMS OPERATIONS ... 25

APPENDIX C MEMBERSHIP MANAGEMENT SERVICE 2.0 OVERVIEW ... 27

C1 MMS DESCRIPTION ... 27

C2 MMS OPERATIONS ... 27

APPENDIX D COURSE MANAGEMENT SERVICE 1.0 OVERVIEW ... 29

D1 CMS DESCRIPTION ... 29

D2 CMS OPERATIONS ... 31

D2.1 CourseTemplateManager Interface Description ... 31

D2.2 CourseOfferingManager Interface Description ... 32

D2.3 CourseSectionManager Interface Description ... 34

D2.4 SectionAssociationManager Interface Description ... 35

APPENDIX E OUTCOMES MANAGEMENT SERVICE 1.0 OVERVIEW ... 36

E1 OMS DESCRIPTION ... 36

E2 OMS OPERATIONS ... 38

E2.1 LineItemManager Interface Description ... 38

E2.2 ResultManager Interface Description ... 39

E2.3 ResultValueManager Interface Description ... 40

APPENDIX F BULK DATA EXCHANGE MANAGEMENT SERVICE 1.0 OVERVIEW ... 42

F1 BDEMS DESCRIPTION ... 42

F2 BDEMS OPERATIONS ... 42

ABOUT THIS DOCUMENT ... 44

LIST OF CONTRIBUTORS ... 44

REVISION HISTORY ... 45

INDEX ... 46

List of Figures

FIGURE 2.1 SCHEMATIC ARCHITECTURE OF THE LEARNER INFORMATION SERVICES. 12

FIGURE 5.1 SCHEMATIC REPRESENTATION OF THE DOCUMENTATION SET. 21

FIGURE A1.1 PERSONMANAGEMENTSERVICE INTERFACE DEFINITION. 23

FIGURE B1.1 GROUPMANAGEMENTSERVICE INTERFACE DEFINITION. 25

FIGURE C1.1 MEMBERSHIPMANAGEMENTSERVICE INTERFACE DEFINITION. 27

FIGURE D1.1 COURSEMANAGEMENTSERVICE COURSETEMPLATEMANAGER INTERFACE DEFINITION. 29

FIGURE D1.2 COURSEMANAGEMENTSERVICE COURSEOFFERINGMANAGER INTERFACE DEFINITION. 30

FIGURE D1.3 COURSEMANAGEMENTSERVICE COURSESECTIONMANAGER INTERFACE DEFINITION. 30

FIGURE D1.4 COURSEMANAGEMENTSERVICE SECTIONASSOCIATIONMANAGER INTERFACE DEFINITION. 31

FIGURE E1.1 OUTCOMESMANAGEMENTSERVICE LINEITEMMANAGER INTERFACE DEFINITIONS. 36

FIGURE E1.2 OUTCOMESMANAGEMENTSERVICE RESULTMANAGER INTERFACE DEFINITIONS. 37

FIGURE E1.3 OUTCOMESMANAGEMENTSERVICE RESULTVALUEMANAGER INTERFACE DEFINITIONS. 37

FIGURE F1.1 BULKDATAEXCHANGEMANAGEMENTSERVICE INTERFACE DEFINITION. 42

List of Tables

TABLE 3.1 VERSION HISTORY FOR ENTERPRISE/LIS SPECIFICATIONS. 14

TABLE A2.1 SUMMARY OF OPERATIONS FOR PERSONMANAGER. 23

TABLE B2.1 SUMMARY OF OPERATIONS FOR GROUPMANAGER. 25

TABLE C2.1 SUMMARY OF OPERATIONS FOR THE MMS. 27

TABLE D2.1 SUMMARY OF OPERATIONS FOR COURSETEMPLATEMANAGER. 31

TABLE D2.2 SUMMARY OF OPERATIONS FOR COURSEOFFERINGMANAGER. 32

TABLE D2.3 SUMMARY OF OPERATIONS FOR COURSESECTIONMANAGER. 34

TABLE D2.4 SUMMARY OF OPERATIONS FOR SECTIONASSOCIATIONMANAGER. 35

TABLE E2.1 SUMMARY OF OPERATIONS FOR LINEITEMMANAGER. 38

TABLE E2.2 SUMMARY OF OPERATIONS FOR RESULTMANAGER. 39

TABLE E2.3 SUMMARY OF OPERATIONS FOR RESULTVALUEMANAGER. 40

TABLE F2.1 SUMMARY OF OPERATIONS FOR BULKDATAEXCHANGEMANAGER. 42

1 Introduction

1.1 Learning Information Service Systems

The Learning Information Services (LIS) specification is the definition of how systems manage the exchange of information that describes people, groups, memberships, courses and outcomes within the context of learning. The Learning Information Services 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 – Learning Information Services focuses on the exchange of information between Learning Information Services systems. There are no assumptions in the specification on how the data is managed within the Learning Information Services systems;
- Service-oriented – Learning Information Services defines the exchange of information in terms of the services being supplied by the collaboration of the systems;
- Component-based – the Learning Information Services are composed of the Person Management Service (PMS), Group Management Service (GMS), Membership Management Services (MMS), Course Management Service (CMS), Outcomes Management Service (OMS) and the Bulk Data Exchange Management Service (BDEMS);
- Behaviors and Data Models – the Learning Information Services are defined in terms of their 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 Learning Information Services information model is to be defined using the Unified Modeling Language (UML). This enables reliable mapping of the information model into a range of different bindings. The bindings of immediate importance are to the Web Services Description Language (WSDL) and the Lightweight Directory Access Protocol (LDAP);
- Adoption – whenever appropriate, the Learning Information Services specification makes use of other IMS GLC and non-IMS GLC standards and specifications.

1.2 The Scope and Context

This document is the IMS GLC Learning Information Services Specification v2.0; an overview is available in [LIS, 11a]. The Learning Information Services specification supersedes the IMS GLC Enterprise Services specification [ES, 04a]. This specification is based upon the aggregation of the Person Management, Group Management, Membership Management, Course Management, Outcomes Management and the Bulk Data Exchange Management Services specifications, namely:

a) IMS GLC Person Management Service Information Model v2.0 [PMS, 11a];

b) IMS GLC Group Management Service Information Model v2.0 [GMS, 11a];

c) IMS GLC Membership Management Service Information Model v2.0 [MMS, 11a];

d) IMS GLC Course Management Service Information Model v1.0 [CMS, 11a];

e) IMS GLC Outcomes Management Service Information Model v1.0 [OMS, 11a];

f) IMS GLC Bulk Data Exchange Management Services Information Model v1.0 [BDEMS, 11a].

The Learning Information Services Specification v2.0 can be implemented using both a Web Services infrastructure (based upon a SOAP/http transport mechanism) and the LDAP. The Web Service bindings are detailed in the corresponding service binding documents [PMS, 11b], [GMS, 11b], [MMS, 11b], [OMS, 11b], [CMS, 11b], [BDEMS, 11b].

1.3 Structure of this Document

The structure of this document is:

2. The Learning Information Services Specification

   Provides an overview of the specification as a whole and explains how the six component services are orchestrated to provide the full range of services;

3. The Historic Perspective

   Provides the historic context for this specification as it has evolved from a pure data model to this current service description. This includes a summary of the changes from the IMS GLC Enterprise Services v1.0 specification;

4. Using the New Version

   Provides an overview of the various ways in which this specification can be used to support different learning administration requirements;

5. The Document Set

   Lists and explains the set of documents produced as part of the specification. This includes an explanation of the relevant set of IMS GLC Specification Development Notes;

Appendix A: Person Management Service 2.0 Overview

An overview of the operations provided in the Person Management Service;

Appendix B: Group Management Service 2.0 Overview

An overview of the operations provided in the Group Management Service;

Appendix C: Membership Management Service 2.0 Overview

An overview of the operations provided in the Membership Management Service;

Appendix D: Course Management Service 1.0 Overview

An overview of the operations provided in the Course Management Service;

Appendix E: Outcomes Management Service 1.0 Overview

An overview of the operations provided in the Outcomes Management Service;

Appendix F: Bulk Data Exchange Management Service 1.0 Overview

An overview of the operations provided in the Bulk Data Exchange Management Service.

1.4 Nomenclature

BDEMS	Bulk Data Exchange Management Service
CMS	Course Management Service
GMS	Group Management Service
IAF	IMS GLC Abstract Framework
I-BAT	IMS GLC Binding Auto-generation Tool-kit
IMS GLC	IMS Global Learning Consortium Inc.
LDAP	Lightweight Directory Access Protocol
LIP	Learner Information Packaging
LIS	Learning Information Services
LMS	Learning Management System
MMS	Membership Management Service
OMS	Outcomes Management Service
PIM	Platform Independent Model
PMS	Person Management Service
PSM	Platform Specific Model

SIS	Student Information System
UML	Unified Modeling Language
URL	Uniform Resource Locator
VDEX	Vocabulary Definition Exchange
WSDL	Web Services Description Language
XSD	XML Schema Description

1.5 References

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

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

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

[CMS, 11b] *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, 04a] *IMS GLC Enterprise Services Specification Final Release v1.0*, C.Smythe and C.Vento, IMS Global Learning Consortium, June 2004.

[ES, 04b] *IMS GLC Enterprise Services Use-cases Final Release v1.0*, C.Smythe and C.Vento, IMS Global Learning Consortium, June 2004.

[GMS, 04] *IMS GLC Group Management Service Information Model v1.0 Final Release*, C.Smythe and C.Vento, IMS Global Learning Consortium, June 2004.

[GMS, 11a] *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.

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

[GWS, 06a] *IMS GLC General Web Services Base Profile Final Release*, C.Schroeder, J.Simon and C.Smythe, V1.0, IMS Global Learning Consortium, January 2006.

[GWS, 06b] *IMS GLC General Web Services WSDL Binding Guidelines Final Release*, C.Schroeder, J.Simon and C.Smythe, V1.0, IMS Global Learning Consortium, January 2006.

[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 Specification Overview Final Release v1.0*, L.Feng, W.Lee and C.Smythe, IMS Global Learning Consortium, June 2011.

[LIS, 11b] *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.

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

[MMS, 04] *IMS GLC Membership Management Service Information Model v1.0 Final Release*, C.Smythe and C.Vento, IMS Global Learning Consortium, June 2004.

[MMS, 11a] *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.

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

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

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

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

[PMS, 11b] *IMS GLC Person Management Service v2.0 WSDL Binding v1.0 Final Release*, L.Feng, W.Lee and C.Smythe, IMS Global Learning Consortium, June 2011.

[SAN11, 06] IMS GLC Profile Definition, Registration and Maintenance Procedures, SAN-11, C.Smythe, IMS Global Learning Consortium, November 2006.

[SDN07, 06] *IMS GLC UML Profile for Platform Independent Model Descriptions of Specifications*, SDN-07, C.Smythe, IMS Global Learning Consortium, October 2006.

[SDN08, 06] *IMS GLC UML Profile for Platform Specific Model Descriptions of Specification Bindings*, SDN-08, C.Smythe, IMS Global Learning Consortium, October 2006.

[SDN11, 06] IMS GLC Vocabulary Definitions, Registration and Maintenance Procedures, SDN-11, C.Smythe, IMS Global Learning Consortium, July 2006.

[VDEX, 04] IMS GLC Vocabulary Definition Exchange Information Model v1.0, A.Cooper, IMS Global Learning Consortium, February 2004. http://www.imsglobal.org/vdex/

2 The Learning Information Services Specification

2.1 Core Use-cases

The core set of use-cases addressed by this specification is:

- Manage and manipulate information about People – to provide data exchange about people who are participating in learning;
- Manage and manipulate enrolment of People in Courses – to control the exchange of information for people attending courses;
- Manage and manipulate organizational structures – to control the exchange of information for people attending courses;
- Manage and manipulate Course structure information – to provide data exchange for information about taught courses;
- Manage and manipulate of Grade-book information – to provide data exchange for outcomes information;
- Batch processing – to provide initialization and synchronization transfer of very large amounts of data.

2.1.1 Management and Manipulate Information about People

People undertake learning and as such attend, or are members of, courses, undertake assessment and obtain grades, and undertake other groups of activities. The specific set of operational needs is:

- Initialize Person, Organization Structure, Enrolment Data;
- Synchronize Person, Organization Structure, Enrolment Data;
- Create Person;
- Change Person Information;
- Update Authentication Credentials for a Person;
- Update Authentication Credentials for all Persons;
- Get All New Persons;
- Get Updated Person Information;
- Get Deleted Persons.

2.1.2 Management and Manipulate Enrolment of People on Courses

The specific set of operational needs is:

- Enroll a Person in a Course Template, Course Offering and Course Section;
- Un-enroll a Person in a Course Template, Course Offering and Course Section;
- Change the Role of a Person in a Course Template, Course Offering and Course Section;
- Get All Enrollment Information for a Person;
- Get All Enrollment Information for All Persons.

2.1.3 Management and Manipulate Organizational Structures

The specific set of operational needs is:

- Create a Parent/Child Relationship in an Organizational Structure;
- Delete a Parent/Child Relationship in an Organizational Structure;
- Get All Persons Enrolled in an Organizational Structure Entity;
- Get All Enrollment Information for an Organizational Structure Entity;
- Use a Learning Context for Several Administrative Contexts;
- Use Differing Kinds of Learning Context for Differing Administrative Contexts.

2.1.4 Management and Manipulate Course Structure Information

The specific set of operational needs is:

- Create a Course Template, Course Offering and Course Section;
- Update Course Template, Course Offering and Course Section information;
- Update Status of Course Template, Course Offering and Course Section;
- Roll-over a Course Template, Course Offering and Course Section;
- Delete a Course Template, Course Offering and Course Section;
- Get Information for a Course Offering;
- Get All Course Offerings for a Semester;
- Get All Active Course Offerings under a Given Organization Structure Entity;
- Get Course Offerings for an Instructor;
- Get Equivalent Course Templates and Course Offerings;
- Get All Enrollment information for a Semester;
- Search for a Course Template or Offering.

2.1.5 Management and Manipulate of Grade Book Information

The specific set of operational needs is:

- Get Grade Book Information for All Persons Enrolled in a Course Offering;
- Get Grade Book Information for a Person;
- Get Grade Book Information for All Persons Enrolled in a Course Section;
- Get Grade Book Information for a Person;
- Get All Final Grade for All Persons Enrolled in a Course Offering;
- Get the Final Grade for All Active Course Offerings for a Given Person.

2.1.6 Batch Processing

There are operational points when the service consumer (the Synchronization Agent) needs to be bulk synchronized or initialized with, or by, the service provider (the Reference Agent). The synchronization/initialization point is typically declared as changes from a particular reference point. Specific synchronization/initialization needs are:

- Batch Initialization and Synchronization of all Person objects;
- Batch Initialization and Synchronization of all Group objects;
- Batch Initialization and Synchronization of all Membership objects;
- Batch Initialization and Synchronization of all Course Template objects;
- Batch Initialization and Synchronization of all Course Offering objects;
- Batch Initialization and Synchronization of all Course Section objects;
- Batch Initialization and Synchronization of all Grade-book objects;
- Batch Initialization of everything.

2.2 An Abstract Representation

The basic architectural model for the LIS specification is shown in Figure 2.1. In this architecture the scope of the data exchange provided by the services is shown as the dotted line. The scope of the interoperability is the data and behavioral models of the objects being exchanged.

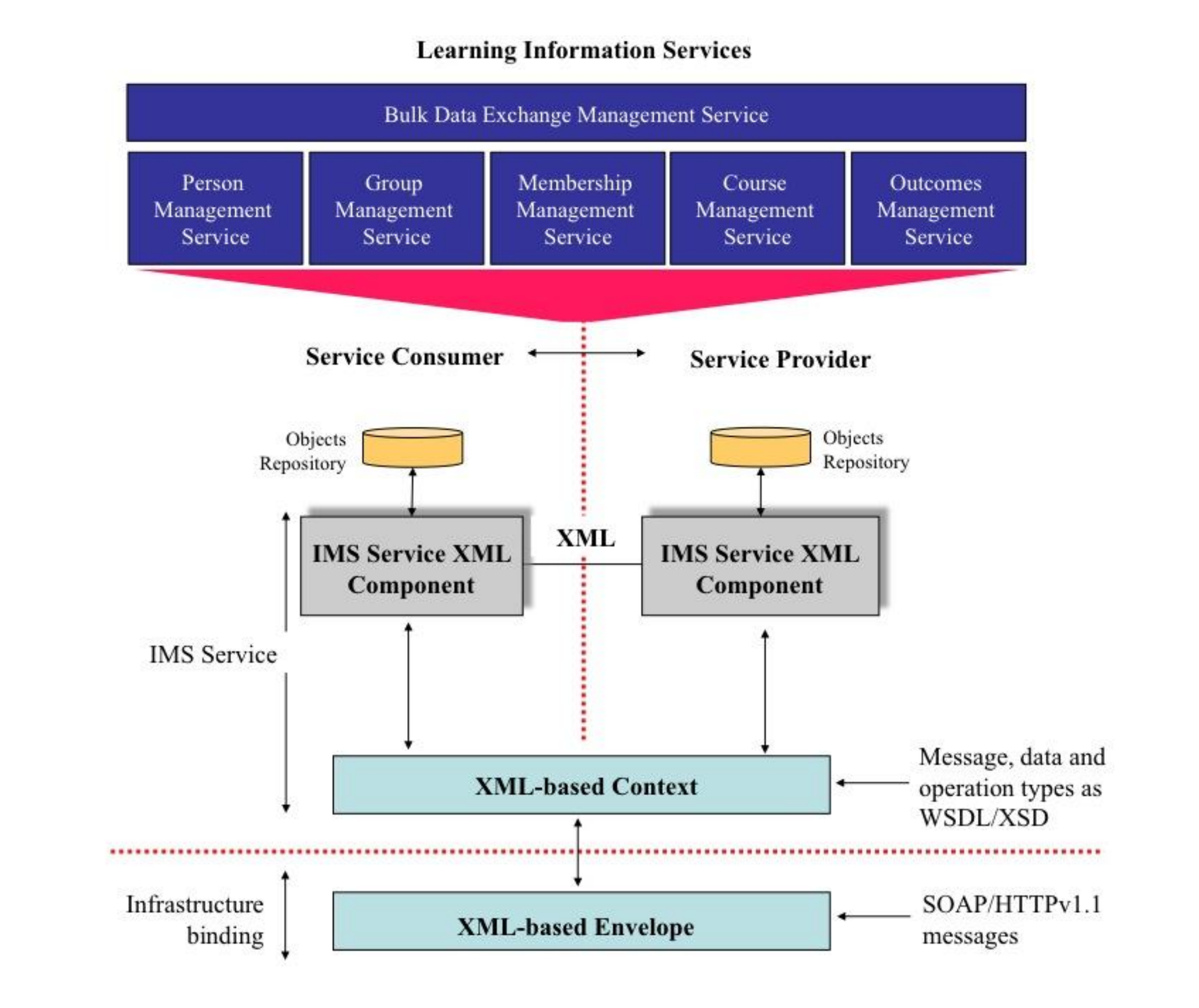

Learning Information Services
Bulk Data Exchange Management Service
Person Management Service
Group Management Service
Membership Management Service
Course Management Service
Outcomes Management Service
Service Consumer
Service Provider
Objects Repository
Objects Repository
IMS Service XML Component
XML
IMS Service 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 Schematic architecture of the learner information services.

Six services are defined. Instances of the data models are stored in the service consumer/provider object repositories. It is the persistence of the data in these repositories that reflects the dynamic changes in the system. The set of services are realized as SOAP messages to exchange the XML-based data object.

2.3 Building Real Systems

An implementation is not required to support each and every service. Neither is an implementation required to support each and every operation. The specific requirements are defined in the corresponding profile[1]. Interoperability is supported between systems that implement the same profile. Cross-profile interoperability may occur but this is a by-product and should NOT be used as the basis for any system realization.

[1] IMS GLC has created one such profile called the ‘Core Profiles’ [LIS, 10c].

One of the outputs of the LIS specification is the set of WSDL/XSD binding files. Each service has its own set of WSDL/XSD files. It is these files that are used by code-generation tools to create the source code that handles the SOAP messages and XML data structures. Some small changes are required to the WSDL files to map the SOAP messages to the actual server-based implementation of the Web Service. These changes can either be manually applied to the WSDL files or to the UML files. The UML files must then be processed using the IMS GLC Binding Auto-generation Tool-kit (I-BAT) to create the new, tailored WSDL/XSD files.

3 The Historic Perspective

3.1 Past Versions

The release history for the LIS is listed in Table 3.1. The original Enterprise v1.0 data model was the second specification released by IMS GLC. It has undergone several revisions in response to feedback from the experience gained in the various implementations.

Table 3.1 Version history for Enterprise/LIS specifications.

Version	Release Date	Description
Enterprise 1.0	November, 1999	Original data model for Enterprise systems interoperability.
Enterprise 1.0.1	January, 2000	Correction of a small number of bugs identified in version 1.
Enterprise 1.1	July, 2002	Revised data model to introduce new functionality based on commonly used extensions.
Enterprise Services v1.0	August, 2004	Introduction of services, based upon the IMS GLC General Web Services, to support the exchange of the information based upon the Enterprise v1.1 data model.

LIS v2.0 is a radical reworking of both the original services and data models.

3.2 A Summary of Changes from Version 1

3.2.1 Person Management Service

The functional changes in version 2 compared to version 1 are:

a) A single service interface is used. With the exception of the ‘ReadPersons’ operation all of the operations in the original ‘Persons Manager’ interface have been removed;

b) The ‘ReadPersons’ operation has been changed such that it returns a single StatusInfo object;

c) New service operations have been added, namely:

- ReadCorePerson – to read information that is defined as the ‘core’ Person
- ReadAllPersonIds – to read all of the SourcedIds allocated in the target system
- ReadPersonIdsFromSavePoint – to read all of the SourcedIds for Person objects that have been altered since the defined reference point
- ReadPersonsFromSavePoint – to read all of the Person objects, that have been altered since the defined reference point
- DiscoverPersonIds – to provide the SourcedIds of the Person objects that are identified by the completion of the requested query operation;

d) The core data model has been extended to support both the PMSv1.0 and the IMS GLC Learner Information Packaging (LIP) v1.0 ‘identification’ data models. The data model has also been modified to use external vocabularies.

3.2.2 Group Management Service

The changes in version 2 compared to version 1 are:

a) A single service interface is used. With the exception of the ‘ReadGroups’ operation all of the operations in the original ‘GroupsManager’ interface have been removed;

b) The ‘ReadGroups’ operation has been changed such that it returns a single StatusInfo object;

c) New service operations have been added, namely:

- ReadAllGroupIds – to read all of the SourcedIds allocated in the target system
- AddGroupRelationship – to add a relationship between two Group objects
- RemoveGroupRelationship – to remove a relationship between two Group objects
- ReadGroupIdsForPerson – to read all of the SourcedIds for Group objects for a specific Person object
- ReadGroupIdsFromSavePoint – to read all of the SourcedIds for Group objects that have been altered since the defined reference point
- ReadGroupsFromSavePoint – to read all of the Group objects, that have been altered since the defined reference point
- DiscoverGroupIds – to provide the SourcedIds of the Group objects that are identified by the completion of the requested query operation;

d) Version 1.0 implementations of the Group Management Service were used to exchange information about courses. For Version 2 this is only permitted for additional features that are added to the Course Management Service capabilities (see [CMS, 11a] for more details).

3.2.3 Membership Management Service

The changes in version 2 compared to version 1 are:

a) A single service interface is used. With the exception of the 'ReadMemberships' operation all of the operations in the original 'MembershipsManager' interface have been removed;

b) The 'ReadMemberships' operation has been changed such that it returns a single StatusInfo object;

c) New service operations have been added, namely:

- ReadAllMembershipIds – to read all of the SourcedIds allocated in the target system to a Membership object
- ReadMembershipIdsForPerson – to read all of the SourcedIds for Membership objects for a specific Person object
- ReadMembershipIdsForPersonWithRole – to read all of the SourcedIds for Membership objects for a specific Person object with a specific role
- ReadMembershipIdsForCollection – to read all of the SourcedIds for Membership objects for a specific type of object i.e., Group, CourseTemplate, CourseOffering, CourseSection and SectionAssociation
- ReadMembershipIdsFromSavePoint – to read all of the SourcedIds for Membership objects that have been altered since the defined reference point
- ReadMembershipsFromSavePoint – to read all of the Membership objects, that have been altered since the defined reference point
- DiscoverMembershipIds – to provide the SourcedIds of the Membership objects that are selected by the application of the requested query operation;

d) The data model has been modified such that:

- The final and interim results structures have been removed (these are now supported using the Outcome Management Service [OMS, 11a])
- The 'recordInfo' attribute has been redefined as a type of meta-data
- A Group cannot have a membership of a Group. Therefore, the 'memberIdType' attribute has been removed because it is now unnecessary i.e., only a Person object can be a member of a Group, etc.

3.2.4 Course Management Service

The Course Management Service was not part of the IMS GLC Enterprise Services v1.0 specification [ES, 04a]. Instead, this functionality was supported using the IMS GLC Group Management Service v1.0 [GMS, 04] specification in a variety of different ways. This created interoperability problems hence the creation of the CMS specification. The Course Management Service v1.0 is closely linked to the Group Management Service v2.0 and Membership Management Service (MMS) v2.0. 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.

3.2.5 Outcomes Management Service

The Outcomes Management Service was not developed in the IMS GLC Enterprise Services v1.0 specification [ES, 04a]. Instead, a simplified form of functionality was supported using the IMS GLC Membership Management Services (MMS) v1.0 [MMS, 04]. In general, there is NO backwards compatibility between the usage of the OMSv1.0 and the ways in which MMSv1.0 has been implemented to support outcomes management. Vendors may define compatibility bridges for their own implementations but these are outside the scope of this specification.

3.2.6 Bulk Data Exchange Management Service

The Bulk Data Exchange Management Service was not developed in the IMS GLC Enterprise Services v1.0 specification [ES, 04a]. Instead a series of best practices were defined to achieve data synchronization and initialization. These are now replaced by the use of the BDEMS.

3.3 Version 2 and Version 1 Compatibility

The release of the Learning Information Services 2.0 creates the issue of compatibility between version 1 and version 2 implementations. Compatibility issues occur when:

a) A version 1 service implementation initiates data exchange with a version 2 implementation;

b) A version 2 service implementation initiates data exchange with a version 1 implementation.

The binding of the Information Model recommends that the Uniform Resource Locator (URL) for the messaging actions is dependent on the type and version number of the source specification: in such a case it is not possible for cross-interaction between implementations of version 1 and 2. However, if a common URL is used then cross-interaction becomes possible. The definition of the behavior for interactions between different versions is beyond the scope of this specification.

4 Using the New Version

4.1 Building an Interoperable Solution

The LIS Best Practices & implementation Guide [LIS, 11b] provides a detailed set of guidance notes on issues to consider when implementing the LIS specification. The LIS specification describes **interoperability** and so it is silent on many aspects of deploying operational systems. However the most essential interoperability features to address in an implementation of LIS are:

- LIS consists of six services and an implementation may support one, many or all of them. Correct operation of some of the services depends on other services (or the equivalent functionality being supplied). For example, the MMS requires the PMS and either the GMS or the CMS: membership is the act of a Person becoming a member of a Group or Course and so the creation of a Membership can only be achieved if the corresponding Person and Group/Course objects have already been created. The specification does *not* address this inter-service choreography, in terms of messaging sequencing, but it must be addressed as part of the creation of an operational end system;

- The Web Services implementation for each of the services is based upon the IMS General Web Services (GWS) specification (this is a profile of the WS-I Basic Profile v1.1). This describes how the SOAP messages must be constructed and includes a simple request/response message correlation mechanism. It also includes the mechanism by which the end-system transaction status information is reported. It is essential that APIs that implement the GWS and the LIS pass this information, in suitable form, to the corresponding end-system applications. Furthermore, conformance requires that the defined status information is returned for requests that are made on services or operations that are not supported by an implementation i.e., a transaction request *must always* receive a transaction response;

- The specification uses the terms Ref Agent, Sync Agent, Service Provider and Service Consumer. A Service Consumer is an end-system that calls a service i.e., issues the request message. A Service Provider is an end-system that provides the service and returns a response message. A Ref Agent is a business function description for the end-system that is to be used as the definitive source of information i.e., the reference. The Sync Agent is the corresponding end-system that is to be synchronized with the Ref Agent. With the exception of the BDEMS the Ref Agent pushes information out (create and write data) and the Sync Agent pulls the data in (read data). When the Ref Agent pushes data out the Sync Agent is the Service Provider, whereas, when the Sync Agent pulls the data in the Ref Agent is the Service Provider. In the case of system exchanges, such as the SIS and LMS, then depending on the nature of the service they may be either the Ref Agent or the Sync Agent (hence in an extensive implementation of LIS an end system may be both a Ref and Sync Agent depending on the service). For example, in the case of course enrollment the SIS is the Ref Agent and the LMS the Sync Agent whereas for grade reporting the LMS is the Ref Agent and the SIS the Sync Agent;

- Apart from the BDEMS, the services operate in real-time i.e., a request message is sent and the calling end system (the service consumer) waits (formally termed blocked) until the corresponding response message is received (an implementation should consider timeout issues as part of error handling). This is also termed a synchronous service. The BDEMS operates as a batch service (this is also termed asynchronous). In this case a request for data file transfer is made, this request is acknowledged, at some time later the data file is made available and downloaded, and finally a report of the data file downloading is reported. During this operation, the real-time/synchronous service calls are blocked i.e., not permitted to the service provider. However real-time exchanges with other service providers are permitted. Real-time processing recommences once the batch exchanges have been completed or cancelled;

- The data models have two uses. The first is to define the permitted contents in the SOAP messages and the second is to inform the end-system data storage implications. In general, all of the contents in a SOAP message are optional. This is because there can never be a guarantee that a required piece of data is available e.g., for a failed read attempt there will be no data and so any ‘mandatory’ data will be missing. In the case of end-system provisioning, an end-system must support one instance of any object that is either required or optional and has a multiplicity of one. In the case of multiplicities of greater than one then the maximum number of supported instantiations is implementation dependent. In the case of a profile, whenever a smallest permitted maximum is defined then an end system implementation must support at least that number of instantiations. If a request attempts to exceed either a service consumer or service provider implementation constraint then the corresponding status code must be returned;

- SourcedIds are used to uniquely identify object instances. In an ideal implementation the SourcedIds would be globally unique identifiers. However, practical constraints normally make it impossible to make this assumption. A SourcedId must be unique between the communicating end-systems. It is possible that SourcedIds could be reused in different services because the use of a SourcedId is always presented in the context of a specific service and its associated data model. It is recommended that an implementation assume that a SourcedId is only unique to a service i.e., all SourcedId processing should be accompanied by the service context also being used. This provides the most robust implementation;
- It is important to note that the use of the LIS specification does not provide 'out-of-the-box' interoperability. This is not a failure of the specification but a reflection of the pragmatic nature of agreeing such a specification and enabling commercial competitiveness. Many system architecture aspects are not addressed by the specification e.g., service link initialization by which two end systems become aware of the capabilities supported by the other, service integrity and security, etc. This type of information is exchanged using either an out-of-band set of services or as part of service deployment and configuration.

4.2 The Core Profiles

The Core Profiles identify the minimal subset of the functionality that must be supported by systems developed for Student Information System/Learning Management System (SIS/LMS) deployment. These Profiles (there is a Core plus several Additions) define the set of operations and data models that must be supported by the systems supporting the set of services within the LIS. A system can support greater functionality but there is no guarantee of interoperability for those extra features. Interoperability is only guaranteed for the functionality described in the Core Profiles.

The Core Profile consists of:

- The deletePerson, readPerson (required for conformance testing only) and replacePerson operations for the Person Management Service;
- The deleteMembership, readMembership (required for conformance testing only) and replaceMembership operations for the Membership Management Service;
- The deleteCourseSection, readCourseSection (required for conformance testing only) and replaceCourseSection operations for the Course Management Service;
- The announceBulkDataExchange, reportBulkDataExchange, ignoreBulkDataExchange and SOAP-based file download operations for the Bulk Data Exchange Management Service.

The three Addition Profiles are:

- Final Grade – provides support for the management of grades and grade books using the Outcomes Management Service;
- Combined Sections – provides support for the management of SectionAssociations using the Course Management Service
- Full Course Hierarchy – provides support for the management of the full course structure (templates, offerings, sections and section associations) using the Course Management Service.

4.3 Creating a Profile

If the Core Profile, either alone or in combination with one or more of the Addition Profiles, is inadequate then a new profile can be created and registered with IMS GLC [SAN11, 06]. The recommended steps for creating a profile of the LIS are:

- Identify which of the six services are to be included in the Profile;
- For each service identify the set of operations that are to be included in the Profile of the service;
- For each operation identify the set of status codes that are to be supported in the Profile of the operation;

- For each data model in each service identify which parts of any vocabulary are to be supported in the Profile of the vocabulary. If appropriate, or necessary, new vocabularies should be defined to replace the corresponding default vocabularies;
- For each data model identify any increased constraints on the multiplicity of any attributes i.e., optional can become mandatory, optional can be prohibited and many can be reduced to one.

It is recommended that the profile be documented using the original Unified Modeling Language (UML) Platform Independent Model (PIM)/Platform Specific Model (PSM) descriptions. Each change should have a documented comment. Once the Profile has been completed, the I-BAT should be used to generate the new WSDL/XSD from the profiled UML PSM.

4.4 Conformance to a Profile

Conformance to LIS is non-trivial even when this is against a Profile. An inclusive approach to conformance has been adopted. The conformance statement for a profile is described as the set of business capabilities that are supported by the Profile e.g., student enrolment in a course, reporting of a student result, etc. An implementation claiming conformance must identify which of the capabilities are supported; this takes the form of a checklist. Therefore, interoperability between two systems is determined by the extent of the overlap in the corresponding checklists.

As an aid to implementers, once the business capability checklist has been produced a corresponding functional map is produced. This functional map identifies the set of services, operations and data model features that must be supported to realize the business capability. Any list of business capabilities will require a common set of LIS functional capabilities (the extent defines how much of the LIS is required to support a Profile). This list of functionality defines how a business capability MUST be realized using the LIS specification.

This approach has been used to allow incremental adoption of the LIS specification. Therefore, there are degrees of interoperability and it becomes the responsibility of developers to clearly identify which features of the LIS specification are implemented. Over time, the degree of interoperability across the community should increase.

5 The Document Set

5.1 The Set of Documents

5.1.1 The LIS Specification

This document. It describes how the LIS is composed using its six component services. There is an overview version of this document [LIS, 11a]. A schematic representation of the document set is shown in Figure 5.1.

5.1.2 Information Models

The Information Model documents contain the normative description of the various service definitions, data structures and their relationships. These descriptions use the UML, and in particular, the *IMS GLC UML Profile Platform Independent Model (PIM) Description for Specifications* [SDN07, 06]. Each of the six services has its own Information Model [PMS, 11a], [GMS, 11a], [MMS, 11a], [CMS, 11a], [OMS, 11a] and [BDEMS, 11a].

5.1.3 WSDL Bindings

The WSDL Binding documents contain the PSM for the service. This PSM has been transformed into the corresponding Web Services Description Language (WSDL) and XML Schema Definition (XSD) files using the I-BAT. The Binding document describes the underlying structure of the WSDL/XSD, the associated Vocabulary Definition Exchange (VDEX) vocabulary files and the formats of the corresponding SOAP messages.

5.1.4 Best Practice & Implementation Guide

The IMS GLC LIS Best Practice and Implementation Guide [LIS, 11b] is intended to provide vendors with an overall understanding of the IMS GLC LIS Specification, the relationship of the specification with other IMS GLC specifications, and a best practices guide derived from experiences of those using the specification[2]. The guide also includes a several actual examples that describe how vendors can make the best use of the IMS LIS Specification.

5.1.5 Core Profile & Conformance Specification

Core Profiles for the LIS have been created [LIS, 11c]. The Core Profiles identifies the minimal subset of the functionality that must be supported by systems developed for SIS/LMS interaction. This Profile (there is a Core plus several Additions) defines the set of operations and data models that must be supported by the systems implementing the set of services within the LIS. A system can support greater functionality but there is no guarantee of interoperability for those extra features. Interoperability is only guaranteed for the functionality described in the HE Profiles.

For the Core and the set of Additions there is a set of conformance statements. These statements constitute the Conformance Specification for each profile.

5.1.6 Related Documents

The following related documents were used to support the development of this specification:

- *IMS GLC General Web Services Base Profile Final Release* [GWS, 06a] – provides the definition of how a service is to be implemented using SOAP-based Web Services
- *IMS GLC General Web Services WSDL Binding Guidelines Final Release* [GWS, 06b] – provides the context by which the WSDL description is derived from the UML-based Information Model;
- *IMS GLC SDN 07: UML Profile for PIM Descriptions of Specifications* [SDN07, 06] – provides the definition and description of the syntax and semantic of the UML profile used for the PIM description of the data models in the set of information model documents;

[2] We recommend that new users of the IMS GLC LIS Specification start with the *IMS GLC LIS Best Practices & Implementation Guide*. The examples in this document show how we intend the specification to be used, whereas the Information Model and WSDL Binding documents contain the formal description of the services, data structures, their syntax and semantics.

- *IMS GLC SDN 08: UML Profile for PSM Descriptions of Specification Bindings* [SDN08, 06] – provides the definition and description of the syntax and semantics of the UML Profile used for the PSM description of the XML Binding aspects of the service definitions;
- *IMS GLC SDN 11: Vocabulary Definition, Registration and Maintenance Procedures* [SDN11, 06] – provides the definition and description of how the set of external attribute vocabularies are expressed as instances of the IMS Vocabulary Definition Exchange (VDEX) specification [VDEX, 04].

5.2 The Binding Files

5.2.1 WSDL Files

Each of the services has a WSDL file that describes the nature of the service. Each service has a single combined WSDL/XSD file. Separate WSDL and XSD files are also available.

5.2.2 XSD Files

Apart from the separate XSD file that complements the corresponding separate WSDL file, there are other XSD files. These files are used to validate the content of external data files that are exchanged in the BDEMS using an appropriate file transfer protocol.

5.2.3 Vocabulary Files

Each of the services has a set of VDEX files that contain the set of default vocabularies defined in the Information Model. The VDEX files conform to the IMS GLC VDEX specification [VDEX, 04].

5.3 Using the Documents

The documentation set of the IMS GLC Learning Information Services Specification is daunting. However, a few simple guidelines make it considerably easier for even a newcomer to work through the documents.

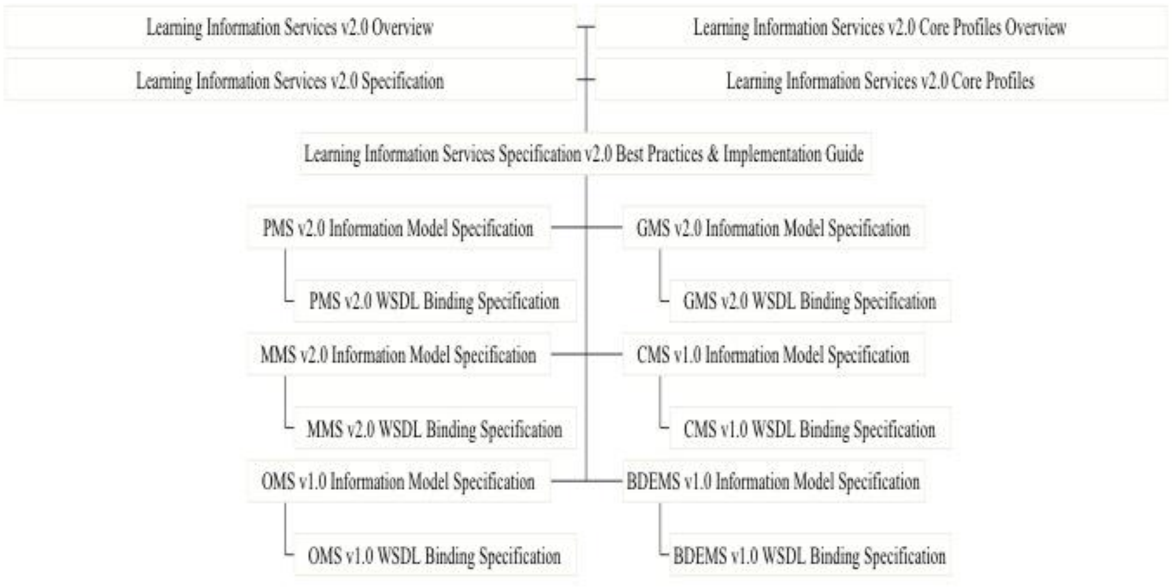

Learning Information Services v2.0 Overview
Learning Information Services v2.0 Core Profiles Overview
Learning Information Services v2.0 Specification
Learning Information Services v2.0 Core Profiles
Learning Information Services Specification v2.0 Best Practices & Implementation Guide
PMS v2.0 Information Model Specification
GMS v2.0 Information Model Specification
PMS v2.0 WSDL Binding Specification
GMS v2.0 WSDL Binding Specification
MMS v2.0 Information Model Specification
CMS v1.0 Information Model Specification
MMS v2.0 WSDL Binding Specification
CMS v1.0 WSDL Binding Specification
OMS v1.0 Information Model Specification
BDEMS v1.0 Information Model Specification
OMS v1.0 WSDL Binding Specification
BDEMS v1.0 WSDL Binding Specification

Figure 5.1 Schematic representation of the documentation set.

A schematic representation of the document set is shown in Figure 5.1 and the recommended approach to reviewing the set is:

a) Always start with the specification overview (the shorter version of this document);

b) The *IMS GLC LIS Best Practices & Implementation Guide* is an excellent way to understand the why, what, and how of a specification. The set of examples described in the guide is an excellent way to understand what is being created by the specification. The guide should always be read before attempting to work through either the Information Models or the WSDL Bindings;

c) Once the set of examples has been digested, it is time to work through the Information Models. These models provide the formal definition of all of the service operations, data structures and their behaviors and relationships;

d) Finally, work through the WSDL Binding documents. Often, only the implementation/engineering team needs to understand the details of these documents. Each binding document is the definitive statement of how interoperability is achieved using WSDL. The WSDL Bindings are formally realized as the WSDL/XSD files (these files can be used to generate the corresponding SOAP messages);

The *IMS GLC LIS Core Profiles* [LIS, 11c] should be used as the basis for providing the core service capability. This profile identifies the minimal set of service operations that are required to create a ‘useful’ learning information service (the profile identified less than 5% of the full set of service operations as necessary).

Appendix A Person Management Service 2.0 Overview

A1 PMS Description

The Person Management Service is used to model the service responsible for manipulating information about people. The PersonManagementService interface is shown in Figure A1.1.

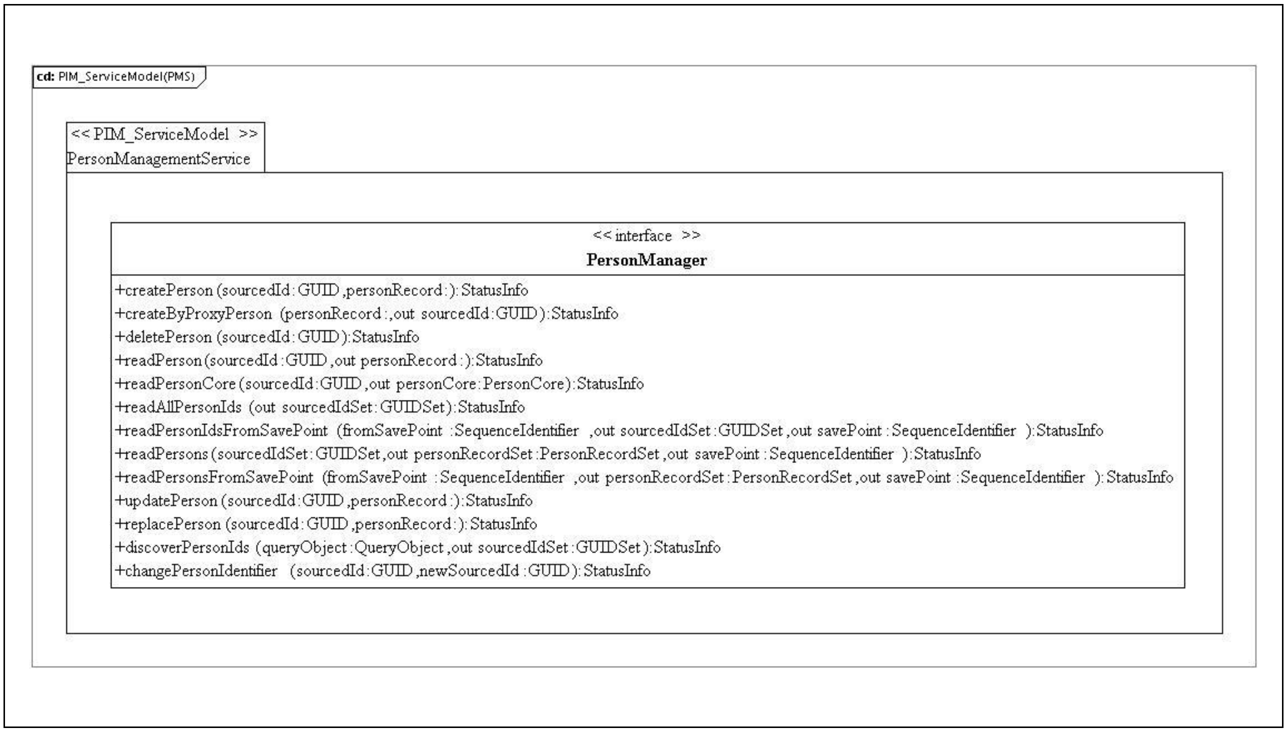

cd: PIM_ServiceModel(PMS)
<< PIM_ServiceModel >>
PersonManagementService
<< interface >>
PersonManager
+createPerson (sourcedId : GUID ,personRecord :): StatusInfo
+createByProxyPerson (personRecord : ,out sourcedId : GUID): StatusInfo
+deletePerson (sourcedId : GUID): StatusInfo
+readPerson (sourcedId : GUID ,out personRecord :): StatusInfo
+readPersonCore (sourcedId : GUID ,out personCore : PersonCore): StatusInfo
+readAllPersonIds (out sourcedIdSet : GUIDSet): StatusInfo
+readPersonIdsFromSavePoint (fromSavePoint : SequenceIdentifier ,out sourcedIdSet : GUIDSet ,out savePoint : SequenceIdentifier): StatusInfo
+readPersons (sourcedIdSet : GUIDSet ,out personRecordSet : PersonRecordSet ,out savePoint : SequenceIdentifier): StatusInfo
+readPersonsFromSavePoint (fromSavePoint : SequenceIdentifier ,out personRecordSet : PersonRecordSet ,out savePoint : SequenceIdentifier): StatusInfo
+updatePerson (sourcedId : GUID ,personRecord :): StatusInfo
+replacePerson (sourcedId : GUID ,personRecord :): StatusInfo
+discoverPersonIds (queryObject : QueryObject ,out sourcedIdSet : GUIDSet): StatusInfo
+changePersonIdentifier (sourcedId : GUID ,newSourcedId : GUID): StatusInfo

Figure A1.1 PersonManagementService interface definition.

A2 PMS Operations

The PersonManager interface class describes the operations that are permitted on Person objects. These operations are based upon the classic Create/Read/Update/Delete model with variations defined to differentiate subtleties of functionality. The interface stereotype indicates that there are no attributes for this class. The set of operations are summarized in Table A2.1.

Table A2.1 Summary of operations for PersonManager.

Operation	Description
createPerson	To request the creation of a populated Person object on the target system where the source is responsible for the allocation of the unique identifier.
createByProxyPerson	To request the creation of a populated Person object on the target system where the target is responsible for the allocation of the unique identifier.
deletePerson	To request the deletion of a Person object. The Person object is deleted and all of its associated Memberships.
readPerson	To read the full contents of the identified Person object. The target must return all of the data it has for the identified Person object.
readPersonCore	To read the minimal mandatory contents of the identified Person object. Only the data structures that form the core of a Person object must be returned.

Operation	Description
readAllPersonIds	To obtain the set of identifiers which have been assigned to Person objects.
readPersonIdsFromSavePoint	To obtain the set of identifiers for Person 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.
readPersons	To obtain the Person 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.
readPersonsFromSavePoint	To obtain the set of Person 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.
updatePerson	To write new content into the identified Person object. The target must write the new data into the Person object. This is an additive operation.
replacePerson	To replace the content of the identified Person object. The target must write the new data into the Person object. This is a destructive write-over of all of the original information. If the Person object does not exist, this operation acts as per a ‘createPerson’ request.
discoverPersonIds	To obtain the set of identifiers for Person objects whose properties agree with those defined in the query.
changePersonIdentifier	To change the SourcedId of the Person object. The completion of this operation will result in later actions using the original SourcedId reporting an unknown identifier status.

Appendix B Group Management Service 2.0 Overview

B1 GMS Description

The Group Management Service is used to model the service responsible for manipulating information about Groups. The GroupManagementService is shown in Figure B1.1.

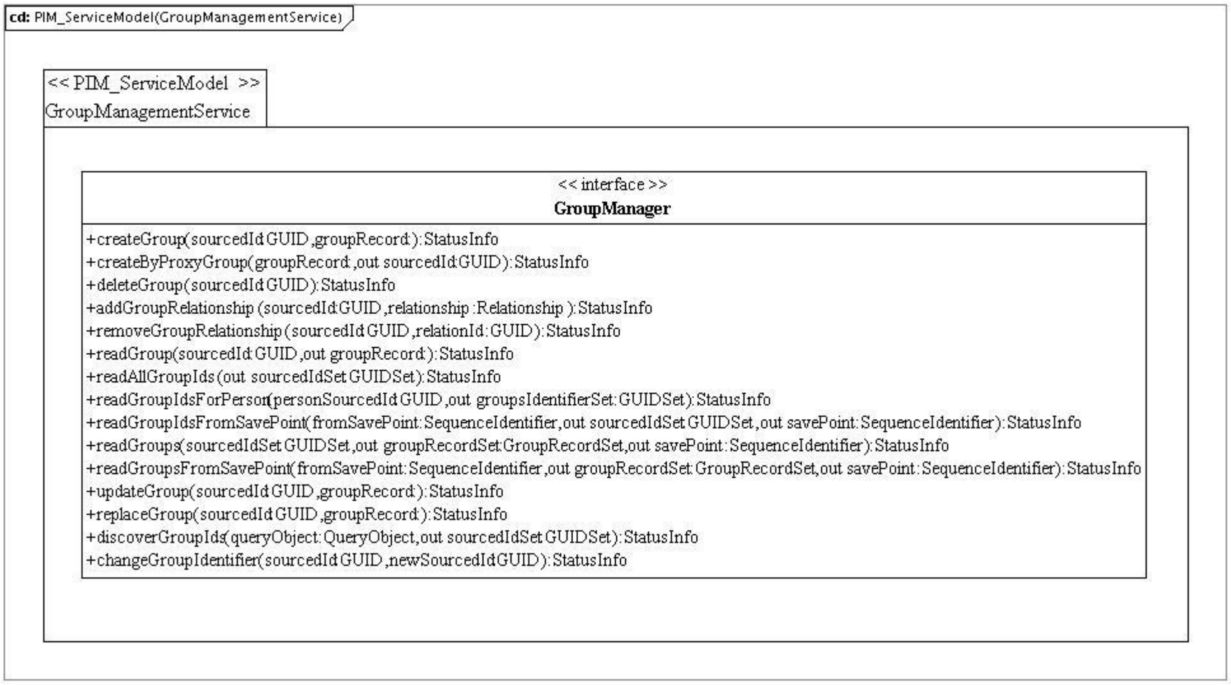

cd: PIM_ServiceModel(GroupManagementService)
<< PIM_ServiceModel >>
GroupManagementService
<< interface >>
GroupManager
+createGroup(sourcedId:GUID,groupRecord:):StatusInfo
+createByProxyGroup(groupRecord:,out sourcedId:GUID):StatusInfo
+deleteGroup(sourcedId:GUID):StatusInfo
+addGroupRelationship (sourcedId:GUID,relationship :Relationship):StatusInfo
+removeGroupRelationship (sourcedId:GUID,relationId: GUID):StatusInfo
+readGroup(sourcedId:GUID,out groupRecord:):StatusInfo
+readAllGroupIds(out sourcedIdSet:GUIDSet):StatusInfo
+readGroupIdsForPerson(personSourcedId:GUID,out groupsIdentifierSet:GUIDSet):StatusInfo
+readGroupIdsFromSavePoint(fromSavePoint:SequenceIdentifier,out sourcedIdSet:GUIDSet,out savePoint:SequenceIdentifier):StatusInfo
+readGroups(sourcedIdSet:GUIDSet,out groupRecordSet:GroupRecordSet,out savePoint:SequenceIdentifier):StatusInfo
+readGroupsFromSavePoint(fromSavePoint:SequenceIdentifier,out groupRecordSet:GroupRecordSet,out savePoint:SequenceIdentifier):StatusInfo
+updateGroup(sourcedId:GUID,groupRecord:):StatusInfo
+replaceGroup(sourcedId:GUID,groupRecord:):StatusInfo
+discoverGroupIds(queryObject:QueryObject,out sourcedIdSet:GUIDSet):StatusInfo
+changeGroupIdentifier(sourcedId:GUID,newSourcedId:GUID):StatusInfo

Figure B1.1 GroupManagementService interface definition.

B2 GMS Operations

The GroupManager interface class describes the operations that are permitted on Group objects. These operations are based upon the classic Create/Read/Update/Delete model with variations defined to differentiate subtleties of functionality. The interface stereotype indicates that there are no attributes for this class. The set of operations are summarized in Table B2.1.

Table B2.1 Summary of operations for GroupManager.

Operation	Description
createGroup	To request the creation of a populated Group object on the target system where the source is responsible for the allocation of the unique identifier.
createByProxyGroup	To request the creation of a populated Group object on the target system where the target is responsible for the allocation of the unique identifier.
deleteGroup	To request the deletion of a Group. All of the associated Membership objects are also deleted.
addGroupRelationship	To request the creation of a relationship between two Group objects. This does not create the Group objects themselves.

Operation	Description
removeGroupRelationship	To request the deletion of a relationship between two Group objects. This does not delete the Group objects themselves.
readGroup	To read the full contents of the identified Group object. The target must return all of the data it has for the identified Group object.
readAllGroupIds	To obtain the set of identifiers which have been assigned to Group objects.
readGroupIdsForPerson	To read the identifiers for all of the Group objects associated with the identified Person object.
readGroupIdsFromSavePoint	To obtain the set of identifiers for Group 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.
readGroups	To obtain the Group 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.
readGroupsFromSavePoint	To obtain the set of Group 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.
updateGroup	To write new content into the identified Group object. The target must write the new data into the Group object. This is an additive operation.
replaceGroup	To replace the content of the identified Group object. The target must write the new data into the Group object. This is a destructive write-over of all of the original information. If the Group object does not exist, this operation acts as per a ‘createGroup’ request.
discoverGroupIds	To obtain the set of identifiers for Group objects whose properties agree with those defined in the query/filter.
changeGroupIdentifier	To change the identifier of the Group object. The completion of this operation will result in later actions using the original identifier reporting an unknown identifier status.

Appendix C Membership Management Service 2.0 Overview

C1 MMS Description

The Membership Management Service is used to model the service responsible for manipulating information about people’s memberships of Groups [GMS, 11a] and Courses [CMS, 11a]. The MembershipManagementService is shown in Figure C1.1.

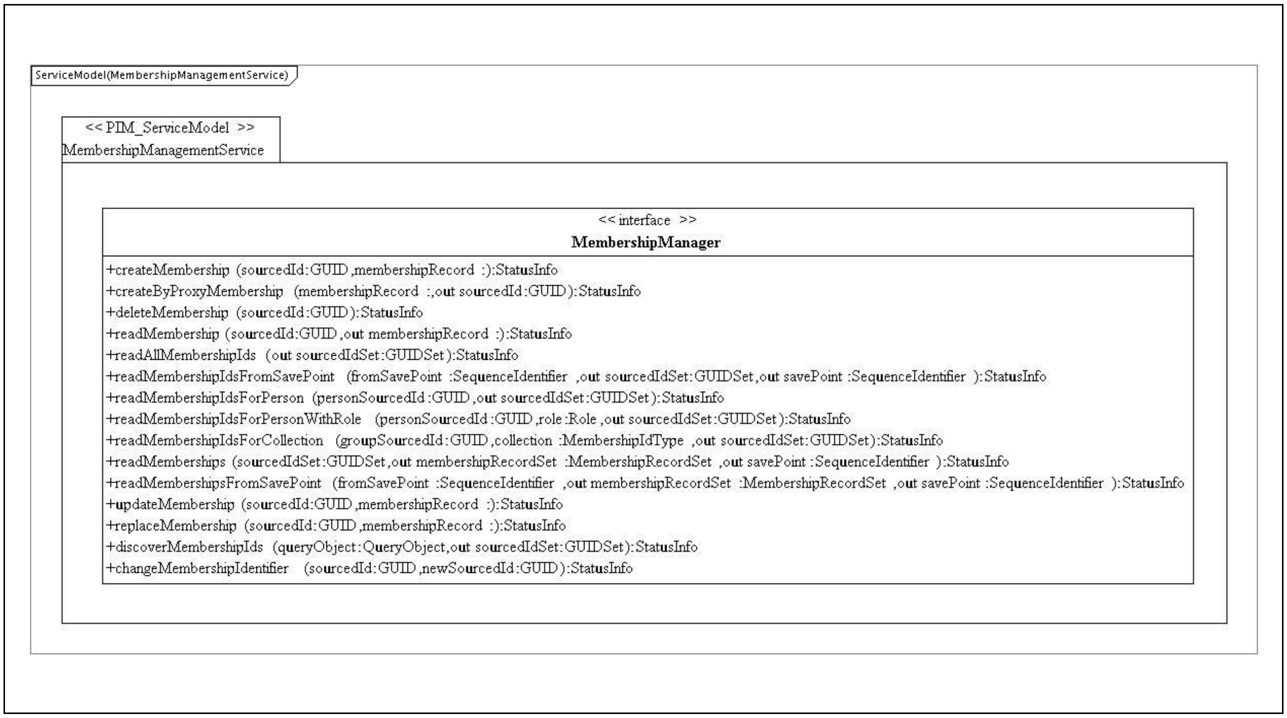

ServiceModel(MembershipManagementService)
<< PIM_ServiceModel >>
MembershipManagementService
<< interface >>
MembershipManager
+createMembership (sourcedId:GUID ,membershipRecord :):StatusInfo
+createByProxyMembership (membershipRecord :,out sourcedId:GUID):StatusInfo
+deleteMembership (sourcedId:GUID):StatusInfo
+readMembership (sourcedId:GUID ,out membershipRecord :):StatusInfo
+readAllMembershipIds (out sourcedIdSet:GUIDSet):StatusInfo
+readMembershipIdsFromSavePoint (fromSavePoint :SequenceIdentifier ,out sourcedIdSet:GUIDSet,out savePoint :SequenceIdentifier):StatusInfo
+readMembershipIdsForPerson (personSourcedId :GUID ,out sourcedIdSet:GUIDSet):StatusInfo
+readMembershipIdsForPersonWithRole (personSourcedId :GUID ,role :Role ,out sourcedIdSet:GUIDSet):StatusInfo
+readMembershipIdsForCollection (groupSourcedId:GUID ,collection :MembershipIdType ,out sourcedIdSet:GUIDSet):StatusInfo
+readMemberships (sourcedIdSet:GUIDSet,out membershipRecordSet :MembershipRecordSet ,out savePoint :SequenceIdentifier):StatusInfo
+readMembershipsFromSavePoint (fromSavePoint :SequenceIdentifier ,out membershipRecordSet :MembershipRecordSet ,out savePoint :SequenceIdentifier):StatusInfo
+updateMembership (sourcedId:GUID ,membershipRecord :):StatusInfo
+replaceMembership (sourcedId:GUID ,membershipRecord :):StatusInfo
+discoverMembershipIds (queryObject:QueryObject,out sourcedIdSet:GUIDSet):StatusInfo
+changeMembershipIdentifier (sourcedId:GUID ,newSourcedId:GUID):StatusInfo

Figure C1.1 MembershipManagementService interface definition.

C2 MMS Operations

The MembershipManager interface class describes the operations that are permitted on Membership objects. These operations are based upon the classic Create/Read/Update/Delete model with variations defined to differentiate subtleties of functionality. The interface stereotype indicates that there are no attributes for this class. The set of operations are summarized in Table C2.1.

Table C2.1 Summary of operations for the MMS.

Operation	Description
createMembership	To request the creation of a populated Membership object on the target system where the source is responsible for the allocation of the unique identifier.
createByProxyMembership	To request the creation of a populated Membership object on the target system where the target is responsible for the allocation of the unique identifier.
deleteMembership	To request the deletion of a Membership object. The Membership object is deleted along with all of its associated relationships (the associated Group, Course and Person objects are not deleted).
readMembership	To read the full contents of the identified Membership object. The target

Operation	Description
	must return all of the data it has for the identified Membership object.
readMembershipIdsForPerson	To obtain the set of identifiers for all of the Membership objects for the identified Person object.
readMembershipIdsForPersonWithRole	To obtain the set of identifiers for all of the Membership objects for the identified Person object with a specific Member Role.
readMembershipIdsForCollection	To obtain the set of identifiers for all of the Membership objects for the identified collection object i.e., Group, CourseTemplate, CourseOffering, CourseSection or SectionAssociation.
readAllMembershipIds	To obtain the set of identifiers which have been assigned to Membership objects.
readMembershipIdsFromSavePoint	To obtain the set of identifiers for Membership 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.
readMemberships	To obtain the Membership 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.
readMembershipsFromSavePoint	To obtain the set of Membership 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.
updateMembership	To write new content into the identified Membership object. The target must write the new data into the Membership object. This is an additive operation.
replaceMembership	To replace the content of the identified Membership object. The target must write the new data into the Membership object. This is a destructive write-over of all of the original information. If the Membership object does not exist, this operation acts as per a ‘createMembership’ request.
discoverMembershipIds	To obtain the set of identifiers for Membership objects whose properties agree with those defined in the query.
changeMembershipIdentifier	To change the SourcedId of the Membership record. The completion of this operation will result in subsequent actions using the original SourcedId reporting an unknown identifier status.

Appendix D Course Management Service 1.0 Overview

D1 CMS Description

The Course Management Service is used to model the service responsible for manipulating information about course structures. The CourseManagementService interfaces are shown in Figures D1.1, D1.2, D1.3 and D1.4.

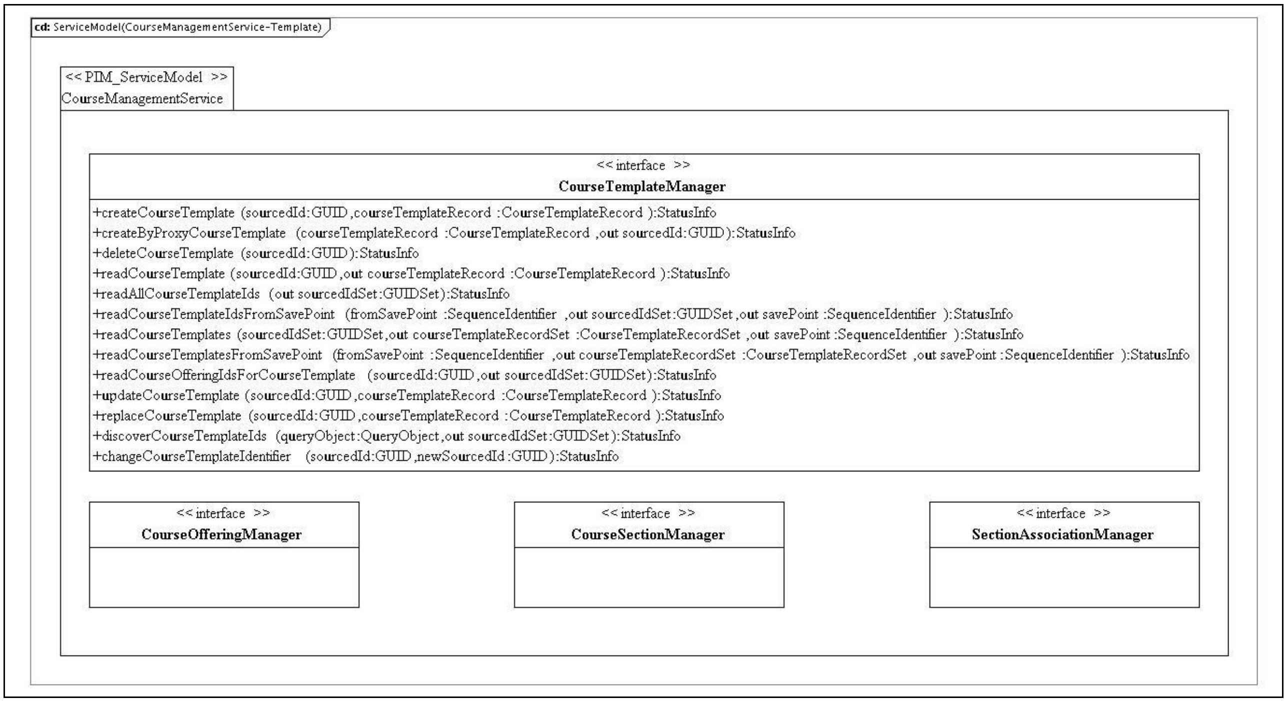

cd: ServiceModel(CourseManagementService-Template)
<< PIM_ServiceModel >>
CourseManagementService
<< interface >>
CourseTemplateManager
+createCourseTemplate (sourcedId:GUID ,courseTemplateRecord :CourseTemplateRecord):StatusInfo
+createByProxyCourseTemplate (courseTemplateRecord :CourseTemplateRecord ,out sourcedId:GUID):StatusInfo
+deleteCourseTemplate (sourcedId:GUID):StatusInfo
+readCourseTemplate (sourcedId:GUID ,out courseTemplateRecord :CourseTemplateRecord):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 :CourseTemplateRecord):StatusInfo
+replaceCourseTemplate (sourcedId:GUID ,courseTemplateRecord :CourseTemplateRecord):StatusInfo
+discoverCourseTemplateIds (queryObject:QueryObject,out sourcedIdSet:GUIDSet):StatusInfo
+changeCourseTemplateIdentifier (sourcedId:GUID ,newSourcedId :GUID):StatusInfo
<< interface >>
CourseOfferingManager
<< interface >>
CourseSectionManager
<< interface >>
SectionAssociationManager

Figure D1.1 CourseManagementService CourseTemplateManager interface definition.

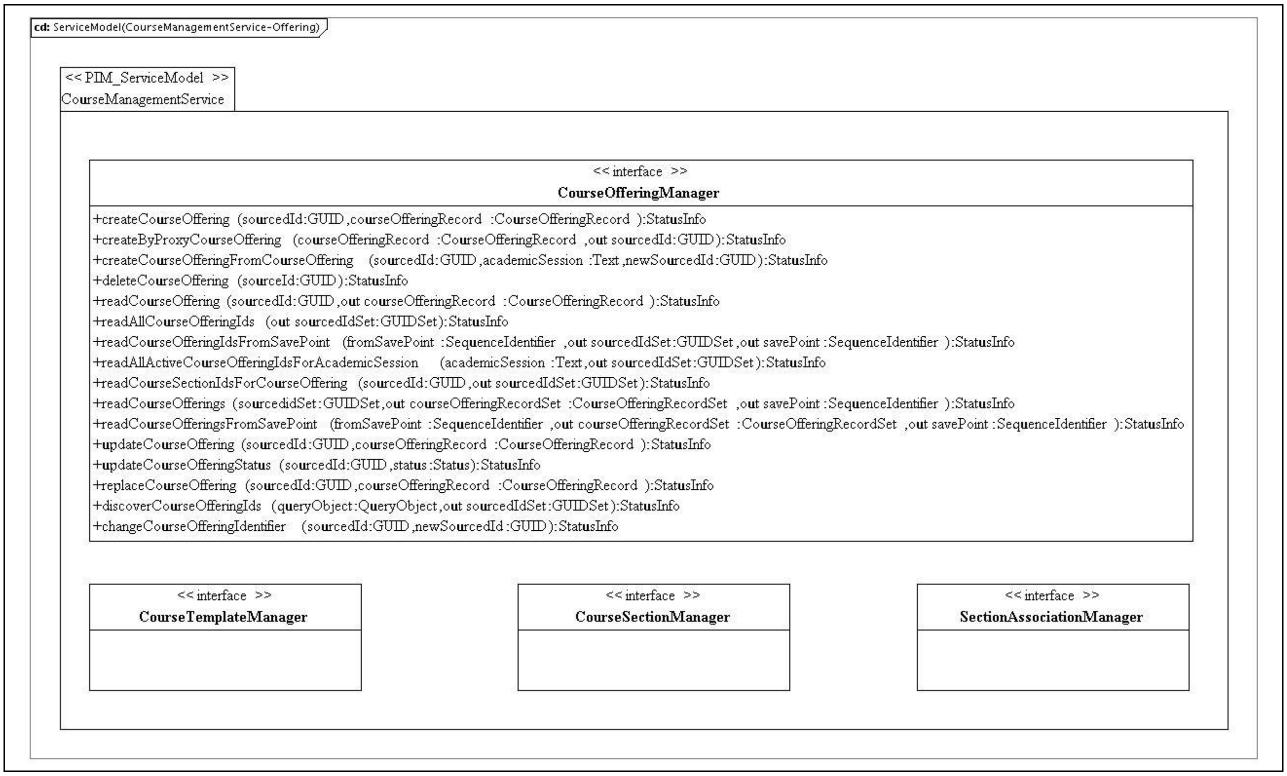

cd: ServiceModel(CourseManagementService-Offering)
<< PIM_ServiceModel >>
CourseManagementService
<< interface >>
CourseOfferingManager
+createCourseOffering (sourcedId:GUID ,courseOfferingRecord :CourseOfferingRecord):StatusInfo
+createByProxyCourseOffering (courseOfferingRecord :CourseOfferingRecord ,out sourcedId:GUID):StatusInfo
+createCourseOfferingFromCourseOffering (sourcedId:GUID ,academicSession :Text ,newSourcedId:GUID):StatusInfo
+deleteCourseOffering (sourceId: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
<< interface >>
CourseTemplateManager
<< interface >>
CourseSectionManager
<< interface >>
SectionAssociationManager

Figure D1.2 CourseManagementService CourseOfferingManager interface definition.

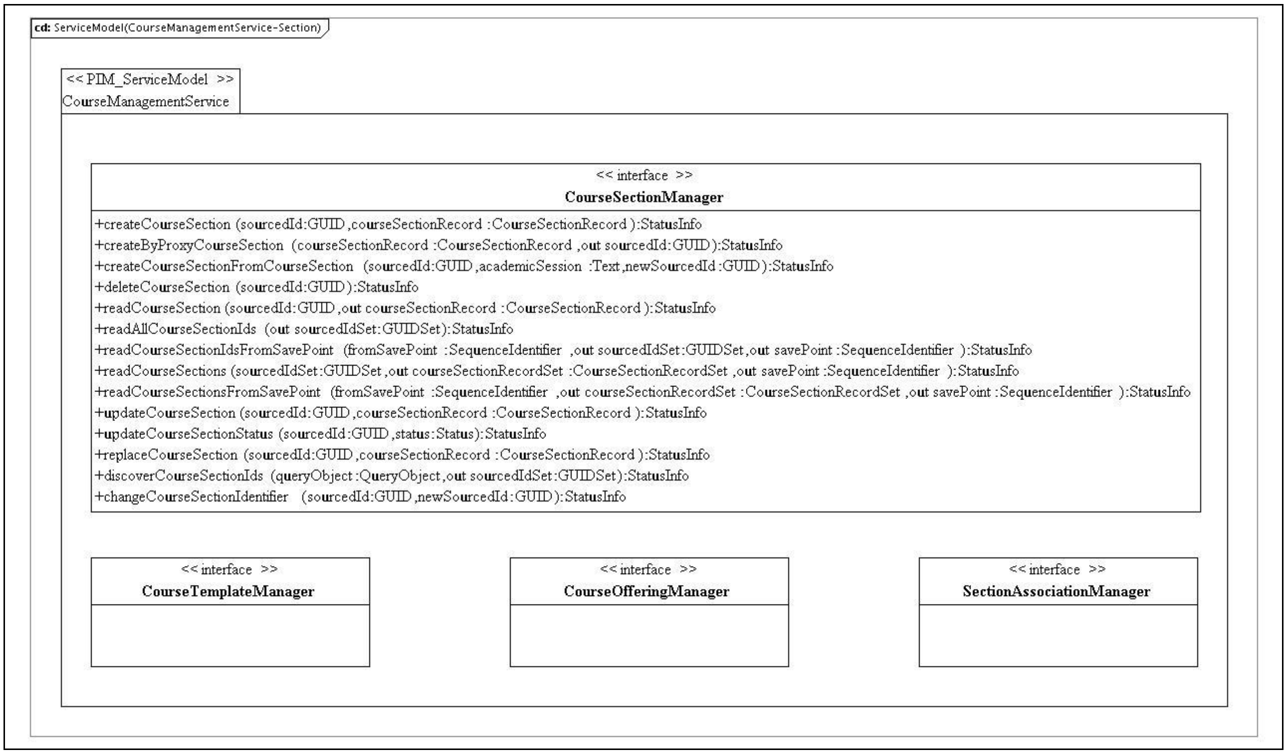

cd: ServiceModel(CourseManagementService-Section)
<< PIM_ServiceModel >>
CourseManagementService
<< interface >>
CourseSectionManager
+createCourseSection (sourcedId:GUID ,courseSectionRecord :CourseSectionRecord):StatusInfo
+createByProxyCourseSection (courseSectionRecord :CourseSectionRecord ,out sourcedId:GUID):StatusInfo
+createCourseSectionFromCourseSection (sourcedId:GUID ,academicSession :Text ,newSourcedId :GUID):StatusInfo
+deleteCourseSection (sourcedId:GUID):StatusInfo
+readCourseSection (sourcedId:GUID ,out courseSectionRecord :CourseSectionRecord):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 :CourseSectionRecord):StatusInfo
+updateCourseSectionStatus (sourcedId :GUID ,status:Status):StatusInfo
+replaceCourseSection (sourcedId:GUID ,courseSectionRecord :CourseSectionRecord):StatusInfo
+discoverCourseSectionIds (queryObject :QueryObject,out sourcedIdSet:GUIDSet):StatusInfo
+changeCourseSectionIdentifier (sourcedId:GUID ,newSourcedId:GUID):StatusInfo
<< interface >>
CourseTemplateManager
<< interface >>
CourseOfferingManager
<< interface >>
SectionAssociationManager

Figure D1.3 CourseManagementService CourseSectionManager interface definition.

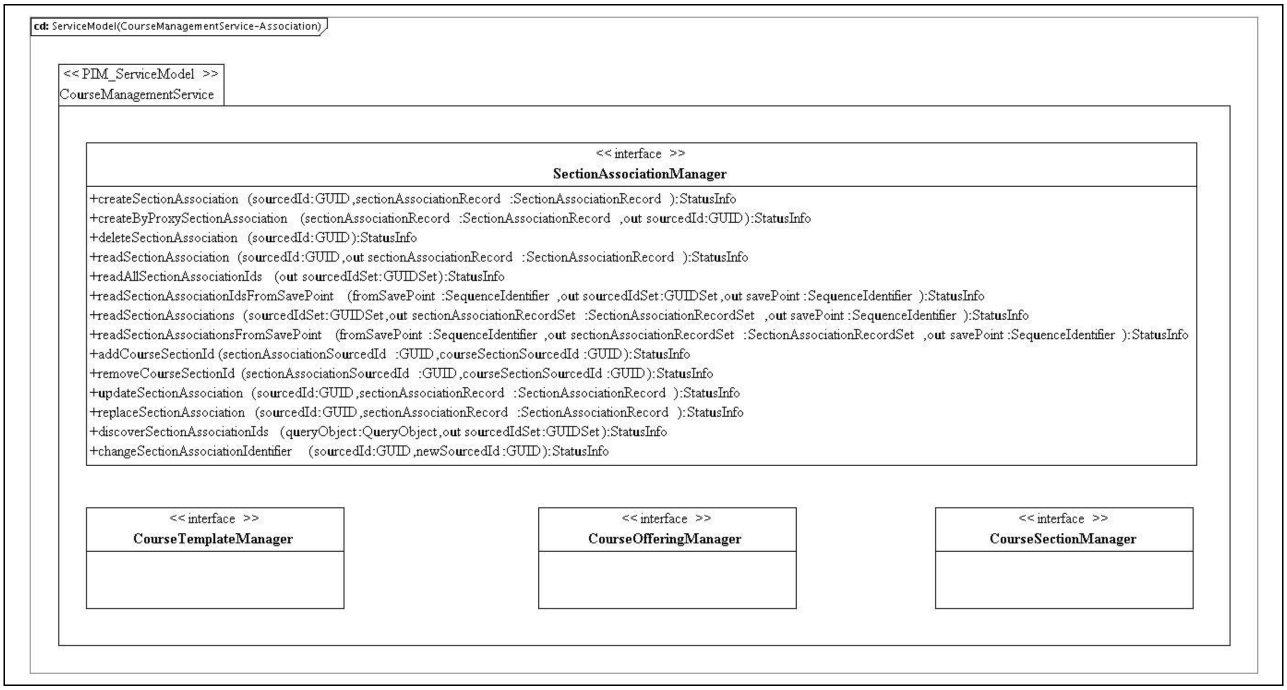

cd: ServiceModel(CourseManagementService-Association)
<< PIM_ServiceModel >>
CourseManagementService
<< interface >>
SectionAssociationManager
+createSectionAssociation (sourcedId:GUID ,sectionAssociationRecord :SectionAssociationRecord):StatusInfo
+createByProxySectionAssociation (sectionAssociationRecord :SectionAssociationRecord ,out sourcedId:GUID):StatusInfo
+deleteSectionAssociation (sourcedId:GUID):StatusInfo
+readSectionAssociation (sourcedId:GUID ,out sectionAssociationRecord :SectionAssociationRecord):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 :SectionAssociationRecord):StatusInfo
+replaceSectionAssociation (sourcedId:GUID ,sectionAssociationRecord :SectionAssociationRecord):StatusInfo
+discoverSectionAssociationIds (queryObject:QueryObject,out sourcedIdSet:GUIDSet):StatusInfo
+changeSectionAssociationIdentifier (sourcedId:GUID ,newSourcedId :GUID):StatusInfo
<< interface >>
CourseTemplateManager
<< interface >>
CourseOfferingManager
<< interface >>
CourseSectionManager

Figure D1.4 CourseManagementService SectionAssociationManager interface definition.

D2 CMS Operations

The Course Management Service is defined by four interfaces: CourseTemplateManager that supports the manipulation of CourseTemplates; CourseOfferingManager that supports the manipulation of CourseOfferings; CourseSectionManager that supports the manipulation of CourseSections; and SectionAssociation Manager that supports the manipulation of SectionAssociations.

D2.1 CourseTemplateManager Interface Description

The CourseTemplateManager interface class 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 D2.1.

Table D2.1 Summary of operations for CourseTemplateManager.

Operation	Description
createCourseTemplate	To request the creation of a populated CourseTemplate object on the target system where the source assigns the unique identifier.
createByProxyCourseTemplate	To request the creation of a populated CourseTemplate object on the target system where the target assigns 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.

Operation	Description
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 get 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. If the CourseTemplate object does not exist, this operation acts as per a 'createCourseTemplate' request.
discoverCourseTemplateIds	To obtain the set of SourcedIds for CourseTemplate objects whose properties agree with those defined in the query.
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.

D2.2 CourseOfferingManager Interface Description

The CourseOfferingManager interface class describes the operations that are permitted on a CourseOffering. The interface stereotype indicates that there are no attributes for this class. The set of operations are summarized in Table D2.2.

Table D2.2 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.
readCourseOffering	To read the full contents of the identified CourseOffering

Operation	Description
	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. If the CourseOffering object does not exist, this operation acts as per a 'createCourseOffering' request.
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.
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.

D2.3 CourseSectionManager Interface Description

The CourseSectionManager interface class describes the operations that are permitted on a CourseSection. The interface stereotype indicates that there are no attributes for this class. The set of operations are summarized in Table D2.3.

Table D2.3 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 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. If the CourseSection object does not exist, this operation acts as per a 'createCourseSection' request.
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.
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.

D2.4 SectionAssociationManager Interface Description

The SectionAssociationManager interface class describes the operations that are permitted on a SectionAssociation. The interface stereotype indicates that there are no attributes. The operations are summarized in Table D2.4.

Table D2.4 Summary of operations for SectionAssociationManager.

Operation	Description
createSectionsAssociation	To request the creation of a populated SectionsAssociation object on the target system where the source is responsible for the allocation of the unique identifier.
createByProxySectionsAssociation	To request the creation of a populated SectionsAssociation 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 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.
addCourseSection	To add a new Course Section identifier to the SectionAssociation.
removeCourseSection	To remove a Course Section 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. If the SectionAssociation object does not exist, this operation acts as per a 'create SectionAssociation' request.
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.
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.

Appendix E Outcomes Management Service 1.0 Overview

E1 OMS Description

The Outcomes Management Service is used to model the service responsible for manipulating information about outcomes. The Outcomes Management Service interfaces are shown in Figures E.1, E.2 and E.3.

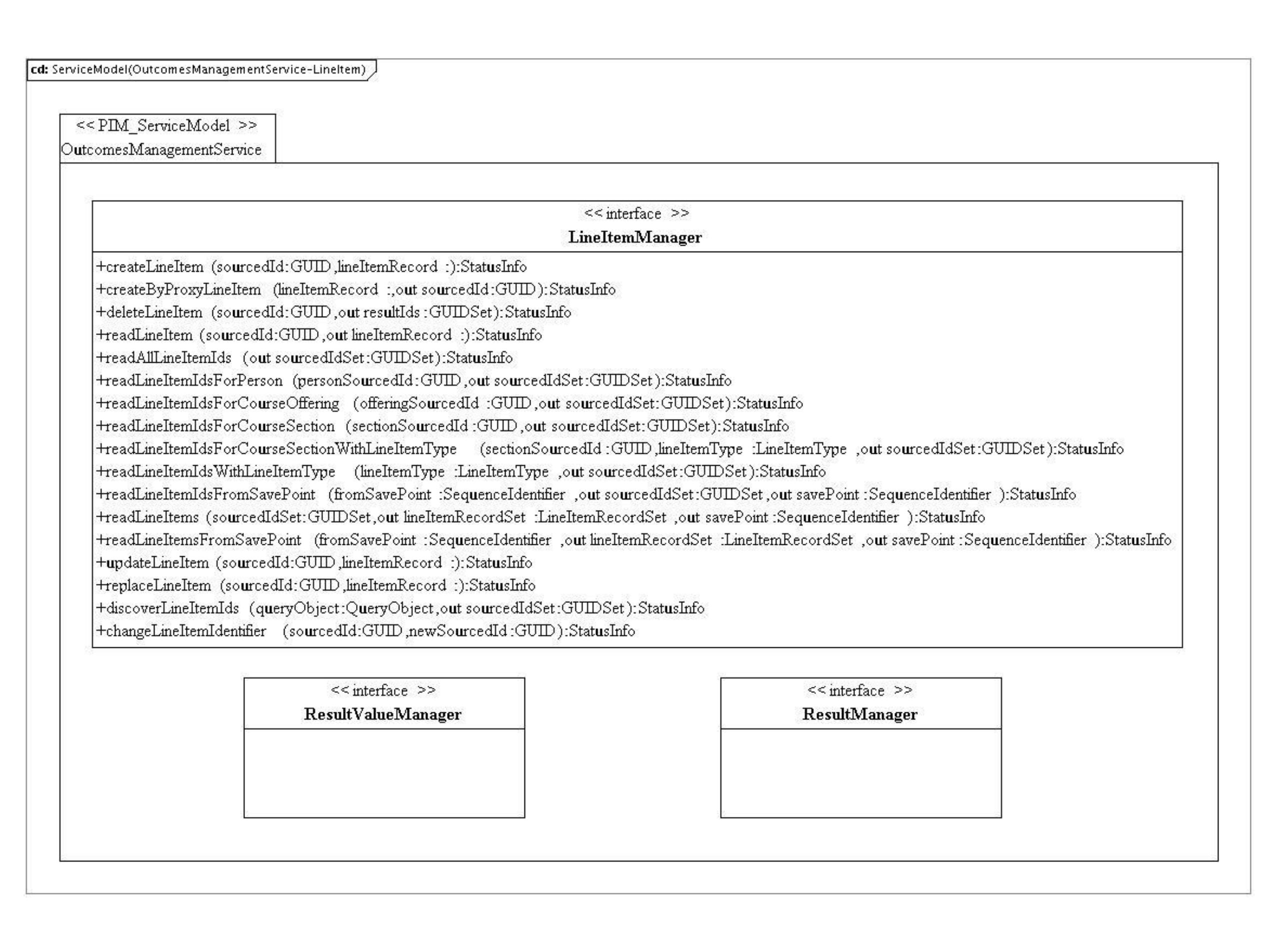

cd: ServiceModel(OutcomesManagementService-LineItem)
<< PIM_ServiceModel >>
OutcomesManagementService
<< interface >>
LineItemManager
+createLineItem (sourcedId:GUID ,lineItemRecord :):StatusInfo
+createByProxyLineItem (lineItemRecord :,out sourcedId:GUID):StatusInfo
+deleteLineItem (sourcedId:GUID ,out resultIds :GUIDSet):StatusInfo
+readLineItem (sourcedId:GUID ,out lineItemRecord :):StatusInfo
+readAllLineItemIds (out sourcedIdSet:GUIDSet):StatusInfo
+readLineItemIdsForPerson (personSourcedId:GUID ,out sourcedIdSet:GUIDSet):StatusInfo
+readLineItemIdsForCourseOffering (offeringSourcedId :GUID ,out sourcedIdSet:GUIDSet):StatusInfo
+readLineItemIdsForCourseSection (sectionSourcedId :GUID ,out sourcedIdSet:GUIDSet):StatusInfo
+readLineItemIdsForCourseSectionWithLineItemType (sectionSourcedId :GUID ,lineItemType :LineItemType ,out sourcedIdSet:GUIDSet):StatusInfo
+readLineItemIdsWithLineItemType (lineItemType :LineItemType ,out sourcedIdSet:GUIDSet):StatusInfo
+readLineItemIdsFromSavePoint (fromSavePoint :SequenceIdentifier ,out sourcedIdSet:GUIDSet ,out savePoint :SequenceIdentifier):StatusInfo
+readLineItems (sourcedIdSet:GUIDSet,out lineItemRecordSet :LineItemRecordSet ,out savePoint :SequenceIdentifier):StatusInfo
+readLineItemsFromSavePoint (fromSavePoint :SequenceIdentifier ,out lineItemRecordSet :LineItemRecordSet ,out savePoint :SequenceIdentifier):StatusInfo
+updateLineItem (sourcedId:GUID ,lineItemRecord :):StatusInfo
+replaceLineItem (sourcedId:GUID ,lineItemRecord :):StatusInfo
+discoverLineItemIds (queryObject:QueryObject,out sourcedIdSet:GUIDSet):StatusInfo
+changeLineItemIdentifier (sourcedId:GUID ,newSourcedId :GUID):StatusInfo
<< interface >>
ResultValueManager
<< interface >>
ResultManager

Figure E1.1 OutcomesManagementService LineItemManager interface definitions.

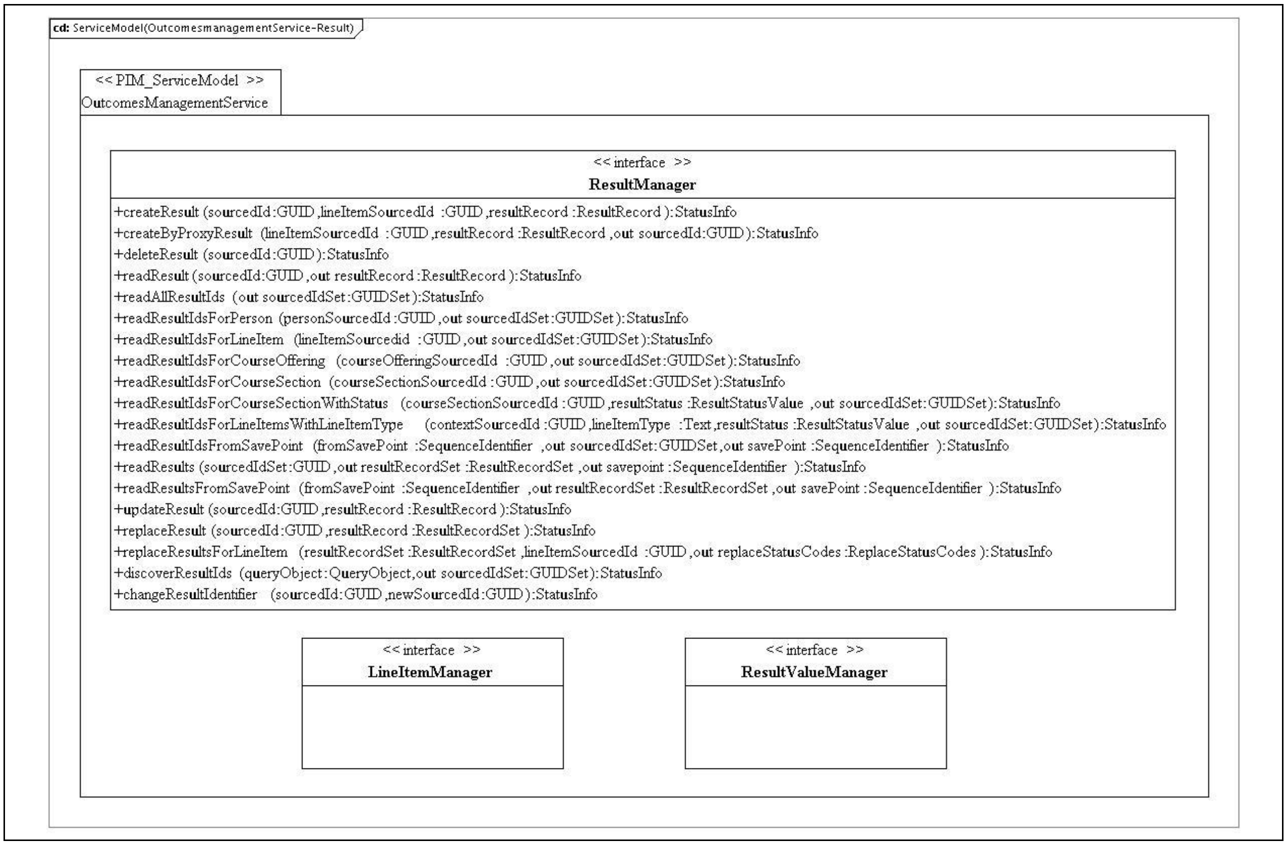

cd: ServiceModel(OutcomesmanagementService-Result)
<< PIM_ServiceModel >>
OutcomesManagementService
<< interface >>
ResultManager
+createResult (sourcedId:GUID ,lineItemSourcedId :GUID ,resultRecord :ResultRecord):StatusInfo
+createByProxyResult (lineItemSourcedId :GUID ,resultRecord :ResultRecord ,out sourcedId:GUID):StatusInfo
+deleteResult (sourcedId:GUID):StatusInfo
+readResult (sourcedId:GUID ,out resultRecord :ResultRecord):StatusInfo
+readAllResultIds (out sourcedIdSet:GUIDSet):StatusInfo
+readResultIdsForPerson (personSourcedId :GUID ,out sourcedIdSet:GUIDSet):StatusInfo
+readResultIdsForLineItem (lineItemSourcedid :GUID ,out sourcedIdSet:GUIDSet):StatusInfo
+readResultIdsForCourseOffering (courseOfferingSourcedId :GUID ,out sourcedIdSet:GUIDSet):StatusInfo
+readResultIdsForCourseSection (courseSectionSourcedId :GUID ,out sourcedIdSet:GUIDSet):StatusInfo
+readResultIdsForCourseSectionWithStatus (courseSectionSourcedId :GUID ,resultStatus :ResultStatusValue ,out sourcedIdSet:GUIDSet):StatusInfo
+readResultIdsForLineItemsWithLineItemType (contextSourcedId :GUID ,lineItemType :Text ,resultStatus :ResultStatusValue ,out sourcedIdSet:GUIDSet):StatusInfo
+readResultIdsFromSavePoint (fromSavePoint :SequenceIdentifier ,out sourcedIdSet:GUIDSet ,out savePoint :SequenceIdentifier):StatusInfo
+readResults (sourcedIdSet:GUID ,out resultRecordSet :ResultRecordSet ,out savepoint :SequenceIdentifier):StatusInfo
+readResultsFromSavePoint (fromSavePoint :SequenceIdentifier ,out resultRecordSet :ResultRecordSet ,out savePoint :SequenceIdentifier):StatusInfo
+updateResult (sourcedId:GUID ,resultRecord :ResultRecord):StatusInfo
+replaceResult (sourcedId:GUID ,resultRecord :ResultRecordSet):StatusInfo
+replaceResultsForLineItem (resultRecordSet :ResultRecordSet ,lineItemSourcedId :GUID ,out replaceStatusCodes :ReplaceStatusCodes):StatusInfo
+discoverResultIds (queryObject:QueryObject,out sourcedIdSet:GUIDSet):StatusInfo
+changeResultIdentifier (sourcedId:GUID ,newSourcedId:GUID):StatusInfo
<< interface >>
LineItemManager
<< interface >>
ResultValueManager

Figure E1.2 OutcomesManagementService ResultManager interface definitions.

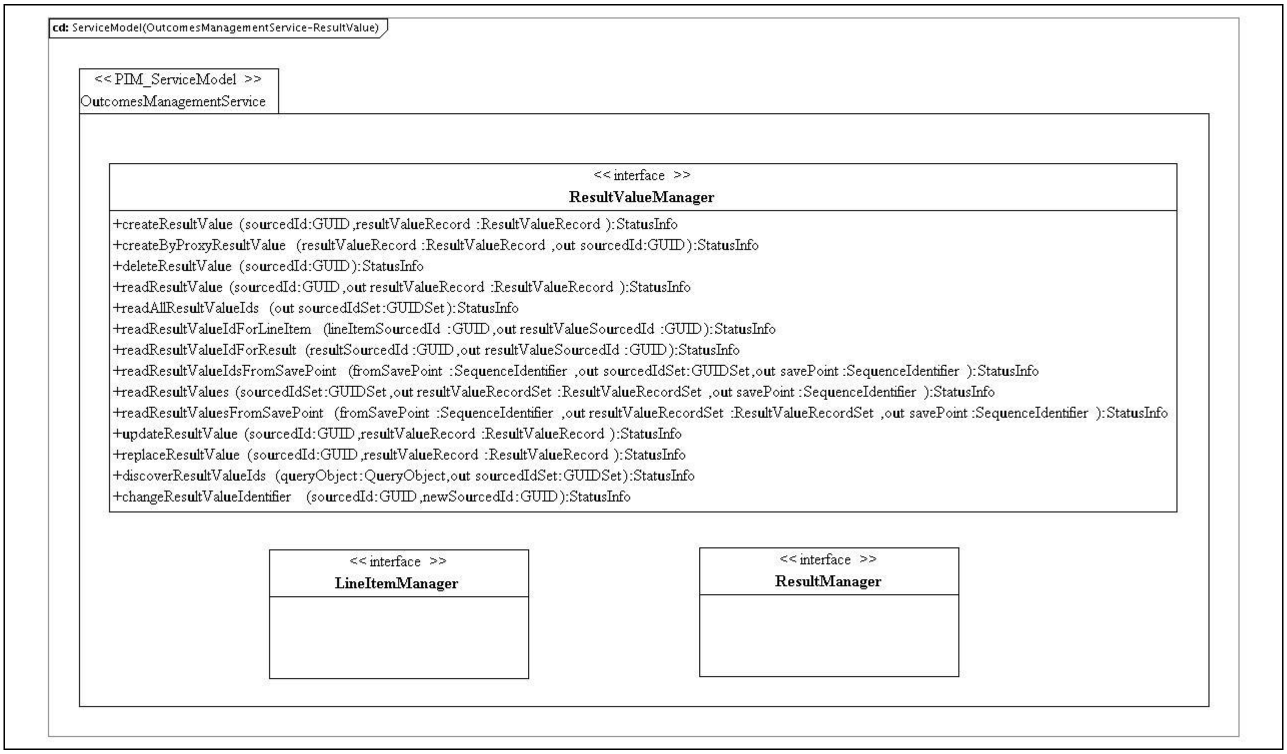

cd: ServiceModel(OutcomesManagementService-ResultValue)
<< PIM_ServiceModel >>
OutcomesManagementService
<< interface >>
ResultValueManager
+createResultValue (sourcedId:GUID ,resultValueRecord :ResultValueRecord):StatusInfo
+createByProxyResultValue (resultValueRecord :ResultValueRecord ,out sourcedId:GUID):StatusInfo
+deleteResultValue (sourcedId:GUID):StatusInfo
+readResultValue (sourcedId:GUID ,out resultValueRecord :ResultValueRecord):StatusInfo
+readAllResultValueIds (out sourcedIdSet:GUIDSet):StatusInfo
+readResultValueIdForLineItem (lineItemSourcedId :GUID ,out resultValueSourcedId :GUID):StatusInfo
+readResultValueIdForResult (resultSourcedId :GUID ,out resultValueSourcedId :GUID):StatusInfo
+readResultValueIdsFromSavePoint (fromSavePoint :SequenceIdentifier ,out sourcedIdSet:GUIDSet ,out savePoint :SequenceIdentifier):StatusInfo
+readResultValues (sourcedIdSet:GUIDSet ,out resultValueRecordSet :ResultValueRecordSet ,out savePoint :SequenceIdentifier):StatusInfo
+readResultValuesFromSavePoint (fromSavePoint :SequenceIdentifier ,out resultValueRecordSet :ResultValueRecordSet ,out savePoint :SequenceIdentifier):StatusInfo
+updateResultValue (sourcedId:GUID ,resultValueRecord :ResultValueRecord):StatusInfo
+replaceResultValue (sourcedId:GUID ,resultValueRecord :ResultValueRecord):StatusInfo
+discoverResultValueIds (queryObject:QueryObject,out sourcedIdSet:GUIDSet):StatusInfo
+changeResultValueIdentifier (sourcedId:GUID ,newSourcedId:GUID):StatusInfo
<< interface >>
LineItemManager
<< interface >>
ResultManager

Figure E1.3 OutcomesManagementService ResultValueManager interface definitions.

E2 OMS Operations

The Outcomes Management Service is split into three interfaces: LineItemManager that supports the manipulation of LineItem objects; ResultManager that supports the manipulation of Result objects; ResultValueManager that supports the manipulation of ResultValue objects.

E2.1 LineItemManager Interface Description

The LineItemManager interface class describes the operations that are permitted on LineItems object (as shown in Figure E1.1). The set of operations are summarized in Table E2.1.

Table E2.1 Summary of operations for LineItemManager.

Operation	Description
createLineItem	To request the creation of a populated LineItem object on the target system where the source is responsible for the allocation of the unique SourcedId.
createByProxyLineItem	To request the creation of a populated LineItem object on the target system where the target is responsible for the allocation of the unique SourcedId.
deleteLineItem	To request the deletion of a LineItem object. The LineItem object is deleted with all of its associated relationships.
readLineItem	To read the full contents of the identified LineItem object. The target must return all of the data it has for the identified LineItem object.
readAllLineItemIds	To obtain all the SourcedIds assigned to LineItem objects.
readLineItemIdsWithLineItemType	To obtain the set of SourcedIds for the LineItem objects which have a particular state.
readLineItemIdsForPerson	To obtain the set of SourcedIds for the LineItem objects which are associated with a particular Person object.
readLineItemIdsForCourseOffering	To obtain the set of SourcedIds for the LineItem objects which are associated with a particular CourseOffering object.
readLineItemIdsForCourseSection	To obtain the set of SourcedIds for the LineItem objects which are associated with a particular CourseSection object.
readLineItemIdsForCourseSectionWithLineItemType	To obtain the set of SourcedIds for the LineItem objects with the required state, which are associated with a particular CourseSection object.
readLineItemIdsFromSavePoint	To obtain the set of SourcedIds for LineItem 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.
readLineItems	To obtain the LineItem 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.
readLineItemsFromSavePoint	To obtain the set of LineItem 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 exchange a large volume of data in the response.
updateLineItem	To write new content into the identified LineItem object. The target must write the new data into the LineItem object. This is an additive operation.
replaceLineItem	To replace the content of the identified LineItem object. The target must write the new data into the LineItem 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 'createLineItem'.
discoverLineItemIds	To obtain the set of SourcedIds for LineItem objects whose properties agree with those defined in the query/filter.
changeLineItemIdentifier	To change the SourcedId for a LineItem object.

E2.2 ResultManager Interface Description

The ResultManager interface class describes the operations that are permitted on Result objects (as shown Figure E1.2). The set of operations are summarized in Table E2.2.

Table E2.2 Summary of operations for ResultManager.

Operation	Description
createResult	To request the creation of a populated Result object on the target system where the source allocates the unique SourcedId. The Result object is tied to a LineItem object.
createByProxyResult	To request the creation of a populated Result object on the target system where the target allocates the unique SourcedId. The Result object is tied to a LineItem object.
deleteResult	To request the deletion of a Result object. The Result object is deleted but the associated ResultValue and LineItems remain.
readResult	To read the full contents of the identified Result object. The target must return all of the data it has for the identified Result object.
readAllResultIds	To obtain the set of SourcedIds assigned to Result objects.
readResultIdsForPerson	To obtain the set of SourcedIds for the Result objects which are associated with a particular Person object.
readResultIdsForLineItem	To obtain the set of SourcedIds for the Result objects which are associated with a particular LineItem object.
readResultIdsForCourseOffering	To obtain the set of SourcedIds for the Result objects which are associated with a particular CourseOffering object.
readResultIdsForCourseSection	To obtain the set of SourcedIds for the Result objects which are associated with a particular CourseSection object.
readResultIdsForCourseSectionWithStatus	To obtain the set of SourcedIds for the Result objects that are associated with a particular CourseSection object and have results of a particular status.
readResultIdsForLineItemWithLineItemType	To obtain the SourcedIds for the Result objects that are associated with a LineItem that has a particular LineItemType state.

readResultIdsFromSavePoint	To obtain the set of identifiers for Result 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.
readResults	To obtain the Result objects for a defined set of identifiers. This results in a single transaction that may exchange of a large volume of data.
readResultsFromSavePoint	To obtain the set of Result 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.
updateResult	To add content to the identified Result object. The target must write the new data into the Result object. This is an additive operation.
replaceResult	To replace the content of the identified Result object. The target must write the new data into the Result 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 ‘createResult’.
replaceResultsForLineItem	To replace the content of the identified Result objects for a given result status for the specified LineItem. The target must write the new data into the Result object. This is a destructive write-over of all of the original information.
discoverResultIds	To obtain the set of SourcedIds for Result objects whose properties agree with those defined in the query/filter.
changeResultIdentifier	To change the SourcedId of a Result object.

E2.3 ResultValueManager Interface Description

The ResultValueManager interface class describes the operations that are permitted on ResultValue objects (as shown Figure E1.3). The set of operations are summarized in Table E2.3.

Table E2.3 Summary of operations for ResultValueManager.

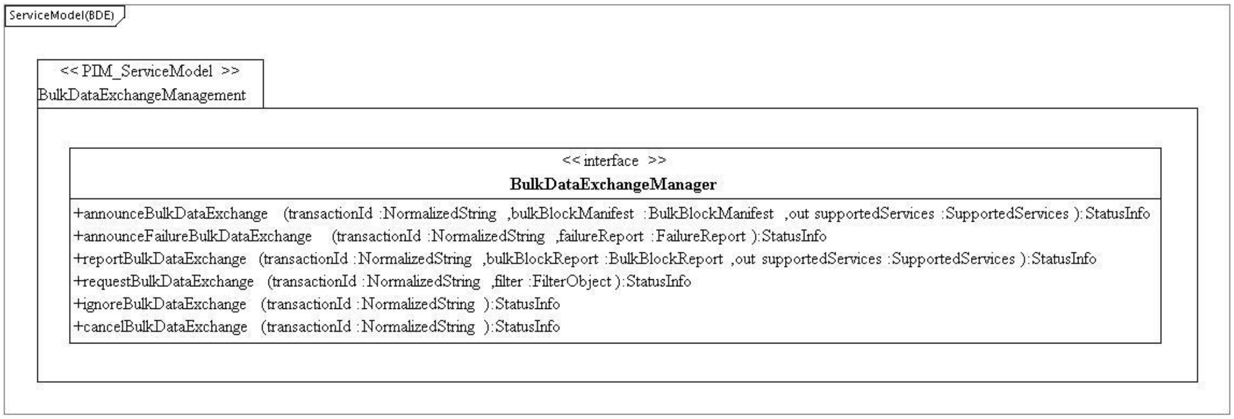
Operation	Description
createResultValue	To request the creation of a populated ResultValue object on the target system. The source is responsible for the allocation of the unique SourcedId.
createByProxyResultValue	To request the creation of a populated ResultValue object on the target system. The target is responsible for the allocation of the unique SourcedId.
deleteResultValue	To request the deletion of a ResultValue object. The ResultValue object is deleted along with all of its associated relationships.
readResultValue	To read the full contents of the identified ResultValue object. The target must return all of the data it has for the identified ResultValue object.
readAllResultValueIds	To obtain the set of SourcedIds that have been assigned to ResultValue objects.
readResultValueIdForLineItem	To obtain the SourcedId for the ResultValue object which has been associated with a particular LineItem object.
readResultValueIdForResult	To obtain the SourcedId for the ResultValue object which is associated with a particular Result object.

readResultValueIdsFromSavePoint	To obtain the set of identifiers for ResultValue 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.
readResultValues	To obtain the ResultValue 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.
readResultValuesFromSavePoint	To obtain the set of ResultValue 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 exchange of a large volume of data in the response message.
updateResultValue	To write new content into the identified ResultValue object. The target must write the new data into the ResultValue object. This is an additive operation.
replaceResultValue	To replace the content of the identified ResultValue object. The target must write the new data into the ResultValue 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 ‘createResultValue’.
discoverResultValueIds	To obtain the set of SourcedIds for ResultValue objects whose properties agree with those defined in the query/filter.
changeResultValueIdentifier	To change the SourcedId for a ResultValue object.

Appendix F Bulk Data Exchange Management Service 1.0 Overview

F1 BDEMS Description

The Bulk Data Exchange Management Service is used to model the service responsible for manipulating information used in Learning Information Services systems. The BulkDataExchangeManagementService interface is shown in Figure F1.1.



```
+announceBulkDataExchange (transactionId :NormalizedString ,bulkBlockManifest :BulkBlockManifest ,out supportedServices :SupportedServices ):StatusInfo
+announceFailureBulkDataExchange (transactionId :NormalizedString ,failureReport :FailureReport ):StatusInfo
+reportBulkDataExchange (transactionId :NormalizedString ,bulkBlockReport :BulkBlockReport ,out supportedServices :SupportedServices ):StatusInfo
+requestBulkDataExchange (transactionId :NormalizedString ,filter :FilterObject ):StatusInfo
+ignoreBulkDataExchange (transactionId :NormalizedString ):StatusInfo
+cancelBulkDataExchange (transactionId :NormalizedString ):StatusInfo
```

Figure F1.1 BulkDataExchangeManagementService interface definition.

F2 BDEMS Operations

The BulkDataExchangeManager interface class describes the operations that are permitted on ‘bulk data’ objects. The interface stereotype indicates that there are no attributes for this class. The set of operations are summarized in Table F2.1.

Table F2.1 Summary of operations for BulkDataExchangeManager.

Operation	Description
announceBulkDataExchange	To announce the availability of a bulk data object that is ready to be retrieved by service consumers. This call suspends real-time event notification
announceFailureBulkDataExchange	To inform the service consumer that a previously issued, and acknowledged, requestBulkDataExchange transaction cannot be serviced by the service provider.
reportBulkDataExchange	To report the retrieval of the bulk data object whose availability was announced previously. This signals real-time event processing should be restarted.
requestBulkDataExchange	Issued by the service consumer to request a bulk data exchange from the service provider.
ignoreBulkDataExchange	Issued by the service consumer to inform the service provider that a previously issued announce bulk data exchange request will be ignored. Real-time processing should be resumed by both consumer and server.

cancelBulkDataExchange	Issued by the service consumer to inform the service provider that a previously requested bulk data exchange request has now been cancelled. Real-time processing should be resumed by both consumer and server.

NOTE: The operation used to download the data file(s) is not defined in Table F2.1. Instead, the specific Web Service binding is recommended in [BDEMS, 11].

About This Document

Title:	IMS GLC Learning Information Services Specification
Editor:	Colin Smythe (IMS GLC)
Co-chairs:	Linda Feng (Oracle) and Bill Lee (Desire2learn)
Version:	2.0
Version Date:	30 June 2011
Release:	Final 1.0
Status:	**Final Release**
Summary:	This document contains the description of the IMS GLC Learning Information Services (LIS) specification. LIS is a collection of six component services that are combined to provide the required functionality. These component services are: Person Management Service; Group Management Service; Membership Management Service; Course Management Service; Outcomes Management Service; and Bulk Data Exchange Management Service. As part of the specification a Core Profile has been created for Learning Management System/Student Information System interaction.
Revision Information:	Original Final 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	Cengage (USA)	Colin Smythe	IMS GLC (UK)
Linda Feng	Oracle (USA)	Reinhold Staudinger	Blackboard (USA)
Chris Hatton	Pearson (USA)	Nick Terrible	University of Wisconsin (USA)
John 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

A

Abstract Framework 6, 7, 8
Addition Profiles 18

B

Binding technologies
  SOAP ..2, 6, 12, 13, 17, 20, 22
  WSDL ..2, 3, 4, 6, 8, 9, 13, 19, 20, 21, 22
  XSD 2, 4, 8, 13, 19, 20, 21, 22
Bulk Data Exchange Management Service .. 2, 3, 4, 6, 7, 8, 16, 17, 18, 20, 21, 42, 43, 44

C

Conformance 2, 3, 4, 19, 20
Core Profile2, 3, 4, 8, 12, 18, 20, 22, 44
Core Profiles 3, 12, 18, 20, 22
Course2, 3, 4, 6, 7, 8, 10, 11, 15, 17, 18, 27, 29, 31, 35, 44
Course Management Service .. 3, 4, 6, 7, 8, 15, 17, 18, 20, 27, 29, 31, 44
Course Structures
  CourseOffering 15, 28, 32, 33, 38, 39
  CourseSection. 15, 28, 34, 38, 39
  CourseTemplate15, 28, 31, 32
  Offering 10, 11
  Section 10, 11, 35
  SectionAssociation 15, 28, 31, 35
  Template 10, 11

G

Group Management Service 3, 4, 6, 7, 8, 14, 15, 25, 44

I

Interface Class
  BulkDataExchangeManager 5, 42
  GroupManager............... 5, 25
  GroupsManager 14
  LineItemManager 4, 5, 36, 38
  MembershipManager......... 27
  MembershipsManager 15
  PersonManager 5, 23
  ResultManager4, 5, 37, 38, 39
  ResultValueManager. 4, 5, 37, 38, 40

L

LDAP.................................6, 7
Learning Information Services 1, 2, 3, 6, 7, 8, 10, 16, 21, 42, 44
Lightweight Directory Access Protocol............................6, 7
LIS
  Bulk Data Exchange Management Service... 2, 3, 4, 6, 7, 8, 16, 17, 18, 20, 21, 42, 43, 44
  Course Management Service 3, 4, 6, 7, 8, 15, 17, 18, 20, 27, 29, 31, 44
  Group Management Service 3, 4, 6, 7, 8, 14, 15, 17, 20, 25, 27, 44
  Membership Management Service 3, 4, 5, 6, 7, 8, 9, 15, 16, 17, 18, 20, 27, 44
  Outcomes Management Service 3, 4, 6, 7, 9, 15, 16, 18, 20, 36, 38, 44
  Person Management Service3, 4, 6, 7, 9, 14, 17, 18, 20, 23, 44

M

Membership Management Service 3, 4, 6, 7, 8, 9, 15, 16, 18, 27, 44

O

Operations
  BDEMS
    announceBulkDataExchange...........................18, 42
    announceFailureBulkDataExchange 42
    cancelBulkDataExchange 43
    ignoreBulkDataExchange 18, 42
    reportBulkDataExchange 18, 42
    requestBulkDataExchange....................................... 42
  Course
    addCourseSection...........35
    changeCourseOfferingIdentifier..............................33
    changeCourseSectionIdentifier..................................34
    changeCourseTemplateIdentifier 32
    changeSectionAssociationIdentifier 35
    createByProxyCourseOffering 32
    createByProxyCourseSection 34
    createByProxyCourseTemplate 31
    createCourseOffering ...32, 33
    createCourseOfferingFromCourseOffering 32
    createCourseSection......34
    createCourseSectionFromCourseSection..............34
    createCourseTemplate 31, 32
    deleteCourseOffering 32
    deleteCourseSection.....18, 34
    deleteCourseTemplate .31
    deleteSectionAssociation 35
    discoverCourseOfferingIds...33
    discoverCourseSectionIds 34
    discoverCourseTemplateIds.....................................32
    discoverSectionAssociation 35
    readAllActiveCourseOfferingIdsForAcademicSession 33
    readAllCourseOfferingIds 33
    readAllCourseSectionIds 34
    readAllCourseTemplateIds...31
    readCourseOffering 32
    readCourseOfferingIdsFromSavePoint 33
    readCourseOfferings 33
    readCourseOfferingsFromSavePoint......................33

- readCourseSection. 18, 34
- readCourseSectionIdsForCourseOffering 33
- readCourseSectionIdsFromSavePoint 34
- readCourseSections 34
- readCourseSectionsFromSavePoint 34
- readCourseTemplate 31
- readCourseTemplateIdsFromSavePoint 32
- readCourseTemplates 32
- readCourseTemplatesFromSavePoint 32
- readSectionAssociation 35
- readSectionAssociationIdsFromSavePoint 35
- readSectionAssociationsFromSavePoint 35
- removeCourseSection 35
- replaceCourseOffering 33
- replaceCourseSection 18, 34
- replaceCourseTemplate 32
- replaceSectionAssociation 35
- updateCourseOffering 33
- updateCourseOfferingStatus 33
- updateCourseSection 34
- updateCourseSectionStatus 34
- updateCourseTemplate 32
- updateSectionAssociation 35

Group
- addGroupRelationship 25
- changeGroupIdentifier 26
- createByProxyGroup 25
- createGroup 25, 26
- deleteGroup 25
- discoverGroupIds 26
- readGroup 26
- readGroupIdsForPerson 26
- readGroupIdsFromSavePoint 26
- readGroups 26
- readGroupsFromSavePoint 26
- removeGroupRelationship 26
- replaceGroup 26
- updateGroup 26

Membership
- changeMembershipIdentifier 28
- createByProxyMembership 27
- createMembership 27, 28
- deleteMembership 18, 27
- discoverMembershipIds 28
- readAllMembershipIds 28
- readMembership 18, 27
- readMembershipIdsForPerson 28
- readMembershipIdsFromSavePoint 28
- readMembershipIdsPersonWithRole 28
- readMemberships 28
- readMembershipsFromSavePoint 28
- replaceMembership 18, 28
- updateMembership 28

Outcomes
- changeLineItemIdentifier 39
- changeResultIdentifier 40
- changeResultValueIdentifier 41
- createByProxyLineItem 38
- createByProxyResult 39
- createByProxyResultValue 40
- createLineItem 38, 39
- createResult 39, 40
- createResultValue 40, 41
- deleteLineItem 38
- deleteResult 39
- deleteResultValue 40
- discoverLineItemIds 39
- discoverResultIds 40
- discoverResultValue 41
- readAllLineItemIds 38
- readAllResultIds 39
- readAllResultValueIds 40
- readLineItem 38
- readLineItemIdsForCourseOffering 38
- readLineItemIdsForCourseSection 38
- readLineItemIdsForPerson 38
- readLineItemIdsFromSavePoint 38
- readLineItems 38
- readLineItemsFromSavePoint 38
- readResult 39
- readResultIdsForCourseOffering 39
- readResultIdsForCourseSection 39
- readResultIdsForCourseSectionWithStatus 39
- readResultIdsForLineItem 39
- readResultIdsForPerson 39
- readResultIdsFromSavePoint 40
- readResults 40
- readResultsFromSavePoint 40
- readResultValue 40
- readResultValueIdsFromSavePoint 41
- readResultValues 41
- readResultValuesFromSavePoint 41
- replaceLineItem 39
- replaceResult 40
- replaceResultValue 41
- updateLineItem 39
- updateResult 40
- updateResultValue 41

Person
- changePersonIdentifier 24
- createByProxyPerson 23
- createPerson 23, 24
- deletePerson 18, 23
- discoverPersonIds 24
- readPerson 18, 23
- readPersonCore 23
- readPersons 24
- replacePerson 18, 24
- updatePerson 24

Outcomes Management Service 3, 4, 6, 7, 9, 16, 18, 36, 38, 44

P

Person Management Service 3, 4, 6, 7, 9, 14, 18, 23, 44

S

SectionAssociation 15, 28, 31, 35

Services

Bulk Data Exchange
Management 2, 3, 4, 6, 7, 8, 16, 18, 42, 44
Course Management Service .3, 4, 6, 7, 8, 15, 18, 29, 31, 44
Group Management 3, 4, 6, 7, 8, 14, 15, 25, 44
Membership Management... 3, 4, 6, 7, 8, 9, 15, 16, 18, 27, 44
Outcomes Management Service 3, 4, 6, 7, 9, 16, 18, 36, 38, 44
Person Management 3, 4, 6, 7, 9, 14, 18, 23, 44
SOAP 2, 6, 12, 13, 17, 20, 22
Specifications
Other
LDAP 6, 7

W

WDSL 2, 3, 4, 6, 8, 9, 13, 19, 20, 21, 22

X

XSD.... 2, 4, 8, 13, 19, 20, 21, 22

IMS Global Learning Consortium, Inc. ("IMS GLC") is publishing the information contained in this document ("Specification") for purposes of scientific, experimental, and scholarly collaboration only.

IMS GLC makes no warranty or representation regarding the accuracy or completeness of the Specification.

This material is provided on an "As Is" and "As Available" basis.

The Specification is at all times subject to change and revision without notice.

It is your sole responsibility to evaluate the usefulness, accuracy, and completeness of the Specification as it relates to you.

IMS GLC would appreciate receiving your comments and suggestions.

Please contact IMS GLC through our website at http://www.imsglobal.org.

Please refer to Document Name: IMS GLC IMS LIS Specification v2.0 Final Release v1.0

Date: 30 June 2011